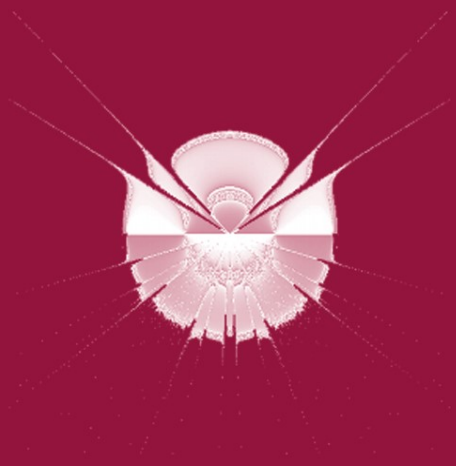
Clara Pizzuti
Marylyn D. Ritchie
Mario Giacobini (Eds.)

LNCS 6023

Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics

8th European Conference, EvoBIO 2010
Istanbul, Turkey, April 2010
Proceedings



Springer

Lecture Notes in Computer Science 6023

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Clara Pizzuti Marylyn D. Ritchie
Mario Giacobini (Eds.)

Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics

8th European Conference, EvoBIO 2010
Istanbul, Turkey, April 7-9, 2010
Proceedings

Springer

Volume Editors

Clara Pizzuti
Institute for High-Performance Computing and Networking (ICAR)
Italian National Research Council (CNR)
Via P. Bucci 41C, 87036 Rende (CS), Italy
E-mail: pizzuti@icar.cnr.it

Marylyn D. Ritchie
Vanderbilt University, Center for Human Genetics Research
Department of Molecular Physiology and Biophysics
519 Light Hall, Nashville, TN 37232, USA
E-mail: ritchie@chgr.mc.vanderbilt.edu

Mario Giacobini
University of Torino, Molecular Biotechnology Center
Department of Animal Production Epidemiology and Ecology
Via Leonardo da Vinci 44, 10095 Grugliasco (TO), Italy
E-mail: mario.giacobini@unito.it

Cover illustration:
"Pelegrina Galathea" by Stayko Chalakov (2009) Aston University, UK

Library of Congress Control Number: 2010922893

CR Subject Classification (1998): J.3, H.2.8, E.1, I.2, F.1, F.2.1

LNCS Sublibrary: SL 1 – Theoretical Computer Science and General Issues

ISSN 0302-9743
ISBN-10 3-642-12210-8 Springer Berlin Heidelberg New York
ISBN-13 978-3-642-12210-1 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper 06/3180

Preface

The field of bioinformatics has two main objectives: the creation and maintenance of biological databases, and the discovery of knowledge from life sciences data in order to unravel the mysteries of biological function, leading to new drugs and therapies for human disease. Life sciences data come in the form of biological sequences, structures, pathways, or literature. One major aspect of discovering biological knowledge is to search, predict, or model specific information in a given dataset in order to generate new interesting knowledge. Computer science methods such as evolutionary computation, machine learning, and data mining all have a great deal to offer the field of bioinformatics. The goal of the 8th European Conference on Evolutionary Computation, Machine Learning, and Data Mining in Bioinformatics (EvoBIO 2010) was to bring together experts in these fields in order to discuss new and novel methods for tackling complex biological problems.

The 8th EvoBIO conference was held in Istanbul, Turkey during April 7–9, 2010 at the Istanbul Technical University. EvoBIO 2010 was held jointly with the 13th European Conference on Genetic Programming (EuroGP 2010), the 10th European Conference on Evolutionary Computation in Combinatorial Optimisation (EvoCOP 2010), and the conference on the applications of evolutionary computation, EvoApplications. Collectively, the conferences are organized under the name Evo* (www.evostar.org). EvoBIO, held annually as a workshop since 2003, became a conference in 2007 and it is now the premiere European event for those interested in the interface between evolutionary computation, machine learning, data mining, bioinformatics, and computational biology. All papers in this book were presented at EvoBIO 2010 in oral or poster presentations and responded to a call for papers that included topics of interest such as biomarker discovery, cell simulation and modeling, ecological modeling, fluxomics, gene networks, biotechnology, metabolomics, microarray analysis, phylogenetics, protein interactions, proteomics, sequence analysis and alignment, and systems biology. A total of 40 papers were submitted to the conference for peer-review. Of those, 15 (37.5%) were accepted for oral presentation and 6 (15%) were accepted for poster presentation. This result is in an overall acceptance rate of 21 (52.5%) papers accepted for publication in the proceedings.

We would first and foremost like to thank all authors who have spent time and effort to produce interesting contributions to this book. We would like to acknowledge Mario Giacobini, of the University of Torino, for his thorough work editing the EvoBIO 2010 volume. We would like to thank the members of the Program Committee for their expert evaluation of the submitted papers, Jennifer Willies from Edinburgh Napier University, for her unfaltering dedication to the coordination of the event over the years, Stephen Dignum, University of Essex, for his excellent work as the Publicity Chair, as well as Cecilia Di Chio,

University of Strathclyde, who assisted Stephen. We would also like to extend special thanks to Şima Uyar from Istanbul Technical University for her outstanding work as the local organizer. Moreover, we would like to thank the following persons and institutes: Computer Engineering Department of Istanbul Technical University, Turkey for local support, Marc Schoenauer INRIA in France and the MyReview team for providing the conference review management system and efficient assistance.

Finally, we want to especially acknowledge the invited speakers who gave two very interesting and inspirational talks: Luca Cavalli-Sforza, Professor Emeritus at the Stanford School of Medicine, and Kevin Warwick, Professor at the University of Reading.

Finally, we hope that you will consider contributing to EvoBIO 2011.

April 2010

Clara Pizzuti
Marylyn D. Ritchie

Organization

EvoBIO 2010, together with EuroGP 2010, EvoCOP 2010, and EvoAPPLICATIONS 2010, was part of EVO* 2010, Europe's premier co-located events in the field of evolutionary computing.

Program Chairs

Clara Pizzuti	Institute for High Performance Computing and Networking National Research Council (ICAR-CNR), Italy
Marylyn D. Ritchie	Vanderbilt University in Nashville, TN, USA

Local Chair

Şima Uyar	Istanbul Technical University, Turkey

Publicity Chair

Stephen Dignum	University of Essex, UK

Proceedings Chair

Mario Giacobini	University of Torino, Italy

Steering Committee

David W. Corne	Heriot-Watt University, Edinburgh, UK
Elena Marchiori	Radboud University, Nijmegen, The Netherlands
Carlos Cotta	University of Malaga, Spain
Jason H. Moore	Dartmouth Medical School in Lebanon, NH, USA
Jagath C. Rajapakse	Nanyang Technological University, Singapore

Program Committee

Jesus S. Aguilar-Ruiz	Pablo de Olavide University, Spain
Wolfgang Banzhaf	Memorial University of Newfoundland, Canada
Jacek Blazewicz	Poznan University of Technology, Poland
Erik Boczko	Vanderbilt University, USA
William Bush	Vanderbilt University, USA

Zehra Catalpete	Instanbul University, Turkey
Carlos Cotta	University of Malaga, Spain
Federico Divina	Pablo de Olavide University, Spain
Jitesh Dundas	Edencore Technologies, USA
Todd Edwards	Miami University, USA
Gary Fogel	Natural Selection, Inc., USA
Raffaele Giancarlo	University of Palermo, Italy
Rosalba Giugno	University of Catania, Italy
Lutz Hamel	University of Rhode Island, USA
Jin-Kao Hao	University of Angers, France
Tom Heskes	Radboud University Nijmegen, The Netherlands
Zhenyu Jia	University of California, Irvine, USA
Mehmet Koyuturk	Case Western Reserve University, USA
Natalio Krasnogor	University of Nottingham, UK
Michael Lones	University of York, UK
Bob MacCallum	Imperial College London, UK
Daniel Marbach	Ecole Polytechnique Fédérale de Lausanne, Switzerland
Elena Marchiori	Radboud University, Nijmegen, The Netherlands
Andrew Martin	University College London, UK
Brett McKinney	University of Alabama, Birmingham, USA
Jason Moore	Dartmouth College, USA
Pablo Moscato	The University of Newcastle, Australia
Alison Motsinger-Reif	North Carolina State University, USA
Vincent Moulton	University of East Anglia, UK
See-Kiong Ng	Institute for Infocomm Research, Singapore
Carlotta Orsenigo	Politecnico di Milano, Italy
Clara Pizzuti	ICAR-CNR, Italy
Michael Raymer	Wright State University, USA
Marylyn Ritchie	Vanderbilt University, USA
Raul Giraldez Rojo	Pablo de Olavide University, Spain
Simona Rombo	University of Calabria, Italy
Jem Rowland	Aberystwyth University, UK
Marc Schoenauer	University of Paris-Sud, France
Ugur Sezerman	Sabanci University, Turkey
Marc L. Smith	Vassar College, USA
El-Ghazali Talbi	University des Sciences et Technologies de Lille, France
Tricia Thornton-Wells	Vanderbilt University, USA
Dave Trudgian	University of Exeter, UK
Alfonso Urso	ICAR-CNR, Italy
Antoine van Kampen	University of Amsterdam, The Netherlands
Carlo Vercellis	Politecnico di Milano, Italy
Tiffani Williams	Texas A&M University, USA

Kai Ye	Leiden Amsterdam Center for Drug Research, The Netherlands
Andreas Zell	University of Tübingen, Germany
Zhongming Zhao	Vanderbilt University, USA
Eckart Zitzler	ETH Zurich, Switzerland
Blaz Zupan	University of Ljubljana, Slovenia

Sponsoring Institutions

- Istanbul Technical University
- Microsoft
- Scientific and Technological Research Council of Turkey
- The Centre for Emergent Computing, Edinburgh Napier University, UK

Table of Contents

Posters

Variable Genetic Operator Search for the Molecular Docking Problem

Salma Mesmoudi[1], Jorge Tavares[2], Laetitia Jourdan[3], and El-Ghazali Talbi[3]

[1] Laboratoire d'Informatique de Paris 6
104 avenue du Président Kennedy
75016 Paris, France
[2] CISUC, Informatics Engineering Department, University of Coimbra
Polo II - Pinhal de Marrocos
3030-252 Coimbra, Portugal
[3] INRIA Lille - Nord Europe Research Centre
40, Avenue Halley Bt.A, Park Plaza
59650 Villeneuve d'Ascq, France
salma.mesmoudi@lip6.fr, jorge.tavares@ieee.org,
{laetitia.jourdan,el-ghazali.talbi}@inria.fr

Abstract. The aim of this work is to present a new hybrid algorithm for the Molecular Docking problem: Variable Genetic Operator Search (VGOS). The proposed method combines an Evolutionary Algorithm with Variable Neighborhood Search. Experimental results show that the algorithm is able to achieve good results, in terms of energy optimization and RMSD values for several molecules when compared with previous approaches. In addition, when hybridized with the L-BFGS local search method it attains very competitive results.

1 Introduction

The protein-ligand complex is the base of drug design. The *in vivo* and *in vitro* development of a new drug is a long and costly process. Thus, docking algorithms have been developed to provide an *in silico* approach to the problem. The aim of protein-ligand docking research is to predict the conformation of a ligand relative to a target protein. To correctly predict a good complex, a docking algorithm must be able to generate several complexes, and recognize which one is the best among these. Therefore, the molecular docking problem can be considered as the optimization of structural and energetic criteria described by an objective function. This function is based on the degrees of flexibility which represent the position and the conformation of the ligand and the receptor (protein).

In the last few years, several docking approaches have been developped [1,2]. More recently, Evolutionary Algorithms (EA) have become a popular choice in molecular docking applications since they usually perform better than other algorithms [3]. The goal of this work is to improve the quality of molecular docking solutions. To accomplish this objective, we propose the hybridization of Variable Neighborhood Search (VNS) [4] with an EA. VNS is a relatively

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2010, LNCS 6023, pp. 1–12, 2010.

recent algorithm which operates by changing the neighborhood of a current solution to explore other regions of the search space. Variable Genetic Operator Search (VGOS) operates in the same manner as the standard VNS with a main difference: instead of using a single solution with works with a population of solutions and each neighborhood is defined by a pair of genetic operators. Later, we will also add a local search method to our approach, to better understand and compare its performance with previous approaches. We apply to and test with this new algorithm to the molecular docking problem.

The rest of the paper is structured as follows. Section 2 presents the VGOS algorithm and related issues. In Section 3 we describe the results obtained with VGOS and the respective analysis. Finally, the main conclusions are discussed in Section 4.

2 Variable Genetic Operator Search

Evolutionary algorithms for the molecular docking problem can be found in the literature since 1993 [5]. A review of these efforts can be found in [6,7].

The original idea of the VGOS is to change genetic operators every time the algorithm starts to converge towards a local minimum. This variation of operators is inspired by the concept used in Variable Neighborhood Search. VNS is a stochastic algorithm in which an ordered set of neighborhood structures $N_k(k = 1, \ldots, n)$ are defined [4]. When a local optimum is obtained, in the neighborhood $N_i(1 \leq i \leq n)$, that is better than the current solution, N_1 becomes the current neighborhood and the local optimum becomes the current solution. Otherwise, if the local optima is worst than the current solution the neighborhood N_{i+1} becomes the current one. These steps are repeated until all neighborhoods are explored without the improvement of the current solution.

2.1 Neighborhood Structures

The important part of the work is supported by the first neighborhood structure (N_1). Therefore, we must be aware that the neighborhood structures order is important and, specifically, the first structure has to be chosen carefully. The other neighborhood structures play the role of checking the first neighborhood structure to escape from local minimum. The way to construct our neighborhoods is very important. In our case, we will base our VGOS on a population of solutions instead of a unique solution and every neighborhood is defined by different crossover and mutation operators. Thus, we make a hybridization between VNS and an EA. Generally, as we increase the number of visited neighborhoods, the chance to escape local optima increases. However, to facilitate the order choice of the neighborhoods, we only use standard genetic operators.

In our work, we adopt an experimental model which uses the main components from [8,9]. The main reason is that *AutoDock* (the EA proposed in [9]) serves as a basis for the large majority of evolutionary-inspired approaches, and thus, the attained results here can be useful on a larger degree.

2.2 Encoding and Evaluation

During the docking process the protein remains rigid whilst the ligand is flexible. An individual represents only the ligand, as the encoding is an indirect representation. A genotype of a candidate solution is encoded by a vector of real-valued numbers which represent the ligand's translation, orientation and torsion angles. Cartesian coordinates represent the translation, three variables in the vector, whereas four variables defining a quaternion represent the orientation. A quaternion can be considered to be a vector (x, y, z) which specifies an axis of rotation with an angle θ of rotation for this axis. For each flexible torsion angle one variable is used. The phenotype of a candidate solution is composed of the atomic coordinates that represent the three-dimensional structure of the ligand. The atomic structure is built from the translation and orientation coordinates in the ligand crystal structure with the application of the torsion angles.

We use the same evaluation function as established in [9]. To identify appropriate binding conformations, it uses an aproximate physical model to compute the energy of a candidate conformation. It uses empirical free energy potentials composed of four commonly used energy terms. The first three terms are pairwise interatomic potentials that account for weak long-range attractive forces and short-range electrostatic repulsive forces. The overall docking energy of a given ligand molecule is expressed as the sum of intramolecular interactions between the complex and the internal energy of the ligand:

$$E_{total} = E_{vdw} + E_{H-bond} + E_{elec} + E_{internal} \tag{1}$$

The first three terms are the intermolecular energies: van der Waals force (E_{vdw}), hydrogen bonding (E_{H-bond}), and electronic potential (E_{elec}). The last term is the internal energy of the ligand. Lower energy means better docking stability. Thus, the aim of any docking method is to minimize the total energy (E_{total}) value.

2.3 Genetic Operators

In this work, genetic operators are important since they define the neighborhoods of the algorithm. In [10,11], several types of mutation and crossover operators are tested when applying them to the molecular docking problem according to their influence in representation properties, such as locality, heritability and heuristic bias. We use operators based on evolutionary strategies. They act in the following way: when undergoing mutation, the new value for a gene x' is obtained from the old value x by adding a random real number sampled from a distribution $U(0, 1)$:

$$x' = x + \sigma \times U(0, 1) \tag{2}$$

The common distribution used for $U(0, 1)$ is the standard Uniform distribution. However, we also replace this distribution with the standard Gaussian distribution,

$N(0,1)$ and, in the same way as the *AutoDock* approach, with a Cauchy distribution. The Cauchy distribution is:

$$C(x,\alpha,\beta) = \frac{\beta}{\pi\beta^2 + (x-\alpha)^2} \tag{3}$$

where $\alpha \leq 0, \beta > 0, -\infty < x < +\infty$ (α and β are parameters that control the mean and spread of the distribution). The two main mutation operators used are Gaussian and Cauchy mutation like in previous works (see e.g., [10,12]). The third operator used is Uniform mutation.

In terms of crossover, we also apply three operators. Several common crossover operators, such as Whole Arithmetical crossover, Discrete crossover, and Blend-α crossover are used. For a complete description of their operation, we refer the reader to the following literature [13,14].

2.4 Algorithm

The order notion is very important in the VGOS algorithm. Because of its importance, the exploration of the first neighborhood was performed by the best operators, thus the choice of the mutation and crossover is a crucial step [11,10]. We use three kinds of mutation and crossover and each EA_{N_k} is based on a specified pair of operators. Each pair is applied on a neighborhood. Thus, the EA_{N_1} is based on Discret crossover and Gaussian mutation, the second EA_{N_2} is constructed from Whole Arithmetic crossover and Cauchy mutation, and finally, the third pair used in EA_{N_3} is Blend-α crossover and Uniform mutation.

Algorithm 1. VARIABLE GENETIC OPERATOR SEARCH

Require: a set of different neighborhood structures N_k for $k = 1, \ldots, max$

1: $pop \Leftarrow$ initial population of random solutions
2: $x_{best} \Leftarrow$ best_element(pop)
3: $k \Leftarrow 1$
4: **repeat**
5: $\quad pop \Leftarrow EA_{N_k}$(pop)
6: $\quad x'_{best} \Leftarrow$ best_element(pop)
7: $\quad$ **if** $f(x'_{best}) > f(x_{best})$ **then**
8: $\quad\quad k \Leftarrow 1$
9: $\quad$ **else**
10: $\quad\quad k \Leftarrow k + 1$
11: $\quad$ **end if**
12: **until** k = max + 1
13: **return** best element found

As described in algorithm 1, from the current population (pop), we identify the best individual (x_{best}) and the current evolutionary algorithm EA_{N_k} is applied to (pop), to obtain a new population. We put in $x\prime_{best}$ the best element from this

new population. If and only if a better solution 'better fitness (f)" has been found (i.e., f($x\prime_{best}$) > $f(x_{best})$), the search procedure EA_{N_1} is restarted from (pop). If no better solution is found (i.e., f($x\prime_{best}$) ≤ $f(x_{best})$) the algorithm moves to the next neighbood (next mutation/crossover) ($EA_{N_{k+1}}$). These operations will be repeated until all neighborhoods (operators) are visited and no improvement is found or the maximum number of generations is reached.

3 Experimentation and Analysis

To perform our experimentation we used instances from the *AutoDock* test suite. The suite is composed of eight protein-ligand complexes. Each complex contains a macromolecule (the protein) and a small substrate or inhibitor molecule (the ligand). The structures of these molecular complexes have been obtained from the Protein Data Bank (PDB). The essential information about the complexes in the test suite are included in Table 1.

Table 1. X-ray crystal structures used in the docking experiments

Protein-ligand complex	PDB	Resolution	Torsion	Size
Alcohol dehydrogenase	1adb	2.4	14	21
Alpha-Thrombin	1bmm	2.6	12	19
Beta-Trypsin	3ptb	1.7	0	7
Carbonic anydrase	1nnb	2.3	9	16
Trypsin	1tnh	1.80	2	9
IGG1-KAPPA DB_3 FAB	2dbl	2.9	6	13
L-Arabinose-binding protein	7abp	1.67	4	11
HIV_{-1} Protease	1hvr	1.80	10	17

We evaluate a resulting ligand conformation by comparing it with the experimental structures using the standard Cartesian root-mean-square deviation (RMSD):

$$RMSD_{lig} = \sqrt{\frac{\sum_{i=1}^{n} dx_i^2 + dy_i^2 + dz_i^2}{n}} \tag{4}$$

where n is the number of atoms in the comparison and dx_i^2, dy_i^2 and dz_i^2 are the deviations between the crystallographic structure and the corresponding coordinates from the predicted structure lig on Cartesian coordinate i. RMSD values below or near 2.0Å can be considered to be a success and the ligand is classified as being docked. On the other hand, a structure with an RMSD just less than 3.0Å is classified as partially docked. Thus, lower values mean that the observed and the predicted structures are similar.

For the first experiments, we consider two versions of a standard evolutionary algorithm. The only difference between them is the local search method whereas the first variant does not include any type of local search, mimicking

the algorithm presented in [15]. The second evolutionary algorithm includes the Solis-Wets algorithm as local search, thus being equivalent to the algorithm contained in the *AutoDock* package [16]. As for the rest of the evolutionary algorithm components, the basic algorithm for our experiments follows what was described in the previous sections in terms of representation, fitness function and genetic operators. The representation and evaluation is the one used by *AutoDock*. The genetic operators used are the Whole Arithmetic crossover and the Gaussian-based mutation operator. Moreover, the evolutionary algorithm is a standard generational algorithm with stochastic tournament selection and weak elitism. Our proposed VGOS algorithm will be tested against these two variants. To elaborate the different neighborhoods, we use three mutation and crossover operators as explained in the previous section.

3.1 Settings

The parameter values were set heuristically, even though we did some additional tests and verified that, within a moderate range, there was no significant difference in the outcomes. In any case, we did not perform a comprehensive study on the influence of different parameter settings and it is possible that a careful fine-tuning of some values could bring slight improvements to the achieved results. For all experiments, the settings of the tested algorithms are the following: Population size: 150, Selection rate: 0.8; Crossover rate: 0.9; Mutation rate: 0.5; Tournament selection rate: 0.95; Tournament replacement rate: 0.9; Number of local search steps: 1000; Maximum number of fitness evalutions: 10 000 000. We should note that, although the maximum number of fitness evaluations is high, in the case that no improvement is found after 10 000 fitness evaluations, the algorithm stops. A local search step counts as a single fitness evaluation. For all experiments the number of runs is equal to 30.

3.2 Experiments

Table 2 shows us the comparison between VGOS and the standard evolutionary approaches, without local search and with the Solis-Wets method. For all molecules the table contains columns with the best energy (Best), the average (Avg) and standard deviation (Std) for the 30 runs. Bold values in the Best and Avg columns indicate the best value for that instance. The most important observation is that VGOS attains the best results for every single instance, in terms of average and for seven instances for the best energy value found. The exception was the complex 1nnb where the best energy was attained by the EA with the Solis-Wets method. These results are encouraging since it shows that VGOS is competitive against the EA that models the evolutionary algorithm used in the *AutoDock* package. The efficiency of VGOS can also be considered superior to the other approaches. For the instances 1adb and 1bmm, the larger ones, VGOS is the only approach to be able to consistently find good solutions since the average is negative in value, i.e., all the solutions found are closer to

Table 2. Summary of optimization results for VGOS according to energy evaluation

Complex		EA + No LS			EA + Solis-Wets			VGOS		
Label	Size	Best	Avg	Std	Best	Avg	Std	Best	Avg	Std
1adb	21	-6.59	335.34	953.64	-3.85	192.36	559.51	**-17.33**	**-15.76**	1.32
1bmm	19	-3.94	5.00	11.68	-4.66	5.04	18.84	**-10.32**	**-4.98**	3.97
1hvr	17	-15.79	-8.67	9.39	-13.81	-10.13	4.07	**-16.32**	**-15.86**	0.27
1nnb	16	-4.53	-2.48	1.69	**-5.18**	-2.16	1.57	-5.17	**-3.74**	1.22
1tnh	9	-5.39	-2.73	1.36	-5.37	-2.49	1.14	**-5.84**	**-4.41**	1.38
2dbl	13	-10.03	-4.66	3.47	-11.43	-4.26	3.06	**-11.74**	**-9.67**	2.84
3ptb	7	-5.49	-3.66	1.02	-5.99	-4.16	1.18	**-6.23**	**-6.05**	0.23
7abp	11	-7.97	-6.78	0.90	-7.90	-6.84	1.14	**-8.87**	**-7.42**	2.64

Table 3. Summary of optimization results for VGOS according to RMSD values

Complex		EA + No LS			EA + Solis-Wets			VGOS		
Label	Size	Best	Best-En	Avg	Best	Best-En	Avg	Best	Best-En	Avg
1adb	21	1.47	1.53	2.78	1.34	1.50	2.79	**0.06**	**0.33**	**0.41**
1bmm	19	2.06	2.06	4.18	1.29	1.29	3.89	**0.09**	**0.25**	**3.74**
1hvr	17	0.29	**0.29**	0.96	0.53	0.65	0.90	**0.24**	0.54	**0.42**
1nnb	16	0.44	0.76	1.84	0.62	**0.64**	2.76	**0.29**	0.65	**1.74**
1tnh	9	0.70	0.73	2.94	0.34	0.34	3.38	**0.07**	**0.16**	**1.32**
2dbl	13	0.57	0.78	2.42	0.53	0.53	2.64	**0.09**	**0.18**	**0.93**
3ptb	7	0.42	0.54	2.07	0.31	0.31	1.45	**0.04**	**0.14**	**0.23**
7abp	11	0.22	0.61	0.65	0.26	**0.45**	0.86	**0.21**	0.79	**0.56**

the best energy value. This is confirmed by the low value of the standard deviation. For the remaining instances, where all the approaches are able to find good solutions, VGOS has an average efficiency of 20% while the EA+Solis-Wets is around 41% and the simple EA is 67%. These values are obtained by calculating the distance of the runs average to the best solution found by the approach.

The same trend is also found when we consider RMSD values. Looking at table 3 we confirm the good performance of VGOS. The table contains for the three approaches and the eight problem instances: the best RMSD value found (Best), the RMSD value associated to the best energy value found contained in the previous table (Best-En) and the average of the RMSD values. The VGOS approach has the best values for the Best and Avg columns for every instance. Only in three occasions VGOS did not attain the best results for Best-En. The exceptions were 1hvr, 1nnb and 7abp. However, the important aspect to consider in this table is, overall, the RMSD values presented are very close zero. This indicates that solutions with a good structure are found. As expected, the instances where this is not verified are the ones where VGOS did not attain the best relationship between energy and RMSD (1hvr, 1nnb and 7abp).

3.3 Experiments with Local Search

The Solis-Wets algorithm, used in *Autodock* [16], is a direct search method with an adaptive step size, which performs a randomized local minimization of a given candidate solution. In [12], it is shown that this method is inefficient for the Molecular Docking problem in the context of evolutionary search. The recommended Local Search algorithm is the Broyden-Fletcher-Goldfarb-Shanno method (L-BFGS) [17].

L-BFGS is a powerful quasi-Newton conjugate gradient method, where both the function to minimize and its gradient must be supplied, but no a priori knowledge about the corresponding Hessian matrix is required. During local search, the maximum number of iterations that can be performed is specified by a parameter of the algorithm. However, the method stops as soon as it finds a local optimum, so the real number of iterations may be smaller than the specified value.

We will now hybridized VGOS with Solis-Wets and L-BFGS to see how the algorithm performs with an additional component. Furthermore, we will compare with the very efficient EA algorithm described in [12].

Table 4. Summary of optimization results for VGOS with LS according to energy evaluation

Complex		EA + L-BFGS			VGOS + L-BFGS			VGOS + Solis-Wets		
Label	Size	Best	Avg	Std	Best	Avg	Std	Best	Avg	Std
1adb	21	-18.30	-14.00	4.80	**-22.80**	**-21.35**	1.50	-17.87	-16.28	3.43
1bmm	19	**-11.99**	-5.52	2.73	-11.91	**-5.64**	4.07	-10.39	-3.95	3.79
1hvr	17	**-32.43**	**-32.13**	0.28	-29.30	-28.80	0.35	-16.37	-16.01	0.52
1nnb	16	**-6.25**	-4.63	1.50	-6.09	**-5.01**	0.90	-5.23	-4.05	-10.39
1tnh	9	**-6.06**	**-5.54**	1.09	-5.83	-4.84	1.35	-5.84	-4.25	1.78
2dbl	13	-12.14	-10.04	2.99	**-12.19**	-9.71	3.70	-11.82	**-10.23**	2.74
3ptb	7	**-6.33**	**-6.22**	0.25	-6.22	-6.09	0.14	-6.23	-5.99	0.48
7abp	11	**-9.12**	**-8.67**	0.37	-8.69	-7.81	0.41	-8.81	-7.84	0.31

Table 4 compares the two variants of VGOS, with Solis-Wets and L-BFGS methods, against the EA with L-BFGS. The columns follow the same structure as table 2. Some important information can be drawn from the presented data. Regarding the best energy values the VGOS approaches are not competitive with the EA. With the exception of the problem instance 1adb, the EA attains all the best results. Although for most of the instances, the VGOS+L-BFGS approach stands close, it is unable to attain a better performance. VGOS with the Solis-Wets performs worse.

However, if we compare the results between VGOS and VGOS hybridized with the L-BFGS method, it is clear that the later performs better. This effect is not observable with the other VGOS hybrid. When looking at the Avg we find a more balanced situation. The EA attains four of the best averages, VGOS+L-BFGS

Table 5. Summary of optimization results for VGOS with LS according to RMSD values

Complex		EA + L-BFGS			VGOS + L-BFGS			VGOS + Solis-Wets		
Label	Size	Best	Best-En	Avg	Best	Best-En	Avg	Best	Best-En	Avg
1adb	21	0.30	0.51	1.64	**0.05**	**0.29**	**0.42**	0.07	0.35	0.46
1bmm	19	0.67	1.10	**3.63**	**0.06**	0.69	3.98	0.08	**0.25**	4.30
1hvr	17	0.50	0.71	0.61	0.25	**0.49**	0.46	**0.21**	0.64	**0.44**
1nnb	16	0.39	0.74	1.84	**0.20**	**0.10**	**1.04**	0.29	0.81	1.74
1tnh	9	0.20	0.94	1.16	0.10	0.67	**1.12**	**0.03**	**0.17**	2.21
2dbl	13	0.35	0.35	1.40	**0.06**	**0.22**	1.06	0.15	**0.22**	**0.75**
3ptb	7	0.24	0.51	0.93	0.02	0.24	**0.23**	**0.01**	**0.14**	0.36
7abp	11	0.35	0.82	0.76	**0.18**	**0.90**	**0.51**	0.25	**0.90**	**0.51**

attains three and VGOS+Solis-Wets one. This primarily reflects the effect of the local search method used. The approaches using L-BFGS show a better performance than the one with the Solis-Wets method. This extendes and supports the findings of [12].

In the same way as before, table 5 displays the overview of the RMSD values, where each line displays the results for a complex. The differences between the three approaches are now between the EA and the VGOS hybrids. Both VGOS approaches attain among themselves the best results., in terms of Best, Best-En and Avg. The EA with the L-BFGS method was unable to obtain a single best result. The only exception was the average for the 1bmm complex. This indicates how well the VGOS algorithm is able to perfom in terms of RMSD, confirming the previous results (see table 3). However, there are no major differences between the two VGOS variants. The influence of a local search method is less visible. It was already shown that the L-BFGS method was more efficient with regards to energy optimization [12]. Thus, it was expected that VGOS with L-BFGS would not attain a significant difference from the other variant.

This table reinforces the importance of the relation between the RMSD value attained with the best energy. The lowest energy found is only good if it corresponds to a low RMSD value. From the Best-En column it is possible to conclude that the distribution of the best values is focused between the two approaches with VGOS. The conclusion we can draw from this is that, although the EA+L-BFGS is more capable to optimize in terms of energy minimization, the other methods (although with higher energies) can reach solutions with a more similar structure to the optimal case.

This is shown by the scatter plots in figures 1 and 2. The plots display for the 1adb and 1bmm complexes the position of the discovered solutions, the relation between energy and RMSD for an optimization run. The closer to the origin of the axis the better. Common patterns can be detected on both plots: there is a clear approximation flow to the origin, whereas the VGOS approaches clearly

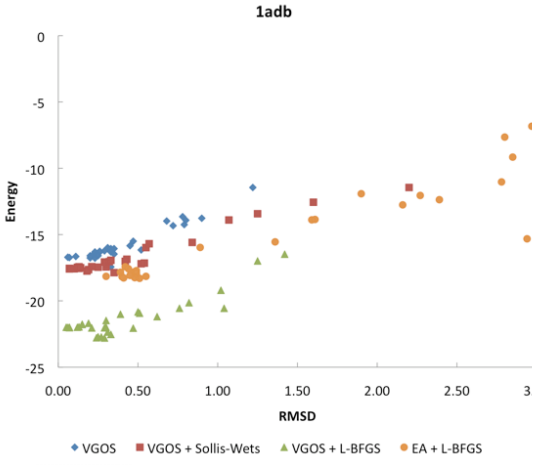


Fig. 1. Scatter plot of an optimization run for the 1adb complex

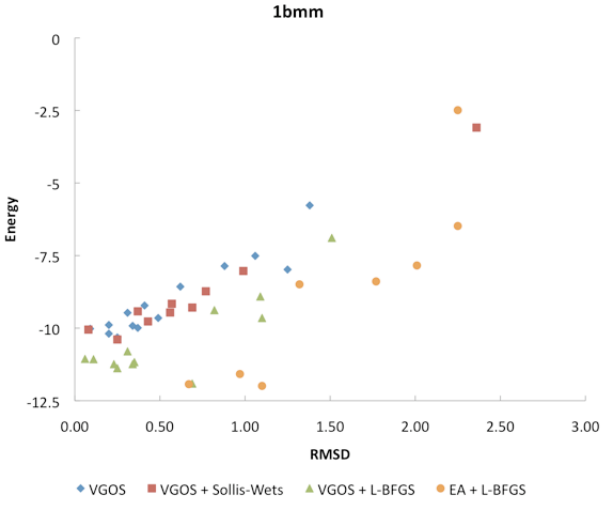


Fig. 2. Scatter plot of an optimization run for the 1bmm complex

approximate more, with a higher concentration. The main differences between the approaches are: 1) VGOS+L-BFGS is the only one that approximates the origin in terms of energy and RMSD (especially for 1adb complex); 2) the majority of the initial solutions for the EA are more distant. The main information from the these plots is that VGOS+L-BFGS drives more solutions with lower energy and RMSD values closer to the desired point.

4 Conclusions and Future Work

We proposed and investigated a new optimization method for the Molecular Docking problem which hybridizes Variable Neighborhood Search with an Evolutionary Algorithm. Variable Genetic Operator Search operates in the same manner as standard VNS , i.e., by changing the neighborhood of a current solution to explore other regions of the search space. The main difference is that instead of working with a single solution, it operates with a population of solutions. Later, we also add local search methods to our approach.

Results show that VGOS is able to attain consistently good results in terms of energy and RMSD values. The method is superior to previous approaches when considering the dual relationship between energy and structural optimization. Moreover, this is achieved by adding the L-BFGS method as local search. In spite of an EA with this local search method provides better results in terms of energy optimization, the VGOS approach is able to discover solutions with better RMSD values. For future work, we intend to study and understand the relation between the molecules topology and operators. With this knowledge, it is our aim to design more suitable operators adapted to each complex.

References

1. Diller, R.M., Verlind, L.M.J.: A critical evaluation of several global optimization algorithms for the purpose of molecular docking. J. Comput. Chem. 20, 1740–1751 (1999)
2. McConkey, B., Sobelove, V., Edlman, M.: The performance of current methods in ligand-protein docking. Current Science 83, 845–855 (2002)
3. Oduguawa, A., Tiwari, A., Fiorentino, S., Roy, R.: Multi-objective optimisation of the protein-ligand docking problem in drug discovery. In: ACM, GECCO 2006 (2006)
4. Maldenovic, N., Hansen, P.: Variable neighborhood search. Computers in Operations Reasearch 24, 1097–1100 (1997)
5. Dixon, J.S.: Flexible docking of ligands to receptor sites using genetic algorithms. In: Proc. of the 9th European Symposium on Structure-Activity Relationships, Leiden, The Netherlands, pp. 412–413. ESCOM Science Publishers (1993)
6. Thomsen, R.: Protein-ligand docking with evolutionary algorithms. In: Fogel, G.B., Corne, D.W., Pan, Y. (eds.) Computational Intelligence in Bioinformatics, pp. 169–195. Wiley-IEEE Press (2008)
7. Moitessier, N., Englebienne, P., Lee, D., Lawandi, J., Corbeil, C.: Towards the development of univeral, fast and highly accurate docking/scoring methods: a long way to go. British Journal of Pharmacology 153, 1–20 (2007)
8. Thomsen, R.: Flexible ligand docking using evolutionary algorithms: investigating the effects of variation operators and local search hybrids. Biosystems 72, 57–73 (2003)
9. Morris, G.M., Goodsell, D.S., Halliday, R.S., Huey, R., Hart, W.E., Belew, R.K., Olson, A.J.: Automated docking using a lamarkian genetic algorithm and an empirical binding free energy function. J. Comput. Chem. 19, 1639–1662 (1998)

10. Tavares, J., Tantar, A.A., Melab, N., Talbi, E.G.: The influence of mutation on protein-ligand docking optimization: a locality analysis. In: Rudolph, G., Jansen, T., Lucas, S., Poloni, C., Beume, N. (eds.) PPSN 2008. LNCS, vol. 5199, pp. 589–598. Springer, Heidelberg (2008)
11. Tavares, J., Melab, N., Talbi, E.G.: An empirical study on the influence of genetic operators for molecular docking optimization. Technical Report RR-6660, INRIA Lille - Nord Europe research Centre (2008)
12. Tavares, J., Mesmoudi, S., Talbi, E.G.: On the efficiency of local search methods for the molecular docking problem. In: Pizzuti, C., Ritchie, M.D., Giacobini, M. (eds.) EvoBIO 2009. LNCS, vol. 5483, pp. 104–115. Springer, Heidelberg (2009)
13. Eiben, A.E., Smith, J.E.: Introduction to Evolutionary Computing (Spring 2003)
14. Lozano, M., Herrera, F., Krasnogor, N., Molina, D.: Real-coded memetic algorithms with crossover hill-climbing. Evolutionary Computation 12, 273–302 (2004)
15. Thomsen, R.: Flexible ligand docking using evolutionary algorithms: investigating the effects of variation operators and local search hybrids. Biosystems 72, 57–73 (2003)
16. Morris, G.M., Goodsell, D.S., Halliday, R.S., Huey, R., Hart, W.E., Belew, R.K., Olson, A.J.: Automated docking using a lamarckian genetic algorithm and empirical binding free energy function. Journal of Computational Chemistry 19, 1639–1662 (1998)
17. Liu, D.C., Nocedal, J.: On the limited memory method for large scale optimization. Mathematical Programming B, 503–528 (1989)

Role of Centrality in Network-Based Prioritization of Disease Genes

Sinan Erten[1] and Mehmet Koyutürk[1,2]

[1] Dept. of Electrical Engineering & Computer Science
[2] Center for Proteomics & Bioinformatics
Case Western Reserve University, Cleveland, OH 44106, USA

Abstract. High-throughput molecular interaction data have been used effectively to prioritize candidate genes that are linked to a disease, based on the notion that the products of genes associated with similar diseases are likely to interact with each other heavily in a network of protein-protein interactions (PPIs). An important challenge for these applications, however, is the incomplete and noisy nature of PPI data. Random walk and network propagation based methods alleviate these problems to a certain extent, by considering indirect interactions and multiplicity of paths. However, as we demonstrate in this paper, such methods are likely to favor highly connected genes, making prioritization sensitive to the skewed degree distribution of PPI networks, as well as ascertainment bias in available interaction and disease association data. Here, we propose several statistical correction schemes that aim to account for the degree distribution of known disease and candidate genes. We show that, while the proposed schemes are very effective in detecting loosely connected disease genes that are missed by existing approaches, this improvement might come at the price of more false negatives for highly connected genes. Motivated by these results, we develop uniform prioritization methods that effectively integrate existing methods with the proposed statistical correction schemes. Comprehensive experimental results on the Online Mendelian Inheritance in Man (OMIM) database show that the resulting hybrid schemes outperform existing methods in prioritizing candidate disease genes.

1 Introduction

Identification of disease-associated genes is an important step toward enhancing our understanding of the cellular mechanisms that drive human diseases, with profound applications in modeling, diagnosis, prognosis, and therapeutic intervention [1]. Genome-wide linkage and association studies in healthy and affected populations provide chromosomal regions containing up to 300 candidate genes possibly associated with genetic diseases [2]. Investigation of these candidates based on sequencing is an expensive task, thus not always a feasible option. Consequently, computational methods are primarily used to prioritize and identify the most likely disease-associated genes by utilizing a variety of data sources such as gene expression [3], functional annotations [4,5], and protein-protein interactions (PPIs) [6,7,8,9,10,3,11]. However, the scope of methods that rely on functional annotations is limited because only a small fraction of genes in the genome are currently annotated.

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2010, LNCS 6023, pp. 13–25, 2010.

In recent years, several algorithms are proposed to incorporate topological properties of PPI networks in understanding genetic diseases [3, 10, 6]. These algorithms mostly focus on prioritization of candidate genes and mainly exploit the notion that the products of genes associated with similar diseases are likely to be close to each other and interact heavily in a network of PPIs. However, an important challenge for these applications is the incomplete and noisy nature of the PPI data [12]. Vast amounts of missing interactions and false positives effect the accuracy of methods based on local network information such as direct interactions and shortest distances. Few global methods based on simulation of random walks [10, 6] and network propagation [11] get around this problem to a certain extent by considering multiple alternate paths and whole topology of PPI networks. Nevertheless, as we demonstrate in this paper, these methods favor genes whose products are highly connected in the network and perform poorly in identifying loosely connected disease genes.

Motivated by this observation, we here propose novel statistical correction methods for network-based disease gene prioritization. These methods aim to assess the significance of the connectivity of a candidate gene to known disease genes with respect to a reference model that takes into account the degree distribution of the PPI network. We show that the proposed correction schemes are very effective in detecting loosely connected disease genes which are generally less studied, thus potentially more interesting for many applications in terms of generating novel biological knowledge. However, we observe that these schemes might perform less favorably in identifying highly connected disease genes. Consequently, we develop several uniform prioritization methods that effectively integrate existing algorithms with the proposed statistical adjustment schemes, with a view to delivering high accuracy irrespective of the network centrality of target disease genes. Comprehensive experimental results show that the resulting hybrid prioritization schemes outperform existing approaches in identifying disease-associated genes.

2 Background and Motivation

There exists a wide range of methods based on the analysis of the topological properties of PPI networks. These methods commonly rely on the expectation that the products of genes that are associated with similar diseases interact heavily with each other. It is important to note that the purpose here is to infer functional associations between genes from functional and physical interactions between their products. For this reason, any reference to interactions between genes in this paper refers to the interactions between their products. Existing methods can be classified into two main categories; (i) localized methods, *i.e.*, methods based on direct interactions and shortest paths between known disease genes and candidate genes [3,7,13], (ii) global methods, *i.e.*, methods that model the information flow in the cell to assess the proximity and connectivity between known disease genes and candidate genes. Several studies show that global approaches, such as random walk and network propagation, clearly outperform local approaches [11,10]. For this reason, we focus on global methods in this paper.

Network-based candidate disease gene prioritization. For a given disease of interest D, the input to the candidate disease gene prioritization problem consists of two sets of

genes, seed set $\mathcal{S}$ and candidate set $\mathcal{C}$. The *seed set* $\mathcal{S}$ specifies prior knowledge on the disease, *i.e.*, it is the set of genes known to be associated with D and diseases similar to D. Each gene $v \in \mathcal{S}$ is also associated with a similarity score $\sigma(v, D)$, indicating the known degree of association between v and D. The similarity score for gene v is computed as the maximum similarity between D and any other disease associated with v (a detailed discussion on computation of similarity scores can be found in [14]). The *candidate set* $\mathcal{C}$ specifies the genes, one or more of which are potentially associated with disease D (e.g., these genes might lie within a linkage interval that is identified by association studies). The overall objective of network based disease prioritization is to use a human PPI network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, to compute a score $\alpha(v, D)$ for each gene $v \in \mathcal{C}$ that represents the likelihood of v to be associated with D.

The PPI network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ consists of a set of gene products $\mathcal{V}$ and a set of undirected interactions $\mathcal{E}$ between these gene products where $uv \in \mathcal{E}$ represents an interaction between $u \in \mathcal{V}$ and $v \in \mathcal{V}$. In this network, the set of interacting partners of a gene product $v \in \mathcal{V}$ is defined as $N(v) = \{u \in \mathcal{V} : uv \in \mathcal{E}\}$. Global prioritization schemes use this network information to compute α by propagating σ over $\mathcal{G}$. Candidate proteins are then ranked according to α and novel genes that are potentially associated with the disease of interest are identified based on this ranking.

Random walk with restarts. This method simulates a random walk on the network to compute the proximity between two nodes by exploiting the global structure of the network [15,16]. It is used in a wide range of applications, including identification of functional modules [17] and modeling the evolution of social networks [18]. Recently, random walk with restarts has also been applied to candidate disease gene prioritization [6,10].

In the context of disease gene prioritization, random walk with restarts is applied as follows. A random walk starts at one of the nodes in $\mathcal{S}$. At each step, the random walk either moves to a randomly chosen neighbor $u \in N$ of the current gene v or it restarts at one of the genes in the seed set $\mathcal{S}$. The probability of restarting at a given time step is a fixed parameter denoted by r. For each restart, the probability of restarting at $v \in \mathcal{S}$ is a function of $\sigma(v, D)$, *i.e.*, the degree of association between v and the disease of interest. After a sufficiently long time, the probability of being at node v at a random time step provides a measure of the functional association between v and the genes known to be associated with D [10,6]. Algorithmically, random-walk based association scores can be computed iteratively as follows:

$$x_{t+1} = (1 - r)P_{\text{RW}}x_t + r\rho. \tag{1}$$

Here, ρ denotes the restart vector with $\rho(u) = \sigma(u, D)/\sum_{v \in \mathcal{S}} \sigma(v, D)$ for $u \in \mathcal{S}$ and 0 otherwise. P_{RW} denotes the stochastic matrix derived from $\mathcal{G}$, *i.e.*, $P_{\text{RW}}(u, v) = 1/|N(v)|$ for $vu \in \mathcal{E}$ and 0 otherwise. For each $v \in \mathcal{V}$, $x_t(v)$ denotes the probability that the random walk will be at v at time t, where $x_0 = \rho$. For each gene v, the resulting random-walk based association score is defined as $\alpha_{\text{RW}}(v, D) = \lim_{t \to \infty} x_t(v)$.

Network propagation. Propagation based models have been previously shown to be effective in network based functional annotation of proteins [19]. In recent work, Vanunu and Sharan [11] propose a network propagation algorithm to compute the association between candidate proteins and known disease genes. They define a prioritization function which models simulation of an information pump that originates at the seed sets.

This idea is very similar to that of random walk with restarts, with one key difference. Namely, in network propagation, the flow of information is normalized by not only the total outgoing flow from each node, but also the total incoming flow into each node. In other words, the matrix P_{RW} is replaced my a matrix P_{NP}, in which each entry is normalized with respect to row and column sums. The resulting propagation based model can also be simulated iteratively as follows:

$$y_{t+1} = (1 - r)P_{\text{NP}}y_t + r\rho. \quad (2)$$

Here, the propagation matrix P_{NP} is computed as $P_{\text{NP}}(u, v) = 1/\sqrt{|N(u)||N(v)|}$ for $uv \in \mathcal{E}$, 0 otherwise. For each $v \in \mathcal{V}$, $y_t(v)$ denotes the amount of disease association information at node v at step t, where $y_0 = \rho$. For each gene v, the resulting network propagation based association score is defined as $\alpha_{\text{NP}}(v, D) = \lim_{t\to\infty} y_t(v)$. In this model, $0 \leq r \leq 1$ is also a user-defined parameter that is used to adjust the relative importance of prior knowledge and network topology.

Role of network centrality. In order to motivate our approach, we evaluate here the performance of random walk with restarts and network propagation with respect to the network degree (number of known interactions) of candidate genes. As shown in Figure 1(a), these methods are clearly biased toward scoring highly connected proteins higher. In this figure, the performance measure is the average rank of the true candidate protein among other 99 proteins in the same linkage interval. As evident in the figure, existing global methods work very well in predicting highly connected proteins, whereas they perform quite poorly for loosely connected proteins, especially for those with degree less than 6. Furthermore, as seen in Figure 1(b), the degree distribution of known disease genes is slightly biased toward highly connected genes, however there exist many disease genes that are loosely connected as well. For this reason, it is at least as important to correctly identify loosely connected disease genes as to identify those that are highly connected, in order to remove the effect of ascertainment bias in PPI data and known disease associations.

The dependency of performance on network degree can be understood by carefully inspecting the formulation of random walk and network propagation models. Random walk with restarts is actually a generalization of Google's well-known page-rank algorithm [20], such that for $r = 0$, α is solely a measure of network centrality. Therefore, for any $r > 0$ (in our experiments, we observe that $r = 0.3$ is optimal for the performance of both algorithms after running the algorithms with small increments of r values; this is also the setting used in Figure 1), $\alpha(v, D)$ contains a component that represents the network centrality of v, in addition to its association with D. Network propagation alleviates this problem by normalizing the incoming flow into a gene, therefore provides a slightly more balanced performance compared to random walk with restarts. However, as evident in the figure, its performance is still influenced heavily by node degrees. Motivated by these insights, we argue that the association scores computed by these algorithms have to be statistically adjusted with respect to reference models that take into account the degree distribution of the network.

3 Methods

In this section, we propose several reference models for assessing the significance of network-based disease association scores. Subsequently, we discuss how these mod-

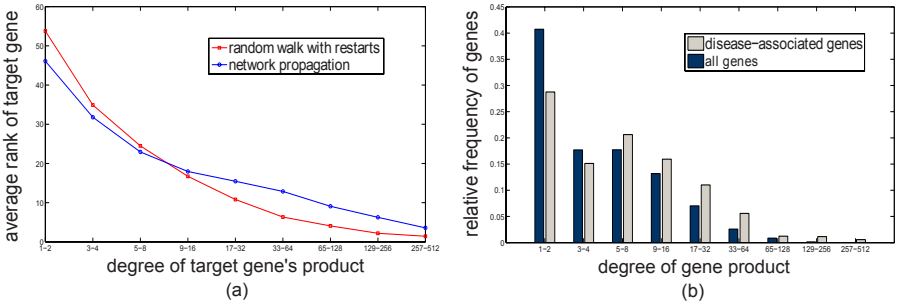


Fig. 1. (a) The effect of degree to the performance of existing global approaches. x-axis is the degree range while y-axis represents the average rank for the true disease genes. (b) Histogram of the degrees of disease genes and all genes in the network.

els can be used in conjunction with existing methods to obtain uniform prioritization schemes that can deliver high accuracy regardless of centrality of candidate genes.

3.1 Reference Models for Statistical Adjustment

Here, we consider three alternate reference models for assessing the significance of disease association scores obtained by random walk with restarts or network propagation: (i) a model that generates a separate background population for each candidate gene based on the degree distribution of the seed set, (ii) a model that generates a background population for each group of candidates with similar degree for a fixed seed set, (iii) a model that assesses the log-likelihood of the association of a gene with the seed set with respect to its network centrality. Here, for the sake of clarity, we describe each model assuming that random walk based restarts is used to compute raw association scores (we also drop the subscript RW from our notation for simplicity).

Reference model based on seed degrees. The objective here is to generate a reference model that captures the degree distribution of seed proteins accurately. To this end, we compare the association score $\alpha(v, D)$ for each protein with scores computed using random seed sets (by preserving the degree distribution of the seed genes). The expectation here is that false positives that correspond to centralized and highly connected proteins will have high association scores even with respect to these randomly generated seed sets. Furthermore, this model aims to balance the effect of highly connected known disease genes with that of loosely connected ones.

Given a disease D, seed set $\mathcal{S}$, and candidate set C, this reference model is implemented as follows:

- We first compute network-based association scores $\alpha(v, D)$ for the original seed set $\mathcal{S}$, using the procedure described in Equation 1.
- Then, based on the original seed set $\mathcal{S}$, we generate a random instance $\mathcal{S}^{(i)}$ that represents $\mathcal{S}$ in terms of degree distribution. $\mathcal{S}^{(i)}$ is generated as follows:
  - First, a bucket $\mathcal{B}(u)$ is created for each protein $u \in \mathcal{S}$.
  - Then, each protein $v \in \mathcal{V}$ is assigned to bucket $\mathcal{B}(u)$ if $|N(v) - N(u)| < |N(v) - N(u')|$ for all $u' \in \mathcal{S}$, where ties are broken randomly.

- Subsequently, $\mathcal{S}^{(i)}$ is generated by choosing a protein from each bucket uniformly at random. It can be observed that each protein in $\mathcal{S}$ is represented by exactly one protein in $\mathcal{S}^{(i)}$, thus the total degree of proteins in $\mathcal{S}^{(i)}$ is expected to be very close to that of $\mathcal{S}$.

– For $1 \leq i \leq n$, the association scores $\alpha^{(i)}$ for seed set $\mathcal{S}^{(i)}$ are computed using Equation 1. Here, n is a sufficiently large number that is used to obtain a representative sampling $\{\alpha^{(1)}, \alpha^{(2)}, \alpha^{(3)}, ..., \alpha^{(n)}\}$ of the population of association scores for seed sets that match the size and degree distribution of $\mathcal{S}$ (we use $n = 1000$ in our experiments).
– We then estimate the mean of this distribution as $\mu_{\mathcal{S}} = \sum_{1 \leq i \leq n} \alpha^{(i)}/n$ and the standard deviation as $\sigma^2_{\mathcal{S}} = \sum_{1 \leq i \leq n}((\alpha^{(i)} - \mu_{\mathcal{S}})(\alpha^{(i)} - \mu_{\mathcal{S}})^T)/(n-1)$.
– Finally, we compute the seed degree adjusted association score for each gene v as $\alpha_{\text{SD}}(v, D) = (\alpha(v, D) - \mu_{\mathcal{S}})/\sigma_{\mathcal{S}}$.

Note that, since the multiple hypotheses being tested here are compared and ranked against each other (as opposed to accepting/rejecting individual hypotheses), it is not necessary to perform correction for multiple hypothesis testing.

Reference model based on candidate degree. This reference model aims to assess the statistical significance of the association score $\alpha(v, D)$ of a gene $v \in \mathcal{V}$ with respect to seed set $\mathcal{S}$ based on a population of association scores that belong to genes with degree similar to that of v. This reference model is generated as follows:

– First, we compute the network-based association vector α with respect to the given seed set $\mathcal{S}$, again using Equation 1.
– Then, for each candidate gene $v \in \mathcal{C}$, we select the n genes in the network with smallest $|N(v) - N(u)|$ to create a representative set $\mathcal{M}(v)$ that contains the n genes most similar to v in terms of their degree ($n = 1000$ in our experiments).
– Subsequently, for each gene $v \in \mathcal{C}$, we estimate the mean association score of its representative population as $\mu(v) = \sum_{u \in \mathcal{M}(v)} \alpha(u)/|\mathcal{M}(v)|$ and the standard deviation of association scores as $\sigma^2(v) = \sum_{u \in \mathcal{M}(v)} (\alpha_{\mathcal{S}}(u) - \mu(v))/(|\mathcal{M}(v)| - 1)$.
– Finally, we compute the candidate degree adjusted association score of each candidate gene v as $\alpha_{\text{CD}}(v, D) = (\alpha_{\mathcal{S}}(v, D) - \mu(v))/\sigma(v)$.

Likelihood-ratio test using eigenvector centrality. Here, we assess the association of a gene with the seed set using a likelihood-ratio test. More precisely, considering $\alpha_{\text{RW}}(v, D)$ as the likelihood of v being associated with the seed set $\mathcal{S}$ for disease D, we compare this likelihood with the likelihood of v being associated with any other gene product in the network. To compute the likelihood of v's association with any other gene in the network, we use eigenvector centrality [20], which is precisely equal to the random walk based association score of v for zero restart probability ($r = 0$). Indeed, setting $r = 0$ corresponds to the case where the seed set is empty, thereby making the resulting association score a function of the gene's network centrality. For each $v \in \mathcal{C}$, the eigenvector centrality based log-likelihood score is computed as:

$$\alpha_{\text{EC}}(v, D) = \log \frac{\alpha^{(r>0)}(v, D)}{\alpha^{(r=0)}(v, D)}. \tag{3}$$

3.2 Uniform Prioritization

As we demonstrate in the next section, the adjustment strategies presented improve the performance of global prioritization algorithms in identifying loosely connected disease genes. However, this comes at the price of increased number of false negatives for highly connected disease genes. Motivated by this observation, we propose several hybrid scoring schemes that aim to take advantage of both raw and statistically adjusted association scores. The idea here is to derive a uniform prioritization method that uses the adjusted scores for loosely connected candidate genes, while using the raw scores for highly connected candidate genes.

For this purpose, we first sort the raw crosstalk scores (α_{RW} or α_{NP}) of candidate genes in descending order. Let $R_{\text{RAW}}(v)$ denote the rank of gene $v \in \mathcal{C}$ in this ordering. Clearly, for $u, v \in \mathcal{C}$, $R_{\text{RAW}}(v) < R_{\text{RAW}}(u)$ indicates v is more likely to be associated with the disease than u is. Similarly, we sort the statistically adjusted association scores (α_{SD}, α_{CD}, or α_{EC}) in descending order, to obtain a rank $R_{\text{ADJ}}(v)$ for each gene $v \in \mathcal{C}$. We propose three alternate strategies for merging these two rankings to obtain a uniform ranking R_{UNI}, where the objective is to have $R_{\text{UNI}}(v) < R_{\text{UNI}}(u)$ if gene v is associated with the disease, while gene u is not. Once $R_{\text{UNI}}(v)$ is obtained using one of the following methods, we map it into the interval $[1, |\mathcal{C}|]$ in the obvious way.

Uniform prioritization based on the degree of candidate gene. This uniform prioritization scheme chooses the ranking of each candidate gene based on its own degree. Namely, for a given user-defined threshold λ, we define $R^{(\text{C})}_{\text{UNI}}$ as:

$$R^{(\text{C})}_{\text{UNI}}(v) = \begin{cases} R_{\text{RAW}}(v) & \text{if } |N(v)| > \lambda \\ R_{\text{ADJ}}(v) & \text{otherwise} \end{cases} \tag{4}$$

for each $v \in \mathcal{C}$. Thus the ranking of a highly-connected gene is based on its raw association score, while that of a loosely-connected gene is based on the statistical significance of its association score. Note that, with respect to this definition, the ranking of two genes can be identical (but there cannot be more than two genes with identical ranking). In this case, the tie is broken based on the unused ranking of each gene.

Optimistic uniform prioritization. This approach uses the best available ranking for each candidate gene, based on the expectation that a true disease gene is more likely to show itself in at least one of the rankings as compared to a candidate gene that is not associated with the disease. Namely, we define $R^{(\text{O})}_{\text{UNI}}$ as:

$$R^{(\text{O})}_{\text{UNI}}(v) = \begin{cases} R_{\text{RAW}}(v) & \text{if } R_{\text{RAW}}(v) < R_{\text{ADJ}}(v) \\ R_{\text{ADJ}}(v) & \text{otherwise} \end{cases} \tag{5}$$

for each $v \in \mathcal{C}$. Again, ties are broken based on the unused rankings.

Uniform prioritization based on degree of known disease genes. Based on the notion that some diseases are studied more in detail compared to other diseases, we expect the degrees of genes associated with similar diseases to be somewhat close to each other. Statistical tests on disease associations currently available in the OMIM (Online Mendelian Inheritance in Man) database confirms this expectation (data not shown). We take advantage of this observation to approximate the network degree of the unknown disease gene in terms of the degrees of the known disease genes. This enables having a

global criterion for choosing the preferred ranking for all genes, as opposed to the gene-specific (or "local") criteria described above. For a given seed set $\mathcal{S}$, we first compute $\overline{d}(\mathcal{S}) = (\sum_{u \in \mathcal{S}} |N(u)|)/|\mathcal{S}|$. Subsequently, if $\overline{d}(\mathcal{S}) > \lambda$ (where λ is defined as above), we set $R^{(S)}_{\text{UNI}}(v) = R_{\text{RAW}}(v)$ for all $v \in \mathcal{C}$, otherwise, we set $R^{(S)}_{\text{UNI}}(v) = R_{\text{ADJ}}(v)$.

4 Results

In this section, we comprehensively evaluate the performance of the methods presented in the previous section.

4.1 Datasets

In our experiments, we use the human PPI data obtained from NCBI Entrez Gene Database [21]. This database integrates interaction data from several other databases available, such as HPRD, BioGrid and BIND. After the removal of nodes with no interactions, the final PPI network contains 8959 proteins and 33528 distinct interactions among these proteins.

We obtain disease information from Online Mendelian Inheritance in Man (OMIM) database. OMIM provides a publicly accessible and comprehensive database of genotype-phenotype relationship in humans. We map genes associated with diseases to our PPI network and remove those diseases for which we are unable to map more than two associated genes. After this step, we have a total of 206 diseases with at least 3 associated genes. Number of genes associated with these diseases ranges from 3 to 36, with the average number of associations for each disease being approximately 6.

4.2 Experimental Setting

In order to evaluate the performance of different methods in terms of accurately prioritizing disease-associated genes, we apply leave-one-out cross-validation. For each gene that is associated with a disease, we conduct the following experiment:

- We remove a gene from the set of genes associated with a particular disease.
- We generate an artificial linkage interval, containing this removed gene with other 99 genes located nearest in terms of the genomic distance. Note that, according to our experiments, the size of candidate set does not have a significant effect on the performance gap between different methods as long as it is greater than 20 (data not shown).
- Using each of the methods described in the previous section, we obtain a ranking of candidate genes and use this ranking to predict disease genes. Note that, due to space considerations, we only use random walk with restarts in conjunction with the proposed statistical correction and uniform prioritization methods, however, these methods can also be applied to network propagation straightforwardly.

In order to systematically compare the performance of different methods, we use the following evaluation criteria:

Average rank. Average rank of the correct disease gene among all candidate genes, computed across all disease. Clearly, a lower average rank indicates better performance.

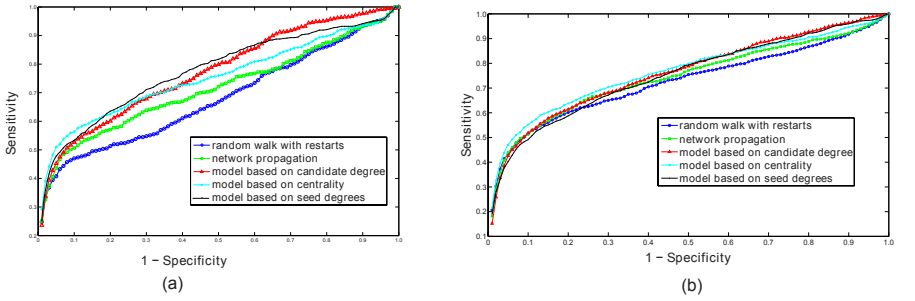


Fig. 2. ROC curves of proposed statistical adjustment schemes and existing methods for (a) cases in which true disease gene has degree at most five, (b) all disease genes. All of the proposed adjustment schemes outperform existing methods.

ROC curves. We also plot ROC curves, *i.e.*, *sensitivity vs.* 1-*specificity*, by thresholding the rank to be considered a "predicted disease gene" from 1 to 100. *Sensitivity* (recall) is defined as the percentage of true disease genes that are ranked above the particular threshold, whereas *specificity* is defined as the percentage of all genes that are ranked below the threshold. The area under ROC curve (AUC) is used as another measure to assess the performance of different methods.

Percentage of the disease genes ranked in top 1% and 5%. Percentages of true disease genes that are ranked as one of the genes in the top 1% (practically, the top gene) and also in the top 5% among all candidates are listed separately.

4.3 Performance of Statistical Adjustment Schemes

As mentioned before, the performance of the global methods is highly biased with the degree of the true candidate protein. The effect of the degree of true disease gene on the performance of global methods is demonstrated in Figure 1. To investigate the effect of the proposed statistical correction schemes on accurate ranking of low-degree proteins, we first compare the ROC curves achieved by different methods by considering the true disease genes with degree $\leq$ 5. These results are shown in Figure 2(a) and Table 1. As seen in the figure, all of the three statistical adjustment schemes outperform existing

Table 1. The effect of statistical adjustment on performance. Average Rank of the true disease genes and AUC values are listed. To demonstrate the effect of connectivity, we also provide separate results for the cases in which the degree of true disease gene is $\leq$ 5 and $>$ 5.

	All Genes		Degrees$\leq$ 5		Degrees$>$ 5	
Method	Avg. Rank	AUROC	Avg. Rank	AUROC	Avg. Rank	AUROC
Network Propagation	26.32	0.74	33.12	0.61	18.29	0.83
Random walk w/ restarts	28.02	0.73	37.73	0.62	17.56	0.84
Based on seed degree	25.55	0.75	26.10	0.73	24.43	0.78
Based on candidate degree	24.62	0.76	26.46	0.72	23.66	0.79
Based on centrality	24.55	0.76	26.27	0.73	23.16	0.79

Table 2. Performance of all combinations of uniform prioritization methods

	Candidate deg.			Seed deg.			Centrality		
	$R^{(C)}_{UNI}$	$R^{(O)}_{UNI}$	$R^{(S)}_{UNI}$	$R^{(C)}_{UNI}$	$R^{(O)}_{UNI}$	$R^{(S)}_{UNI}$	$R^{(C)}_{UNI}$	$R^{(O)}_{UNI}$	$R^{(S)}_{UNI}$
Avg. Rank	23.22	24.33	23.30	25.01	25.29	25.42	24.95	24.92	24.02
AUROC	0.76	0.76	0.77	0.75	0.75	0.76	0.75	0.75	0.76
Perc. ranked in top 1%	21.7	19.4	14.7	18.4	18.5	19.3	20.0	20.5	21.3
Perc. ranked in top 5%	45.1	44.4	42.1	45.5	44.1	41.2	46.3	45.7	47.0

methods for these genes. Furthermore, as evident in Figure 2(b), when all genes are considered, the statistical adjustment schemes still perform better than existing methods. However, as seen in the figure, the performance difference is minor because of the relatively degraded performance of statistical adjustment schemes for highly connected genes. Next, we investigate how the proposed uniform prioritization methods improve the performance of these statistical adjustment schemes.

4.4 Performance of Uniform Prioritization

Here, we systematically investigate the performance of the proposed uniform prioritization methods, by considering the combination of each of these methods with each of the three statistical adjustment methods (a total of nine combinations). In these experiments, the degree threshold λ is set to 5. For convenience, we refer to each uniform prioritization method using the corresponding ranking symbol introduced in the previous section ($R^{(C)}_{UNI}$, $R^{(O)}_{UNI}$, or $R^{(S)}_{UNI}$).

The average rank and AUC for the performance of the nine combinations of proposed methods are listed in Table 2. As seen in the table, while all methods improve upon the performance of raw statistical adjustment schemes, it is difficult to choose between the proposed methods. We suggest that the hybrid method based on candidate degree ($R^{(C)}_{UNI}$), combined with statistical adjustment based on candidate degree, can be considered the "winner", since this approach provides the best accuracy in correctly predicting the disease protein as the top candidate (21.7%) and it provides the lowest

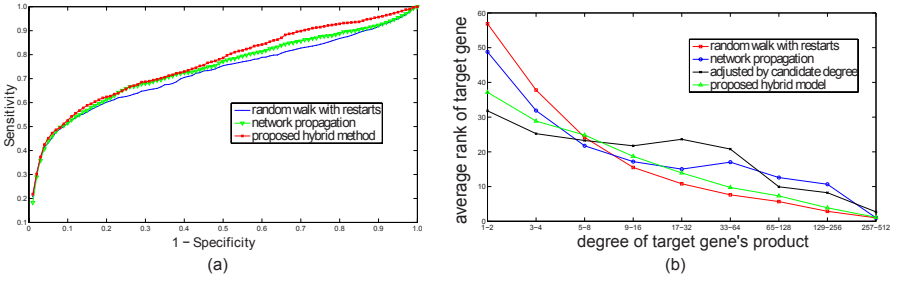


Fig. 3. (a) ROC curves to compare the proposed method with existing global approaches. (b)The effect of the degree of target gene on the performance of existing global approaches, adjusted method based on candidate degrees as well as the our final uniform prioritization method.

average rank of the true candidate gene (23.22). We compare this combination of proposed algorithms with existing global methods in Table 3 and Figure 3(a). These results clearly show that our final uniform prioritization scheme outperforms existing methods with respect to all performance criteria. Furthermore, careful inspection of average rank with respect to the degree of true disease gene in Figure 3(b) shows that, this method almost matches the performance of the best performing algorithm for each degree regime. Namely, if the target gene has low degree, our uniform prioritization method performs close to statistical adjusted random walk, while it performs close to raw random walk for high-degree target genes.

4.5 Case Example

Here, we provide a real example to demonstrate the power of the proposed method in identifying loosely connected disease genes. We focus on *Microphthalmia* which is a disease that has 3 genes directly associated with it in our PPI network, namely *SIX6*, *CHX10* and *BCOR*. In our experiments, we remove *SIX6* and try to predict this gene

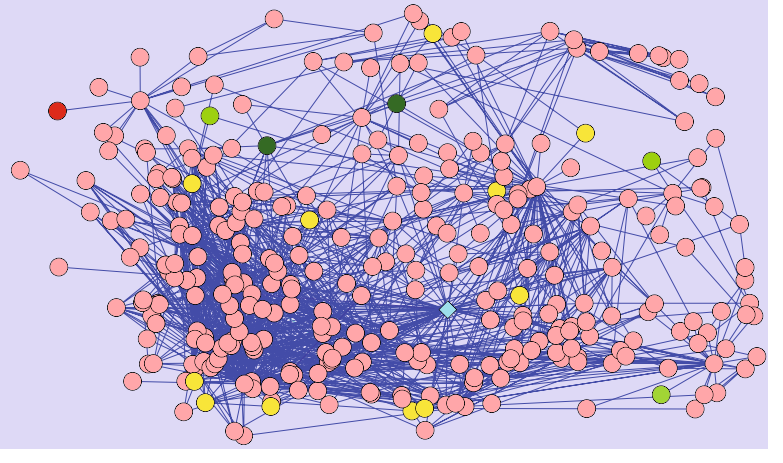


Fig. 4. Case example for the Microphthalmia disease. Products of genes associated with Microphthalmia or a similar disease are shown by green circles, where the intensity of green is proportional to the degree of similarity. The target disease gene that is left out in the experiment and correctly ranked first by our algorithm is represented by a red circle. The gene that is incorrectly ranked first for both of the existing global approaches is shown by a diamond. Other candidate genes that are prioritized are shown by yellow circles.

Table 3. Comparison of the proposed method with existing global approaches. The proposed method outperforms others with respect to all performance criteria.

METHOD	Avg. Rank	AUROC	Perc. Ranked in top 1%	Perc. Ranked in top 5%
Proposed Hybrid Method	23.22	0.76	21.7	45.1
Network propagation	26.32	0.74	18.2	43.2
Random walk w/ restarts	28.02	0.73	20.7	43.9

using the other two genes, as well genes associated with diseases similar to Microphthalmia. This experiment is illustrated in Figure 4. The figure shows the 2-neighborhood of proteins *SIX6*, *CHX10* and *BCOR*. As seen in the figure, the global methods fail because the product of *SIX6* is not a centralized protein with a degree of only 1. Thus, random walk with restarts model ranks this true gene as 26^{th} and network propagation ranks it 16^{th} among 100 candidates. On the other hand, our method is able to correctly rank this gene as the 1^{st} candidate. Both random walk and network propagation rank the gene *AKT1* top among all candidates, which, not surprisingly, is a high degree node (78), also connected to other hub gene products.

5 Conclusion

In this paper, we have shown that approaches based on global network properties in prioritizing disease-associated genes are highly biased by the degree of the candidate gene, thus perform poorly in detecting loosely connected disease genes. We proposed several statistical adjustment strategies that improve the performance, particularly in identifying loosely connected disease genes. We have shown that, when these adjustment schemes are used together with existing global methods, the resulting method outperforms existing approaches significantly. These results clearly demonstrate that, in order to avoid exacerbation of ascertainment bias and propagation of noise, network-based biological inference methods have to be supported by statistical models that take into account the degree distribution.

Acknowledgements

This work is supported in part by NSF CAREER Award CCF-0953195.

References

1. Brunner, H.G., van Driel, M.A.: From syndrome families to functional genomics. Nat. Rev. Genet. 5(7), 545–551 (2004)
2. Glazier, A.M., Nadeau, J.H., Aitman, T.J.: Finding Genes That Underlie Complex Traits. Science 298(5602), 2345–2349 (2002)
3. Lage, K., Karlberg, E., Storling, Z., Olason, P., Pedersen, A., Rigina, O., Hinsby, A., Tumer, Z., Pociot, F., Tommerup, N., Moreau, Y., Brunak, S.: A human phenome-interactome network of protein complexes implicated in genetic disorders. Nat. Bio. 25(3), 309–316 (2007)
4. Adie, E., Adams, R., Evans, K., Porteous, D., Pickard, B.: SUSPECTS: enabling fast and effective prioritization of positional candidates. Bioinformatics 22(6), 773–774 (2006)
5. Turner, F., Clutterbuck, D., Semple, C.: Pocus: mining genomic sequence annotation to predict disease genes. Genome Biology 4(11), R75 (2003)
6. Chen, J., Aronow, B., Jegga, A.: Disease candidate gene identification and prioritization using protein interaction networks. BMC Bioinformatics 10(1), 73 (2009)
7. Oti, M., Snel, B., Huynen, M.A., Brunner, H.G.: Predicting disease genes using protein-protein interactions. J. Med. Genet. (2006), jmg.2006.041376
8. Goh, K.I., Cusick, M.E., Valle, D., Childs, B., Vidal, M., Barabási, A.L.A.A.: The human disease network. PNAS 104(21), 8685–8690 (2007)

9. Ideker, T., Sharan, R.: Protein networks in disease. Genome research 18(4), 644–652 (2008)
10. Köhler, S., Bauer, S., Horn, D., Robinson, P.N.: Walking the interactome for prioritization of candidate disease genes. Am. J. Hum. Genet. 82(4), 949–958 (2008)
11. Vanunu, O., Sharan, R.: A propagation based algorithm for inferring gene-disease associations. In: Proceedings of German Conference on Bioinformatics (2008)
12. Edwards, A.M., Kus, B., Jansen, R., Greenbaum, D., Greenblatt, J., Gerstein, M.: Bridging structural biology and genomics: assessing protein interaction data with known complexes. Trends in Genetics 18(10), 529–536 (2002)
13. George, R.A., Liu, J.Y., Feng, L.L., Bryson-Richardson, R.J., Fatkin, D., Wouters, M.A.: Analysis of protein sequence and interaction data for candidate disease gene prediction. Nucl. Acids Res. 34(19), e130 (2006)
14. van Driel, M.A., Bruggeman, J., Vriend, G., Brunner, H.G., Leunissen, J.A.: A text-mining analysis of the human phenome. EJHG 14(5), 535–542 (2006)
15. Lovász, L.: Random walks on graphs: A survey. Combinatorics, Paul Erdos is Eighty 2, 353–398 (1996)
16. Tong, H., Faloutsos, C., Pan, J.Y.: Random walk with restart: fast solutions and applications. Knowledge and Information Systems 14(3), 327–346 (2008)
17. Macropol, K., Can, T., Singh, A.: Rrw: repeated random walks on genome-scale protein networks for local cluster discovery. BMC Bioinformatics 10(1), 283 (2009)
18. Tong, H., Faloutsos, C.: Center-piece subgraphs: problem definition and fast solutions. In: KDD 2006: Proceedings of the 12th ACM SIGKDD, pp. 404–413. ACM, New York (2006)
19. Nabieva, E., Jim, K., Agarwal, A., Chazelle, B., Singh, M.: Whole-proteome prediction of protein function via graph-theoretic analysis of interaction maps. Bioinf. 21, i302–i310 (2005)
20. Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. Computer Networks and ISDN Systems 30, 107–117 (1998)
21. Maglott, D., Ostell, J., Pruitt, K.D., Tatusova, T.: Entrez Gene: gene-centered information at NCBI. Nucl. Acids Res. 35(suppl. 1), D26–D31 (2007)

Parallel Multi-Objective Approaches for Inferring Phylogenies

Waldo Cancino[1], Laetitia Jourdan[1], El-Ghazali Talbi[1], and Alexandre C.B. Delbem[2]

[1] INRIA Lille Nord Europe
Villeneuve d'Ascq, France
{Waldo.Cancino,Laetitia.Jourdan,El-Ghazali.Talbi}@inria.fr
[2] Institute of Mathematics and Computer Science
University of Sao Paulo, Brazil
acbd@icmc.usp.br

Abstract. The inference of the phylogenetic tree that best express the evolutionary relationships concerning data is one of the central problem of bioinformatics. Several single optimality criterion have been proposed for the phylogenetic reconstruction problem. However, different criteria may lead to conflicting phylogenies. In this scenario, a multi-objective approach can be useful since it could produce a set of optimal trees according to multiple criteria. PhyloMOEA is a multi objective evolutionary approach applied to phylogenetic inference using maximum parsimony and maximum likelihood criteria. On the other hand, the computational power required for phylogenetic inference of large alignments easily surpasses the capabilities of single machines. In this context, the parallelization of the heuristic reconstruction methods can not only help to reduce the inference execution time but also improve the results quality and search robustness. On the other hand, The PhyloMOEA parallelization represents the next development step in order to reduce the execution time. In this paper, we present the PhyloMOEA parallel version developed using the ParadisEO framework. The experiments conducted show significant speedup in the execution time for the employed datasets.

Keywords: Phylogenetic Inference, Multi-Objective Optimization, Parallel Computing.

1 Introduction

Phylogenetic inference is one of the central problems in computational biology. It consists of finding the best tree that explains the evolutionary history of species from a given dataset. Various phylogenetic reconstruction methods have been proposed in the literature [1]. Most of them use one optimality criterion (or objective function) to evaluate possible solutions in order to determine the best tree. The number of possible solutions (trees) grows exponentially with the number of species to be analyzed. Moreover, costly optimality criterion for the evaluation of trees can be also employed. Thus, an exhaustive search is infeasible for

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2010, LNCS 6023, pp. 26–37, 2010.

moderate datasets containing a few hundreds of species. Several heuristic search methods based mainly in hill-climbing [2], evolutionary algorithms (EAs) [3], [4] and other techniques are able to find reasonable solutions in acceptable time.

Numerous studies [5], [6] have shown important differences in the results obtained by applying distinct reconstruction methods to the same input data. In this regard, a multi-objective approach can be a relevant contribution since it can search for phylogenies using more than a single criterion.

One of the first studies that models phylogenetic inference as a multi-objective optimization problem (MOOP) was developed by the author of this paper [7]. In this approach, the multi-objective approach used the maximum parsimony [8] and maximum likelihood [9] as optimality criteria. The proposed multi-objective evolutionary algorithm (MOEA), called PhyloMOEA, produces a set of distinct solutions representing a trade-off between the considered objectives.

On the other hand, the search and evaluation of big datasets largely surpass the memory and processing capability of a single machine. In this context, parallel and distributed computing can be used not only to speedup the search, but also to improve the solution quality, search robustness and to solve larger problem instances [10]. Several studies propose parallel fine-grained and coarse-grained implementations for the tree search and for the evaluation of each solution [11]. In this paper, we present a new parallel PhyloMOEA version developed using the ParadisEO metaheuristic framework [12]. New experiments from this parallel approach are also discussed.

This paper is organized as follows. Section 2 provides relevant information about phylogenetic inference. Section 3 presents key concepts of multi-objective optimization (MOO) and summarizes the main MOO applications in phylogenetic inference. Section 4 describes the parallel PhyloMOEA version. Section 5 presents the experiments involving the new developed PhyloMOEA versions. Finally, Section 6 discusses conclusions and proposes future work.

2 Phylogenetic Reconstruction

The main objective of phylogenetic inference is the determination of the best tree that explains the evolutionary events of the species under analysis. The data used in this analysis usually come from sequence data (nucleotide or aminoacid sequences), morphological features, or other types of data [1]. Due to the absence of information about past species, the phylogenetic reconstruction is only an estimation process since it is based on incomplete information. The evolutionary history of species under analysis is often represented as a leaf-labeled tree, called phylogenetic tree. Taxons (actual species) are represented by the external nodes of the tree while ancestors are referred by internal nodes of the tree. Nodes are connected by branches which may have an associated length value, representing the evolutionary distance between the nodes connected by the branch.

Several phylogenetic reconstruction methods have been proposed in the literature including clustering, optimality criterion and Bayesian methods [1]. Optimality criterion methods defines an objective function to score each possible

solution. A search mechanism, which walks through the tree search space, is necessary to determine the best tree. Maximum parsimony [8] and maximum likelihood [9] are two of the most employed inference criteria in practice. These methods are briefly described bellow.

2.1 Maximum Parsimony

Parsimony methods search for a tree that minimizes the number of character state changes (or evolutionary steps) required by its topology [1]. Let D be a dataset containing n species. Each specie has N sites, where d_{ij} is the character state of specie i at site j. Given tree T with node set $V(T)$ and branch set $E(T)$, the parsimony score of T is defined as:

$$PS(T) = \sum_{j=1}^{N} \sum_{(v,u)\in E(T)} w_j \cdot C(v_j, u_j), \tag{1}$$

where w_j refers to the weight of site j, v_j and u_j are, respectively, the character states of nodes v and u at site j for each branch (u, v) in T and C is the cost matrix, such that $C(v_j, u_j)$ is the cost of changing from state v_j to state u_j. The leaves of T are labeled by character states of species from D, i.e., a leaf representing k-th species has a character state d_{kj} for position j.

The simplest form of parsimony is the Fitch parsimony [8], which assumes a unitary cost matrix such that $C_{xy} = 1$ if $x \neq y$; otherwise $C_{xy} = 0$. The problem of determining the maximum parsimony tree can be separated in two sub-problems:

1. The small parsimony problem, which consists of finding the character states of internal nodes of a given tree T such that $PS(T)$ is minimized.
2. The large parsimony problem, which aims to find the tree T^* such that $PS(T^*)$ is the minimum for the tree search space.

The small parsimony problem can be solved in polynomial time using a divide-and-conquer algorithm [8]. Conversely, the large parsimony problem was proven to be NP-hard [1]. Thus, the search for the best tree topology, for moderate and large datasets, is only feasible using heuristic techniques.

2.2 Maximum Likelihood

The likelihood of a phylogenetic tree, denoted by $L = P(D|T, M)$, is the conditional probability of the sequence data D given a tree T and an evolutionary model M, which contains several parameters related to tree branch lengths and a sequence substitution model [1]. Given a tree T, $P(D|T, M)$ (referred as $L(T)$ henceforth), is calculated from the product of partial likelihoods from all sites:

$$L(T) = \prod_{j=1}^{N} L_j(T), \tag{2}$$

where $L_j(T) = P(D_j/T, M)$ is the likelihood at site j. The site likelihoods can also be expressed as:

$$L_j(T) = \sum_{r_j} L_j^r(r_j) \cdot \pi_{r_j}, \tag{3}$$

where r is the root node of T, r_j refers to any possible state of r at site j, π_{r_j} is the frequency of state r_j, and $L_j^r(r_j)$ is the conditional likelihood of the subtree rooted by r. Specifically, $L_j^r(r_j)$ denotes the probability that everything that is observed from node r to the leaves of T, at site j, given r has state r_j.

A recursive method to calculate L was proposed by Felsenstein [9] using a dynamic programming approach, where L is obtained by a post-order traversal in T. Finding the maximum likelihood tree involves not only the search for the best tree, but the optimization of the parameters of the model M. As in the case of parsimony, the determination of the best tree for moderate or large datasets is only feasible by applying heuristic search techniques.

3 Multi-Objective Approaches for Phylogenetic Inference

A multi-objective optimization problem (MOOP) deals with two or more objective functions that must be simultaneously optimized. When the involved objective functions are conflicting, there is not a single solution but a solution set, called as Pareto-optimal set, whose elements represent a trade-off among objective functions.

In MOOPs, the Pareto dominance concept is commonly used to compare two solutions. A solution x dominates a solution y if x is not worse than y in all objectives and if it is better for at least one. Indeed, solutions in the Pareto optimal set are not dominated by any other solution in the entire search space. The curve formed by plotting these solutions in the objective function space is entitled Pareto front. If there is no preference among the objectives, all Pareto optimal solutions have the same importance. Deb [13] highlights two fundamental goals in MOOP:

1. To find a set of solutions as close as possible to the Pareto optimal front;
2. To find a set of solutions as diverse as possible.

Evolutionary algorithms for multi-objective optimization (MOEAs) and other meta-heuristics have been successfully applied to both theoretical and practical MOOPs [13]. In general, elaborated meta-heuristics models are able to find a distributed Pareto optimal set in a single run [10].

The phylogenetic reconstruction problem can produce conflicting trees in a variety of situations. Rokas et al. [14] show how the selection of the inference method, the employment of different data sources and the assumptions concerning data can produce contradictory tree topologies. In this regard, MOO approaches have been proposed in the literature in order to deal mainly with conflicting datasets [15], [16] and diverse optimality criteria [7].

Cancino and Delbem [7] propose a MOEA approach, called PhyloMOEA, for phylogenetic reconstruction using maximum parsimony and maximum likelihood criteria. PhyloMOEA was tested using four nucleotide datasets and the results indicates that some of the Pareto optimal solutions are not significantly worse than the best trees resulting from a separate analysis. These solutions are, in some cases, consistent with the best solutions obtained from the parsimony and likelihood criteria. A detailed description of PhyloMOEA can be found in previous studies [7].

PhyloMOEA execution consumes significant computational resources. Indeed, the huge tree search space, the costly likelihood evaluation function and the optimization of the model parameters are factors that require the parallelization of the proposed approach. Further details of the parallel PhyloMOEA will be discussed in Section 4.

4 Parallel Strategies for PhyloMOEA

Topological search and tree evaluations (for instance, using the maximum likelihood criterion) are the most expensive tasks of typical phylogenetic inference programs. Bader et al. [11] classify parallel implementations of the aforementioned tasks into three levels: fine-grained, coarse-grained and job level.

Fine-grained strategies aim to parallelize the objective function. For instance, parsimony and likelihood scores can be decomposed due to their site independence assumption (see Equations 1 and 3). The site calculations can be distributed among several processors in SMP architectures.

Coarse-grained parallel approaches are coupled with the tree search mechanism implemented in each algorithm. Typically, heuristic hill-climbing strategies [2] perform evaluation, optimization and local rearrangements for a large number of trees at each iteration. In this case the parallel implementations [11] distribute the aforementioned tasks across several slave workers while a master process dispatches the works and coordinates the search. On the other hand, divide and conquer heuristics [17] decompose the phylogenetic inference problem in small sub-problems and ensemble the partial solutions in a single tree. The corresponding parallel versions for these methods [18] follow a master/slave approach where the master distributes the sub-problems to be solved by the workers. Finally, EA-based phylogenetic strategies [3], [4] work with a set of solutions (population) which are evolved according the genetic operators (selection, recombination, mutation). Parallel EA strategies reported in the literature involve a distributed evaluation of the population solutions [19] and cooperative models [4] with several EAs instances collaborating in the tree search.

Job-level parallelism involves multiple phylogenetic analyses that are performed simultaneously. For instance, several phylogenetic search can be dispatched on several cluster nodes using different starting trees.

In order to describe the parallel approach implemented in PhyloMOEA, we adopt here three levels of parallelism proposed for population-based metaheuristics [10]:

– Algorithmic-level: where independent or collaborative algorithms are running in parallel.
– Iteration-level: in this model, each iteration of metaheuristic is parallelized in order to speedup the algorithm and reduce the search time.
– Solution-level: focused on the parallelization of a single solution.

The first step for parallelizing PhyloMOEA was carried out at iteration level using a master/slave scheme. The master process is responsible for distributing the evaluation of solutions from the population to the worker processes. The slaves performs the likelihood and parsimony evaluations and return the results to the master. Once all evaluations are collected by the master, it performs the selection, recombination and generates the new population. The parallelization of PhyloMOEA using ParadisEO was straightforward due to this component takes into account all communications needed (using MPI) to distribute the function evaluations. The branch length optimization of the solutions, performed at the end of PhyloMOEA execution, was also distributed in a similar fashion.

A profile analysis of PhyloMOEA serial code reveals that around 90% of its execution time is consumed evaluating the likelihood function. Similar results were obtained from other maximum likelihood inference programs like RAxML [2]. The solution level parallelism implemented in PhyloMOEA addresses this problem by distributing the likelihood site calculations across several threads. Likelihood functions implementations collect the per site evaluations in a loop. As OpenMP is well-suited for automatic loop parallelization, it was the natural choice to develop the multi-threaded version of the likelihood function. It is important to remark that the parsimony function parallelization did not produce significant speedup in preliminary experiments. The schema of the PhyloMOEA parallel version is shown in Fig. 1.

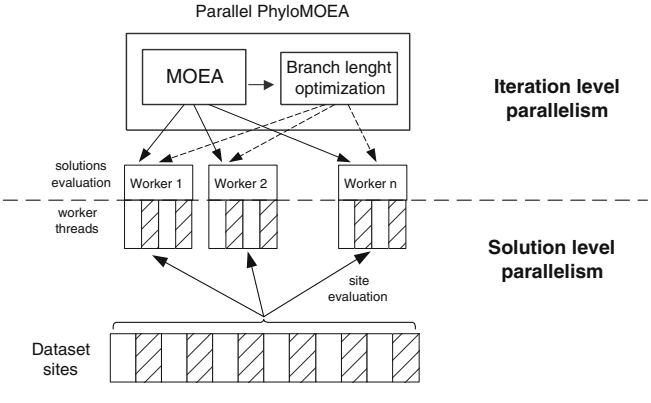


Fig. 1. Iteration and solution level parallelization in PhyloMOEA

Solution and iteration levels can be hybridized to take advantage of multicore per node clusters. Then, for each node, it is possible to execute a pure MPI worker or an hybrid MPI/OpenMP worker distributed on all processor cores.

5 Results

The evaluation of parallel PhyloMOEA performance was carried out in two stages. First, we evaluate the scalability of the multi-thread likelihood implementation. Finally, the overall performance of the MPI and hybrid MPI/OpenMP configuration are compared.

5.1 Multi-Threaded Likelihood Function Scalability

The relative speedup of the OpenMP multi-threaded likelihood function was evaluated using the following datasets taken from the literature [20]:

- $d50\_5000$, $d50\_50000$ and $d50\_500000$: each dataset contains 50 taxa with 5000, 50000 and 500000 sites, respectively.
- $d250\_5000$, $d250\_50000$ comprising 250 taxa with 50000 and 500000 sites, respectively.
- $d500\_5000$ with 500 taxa and 5000 sites.

For each dataset, 20 randomly trees were generated and evaluated using the single and multi-thread versions of the likelihood function. Each experiment was repeated ten times. The average execution time for all runs was used to calculate the speedup in each dataset. Note that in these experiments we only evaluate trees and do not perform tree search.

The experiments were carried out in 64-bits quad-core (AMD Intel Xeon E530) and eight-core (Intel Xeon E5420 QC) architectures running Debian Linux. We tested 2 and 4 threads configuration for the quad-core and 2, 4 and 8 threads configuration for the eight core architectures. The scalability of the multi-thread likelihood function is measured by:

- Increasing the dataset size with larger number of sites for alignments comprising 50 and 250 species.
- Increasing the number of species (50, 250 and 500 taxa) for the datasets with 5.000 sites. In this case, an additional amount of computation is required since deeper recursive calls in the likelihood function.

Figs. 2 and 3 show the speedup obtained when varying the number of sites and species for each configuration tested. The increase of the dataset size (number of sites) for the 50 and 250 species alignments does not produce a significant gain in the quad-core architecture (see Figs. 2(a) and 2(c)). However, in the eight-core architecture, an increase of the speedup for the 50 species datasets is observed (see Fig. 2(b)). Finally, the 250 species datasets does not show a significant speedup improvement due the change of the number of sites.

The increase of the number of species has a negative effect on the speedup in the quad-core architecture (see Figs. 3(a)). This can be caused by the overhead involved due to the additional recursive calls for threads and their synchronization at the end of each evaluation. Conversely, Fig. 3(a) suggests that the extra computation results in speedup increases for the eight-core architecture.

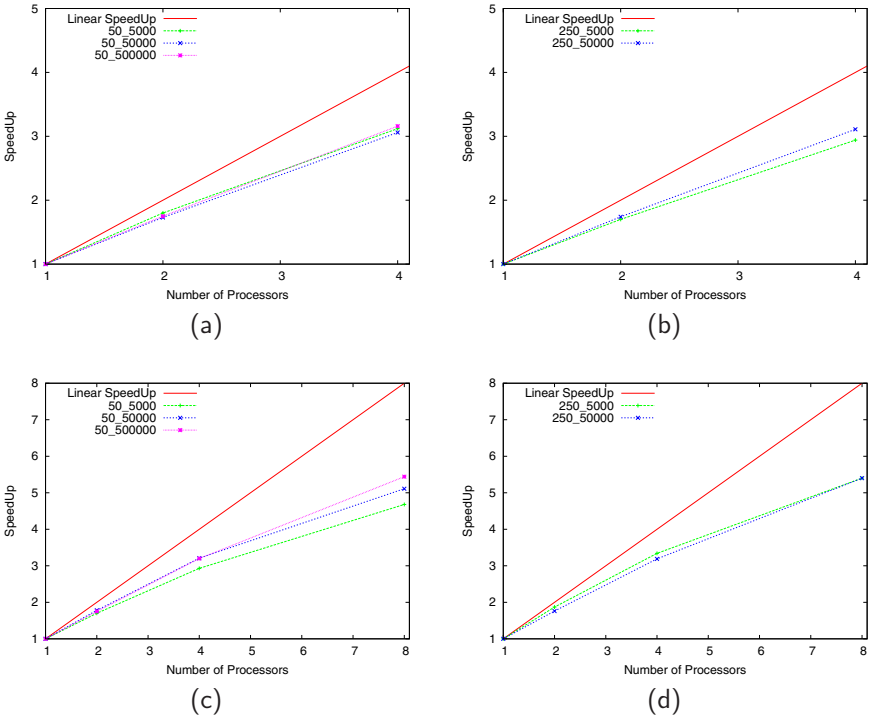


Fig. 2. Likelihood function speedup for 50 and 250 species datasets with increasing number of sites for quad core (a),(c) and eight core (b),(d) architectures

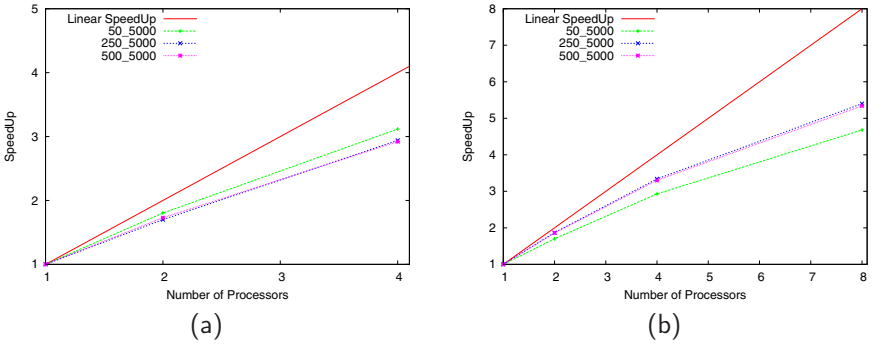


Fig. 3. Likelihood function speedup for datasets comprising 5.000 with increasing number of species for quad core (a) and eight core (b) architectures

There are other studies [21] that measure the effects of the increased dataset size in the multi-threaded likelihood functions. Relative speedup reported in these works are similar with the results present here. It is important to note that our results are not directly comparable with those published elsewhere. In all the cases, the code, the compiler used, the target architectures and even the datasets tested are different. However, it is suggested that the increase in the number of sites produces speedup gain while the increase of number of species have a negative effect in the scalability [21].

5.2 Parallel PhyloMOEA Scalability

Relative speedups of MPI (PhyloMOEA-MPI) and MPI/OpenMP (PhyloMOEA-Hybrid) parallel versions of PhyloMOEA were tested with the following datasets:

1. The $rbcL\_55$ dataset comprises 55 sequences (each sequence has 1314 sites) of the rbcL chloroplast gene from green plants;
2. The $mtDNA\_186$ dataset contains 186 human mitochondrial DNA sequences (each sequence has 16608 sites) obtained from The Human Mitochondrial Genome Database (mtDB);
3. The $RDPII\_218$ dataset comprises 218 prokaryotic sequences of RNA (each sequence has 4182 sites) taken from the Ribosomal Database Project II;
4. Finally, the $ZILLA\_500$ dataset includes 500 rbcL sequences (each sequence has 1428 sites) from plant plastids.

We employed these 4 datasets for testing complete PhyloMOEA execution due limited wall time in the Grid'5000 computer resources. The number of MOEA iterations was restricted to 50 for each dataset. In all experiments, we use 5 nodes (quad-core AMD Opteron 2218) from the Grid'5000 Bordeaux cluster. The following configurations were used for each dataset:

1. PhyloMOEA-MPI: 2, 4, 8, 12 and 16 workers (denoted by 2w, 4w, 8w and 16w, respectively).
2. PhyloMOEA-Hybrid: 1 worker (2 and 4 threads, denoted by 1w-2t and 1w-4t) and 2, 3, 4 workers (4 threads each, denoted by 2w-8t, 3w-12t and 4w-16t)

These configurations use one node for the master tasks and the rest of the nodes run the workers. Fig. 5.2 show the speedup values obtained by both PhyloMOEA versions for $rbcL\_55$, $mtDNA\_186$, $RDPII\_218$ and $ZILLA\_500$.

For all datasets, the 2w, 4w and 12w variants have better speedup values than the 1w-2t, 1w-4t and 3w-8t configurations. However, the 4w-16t configuration is faster than the 4w variant for all datasets except the $ZILLA\_500$. A super-linear speedup is achieved for PhyloMOEA-MPI version for 2w and 4w while PhyloMOEA-Hybrid version only reach similar results with the 2w variant. The speedup increasing rate of the PhyloMOEA-MPI and PhyloMOEA-Hybrid are severely affected with 16 workers (although PhyloMOEA-Hybrid speedup decreases slowly).

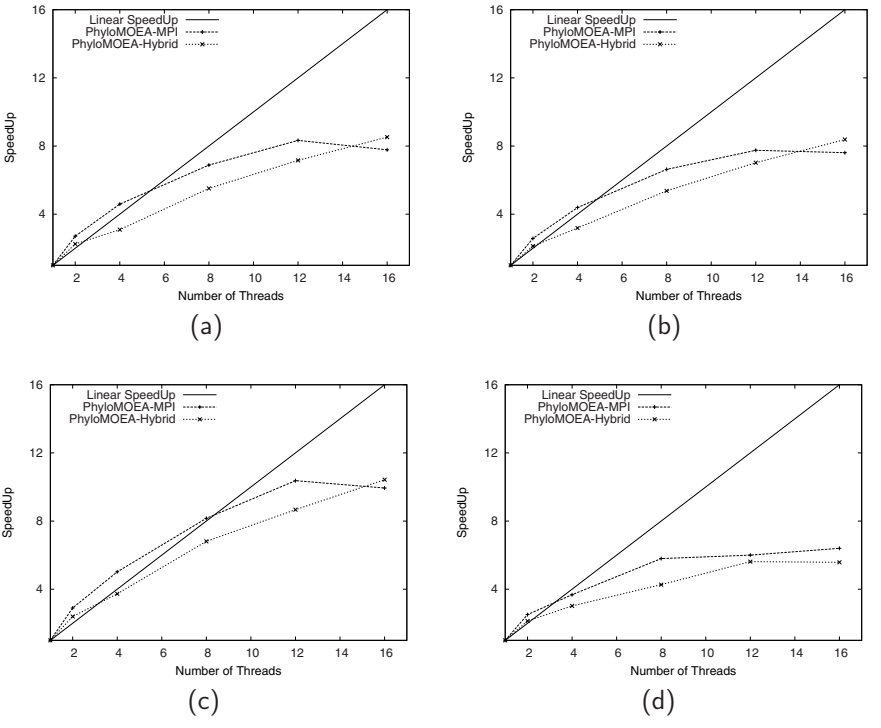


Fig. 4. PhyloMOEA-MPI and PhyloMOEA-Hybrid speedup for the $rbcL\_55$ (a), $mtDNA\_186$ (b), $RDPII\_218$ (c) and $ZILLA\_500$ (d) datasets

There are two factors that penalize the scalability of both PhyloMOEA parallel version: communication and thread synchronization. Former costs are noticeable for PhyloMOEA-MPI version as workers send trees to the master (using the Newick format, the data transmitted increases with the number of species). The increasing communication costs become evident with 16 workers configuration for the $rbcL\_55$, $mtDNA\_186$, $RDPII\_218$ datasets (see Figs. 4(a), 4(b) and 4(c), respectively). Finally, Fig. 4(d) shows that the speedup is affected with more than 8 workers for the $ZILLA\_500$ dataset.

The overhead introduced by synchronizing threads in parallel likelihood calculation affects the speedup of the PhyloMOEA-Hybrid version. This effect is notorious for the $ZILLA\_500$ dataset due to the reduced speedup for the 4w-16t configuration (see Fig. 4(d)). Moreover, the PhyloMOEA-Hybrid version does not show superlinear speedup as in the PhyloMOEA-MPI version. OpenMP thread synchronization overhead could play an important role in this scenario. Conversely, the speedup saturation point is early reached in the PhyloMOEA-MPI than in PhyloMOEA-Hybrid versions for all datasets. The execution times for the complete serial and parallel PhyloMOEA executions for the $ZILLA\_500$ dataset was reduced from 50 to 6 hours.

6 Final Remarks

Inferring phylogenetic trees using several optimality criteria is the main motivation for PhyloMOEA development. Indeed, several studies in the literature [5], [6] point out that the use of various phylogenetic inference methods can lead to inconsistent trees in some cases. Previous experiments using PhyloMOEA show that maximum parsimony and maximum likelihood criteria yield to different trees when they are applied separately [7].

The search for the optimal phylogenetic inference is a very complicated task for moderate and large datasets. The huge tree search space, the evaluation of the trees and the optimization of parameters of sequence model of evolution represent the main bottleneck of current state-of-the-art heuristic phylogenetic approaches [2]. In this regard, several parallel approaches focused on the parallelization of the optimality function and the search procedure have been proposed [4], [18], [20].

The main objective of this paper is to describe the two PhyloMOEA parallel version developed. The first version, called PhyloMOEA-MPI, distributes the evaluation functions (parsimony likelihood) across the several workers. The second version, called PhyloMOEA-Hybrid, is focused on parallelizing the objective functions using a multi-thread approach provided by OpenMP.

Results from both parallel PhyloMOEA versions show sub-linear speedup in most of the cases. However, the execution time reduction compared to the serial version was significant. Similar multi-thread likelihood evaluation function have been proposed elsewhere [18], [20]. In our study, we obtain comparable relative speedup employing bigger alignments and study speedup penalties due to the increase of number of species and sites. The hybrid MPI/OpenMP scheme has also been previously reported [18]. To the best of our knowledge, this scheme has not been proposed for EA-based phylogenetic reconstruction.

Acknowledgment

Experiments presented in this paper were carried out using the Grid'5000 experimental testbed, being developed under the INRIA ALADDIN development action with support from CNRS, RENATER and several Universities as well as other funding bodies (see https://www.grid5000.fr).

References

1. Felsenstein, J.: Inferring Phylogenies. Sinauer, Sunderland (2004)
2. Stamatakis, A.: Raxml-vi-hpc: Maximum likelihood-based phylogenetic analyses with thousands of taxa and mixed models. Bioinformatics 22(21), 2688–2690 (2006)
3. Lewis, P.O.: A Genetic Algorithm for Maximum-Likelihood Phylogeny Inference Using Nucleotide Sequence Data. Molecular Biology and Evolution 15(3), 277–283 (1998)

4. Zwickl, D.: Genetic Algorithm Approaches for the Phylogenetic Analysis of Large Biological Sequence Datasets under the Maximum Likelihood Criterion. PhD thesis, Faculty of the Graduate School. University of Texas (2006)
5. Huelsenbeck, J.: Performance of Phylogenetic Methods in Simulation. Systematic Biology 44, 17–48 (1995)
6. Tateno, Y., Takezaki, N., Nei, M.: Relative Efficiences of the Maximum-Likelihood, Neighbor-Joining, and Maximum Parsimony Methods when Substitution Rate Varies with Site. Molecular Biology and Evolution 11, 261–267 (1994)
7. Cancino, W., Delbem, A.: Multi-criterion phylogenetic inference using evolutionary algorithms. In: IEEE Symposium on Computational Intelligence and Bioinformatics and Computational Biology, CIBCB 2007, pp. 351–358 (2007)
8. Fitch, W.: Toward Defining the Course of Evolution: Minimum Change for a Specific Tree Topology. Systematic Zoology 20(4), 406–416 (1972)
9. Felsenstein, J.: Evolutionary Trees from DNA Sequences: A Maximum Likelihood Approach. Journal of Molecular Evolution 17, 368–376 (1981)
10. Talbi, E.: Metaheuristics: from design to implementation. Wiley, Chichester (2009)
11. Bader, D., Roshan, U., Stamatakis, A.: Computational Grand Challenges in Assembling the Tree of Life: Problems and Solutions. Advances in Computers 68, 128 (2006)
12. Cahon, S., Melab, N., Talbi, E.: Paradiseo: a framework for the flexible design of parallel and distributed hybrid metaheuristics. Journal of Heuristics 10, 357–380 (2004)
13. Deb, K.: Multi-Objective Optimization using Evolutionary Algorithms. John Wiley & Sons, New York (2001)
14. Rokas, A., Wiliams, B., King, N., Carroll, S.: Genome-Scale Approaches to Resolving Incongrounce in Molecular Phylogenies. Nature 425(23), 798–804 (2003)
15. Poladian, L., Jermiin, L.: Multi-Objective Evolutionary Algorithms and Phylogenetic Inference with Multiple Data Sets. Soft. Computing 10(4), 359–368 (2006)
16. Jayaswal, V., Poladian, L., Jermiin, L.: Single- and multi-objective phylogenetic analysis of primate evolution using a genetic algorithm. In: IEEE Congress on Evolutionary Computation, CEC 2007, pp. 4146–4153 (2007)
17. Vinh, L., von Haeseler, A.: Iqpnni: Moving fast through tree space and stopping in time. Molecular Biology and Evolution 21(8), 1565–1571 (2004)
18. Minh, B., Vinh, L., von Haeseler, A., Schmidt, H.: pIQPNNI: parallel reconstruction of large maximum likelihood phylogenies. Bioinformatics 21(19), 3794–3796 (2005)
19. Brauer, M.J., Holder, M.T., Dries, L.A., Zwickl, D.J., Lewis, P.O., Hillis, D.M.: Genetic algorithms and parallel processing in maximum-likelihood phylogeny inference. Molecular Biology and Evolution 19(10), 1717–1726 (2002)
20. Stamatakis, A., Ott, M.: Exploiting Fine-Grained Parallelism in the Phylogenetic Likelihood Function with MPI, Pthreads, and OpenMP: A Performance Study. In: Chetty, M., Ngom, A., Ahmad, S. (eds.) PRIB 2008. LNCS (LNBI), vol. 5265, pp. 424–435. Springer, Heidelberg (2008)
21. Pratas, F., Trancoso, P., Stamatakis, A., Sousa, L.: Fine-grain Parallelism using Multi-core, Cell/BE, and GPU Systems: Accelerating the Phylogenetic Likelihood Function. In: 38th International Conference on Parallel Processing (2009) (accepted for publication)

An Evolutionary Model Based on Hill-Climbing Search Operators for Protein Structure Prediction

Camelia Chira[1], Dragos Horvath[2], and Dumitru Dumitrescu[1]

[1] Babes-Bolyai University, M Kogalniceanu 1, 400084 Cluj-Napoca, Romania
[2] Laboratoire d'Infochimie, UMR 7177, University Strasbourg, France

Abstract. The prediction of a minimum-energy protein structure from its amino-acid sequence represents one of the most important and challenging problems in computational biology. A new evolutionary model based on hill-climbing genetic operators is proposed to address the hydrophobic - polar model of the protein folding problem. The introduced model ensures an efficient exploration of the search space by implementing a problem-specific crossover operator and enforcing an explicit diversification stage during the evolution. The mutation operator engaged in the proposed model refers to the pull-move operation by which a single residue is moved diagonally causing the potential transition of connecting residues in the same direction in order to maintain a valid protein configuration. Both crossover and mutation are applied using a steepest-ascent hill-climbing approach. The resulting evolutionary algorithm with hill-climbing operators is successfully applied to the protein structure prediction problem for a set of difficult bidimensional instances from lattice models.

1 Introduction

Proteins are complex structures composed of amino acid sequences playing key roles in nature. The protein folding problem is of significant importance in many fields including biochemistry, molecular biology and biophysics. Starting from an initially unfolded chain of amino acids, protein folding simulations aim to find a final protein structure having minimum energy. Detecting such a structure represents an NP-hard problem [3,5] even in simplified lattice models which abstract away many of the details of protein folding.

Simplified lattice models such as the hydrophobic-polar (HP) model [4] have been extensively used as benchmarks for computational approaches to protein structure prediction. These include evolutionary search [1,2,12,13], ant colony optimization [11], memetic algorithms [8], tabu search [9] and Monte Carlo approximation algorithms [6].

In this paper, an evolutionary model based on hill-climbing search operators is proposed to address the problem of protein structure prediction in the HP model. The introduced model evolves a population of protein configurations for

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2010, LNCS 6023, pp. 38–49, 2010.

a given HP sequence and relies on hill-climbing recombination and mutation to sustain the search process. Hill-climbing crossover is applied in a dynamic way and offspring are asynchronously inserted in the population during the same generation. The mutation operator engaged in the proposed model is problem-specific and is applied in a steepest-ascent hill-climbing manner. For this purpose, the pull-move transformation [9] by which a single residue is moved diagonally, causing the transition of connecting residues, is engaged. Hill-climbing mutation aims to improve a configuration by applying pull-move transformations in all possible positions. The hill-climbing approach taken in this model enhances the exploitation capabilities of the search, vital for good results in protein folding problems.

The proposed evolutionary model employs hill-climbing driven specialized crossovers and mutations as the main stage of the search process. This is a simpler scheme compared to memetic algorithms, which use a local search stage (sometimes reinforced by hill-climbing) in addition to the standard evolutionary scheme of crossover, mutation and selection. In multimeme algorithms [8] for protein structure prediction, a cycle of mating, mutation, local search (optimization based on memes being aplied to each individual) and replacement is performed each generation. Here, the cycle of hill-climbing crossover and mutation is engaged for a variable number of individuals each generation.

The experiments presented in the current paper focus on various bidimensional HP lattice protein sequences. Numerical results indicate a competitive performance of the evolutionary algorithm based on hill-climbing operators.

The paper is organised as follows: the protein structure prediction problem in the HP model is briefly described, related work on computer-based methods for addressing this problem is discussed, the proposed hill-climbing evolutionary model is presented and numerical results and comparisons are given.

2 The Bidimensional HP Protein Folding Problem

Lattice protein models like the HP model are simplified instances of the generic class of cooperative chain folding processes, which include the actual folding of biological macromolecules. Although HP lattices cannot make actual predictions about real biological macromolecules, their fitness landscape and process dynamics share common traits with real-life processes and therefore may serve to characterize generic features of protein folding. The HP model emphasizes hydrophobicity as the most important difference between amino acids and represents the simplest - yet non-trivial - abstraction for the protein structure prediction problem [14].

The HP model considers a protein structure with n amino acids as a sequence $S = s_1...s_n$ where each residue $s_i, \forall i$ can be either H (hydrophobic or non-polar) or P (hydrophilic or polar). A valid protein configuration forms a self-avoiding path on a regular lattice with vertices labelled by amino acids (see Figure 1 for an example).

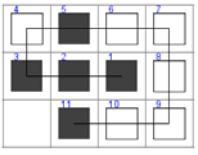


Fig. 1. A protein configuration for sequence $SE = HHHPHPPPPPH$ in the square lattice having the energy value of -2. Black squares represent H residues and white squares are P residues.

Elements of a given pair of residues are considered topological neighbors if they are adjacent (either horizontally or vertically) in the lattice and not consecutive in the sequence.

In the HP model, the energy associated to a protein conformation takes into account every pair of H residues which are topological neighbors. Every H-H topological contact contributes -1 to the energy function. The aim is to find the protein conformation with minimum energy $E^* = min\{E(c)|c \in C(S)\}$, where $C(S)$ contains all valid conformations for amino-acid sequence S (this will correspond to the protein configuration with the maximal number of H-H topological contacts).

3 Related Work

The protein structure prediction problem for the HP model has been shown to be NP-hard [3,5] and many approximation methods and heuristics for addressing it have been proposed [5,14].

Local search methods rely on the idea of iteratively improving a protein conformation based on the exploration of its local neighborhood. However, traditional Monte Carlo methods for protein folding simulations easily get trapped in local optima due to the problem-specific characteristics of the search landscape. Chain growth methods have been proposed to cope with this problem. The pruned-enriched Rosenbluth method (PERM) [6] grows a sequence by sequentially adding one individual particle at a time. The growth is guided towards configurations with lower energies generating good results for the HP problem in 2D and 3D lattices. The main drawbacks refer to the need to incorporate heuristic knowledge and the usage of a significant number of weight thresholds [14].

Lesh et al [9] introduce a local search strategy called pull move for the bidimensional HP model. The pull move transformations are incorporated in a tabu search algorithm able to detect new lowest energy configurations for large HP sequences (having 85 and 100 amino-acids). A pull move operation starts by moving a single residue diagonally to an available location. A valid configuration is maintained by pulling the chain along the same direction (not necessarily

until the end of the chain is reached - a valid conformation can potentially be obtained sooner). The authors also prove that the class of pull moves introduced is reversible and complete [9].

Genetic algorithms (GAs) for protein structure prediction have been initially used by Unger and Moult [13] and proved to obtain better results than traditional Monte Carlo methods. Chromosomes are encoded using internal coordinates with absolute moves and a population of valid conformations is evolved by mutation and crossover. The performance of the 'simplest' genetic algorithm is investigated in [7] where the importance of high resolution building blocks (facilitated by multi-point crossovers) and of local dynamics operator is emphasized. A hybridization between GA and a backtracking algorithm is investigated in [2]. The use of a backtracking-based repairing procedure and of evolutionary search operators constraining the search to the space of valid conformations produces good results for the 3D HP problem. In [12], some specialized genetic operators (called symmetric and cornerchange operators) are introduced. The resulting GA is applied for HP sequences having a length up to 50 residues. In [1], the results of standard GA for protein structure prediction are improved by a GA using pull moves [9] as a local search genetic operation in addition to standard crossover and mutation.

Multimeme algorithms (MMAs) [8] combine GAs with a set of local search heuristics enforcing various neighborhoods for memetic algorithm search. In MMAs, each individual incorporates genetic and memetic material. Crossover, mutation, local search and replacement are performed each generation. MMAs further rely on a contact map memory of already visited solutions based on the topological features of the conformations. The MMA was successfully applied to both HP and functional model proteins.

The protein folding problem has also been tackled using nature-inspired metaheuristics which rely on the model of the search space (such as ant colony systems). Shmygelska et al [11] use Ant Colony Optimization (ACO) combined with a local search mechanism to construct protein conformations. Artificial ants iteratively construct solutions based on the quality of already determined solutions (through the indirect influence of pheromone updates in the search space).

Although evolutionary algorithms have been extensively engaged as robust and efficient global optimization methods for this problem, their computing efficiency needs further improving. Evolutionary approaches to the protein structure prediction problem suffer from the limitations of the genetic operators for this particular problem search space. Crossover and mutation can easily produce invalid configurations due to potential collisions generated by changing various parts of a chromosome. This weak performance of standard genetic operators has a direct impact on the effectiveness of the evolutionary search process.

4 An Evolutionary Model with Hill-Climbing Operators

An evolutionary model relying on hill-climbing genetic operators is proposed to address the protein structure prediction problem. A chromosome represents a

possible protein configuration for a given HP sequence. A population of configurations is evolved by hill-climbing crossover and mutation.

Offspring replace parents if they have a better fitness value. The search scheme is asynchronous in the sense that a new chromosome created as a result of crossover or mutation can potentially be exploited within the same generation by the search operators. Premature convergence is addressed by a *diversification scheme* in which similar individuals are identified and some of them are replaced by new genetic material. Besides the hill-climbing operators and the diversification scheme, the proposed evolutionary model does not require any other phase such as explicit selection or standard mutation.

The distinct features of the introduced model can be summarized as follows:

1. The poulation size is fixed and offspring are asynchronously inserted in the population replacing the worst parent within the same generation.
2. Crossover is applied to randomly selected pairs of individuals in a hill-climbing mode. A number of k offspring are iteratively generated from the same parents. The best-fitted offspring (or its random hill-climbing mutation if better) replaces the worst parent within the same generation. If no better offspring is identified, both parents are replaced by new randomly selected chromosomes. The process continues until the maximum number of hill-climbing iterations is reached.
3. Mutation implements a steepest ascent hill-climbing procedure using the pull move operation [9]. This process is able to generate a variable number of new individuals which replace parents within the same generation (if they have a better fitness value).
4. Diversification ensures the existence of sufficiently heterogeneous genetic material by periodically checking the similarity between individuals having the same energy and replacing similar inidviduals with newly generated ones.

The general scheme of the proposed hill-climbing evolutionary model is given below.

Main Scheme of Evolutionary Algorithm based on Hill-Climbing Operators

Generate $P(0)$ with pop_size individuals randomly
while (maximum number of generations not reached) **do**
  Hill-climbing crossover for k offspring and hc iterations
  Hill-climbing mutation for hc iterations
  Diversification every kd generations
end while

The evolutionary algorithm presented in this paper takes a standard approach to problem representation and fitness function in order to keep the emphasis of the obtained results on the hill-climbing genetic operators implemented.

A chromosome is encoded using an internal coordinates representation. For a protein HP sequence with n residues $S = s_1...s_n$, the chromosome length is $n-1$

and each position in the chromosome encodes the direction $L(Left)$, $U(Up)$, $R(Right)$ or $D(Down)$ towards the location of the current residue relative to the previous one.

The fitness function used corresponds to the energy value of the protein configuration (as given in section 2).

4.1 Hill-Climbing Mutation

The pull move operation proposed in [9] is used as a specialized mutation operator. A pull move transformation can be applied at a given position i from the considered HP sequence.

Let (x_i, y_i) be the coordinates in the square lattice of residue i at time t. Let L denote a free location diagonally adjacent to (x_i, y_i) and adjacent (either horizontally or vertically) to (x_{i+1}, y_{i+1}). Location C denotes the fourth corner of the square formed by the three locations: L, (x_i, y_i) and (x_{i+1}, y_{i+1}). A pull move is possible if location C is free or equals (x_{i-1}, y_{i-1}). In the latter case, the pull move transformation consists of moving the residue from location (x_i, y_i) to location L. In the case that C is a free location, the first step is to move residue from position i to location L and the residue from position $(i-1)$ to location C. The pull move transformation continues by moving all residues from $(i-2)$ down to 1 two locations up the chain until a valid configuration is reached.

Figure 2 presents an example of a pull move transformation for HP sequence $SE = HHHPHPPPPPH$ having the chromosome value of $RRUURURDDD$. The pull move is applied for residue H at position $i = 3$ for which a free location L horizontally adjacent to residue $i+1$ (between residues 4 and 10 in Figure 2.a) is identified. Location C (the location between residues 3 and 11 in Figure 2.a) is free in this example and therefore the pull move will cause moving the residue 3 to location L and residue 2 to location C. The remaining residue 1 (only one in this example) is moved up the chain two positions producing the new chromosome value of $RULURURDDD$ (see Figure 2.b).

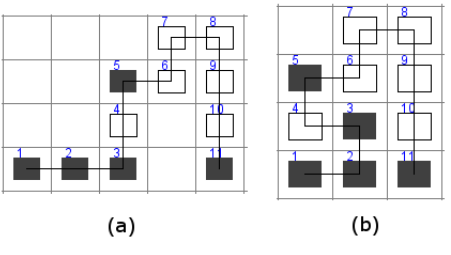


Fig. 2. Pull move transformation for HP sequence $HHHPHPPPPPH$ represented by the chromosome $RRUURURDDD$ (a) at position 3. (b) represents the new chromosome $RULURURDDD$ obtained after the pull move transformation.

In the proposed algorithm, pull moves are applied within a steepest ascent hill climbing procedure each generation. Hill-climbing mutation starts by randomly selecting one individual from the current population and setting it as the *current_hilltop*. Pull moves are applied at each position $i, i = 1, ..., n$ (where n is the length of the HP sequence) resulting in the generation of n new chromosomes. If any of them has a better fitness value than the *current_hilltop* it replaces the latter one. If no improvement is achieved and the maximum number of hill-climbing iterations has not been reached, the *current_hilltop* is reinitialized with a new individual randomly selected from the population.

The procedure for hill-climbing mutation is given below.

Hill-Climbing Mutation Procedure

Set *current_hilltop* to a randomly selected individual *rand_c*
Set *best_c* to *current_hilltop*
while (maximum number of *hc* iterations not reached) **do**
  for i=1 to n **do**
    Generate new chromosome c_i by applying a
    pull move transformation at position i in *current_hilltop*
    if (c_i has better fitness than *best_c*) **then**
      Set *best_c* to c_i
    end if
  end for
  if (better chromosome *best_c* found) **then**
    Set *current_hilltop* to *best_c*
  else
    Replace *rand_c* with *best_c* in the current population
    Set *rand_c* to a new randomly selected individual
    Set *current_hilltop* and *best_c* to *rand_c*
  end if
end while

It should be emphasized that the number of individuals that will undergo hill-climbing mutation within one generation is dynamic. The hill-climbing mutation procedure operates on the same individual by pull move mutation until no further improvement is achieved. The mutated chromosome obtained in this process replaces the original parent in the population during the current generation. The hill-climbing mutation procedure continues with a new individual randomly selected. The process of improving a chromosome by pull moves can last a variable number of hill-climbing iterations for each individual.

4.2 Hill-Climbing Crossover

For the recombination of genetic material, a one-point crossover operator is specified. Given two parent chromosomes p_1 and p_2 and a randomly generated cut point χ, two offspring are created as follows:

1. The genes before the cut point χ are copied from one parent: $c_i^{offspring} = c_i^{p_1}$, for $i = 0, \chi - 1$;
2. For the second part of the offspring, $c_i^{offspring} = c_i^{p_2}$, for $i = \chi, n-1$ unless this move forces residue i to overlap with one of the $i-1$ previous ones. If a collision occurs then a random direction leading to a valid position is selected.

Crossover is applied following a hill-climbing strategy. In every generation, a variable number of chromosome pairs are selected for crossover and better generated offspring replace individuals in the current population.

The hill-cllimbing crossover procedure is detailed below. This procedure is inspired by the crossover-hill-climbing scheme proposed in [10].

Hill-Climbing Crossover Procedure

Set p_1 and p_2 to randomly selected individuals from current population
Set *best_o* to $best(p_1, p_2)$
do
  for $i = 1$ to k **do**
    Generate a random cut point χ (from 1 to chromosome length $n-1$)
    Set o to the best of two offspring obtained from $crossover(p_1, p_2, \chi)$
    if (o has better fitness than *best_o*) **then**
      Set *best_o* to o
    end if
  end for
  if (new *best_o* found) **then**
    Set *rhcm_best* to *random_hill_climbing_mutation*(*best_o*)
    Replace $worst(p_1, p_2)$ with $best(best\_o, rhcm\_best)$
  else
    Set p_1 and p_2 to new individuals randomly selected
    from current population
  end if
while (maximum number of *hc* iterations not reached)

For each pair of chromosomes selected for recombination, a number of k offspring is generated via crossover. The best offspring resulted from this process is mutated using a pull move transformation within a random hill climbing (RHC) procedure. This RHC mutation is similar to the hill-climbing mutation presented in section 4.1 but it has the following distinctive features: (i) only one chromosome is being mutated and when no further improvement is obtained by pull moves the procedure stops; and (ii) at each hill-climbing iteration, only one pull move transformation is applied for a position randomly selected.

The parent having the highest energy is replaced with the best of the two chromosomes generated (best offspring from crossover and its RHC mutated version). This new individual is engaged as a parent in the next hill-climbing iteration. When no better offspring is generated, a new pair of parent chromosomes is randomly selected from the current population and undergoes the same steps until the maximum number of hill-climbing iterations is reached. Similar

to the mutation procedure presented in section 4.1, the number of individuals selected for recombination varies from one generation to another depending on the improvements that can be generated by the same pair of chromosomes.

4.3 Diversification

In order to ensure the maintainance of sufficiently diverse genetic material, it is proposed to explicitly reinforce diversity every kd generations, where kd is a parameter of the algorithm. The diversification stage works as follows:

1. The individuals from the current population are grouped based on their fitness (one group for each fitness value).
2. For each group identified, subgroups of similar individuals are constructed based on the Hamming distance. Individuals are considered similar if the Hamming distance (i.e. the number of different position values in the chromosomes) is less than $(n-1)/4$, where $(n-1)$ is the length of the chromosome.
3. For each subgroup of similar individuals, one of them is kept in the current population and the rest of individuals are replaced by new randomly generated chromosomes (improved by a hill-climbing mutation).

Diversification has the potential to avoid the search process to get trapped in local optima by explicitly introducing new genetic material in the population. The exploration of new search space regions is therefore facilitated in addition to the efficient exploitation performed by the hill-climbing procedures.

5 Numerical Experiments

Experiments focus on 2D HP protein sequences (commonly used as benchmarks) with lengths from 20 to 64.

The parameter setting for the proposed evolutionary algorithm with hill-climbing operators is based on the results of several experiment sets:

1. The population size is 100 and the number of generations is 300;
2. The number of hill-climbing iterations for both crossover and mutation is set to 100;
3. For hill-climbing crossover, a number of 50 offspring are generated for a pair of chromosomes each hill-climbing iteration;
4. Diversification is engaged every 30 generations (generally calculated as 10% of the number of generations).

The initial population contains randomly generated chromosomes representing valid configurations (each chromosome is iteratively generated in a random manner until a conformation free of collisions in the HP square lattice model is found). For each HP sequence considered, the proposed algorithm was run 10 times and the results from one of the most efficient runs are reported.

Table 1. Results obtained by the evolutionary algorithm based on hill-climbing search operators (last column) for standard 2D HP instances

Inst.	Length	Sequence	E^*	Energy
S1	20	1H 1P 1H 2P 2H 1P 1H 2P 1H 1P 2H 2P 1H 1P 1H	-9	-9
S2	24	2H 2P 1H 2P 1H 2P 1H 2P 1H 2P 1H 2P 1H 2P 2H	-9	-9
S3	25	2P 1H 2P 2H 4P 2H 4P 2H 4P 2H	-8	-8
S4	36	3P 2H 2P 2H 5P 7H 2P 2H 4P 2H 2P 1H 2P	-14	-14
S5	48	2P 1H 2P 2H 2P 2H 5P 10H 6P 2H 2P 2H 2P 1H 2P 5H	-23	-23
S6	50	2H 1P 1H 1P 1H 1P 1H 1P 4H 1P 1H 3P 1H 3P 1H 4P 1H 3P 1H 3P 1H 1P 4H 1P 1H 1P 1H 1P 1H 1P 1H 1H	-21	-21
S7	60	2P 3H 1P 8H 3P 10H 1P 1H 3P 12H 4P 6H 1P 2H 1P 1H 1P	-36	-35
S8	64	12H 1P 1H 1P 1H 2P 2H 2P 2H 2P 1H 2P 2H 2P 2H 2P 1H 2P 2H 2P 2H 2P 1H 1P 1H 1P 12H	-42	-39

Table 1 presents the energy values obtained for the 2D HP benchmarks considered. The HP sequence and the known optimum value (in the column labelled E^*) for each instance are given. The last column in table 1 contains the energy values detected by the proposed method.

The introduced model is able to identify the protein configurations having the best known optimum energy for sequences $S1$ to $S6$. For the first three HP instances considered, optimum energy conformations are found very early in the evolution process (usually during the first 20 generations) and therefore, a less computationally expensive implementation of the algorithm (with fewer generations and probably less hill-climbing iterations) would have been able to generate the optimum. The proposed algorithm fails to find the optimum for the larger instances $S7$ and $S8$. Most of the runs of the algorithm for sequence $S7$ detect the suboptimal solution having the energy -35. We expect to improve the performance of the proposed model for large instances by extending the diversification stage to consider other metrics for calculating the similarity between two chromosomes. For example, a *fingerprint* of the protein configuration (which includes topological information) can potentially provide a more accurate comparison between same-energy individuals so that the diversification stage would result in the replacement of meaningfully similar chromosomes.

The performance of the proposed model is compared to the best results obtained by other evolutionary models and memetic algorithms for protein structure prediction. Table 2 presents the results as follows: the known optimum for each HP instance, the energy found by the proposed method (in the third column), the results of standard GA [13], Pull-Move GA (PMGA) [1] and multimeme algorithms (MMA) [8]. The results of GA [13] are based on a population of 200 structures evolved over 300 generations. The Pull-Move GA (PMGA) proposed in [1] is able to improve the results of standard GAs by using pull move transformations in addition to the standard genetic operators but not in a hill-climbing mode. MMAs represent an interesting approach to be compared with the one proposed in this paper as both models use local search but according to

Table 2. Comparison of results achieved by different evolutionary methods for the HP problem

Inst.	E^*	Proposed Method	Genetic Algorithms	Pull Move-GAs	Multimeme Algorithms
S1	-9	-9	-9	-9	-9
S2	-9	-9	-9	-9	
S3	-8	-8	-8	-8	-8
S4	-14	-14	-14	-14	-14
S5	-23	-23	-22	-22	-22
S6	-21	-21	-21	-21	-21
S7	-36	-35	-34	-34	
S8	-42	-39	-37	-38	-39

different strategies. For GA and MMA the best solution from 5 runs is selected while the PMGA reports the best solution from 10 runs.

A direct comparison between energy values obtained by different evolutionary models emphasizes a good and competitive performance of the proposed method. The results are better than those of GAs [13] and PMGAs [1]. The effect of hill-climbing search based on pull moves is clearly benefic as opposed to applying pull moves in addition to mutation as in PMGA (the proposed model detects better energies for instances $S5$, $S7$ and $S8$ compared to PMGA). The results are competitive with those of MMA (the proposed model obtains a better solution for instance $S5$ when compared to MMA). This is a promising result for the proposed method emphasizing the power of hill-climbing search procedures based on specialized genetic operators. In MMAs, optimization is based on the memes available individually (pivot moves, substructure stretching, random macro-mutation of a substructure, reflection of a sub-structure, non local k-opt and local k-opt) [8]. The proposed model uses a scheme by which a dynamic number of individuals are affected each generation by hill-climbing search operators and is able to detect similar or better results compared to MMAs where optimization (based on the six memes mentioned above) is applied for every individual in the population in addition to standard evolutionary search.

6 Conclusions and Future Work

A new evolutionary model relying on a hill-climbing search scheme is presented and engaged to address the protein structure prediction problem in the HP model. The main feature of the proposed model refers to the application of crossover and pull move transformations (as mutation) within hill-climbing search procedures. Better ofsspring are inserted in the population within the same generation making the selection process intrinsic to hill-climbing crossover and mutation. An explicit diversification process is engaged periodically to replace similar chromosomes with new genetic material. The results obtained for several bidimensional HP instances are promising and competitive with the best results of other evolutionary models.

It is planned to extend numerical experiments for other larger HP sequences to further test the performance of the proposed model. More complex approaches to the calculation of the fitness energy will be investigated. Furthermore, the proposed evolutionary model can highly benefit from the improvement of diversification using other mechanisms for checking the similarities between individuals (such as the fingerprint of protein conformations).

References

1. Berenboym, I., Avigal, M.: Genetic algorithms with local search optimization for protein structure prediction problem. In: GECCO 2008, pp. 1097–1098 (2008)
2. Cotta, C.: Protein Structure Prediction Using Evolutionary Algorithms Hybridized with Backtracking. In: Mira, J., Álvarez, J.R. (eds.) IWANN 2003. LNCS, vol. 2687, pp. 321–328. Springer, Heidelberg (2003)
3. Crescenzi, P., Goldman, D., Papadimitriou, C.H., Piccolboni, A., Yannakakis, M.: On the Complexity of Protein Folding. Journal of Computational Biology 50, 423–466 (1998)
4. Dill, K.A.: Theory for the folding and stability of globular proteins. Biochemistry 24(6), 1501–1509 (1985)
5. Hart, W., Newman, A.: Protein Structure Prediction with Lattice Models. In: Handbook of Computational Molecular Biology. Chapman & Hall CRC Computer and Information Science Series (2006)
6. Hsu, H.P., Mehra, V., Nadler, W., Grassberger, P.: Growth algorithms for lattice heteropolymers at low temperatures. J. Chem. Phys. 118(1), 444–451 (2003)
7. Khimasia, M.M., Coveney, P.V.: Protein structure prediction as a hard optimization problem: the genetic algorithm approach. Molecular Simulation 19, 205–226 (1997)
8. Krasnogor, N., Blackburnem, B., Hirst, J.D., Burke, E.K.: Multimeme algorithms for protein structure prediction. In: Guervós, J.J.M., Adamidis, P.A., Beyer, H.-G., Fernández-Villacañas, J.-L., Schwefel, H.-P. (eds.) PPSN 2002. LNCS, vol. 2439, pp. 769–778. Springer, Heidelberg (2002)
9. Lesh, N., Mitzenmacher, M., Whitesides, S.: A complete and effective move set for simplified protein folding. In: RECOMB 2003: Proceedings of the seventh annual international conference on Research in computational molecular biology, pp. 188–195. ACM, New York (2003)
10. Lozano, M., Herrera, F., Krasnogor, N., Molina, D.: Real-coded memetic algorithms with crossover hill-climbing. Evol. Comput. 12(3), 273–302 (2004)
11. Shmygelska, A., Hernandez, R., Hoos, H.H.: An Ant Colony Algorithm for the 2D HP Protein Folding Problem. In: Dorigo, M., Di Caro, G.A., Sampels, M. (eds.) Ant Algorithms 2002. LNCS, vol. 2463, pp. 40–53. Springer, Heidelberg (2002)
12. Song, J., Cheng, J., Zheng, T., Mao, J.: A Novel Genetic Algorithm for HP Model Protein Folding. In: PDCAT 2005: Proceedings of the Sixth International Conference on Parallel and Distributed Computing Applications and Technologies, pp. 935–937. IEEE Computer Society, Los Alamitos (2005)
13. Unger, R., Moult, J.: Genetic algorithms for protein folding simulations. J. Molec. Biol. 231, 75–81 (1993)
14. Zhao, X.: Advances on protein folding simulations based on the lattice HP models with natural computing. Appl. Soft. Comput. 8(2), 1029–1040 (2008)

Finding Gapped Motifs by a Novel Evolutionary Algorithm

Chengwei Lei and Jianhua Ruan

Department of Computer Science
The University of Texas at San Antonio
San Antonio, TX 78249, USA
{clei,jruan}@cs.utsa.edu

Abstract. Identifying approximately repeated patterns, or motifs, in biological sequences from a set of co-regulated genes is an important step towards deciphering the complex gene regulatory networks and understanding gene functions. In this work, we develop a novel motif finding algorithm based on a population-based stochastic optimization technique called Particle Swarm Optimization (PSO), which has been shown to be effective in optimizing difficult multidimensional problems in continuous domains. We propose a modification of the standard PSO algorithm to handle discrete values, such as characters in DNA sequences. Our algorithm also provides several unique features. First, we use both consensus and position-specific weight matrix representations in our algorithm, taking advantage of the efficiency of the former and the accuracy of the later. Furthermore, many real motifs contain gaps, but the existing methods usually ignore them or assume a user know their exact locations and lengths, which is usually impractical for real applications. In comparison, our method models gaps explicitly, and provides an easy solution to find gapped motifs without any detailed knowledge of gaps. Our method also allows some input sequences to contain zero or multiple binding sites. Experimental results on synthetic challenge problems as well as real biological sequences show that our method is both more efficient and more accurate than several existing algorithms, especially when gaps are present in the motifs.

Keywords: DNA motif; optimization; PSO; evolutionary algorithm.

1 Introduction

Computational prediction of transcription factor binding sites (TFBS) from co-expressed / co-regulated genes is an important step towards deciphering complex gene regulatory networks and understanding gene functions. Given the promoter sequences of a set of co-expressed / co-regulated genes, the goal is to find short DNA sequences ("motifs") whose occurrences (with allowed mismatches) in the sequences cannot be explained by a background model. An accurate identification of such motifs is computationally challenging, as they are typically very short (8-15 bases) compared to the promoter sequences (hundreds to thousands bases).

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2010, LNCS 6023, pp. 50–61, 2010.

Furthermore, there is often a great variability among the binding sites of any given TF, and the biological nature of the variability is not yet well understood. Finally, in many cases, the TFBS may appear only in a subset of the putatively co-regulated genes.

Despite the challenge, many computational methods have been developed and have been proven useful in predicting real binding sites [1]. The existing algorithms can be roughly classified into two broad categories according to the motif representations: those based on position-specific weight matrices (PWMs), and those based on consensus sequences. Examples of the former include well-known programs such as MEME [2], AlignACE [3], GibbsSampler [4], and BioProspector [5]. The latter category includes Weeder [6], YMF [7], MultiProfiler [8], and Projection [9]. In general, PWM offers a more accurate description of motifs than consensus sequences, but is more difficult to optimize. On the other hand, consensus-based algorithms often rely on enumerating short subsequences, which may be impossible for longer motifs. For an excellent survey of the existing methods and an assessment of their relative performance, see [1,8].

Recently several consensus-based motif finding algorithms have been developed using evolutionary algorithms, because of their efficiency in searching over multidimensional solution spaces. For example, GAME [10] and GALFP [11] are based on genetic algorithms, and have been shown to outperform many PWM-based algorithms. In a previous work, we proposed a motif finding algorithm based on the classical Particle Swarm Optimization (PSO) strategy [12], where we used the set of positions on each sequence together as a solution, and searched the solution space by PSO algorithm. To keep the solution space continuous, we restructured the original sequences using a sequence mapping. Although the algorithm shows a good performance on small input size (for example 20 sequences and 1000 base for each sequence), the algorithm becomes slow for larger data set, as the number of possible motif positions grows exponentially as the number of sequences increases. Several other motif finding methods have also been developed based on PSO, for example, Hybrid-PSO [13] and PSO-EM [14]. Hybrid-PSO uses a similar basic idea as our previous work [12], and therefore has the same problem we mentioned above. PSO-EM simply uses PSO to find candidate motifs, which are then used as seeds by other expectation-maximization based motif finding algorithms, such as MEME [2].

In this paper, we develop a novel algorithm, called PSO+, for finding motifs. This new method has the following contributions. First and most importantly, PSO+ differs from other motif finding algorithms by explicitly modeling gaps, which provides an easy solution to find gapped motifs. Many real motifs contain positions of low information (gaps), but the existing algorithms usually do not allow gaps, or require a user to specify the exact location and length of gaps, which is often impractical for real applications. Second, we use both consensus and PWM representations in our algorithm, taking advantage of the efficiency of consensus and the accuracy of PWMs. Finally, our method also allows some input sequences to contain zero or multiple binding sites, which is common in real biology data set, but ignored by some of the algorithms. Finally, we propose a

```
Algorithm PSO_Motif_Plus
fitness(final_consensus) = -infinity;
for i=1 to MAX_RESET do {//loop 1}
   Initialize a random solution (current) for each agent
   fitness(pbest) = -infinity for all agents
   fitness(gbest) = -infinity;
   for j=1 to MAX_ITERATION do {//loop 2}
      for k=1 to NUM_AGENTS do {//loop 3}
         Scan each sequence to find a best match to current_k;
         Use the matches to calculate fitness(current_k);
         if fitness(current_k) > fitness(pbest_k) then
            pbest_k = current_k;
         end if
         if fitness(current_k) > gbest then
            gbest = current_k;
         end if
      end for
      Check Shift;
      Update current for each agent based on the update rule;
      if j > MIN_ITERATION and no update on gbest occurred in past N iterations
      then
         End Loop 2;
      end if
   end for
   if fitness(gbest) > fitness(final_consensus) then
      final_consensus = gbest;
   end if
end for
Post-processing;
```

Fig. 1. Pseudo-code of our algorithm

novel modification to the PSO update rule to accommodate discrete values, such as characters in DNA sequences, which may also be useful in other applications.

The remaining sections are organized as follows. In Section II, we present the details of our algorithm. In Section III, we use both synthetic and real biological sequences to show that our method is more efficient and more accurate than several existing algorithms, especially when gaps are present in the motifs. We conclude in Section IV.

2 Algorithm

2.1 Introduction to Particle Swarm Optimization

Particle Swarm Optimization (PSO), which has been shown to be effective in optimizing difficult multidimensional problems in many fields, is a population-based stochastic optimization technique for problem solving that is inspired by

the social behaviors of organisms such as bird flocking [15]. The system is initialized with a population of random solutions and searches for the optimal solution by updating iteratively. Each potential solution, called particle (or agent), is represented by a point in the multiple-dimensional solution space. When searching for the optimum solution, particles fly around the solution space with a certain velocity (speed and direction). During flight, each particle adjusts its position and velocity according to its own experience and the experience of its neighbors. Specifically, each particle keeps track of the best solution it has encountered so far. This solution is called *pbest*, which stands for personal best. The system also keeps track of the global optimum of all the particles, hence called *gbest*. The fundamental concept of PSO consists of changing the velocity of each particle at each time step toward its *pbest* and *gbest* locations [15].

2.2 Method Overview

Fig. 1 shows the main structure of the algorithm, which contains three loops. The most inside loop, loop3, evaluates the fitness value of each agent and update information for the whole system. Using its *current* solution, an agent first finds out a best match from each sequence, calculates the fitness value (see below), and updates *pbest*, *gbest* if necessary. Loop2 is the main part of the PSO+ algorithm. From a random initial solution, each agent continuously searches for better solutions in the neighborhood, taking information from its own experience (*pbest*), and the experience of all agents (*gbest*). The actual movement of each agent is determined by the update rule (see below). Finally, as a stochastic algorithm, the final solution of PSO+ depends on its starting solutions. The purpose of loop1 is therefore to restart the system several times, from independent random solutions, to ensure a high overall success rate. At the final step, we use post-processing to remove and/or add some binding sites, therefore allowing zero or multiple binding sites on each sequence.

2.3 Solution Space and Fitness Function

To utilize the PSO+ algorithm, we need to represent a solution as a vector, and determine a fitness function appropriate for the problem. As discussed in Introduction, a motif of length l can be represented either as a position-specific weight matrix (PWM), which is a 4xl matrix of real numbers specifying the probability of each base at each position, or a consensus describing the most dominate base at each position. The matrix can be converted into a vector of length $4l$, although some care needs to be exercised to ensure proper normalization. The consensus representation is more efficient in searching for new instances, and may lead to faster convergence, while the PWM is more powerful in representing weaker motifs and is more accurate in evaluating the motif quality. We decided to use both representations in our algorithm, to take advantage of both forms. The solution is initialized as a consensus. During the scanning stage, we use the consensus representation. After the best matches to the consensus are found from all the

sequences, we compute a PWM based on the set of matches, and compute the fitness of the solution based on the PWM.

Given a solution, i.e., a consensus, it is first used to scan the input sequences to find a best match on each sequence. Let $X = (x_1 x_2 \dots x_l)$ be the consensus sequence and $Y = (y_1 y_2 \dots y_l)$ be a putative binding site. The matching score between X and Y is computed using the following equations:

$$M(X, Y) = \sum_i \sigma(x_i, y_i), \text{and}$$

$$\sigma(a, b) = \begin{cases} 1 + \log_4 (0.25/p_a) & \text{if } a = b \\ \log_4 (0.25/\sqrt{p_a p_b}) & \text{otherwise} \end{cases}$$

where p_a is the background frequency for base a in the input sequences or in the whole genome. As most genomes contain more AT's than CG's, this formula gives unequal weight to different types of match/mismatches. A match between two bases with lower background frequency would have higher score than that between two bases with higher background frequency. For uniform base frequency $p_A = p_C = p_G = p_T = 1/4$, $\sigma(a, b) = 1$ if $a = b$ and 0 otherwise, corresponding to an intuitive match/mismatch score.

Given a set of matches of a consensus, $W = w_1, w_2, \dots, w_n$ where each w_i is a subsequence with length l , we compute the fitness of the consensus using the information content (IC) score:

$$IC = \sum_{j=1}^{l} \sum_b f_b(j) \log_2(f_b(j)/p_b),$$

where $f_b(j)$ is the normalized frequency of nucleotide b on the column j of all instances in W and p_b is the background frequency of b.

2.4 Initial Solutions and Number of Agents

Similar to all stochastic algorithms, the performance of PSO+ partially depends on the initial solutions. The final solution of the algorithm can be significantly improved if at least one of the agents has an initial solution near the optimal solution. In this work we consider two strategies. The first strategy is to simply generate a set of random consensus. The second strategy is to randomly choose a subsequence from the input sequence as an initial solution. Although the first strategy would allow the maximum coverage, the probability that any randomly generated consensus is near the optimal solution is very low, as there are 4^l possible solutions for a motif of length l. For the second strategy, we assume that the actual binding site is closer to the consensus than is some random sequence. Since there is usually about one binding site per sequence, it is very likely that some agents may select a binding site as an initial solution. More precisely, assuming that the average sequence length is L and the motif length is l, the probability that a randomly selected sequence is a binding site is $1/(L - l + 1)$.

Also because of the Check Shift step in the algorithm, a random solution that contains a large suffix or prefix of the binding site can often lead to the recovery of the real motif quickly. We usually allow a binding site to be shifted by two bases to its left and right, respectively. Therefore, each true binding site can provide up to 5 initial solutions that are similar to the real motif. For this reason, we suggest the minimum number of agents to be $(L - l + 1)/5$ to ensure a high success rate. In our experiments on both synthetic and real sequences, we have found that the second strategy usually leads to much faster convergence and therefore is implemented as the default option.

2.5 Modified PSO+ Update Rule for Discrete Problems

After each iteration, each agent needs to update its *current* solution based on the old *current*, its own *pbest*, and *gbest* of the system, each of which is a vector. The standard PSO algorithm is designed for optimization problems in the continuous domain; therefore, a new solution can be easily obtained by a sum of the three solution vectors multiplied by some random weights. In our case, however, each solution is a vector of ACGT's and they cannot be manipulated by simple algebra such as multiplication and summation. In order to generate a new solution, we use the following rule, which is applied independently to each position of the motif:

$$i^* = \text{argmax}_i(c_i r_i \text{weight}(x_i)) \text{ and } x' = x_{i^*},$$

where x' is the new character being generated, x_1, x_2, x_3 are the characters in *current*, *pbest*, and *gbest*, respectively, x_4 is a random character from ACGT, c_i is a scaling factor to determine the relative importance of the four terms, r_i is a uniform random number, and the function weight gives a higher weight to characters having lower background frequency. This strategy effectively suppress characters with lower occurrence, when compared to an alternative strategy that randomly pick a character proportional to its number of occurrence in x_1, x_2, x_3 and x_4. For example, assuming $c_i = 1$ and the weight function returns a constant, if $x_1 = x_2 = x_3 =$ A and $x_4 =$ C, the alternative strategy would select A with 3/4 probability and C with 1/4 probability; with our strategy, we would select A with 7/8 probability and C with 1/8 probability. This reduces the probability of drifting away when a solution is near the optimal solution.

All agents stop their movements if the number of iterations exceeds an upper limit, or if there has been no update for a certain number of iterations and a minimum number of iterations has passed.

2.6 Check Shift

Similar to many motif finding algorithms, the output of PSO+ algorithm may have a shift issue: the start positions of the binding sites may be one or two positions away from the real positions, and it is difficult for the algorithm to escape from such local optima. To circumvent this problem, we periodically check whether shifting the binding sites by a small number (up to 3 bases) can improve the quality of the solution. This is done in the CHECK SHIFT step.

2.7 Gapped Motifs

Many real motifs contain *do-not-care* positions in their consensus. Furthermore, these *do-not-care* positions are often consecutive, forming a motif with two short conserved regions with some fixed distance in between. We call these *do-not-care* positions gaps and the motifs gapped motifs. Most motif finding algorithms do not consider gaps explicitly. For consensus-based algorithms, ignoring gaps can lead to serious mistakes when a consensus is forced to be selected for a gap position, which has no dominant characters.

We solve this problem by asking the users to provide two parameters, motif length l, and gap length k. Importantly, when we search for gapped motifs, we allow gaps to appear as the suffix or prefix of a motif. This is very useful if the actual gap is shorter than k, or if the non-gap region of the motif is shorter than $l - k$. Furthermore, if a real motif contains no gaps, our algorithm will automatically put all gaps in the flank regions. Therefore, although it seems that by asking a user to provide an addition parameter, gap length, our algorithm may increases the guesswork from the user, it actually gives the user more flexibility in determining the appropriate motif and gap lengths.

To find gapped motifs, we introduce a bit vector of length l. The bit vector contains exactly k 0's and $l - k$ 1's, where a 0 means the position falls in a gap. This vector is initialized randomly and later updated by masking out the columns in the PWM with low IC values. Given this vector, when calculating the match score between a consensus and a potential binding site, we only consider the columns with 1's. When we compute the fitness (IC score) of a motif given all the binding sites, we consider all columns, because in this case we have all the information to derive a PWM.

We consider two types of gaps. The first type of gaps can be anywhere in the motif and do not need to be consecutive. To find this type of gapped motifs, we simply mark the columns with the lowest IC value as gaps. The second type of gaps are consecutive and are located in the center of a motif. For this type of gapped motifs, we require a motif to contain at most two consecutive non-gapped regions, while gaps can appear either as a prefix, a suffix, or in the center of the motif. An algorithm using the second type of gaps is less efficient, but can often obtain better results for real motifs. This is the default option of our algorithm.

Although it seems that asking a user to provide an addition parameter, gap length, increases the guesswork from the user, this actually gives the user more flexibility in determining the appropriate motif/gap length. If the actual gap is shorter than the given number, or if the motif is shorter than expected, gaps can appear as suffix and/or prefix of the motif. Furthermore, the algorithm will automatically put all gaps in the flank regions if the real motif contains no gaps.

By default, we use *8-base motif with 0 gap* for short motifs, *12-base motif with 2 gaps* for medium-length motifs, and *16-base motif with 4 gaps* for long motifs. This strategy works well for the unknown length motif finding problems in this paper. The program allows a user to change the parameters by themselves.

Table 1. Running time (seconds) on (l,d)-motif challenge problems

Sequence length	400	500	600	800	1000	600	600	600	600
(l,d)	(15,4)	(15,4)	(15,4)	(15,4)	(15,4)	(11,2)	(13,3)	(17,5)	(19,6)
Weeder	60	125	200	450	900	-	-	-	-
Projection	9	23	42	162	418	4	13	94	174
MotifEnumerator	-	-	-	-	-	5	119	-	-
PSO	18	34	57	137	288	72	58	61	54
GALFP	100	123	161	212	286	127	137	162	172
GAME	27	30	32	36	41	23	28	34	42
PSO+	1.1	3.1	3.4	7.3	19.4	4.9	10.6	2.3	3.8

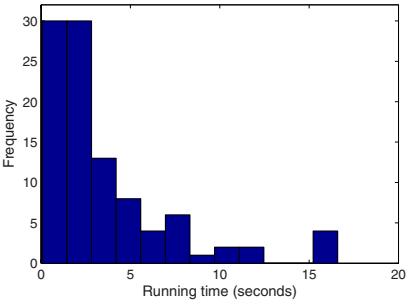


Fig. 2. Running time distribution of PSO+ on (15,4)-motif challenge problem (sequence length = 600). Results are based on 100 runs on independent sequences.

2.8 Post-processing

The basic algorithm described above assumes that there is one and only one binding site on each sequence, which is certainly not always true. To address this problem, we use a statistically-inspired strategy to refine the binding sites. We assume that at least a good fraction of the sequences contain at least one binding site. We calculate the match score for each putative binding site returned from the basic algorithm. Let Q1, Q2, and Q3 represent the lower quartile, median, and upper quartile of the match scores. The inter-quartile range (IQR) of the match scores is then computed by Q3 - Q1. All binding sites with a match score below Q1 - IQR are dropped as false binding sites. We also rescan the input sequences using the consensus for additional putative binding sites. A binding site with match score higher than Q2 is considered a true binding site. A PWM is constructed using the final set of binding sites.

3 Experimental Results

To evaluate the performance of our algorithm, we tested it on two types of sequences. The first type of test data consists of synthetic DNA sequences, also

known as the (l, d)-motif challenging problem [16]. The second type of data contains real promoter sequences. The algorithm is implemented in C.

3.1 Evaluation Using Synthetic Data Sets

We tested our algorithm on the (l, d)-motif challenge problem. Each challenge problem includes n sequences of length L, each of which contains a variant of a pre-defined consensus of length l. The variant were generated by choosing d positions randomly from the consensus and changing them to random bases.

In our first experiment, we focused on the (15, 4)-motif challenge problem, which is one of the most popular benchmarks for motif finding programs. We chose $n = 20$, and varied L from 400 to 1000. Table 1 (left half) shows the running time of our current algorithm (PSO+), another PSO-based algorithm we developed recently (PSO) [12], two evolutionary algorithms (GAME [10] and GALFP [11]), and three well-known combinatorial-search algorithms (Projection [9], Weeder [6], and MotifEnumerator [17]). We were not able to test Weeder by ourselves because the currently available implementation of the algorithm can only handle motifs of even lengths up to 12. Its running time was taken from the original publication and was based on an 89% success probability [6]. The results of the other algorithms were obtained by downloading the programs from the original authors' websites and running with parameters that can recover the embedded motifs with 100% accuracy. Running time was based on the average of 10 runs on 5 sets of sequences.

As shown in Table 1, PSO+ is significantly more efficient than the other algorithms. Our previous PSO algorithm was based on a very different problem formulation and is considerably slower than PSO+. It is worth noting that the running time of the evolutionary algorithms, GAME and GALFP, are much slower than PSO+ in general, but their running time only increases slightly with sequence lengths. This might be because these two algorithms spend a significant amount of time in initialization, independent of sequence lengths.

Next, we compared these algorithms on their performance on challenge problems with varying motif lengths and number of errors. Sequence lengths were fixed at 600. The current algorithm outperforms the existing algorithms again on these test sequences (Table 1, right half). Similar to our previous PSO-based algorithm, the running time of PSO+ is relatively independent of motif lengths.

Finally, we run PSO+ 100 times on independent sequences, and plotted the running time in Fig. 2. As shown, the running time has a long-tail distribution, meaning in some rare cases the algorithm may need extremely long time to converge. Because of this, the algorithm runs faster than the average running time in most cases. A better strategy for detecting local optima may eliminate some of these rare cases and improve the overall efficiency.

3.2 Experiments on Real Biological Sequences

To test the performance of our algorithm on real biological sequences, we used the same eight representative test cases used in [11], which covered different

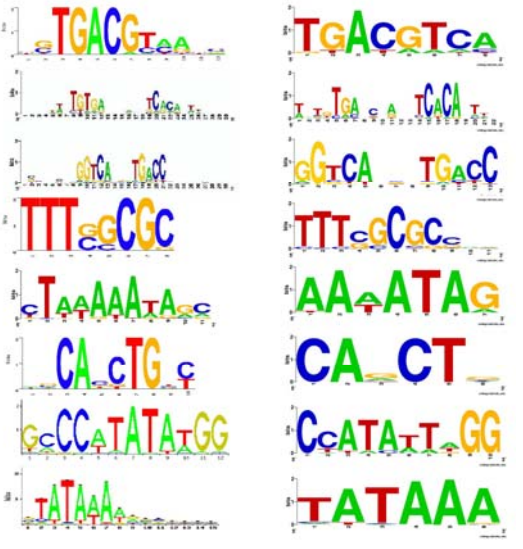


Fig. 3. Motif Logos. Left: real motifs; right: motifs found by our algorithm. The motifs are listed in the same order as in Table 2.

lengths of motifs and both gapped and non-gapped motifs. In these data sets, some sequences may contain zero or more binding sites. Details of the data sets are available in [11]. Based on these data sets, it has been reported that two genetic algorithms, GALFP [11] and GAME [10], are significantly more accurate than five popular algorithms: MEME [2], Bioprospector (BP) [5], BioOptimizers based on MEME (BOM) and BioOptimizers based on Bioprospector (BOB) [18]. Therefore, we only compared PSO+ with GALFP and GAME directly.

As in [11], we measure accuracy by *precision*, *recall*, and *F-score*. Precision is defined as c/p and recall is defined as c/t, where c, p and t are the number of correctly predicted binding sites, the number of predicted binding sites and the number of true binding sites in the sequences, respectively. The *F-score* combining both *precision* and *recall* is defined as $F = 2*\text{Precision}*\text{Recall}/(\text{Precision}+\text{Recall})$. Table 2 shows the average results of PSO+, GALFP [11] and GAME [10] on 20 runs (the bolded entries are the winners). As shown, PSO+ and GALFP have about the same accuracy. PSO+ has the best *F-scores* on 4 of 8 test cases while GALFP has the best *F-scores* on 3 test cases. More in-depth investigation reveals that PSO+ generally has the highest precision (5 out of 8), while GALFP has the highest recall (3 out of 8). This indicates that PSO+ reported less but more accurate binding sites than GALFP.

Interestingly, two of the eight cases (CRP, ERE) are clearly gapped motifs, as shown in Fig. 3. PSO+ won in both cases, especially in ERE (average F-score: 0.92, 0.79, and 0.62, for PSO+, GALFP and GAME, respectively), indicating that the gap model in our algorithm is effective. On the other hand, even though our algorithm has a much larger search space by allowing gaps, its performance

Table 2. Comparisons of average performance on the 8 real datasets

	GAME			GALFP			PSO+		
	Precision	*Recall*	*F-score*	*Precision*	*Recall*	*F-score*	*Precision*	*Recall*	*F-score*
CREB	0.43	0.42	0.42	0.70	**0.84**	**0.76**	**0.76**	0.68	0.72
CRP	0.79	**0.78**	0.78	0.99	0.73	0.84	**1**	**0.78**	**0.88**
ERE	0.52	0.78	0.62	0.82	0.76	0.79	**0.92**	**0.92**	**0.92**
E2F	**0.79**	**0.87**	**0.83**	0.77	0.85	0.81	0.68	0.7	0.69
MEF2	0.52	0.55	0.53	0.91	0.98	0.95	**1**	**1**	**1**
MYOD	0.14	0.14	0.14	**0.57**	**1**	**0.72**	0.20	0.43	0.27
SRF	0.71	0.86	0.78	0.75	**0.89**	**0.82**	**0.80**	0.56	0.66
TBP	0.81	0.74	0.77	**0.87**	0.87	0.87	0.86	**0.91**	**0.88**

Fig. 4. Logos for the ERE motif. Left: real motif; middle: motif found by PSO+ with the gap option turned off; right: motif found by PSO+ with the gap option turned on.

on the other six motifs that do not have gaps is still among the best. Fig. 4 shows the results of PSO+ on the ERE motif, with or without the gap option. Without the gap option, our algorithm attempts to maximize the total information content of all positions, which results in a motif with relatively uniform information content across all positions. In contrast, with the gap option, our algorithm automatically determines the low-information positions and treats them as gaps, and therefore improves the accuracy of the final result. With the gap option turned on, the Pearson correlation coefficient between the predicted and true motif PWMs is improved from 0.92 ($p = 10^{-22}$) to 0.97 ($p = 10^{-33}$).

Finally, it takes our algorithm 20 to 60 seconds for each of the 8 test cases. In comparison, the average running time is about 62 seconds for GALFP, and 291 seconds for GAME.

4 Conclusions and Discussion

In this work, we have proposed a novel algorithm for finding DNA motifs based on Particle Swarm Optimization (PSO). Our contributions include a novel modification of the PSO update rule to allow discrete variables, a model to allow gapped motifs, and a simple method to fine-tune the motif when some sequences contain zero or multiple binding sites. Experimental results on synthetic challenge problems as well as real biological sequences show that our method is both more efficient and more accurate than several existing algorithms, especially when gaps are present in the motifs. We are working to finalize our program, which will be freely available to the research community soon.

Acknowledgments

This work was supported in part by a NIH grant 1SC3GM086305-01 to JR.

References

1. Tompa, M., Li, N., Bailey, T., Church, G., De Moor, B., Eskin, E., Favorov, A., Frith, M., Fu, Y., Kent, W., Makeev, V., Mironov, A., Noble, W., Pavesi, G., Pesole, G., Régnier, M., Simonis, N., Sinha, S., Thijs, G., van Helden, J., Vandenbogaert, M., Weng, Z., Workman, C., Ye, C., Zhu, Z.: Assessing computational tools for the discovery of transcription factor binding sites. Nat. Biotechnol. 23, 137–144 (2005)
2. Bailey, T., Elkan, C.: Fitting a mixture model by expectation maximization to discover motifs in biopolymers. Proc. Int. Conf. Intell. Syst. Mol. Biol. 2, 28–36 (1994)
3. Roth, F., Hughes, J., Estep, P., Church, G.: Finding DNA regulatory motifs within unaligned noncoding sequences clustered by whole-genome mrna quantitation. Nat. Biotechnol. 16, 939–945 (1998)
4. Lawrence, C., Altschul, S., Boguski, M., Liu, J., Neuwald, A., Wootton, J.: Detecting subtle sequence signals: a gibbs sampling strategy for multiple alignment. Science 262, 208–214 (1993)
5. Liu, X., Brutlag, D., Liu, J.: Bioprospector: discovering conserved DNA motifs in upstream regulatory regions of co-expressed genes. In: Pac. Symp. Biocomput., pp. 127–138 (2001)
6. Pavesi, G., Mauri, G., Pesole, G.: An algorithm for finding signals of unknown length in DNA sequences. Bioinformatics 17, S207–S214 (2001)
7. Sinha, S., Tompa, M.: Discovery of novel transcription factor binding sites by statistical overrepresentation. Nucleic Acids Res. 30, 5549–5560 (2002)
8. Keich, U., Pevzner, P.: Finding motifs in the twilight zone. Bioinformatics 18, 1374–1381 (2002)
9. Buhler, J., Tompa, M.: Finding motifs using random projections. J. Comput. Biol. 9, 225–242 (2002)
10. Wei, Z., Jensen, S.T.: GAME: detecting cis-regulatory elements using a genetic algorithm. Bioinformatics 22, 1577–1584 (2006)
11. Chan, T.M., Leung, K.S., Lee, K.H.: TFBS identification based on genetic algorithm with combined representations and adaptive post-processing. Bioinformatics 24(3), 341–349 (2008)
12. Lei, C., Ruan, J.: A novel swarm intelligence algorithm for finding DNA motifs. International Journal of Computational Biology and Drug Design 2, 323–339 (2009)
13. Zhou, W., Zhou, C., Liu, G., Huang, Y.: Identification of transcription factor binding sites using hybrid particle swarm optimization. In: Ślęzak, D., Yao, J., Peters, J.F., Ziarko, W.P., Hu, X. (eds.) RSFDGrC 2005. LNCS (LNAI), vol. 3642, pp. 438–445. Springer, Heidelberg (2005)
14. Hardin, C., Rouchka, E.: DNA motif detection using particle swarm optimization and expectation-maximization. In: Proceedings of the 2005 IEEE Swarm Intelligence Symposium (2005)
15. Eberhart, R., Shi, Y., Kennedy, J.: Swarm Intelligence. Morgan Kaufmann, San Francisco (2001)
16. Pevzner, P., Sze, S.: Combinatorial approaches to finding subtle signals in DNA sequences. Proc. Int. Conf. Intell. Syst. Mol. Biol. 8, 269–278 (2000)
17. Sze, S.H., Zhao, X.: Improved pattern-driven algorithms for motif finding in DNA sequences, pp. 198–211 (2006)
18. Jensen, S.T., Liu, J.S.: Biooptimizer: a bayesian scoring function approach to motif discovery. Bioinformatics 20(10), 1557–1564 (2004)

Top-Down Induction of Phylogenetic Trees

Celine Vens[1,2], Eduardo Costa[1], and Hendrik Blockeel[1,3]

[1] Department of Computer Science, Katholieke Universiteit Leuven,
Celestijnenlaan 200A, 3001 Leuven, Belgium
[2] Institut National de Recherche Agronomique, UMR 1301,
400 Route des Chappes, 06903 Sophia-Antipolis, France
[3] Leiden Institute of Advanced Computer Science, Universiteit Leiden,
Niels Bohrweg 1, 2333 CA Leiden, The Netherlands
{celine.vens,eduardo.costa,hendrik.blockeel}@cs.kuleuven.be

Abstract. We propose a novel distance based method for phylogenetic tree reconstruction. Our method is based on a conceptual clustering method that extends the well-known decision tree learning approach. It starts from a single cluster and repeatedly splits it into subclusters until all sequences form a different cluster. We assume that a split can be described by referring to particular polymorphic locations, which makes such a divisive method computationally feasible. To define the best split, we use a criterion that is close to Neighbor Joining's optimization criterion, namely, minimizing total branch length. A thorough experimental evaluation shows that our method yields phylogenetic trees with an accuracy comparable to that of existing methods. Moreover, it has a number of important advantages. First, by listing the polymorphic locations at the internal nodes, it provides an explanation for the resulting tree topology. Second, the top-down tree growing process can be stopped before a complete tree is generated, yielding an efficient gene or protein subfamily identification approach. Third, the resulting trees can be used as classification trees to classify new sequences into subfamilies.

1 Introduction

We consider the problem of deriving from a set of aligned DNA or protein sequences the most likely phylogenetic tree. Many methods have been proposed for this. One popular approach consists of computing a dissimilarity measure between each pair of sequences, and then using the resulting matrix to infer the tree. Methods that use this approach are called distance based methods, and are represented by the well-known Neighbor Joining (NJ) [1,2] algorithm.

NJ is essentially a so-called agglomerative hierarchical clustering algorithm: starting from one cluster per sequence, it iteratively merges clusters of sequences until a single cluster is obtained. While agglomerative clustering algorithms are among the most popular ones for clustering, many other clustering algorithms exist. Among these algorithms, one can distinguish extensional and conceptual clustering algorithms. In extensional clustering, a cluster is described by enumerating its elements, whereas in conceptual clustering, a cluster is described

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2010, LNCS 6023, pp. 62–73, 2010.

by listing properties of the cluster's elements, using a particular description language. In the context of phylogenetic analysis, a natural conceptual language would be one that refers to polymorphic locations; for instance, one cluster might be described as "all sequences having a C at position 72 and A at position 31".

If we assume that it must be possible to describe clusters using a particular language, the space of all possible clusterings is greatly reduced. This enables us to use a so-called divisive clustering method, which starts from a single cluster and repeatedly divides it into subclusters until all sequences form a different cluster. Normally, given a set of N sequences, there are 2^N ways to split it into two subsets. But if we assume that the split can be described by referring to a particular polymorphic location, the number of splits is linear in the length of the sequences, and constant in the size of the set, making such a divisive method computationally feasible, and potentially faster than agglomerative methods. A similar observation was made by [3], who were the first to propose a top-down clustering method for phylogenetic tree reconstruction. Differences between their algorithm and our work are described in the related work section.

Blockeel et al. [4] discuss how a simple extension of decision tree learning leads to a (general-purpose) divisive conceptual clustering algorithm. In this paper we address the question to what extent this approach lends itself to phylogenetic tree reconstruction. If it works well, that is, if it yields phylogenetic trees with an accuracy comparable to that of existing methods, such an approach would have a number of important advantages over existing methods. First, as just argued, it may be faster than methods based on agglomerative clustering. Second, as each division into subclusters is defined by polymorphic locations, the resulting tree immediately gives an evolutionary trace, which can be useful for recognizing functional sites [5]; current methods for this typically involve a two-step process. Third, by using different stopping criteria, the divisive method can be used not only to reconstruct complete phylogenetic trees, but also to identify subfamilies of genes or proteins, without having to grow a complete tree and cutting it afterwards [6]. Finally, since the phylogenetic tree has the form of a classification tree, it can be used directly to classify newly available sequences into such subfamilies, by simply sorting them down the tree into the leaf nodes.

In order to study the top-down conceptual clustering approach in the context of phylogenetic tree reconstruction, we propose a method that is strongly based on an existing decision tree learner; the only change made to it is the heuristic used for splitting nodes. We thoroughly evaluate this new method with respect to accuracy and efficiency. The conclusion is that the method achieves comparable accuracy as existing state-of-the-art methods, with a tendency to perform better for highly symmetric trees, and somewhat worse for highly asymmetric trees.

2 Background and Related Work

The Neighbor Joining (NJ) algorithm [1] is one of the most widely used methods for phylogenetic tree reconstruction. It is a heuristic estimation of the minimum evolution tree, which is the tree with minimal sum of branch lengths, and assumes

a distance matrix with pairwise distances between all sequences is given. It uses a hierarchical clustering approach, at each stage of the clustering grouping two OTUs (Operational Taxonomic Units) together. The original Neighbor Joining algorithm has a complexity $O(N^5)$ [7]. However, Studier and Keppler [2] have presented an alternative version, which is $O(N^3)$. This makes NJ one of the most efficient algorithms for phylogenetic tree reconstruction [8].

Another efficient method, and among all methods the one that most closely resembles ours, is the PTDC (Phylogeny by Top Down Clustering) algorithm [3]. This algorithm shares with our proposal the idea of recursively partitioning clusters. The main differences are: (1) our approach makes the link to the decision tree learning framework, which allows us to exploit the highly developed state of the art in that area; (2) PTDC's heuristic maximizes the average of all pairwise distances between sequences of different clusters, making it similar to the UPGMA algorithm, from which it inherits some undesirable characteristics such as sensitivity to unequal substitution rates in different lineages [9], whereas our method uses a heuristic that approximates the Neighbor Joining criterion; (3) while PTDC splits clusters based on equality of subsequences, our approach creates splits based on a single, most informative, position; this gives it an efficiency advantage.

3 Method

As said, we build on the state of the art in decision tree learning. Decision trees are typically built top-down using a recursive partitioning method. Given a dataset and a set of tests (where a test gives one of several possible outcomes when applied to a single data element), they find the test that optimally partitions the dataset into subsets (for some definition of optimal), and keep repeating this procedure on the subsets until subsets of size one are obtained, or another stopping criterion is fulfilled. The procedure is best known in the context of classification trees [10], but it can equally be applied for conceptual clustering [4], in which case tests are considered better if the similarity within the created subsets is higher.

In the context of phylogenetic tree construction, data elements are sequences. A test can for instance check for the occurrence of a particular nucleotide or amino acid at a particular location; this is the kind of tests we use in our method. More specifically, we allow tests of the form "$p(x) = y$" or "$p(x) \in Y$", where $p(x)$ returns the nucleotide or amino acid at position x, and where y is a nucleotide or amino acid, and Y a set of them. The test "$p(5) \in \{A, T\}$", for instance, creates two subsets, one containing all sequences with an A or T at position 5 and one containing all other sequences.

Decision tree learners typically stop partitioning sets when all elements of the set are sufficiently similar, e.g., for classification trees, when all elements belong to the same class. For phylogenetic trees, one could do something similar, if the tree need only be built up to the level of subfamilies; here, however, we will focus on building complete trees, i.e., we keep splitting until all subsets are singletons.

The only question that remains, then, is how to determine the quality of a split. Intuitively, one could argue that an "old" mutation, i.e., one that occurred a long

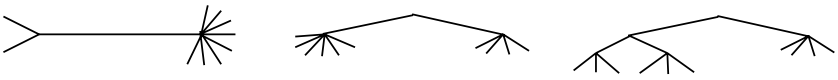


Fig. 1. Split topologies. Left: NJ, center: Clus-φ (root), right: Clus-φ (non-root).

time ago in evolutionary history, is more likely to have led to two different branches that by now have grown far apart, than a more recent mutation. As we are building a rooted tree, where we hope to find older mutations nearer to the root, we should prefer tests that create subsets that are far apart. A natural heuristic for selecting tests is then simply: compute the average distance between elements from different subsets induced by the test; choose the test that maximizes that average distance. This is the heuristic that Arslan et al. use [3]. Alternatively, one could use an extension of the information gain heuristic that is commonly used for classification trees; Clare et al. [11] define such an extension for multilabel classification trees, and this heuristic is available in the decision tree learner on which our system is based.

These standard heuristics, however, do not take into account the particular requirements of phylogenetic tree reconstruction. Using the average distance heuristic, one essentially gets the top-down counterpart of the UPGMA algorithm, which is known to have some undesirable behavior [8]. NJ works in essentially the same way as UPGMA, but uses a more advanced distance metric for deciding which two clusters to merge, and this yields better results. This raises the question whether a top-down counterpart of NJ's heuristic can be developed.

Such a heuristic would have to estimate the total branch length of the tree that will finally be constructed, using the pairwise distances provided in a distance matrix D. Using the same reasoning as in the derivation of NJ's heuristic [1], we can define an equivalent heuristic function for splitting the root node:

$$H(T_l, T_r) = \frac{1}{|T_l||T_r|} \sum_{\substack{x_i \in T_l \\ x_j \in T_r}} D_{x_i x_j} + \frac{1}{|T_l|} \sum_{\substack{x_i, x_j \in T_l \\ i<j}} D_{x_i x_j} + \frac{1}{|T_r|} \sum_{\substack{x_i, x_j \in T_r \\ i<j}} D_{x_i x_j}, \qquad (1)$$

where T_l and T_r denote the set of sequences in the left and right subtrees of the split, respectively. This formula computes the total branch length of a "double star"-shaped topology (see Fig 1, center), which is an upper bound for the actual total branch length of the final tree. It is a generalization of the NJ heuristic, which has $|T_l| = 2$ and $|T_r| = N-2$, with N the total number of sequences (Fig 1, left). The previous equation can be rewritten into a more efficient formulation,

$$H(T_l, T_r) = \frac{1}{|T_l||T_r|} \Big(\sum_{\substack{x_i, x_j \in T_l \cup T_r \\ i<j}} D_{x_i x_j} + (|T_r| - 1) \sum_{\substack{x_i, x_j \in T_l \\ i<j}} D_{x_i x_j} + (|T_l| - 1) \sum_{\substack{x_i, x_j \in T_r \\ i<j}} D_{x_i x_j} \Big), \qquad (2)$$

where the first term is constant for the node to be split.

For splitting the other internal nodes, a more complex heuristic function is needed, since the particular split influences the length of other branches in the tree, and hence, the total branch length (Fig 1, right). Again, using a similar reasoning as applied by Saitou and Nei [1], it can be verified that the function to minimize is given by

$$H(T_l, T_r, T_o) = \frac{1}{|T_l||T_r|}\Big(\sum_{\substack{x_i,x_j \in T_l \cup T_r \\ i<j}} D_{x_i x_j} + (2|T_r|-1) \sum_{\substack{x_i,x_j \in T_l \\ i<j}} D_{x_i x_j} + (2|T_l|-1) \sum_{\substack{x_i,x_j \in T_r \\ i<j}} D_{x_i x_j} + \frac{|T_r|}{|T_o|} \sum_{\substack{x_i \in T_o \\ x_j \in T_l}} D_{x_i x_j} + \frac{|T_l|}{|T_o|} \sum_{\substack{x_i \in T_o \\ x_j \in T_r}} D_{x_i x_j} \Big), \quad (3)$$

with T_o denoting the set of other sequences in the tree (i.e., not in $T_l \cup T_r$).

The computational complexity of a decision tree learning method is roughly $O(aN \log N)$ with a the number of tests and N the number of elements in the original dataset, under the assumption that a reasonably symmetric tree is built (the depth of which is logarithmic in the number of leaves) and the evaluation of a single test takes linear time in the size of the dataset. This scales much better in N than agglomerative methods, which have complexity $O(N^3)$ [2]. As such, such a divisive method may be much more efficient when analyzing large sets of sequences. The NJ-based heuristic, however, is more expensive to compute than the average distance or information gain heuristics; it is quadratic, rather than linear, in the size of the dataset being subdivided, which increases the overall complexity of the method to $O(aN^2 \log N)$.

Our algorithm is called Clus-φ and is implemented in the Clus system (`http://www.cs.kuleuven.be/~dtai/clus`).

4 Experiments

We have tested our alternative method for phylogenetic tree construction on a number of datasets. In all experiments, we used the Jukes-Cantor correction formula [12] to compute the genetic distance between two DNA sequences, and the Jones-Taylor-Thornton matrix [13] for protein sequences. We measure differences between trees using the quartet distance [14]. This distance gives the number of quartets, i.e. subtrees induced by four leaves, that differ between the two trees being compared.

4.1 Real Datasets

In a first set of experiments, we check how much Clus-φ trees differ from the ones returned by Neighbor Joining (NJ). As a reference point, we include the difference between parsimony methods and NJ[1]. To construct the NJ and parsimony trees we used the programs neighbor and dnapars/protpars from the Phylip software package [15], respectively.

[1] When parsimony analysis returns multiple trees, we report the average difference.

Table 1. Quartet distance for real datasets

Dataset	Type	Size	Length	NJ vs. Clus-φ	NJ vs. Pars
Chimp	DNA	5	896	0	2
cynmix	DNA	32	3080	11680	16543
Primates	DNA	21	1500	184	632
SIV	DNA	41	6042	21350	10608
hivALN	DNA	14	2352	0	156
InvertebrateEF	DNA	16	1050	200	152
mtDNA	DNA	17	1998	218	26
VertebrateMtCOI	DNA	8	1509	178	31
g3pdh	Protein	14	313	180	65.33
gpd	Protein	12	234	52	18
gdpAA	Protein	12	422	73	97.50

Table 1 reports the quartet distance for 11 datasets used in [8]. While the trees generated by Clus-φ are very similar to those generated by NJ for some datasets (for datasets Chimp and hivALN, Clus-φ and NJ generated identical phylogenetic trees), there are larger differences for other datasets. However, this variation is comparable to the variation between the parsimony method and NJ.

The question is then whether, in those cases where NJ and Clus-φ differ, any of them is more likely to be correct than the other one. To check this, we tested Clus-φ on a number of synthetic datasets, where the correct tree is known. These experiments are discussed in the next section.

4.2 Synthetic Datasets

The synthetic datasets were generated by simulating an evolutionary process, using the coding sequence simulation program EvolveAGene3 [16], with a randomly generated DNA sequence as input. By default, this software produces symmetric trees, i.e., binary trees that have all leaves at the same depth. However, also random tree topologies can be produced. The average branch length is set to 0.05 in all experiments, meaning that each branch has an average mutation rate of 5%. All other parameters are set to their default value.

First, we evaluate our proposed heuristic (see Section 3). Second, we compare the trees obtained by Clus-φ to those obtained by NJ in terms of similarity with the true tree topology. Finally, we investigate the influence of the number of sequences on quartet distance and computational cost for both NJ and Clus-φ.

Analysis of the Heuristic

To evaluate our proposed heuristic, we compare trees constructed using our heuristic to trees constructed with two other heuristics for decision trees. The first heuristic is an extension of the standard information gain heuristic towards multilabel classification [11]; we call this implementation Clus-IG. The second heuristic is similar[2] to the one used in the PTDC algorithm [3]. It recursively

[2] The only difference between our implementation and the one discussed in [3] is that we create splits based on a single position instead of based on subsequences.

Table 2. Number of wins/losses of Clus-φ's heuristic compared to information gain (Clus-IG) and inter-cluster distance (Clus-PTDC)

	Clus-φ vs Clus-IG	Clus-φ vs Clus-PTDC
Symmetric	113/78 (9 ties)	200/0
Random	113/87	200/0

splits the data by looking, in each step, for the two subclusters with the largest average inter-cluster distance. We call this implementation Clus-PTDC.

In order to perform this experiment, we generated 400 synthetic datasets: 200 based on symmetric topologies and 200 based on random topologies, with an input sequence length of 250 (200 datasets) and 900 (200 datasets)[3], each dataset containing 128 sequences. Table 2 presents a summary of the results, in terms of the number of wins/losses of Clus-φ in the comparison with Clus-IG and Clus-PTDC, according to the quartet distance from the inferred trees to the true tree. As can be seen from the results, Clus-φ presents a higher number of wins than Clus-IG, and is better than Clus-PTDC for all datasets. It shows that, in the context of phylogenetic tree reconstruction, the new heuristic yields better results than standard heuristics for decision trees.

Comparison between Clus-φ and NJ in terms of similarity to the true tree topology

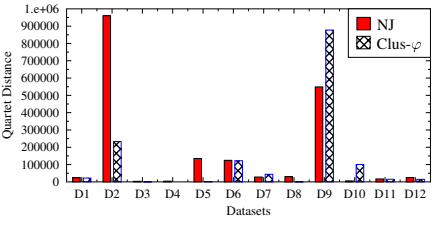
In this experiment we compare the similarity with the true tree for NJ and Clus-φ. We generated 100 synthetic datasets based on symmetric topologies and 100 datasets based on random topologies, containing 128 sequences and using an input sequence of length 900. A summary of the results, in terms of the number of datasets for which each method presents the best performance, is shown in Table 3. On average, NJ performs slightly better than Clus-φ; however the table shows a large difference in the results for symmetric and random tree topologies. For symmetric trees, the Clus-φ tree is in general closer to the true tree than the tree produced by NJ. For the random tree topologies, on the other hand, Clus-φ is closer to the true tree in only 5 cases. The reason for this difference in behavior is currently unclear.

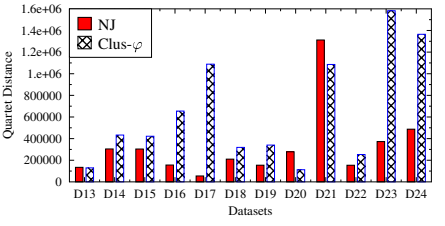
Interestingly, we noted that in a few cases, the trees produced by Clus-φ are identical to the true topologies; this occurred for 5 out of 100 cases for the datasets generated from symmetric topologies. For NJ, that is never the case. The probability of observing at most 0 correct predictions for NJ and at least 5 for Clus-φ, under the null hypothesis that both have the same probability p of making a correct prediction, depends on p but is always less than 0.01; this implies that the hypothesis that Clus-φ does not have a higher probability of predicting the correct tree than NJ is rejected at the 1% significance level.

[3] As insertions and deletions are allowed in EvolveAGene3, the final sequence length is usually larger than the input sequence length, e.g. for an input sequence of length 900, the length of the final sequences is around 1000.

Table 3. Number of wins of NJ and Clus-φ for synthetic datasets

	NJ	Clus-φ
Symmetric	32	**68**
Random	**95**	5





D1	D2	D3	D4	D5	D6
0.91	0.24	0.16	0.00	0.01	0.98
D7	**D8**	**D9**	**D10**	**D11**	**D12**
1.57	0.02	1.60	17.63	0.87	0.59

D13	D14	D15	D16	D17	D18
0.96	1.42	1.39	4.20	19.88	1.52
D19	**D20**	**D21**	**D22**	**D23**	**D24**
2.20	0.40	0.83	1.64	4.25	2.80

Fig. 2. Quartet distance to real tree, for symmetric (left) and random (right) trees. Graphs: absolute values; Tables: quartet distance of Clus-φ relative to that of NJ.

In Figure 2 we show the quartet distance for 12 datasets of each kind, in order to analyze the results in more detail. We can see that for random trees the differences between the two methods are larger. The graphs also show the strong variation of tree quality with the dataset: the dataset has a larger influence on the overall performance than the choice of method.

From the results shown in Table 3 and Fig. 2 we can conclude that Clus-φ tends to perform better for symmetric tree topologies, while NJ tends to perform better for random topologies. In a sense, these settings are at both ends of a spectrum. The question is then how the results differ for datasets based on trees that are neither perfectly symmetric nor completely random, which is what we expect to occur in nature.

To investigate this question, we have generated datasets based on a series of tree topologies, starting from a perfectly symmetric tree, and gradually adding more random tree structure. More precisely, we considered a tree operation that takes two subtrees, one consisting of a single leaf, and another one consisting of one internal node and two leaves, and switches them. Table 4 reports the results for an experiment that counts the number of wins for NJ and Clus-φ on 100 datasets with an increasing number of operations. Each dataset again has 128 sequences generated from a DNA input sequence of 900 positions.

As can be seen from the results, the performance of Clus-φ, compared to NJ, decreases with the increase of the number of modifications in the symmetric tree. However, it is important to notice that this decrease of the performance of Clus-φ occurs gradually, which means that it presents good results not only for

Table 4. Analysis of the effect of the symmetry of the tree on the performance of NJ and Clus-φ. The numbers in bold represent the largest number of wins for each line.

Dataset	NJ	Clus-φ	Dataset	NJ	Clus-φ
Symmetric	32	**68**	35 steps	**65**	35
5 steps	**56**	44	40 steps	**66**	34
10 steps	**56**	44	45 steps	**77**	23
15 steps	**53**	47	50 steps	**76**	24
20 steps	**68**	32	55 steps	**77**	23
25 steps	**65**	35	60 steps	**81**	19
30 steps	**63**	37	Random	**95**	5

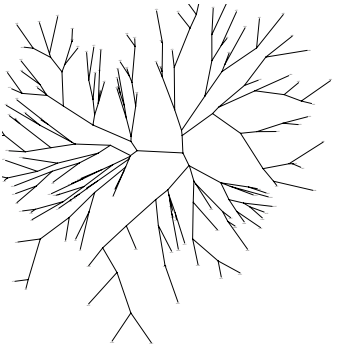


Fig. 3. Tree topology with 40 steps from symmetry

completely symmetric trees, but also for almost symmetric trees: up till 40 steps, Clus-φ finds a better approximation of the true tree in one out of three datasets. As an illustration, Fig. 3 shows a topology with 40 steps from symmetry.

Analysis of the Influence of the Number of Sequences on Quartet Distance and Computational Cost

In this section, we analyze the effect of the number of sequences on the performance of the algorithms, both in terms of tree reconstruction, and computational cost. For this analysis, we use datasets based on symmetric tree topologies.

To analyze how NJ and Clus-φ scale in the number of sequences, a number of synthetic datasets that contain sequences of the same length[4], but with an increasing number of sequences were generated.

Figure 4 (left) shows run times for datasets with 300 nucleotides. Each point in the curve shows the average run times over 20 datasets of the specified size. As we can see, between 1000 and 2000 sequences, the run times of Clus-φ become smaller than those of NJ. For 8000 sequences, Clus-φ is about 5 times faster (the exact values are 1517 seconds for Clus-φ and 7990 seconds for NJ).

[4] To ensure an equal dataset length, we disabled insertions and deletions here.

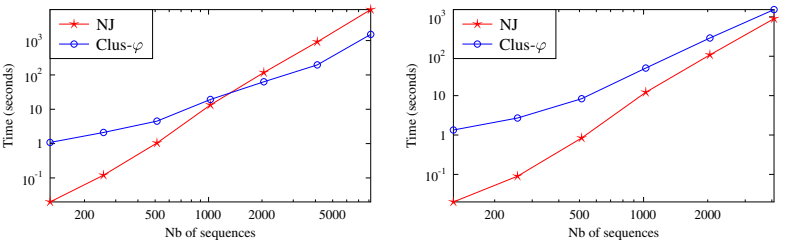


Fig. 4. Running times for Clus-φ and NJ (logarithmic scale). Sequence length: 300 (left) and 900 (right) nucleotides.

Table 5. Number of wins of NJ and Clus-φ for synthetic datasets with sequences of length 900 according to the quartet distance.

Nb Sequences	NJ	Clus-φ	Ties
8	0	**1**	19
16	2	**7**	11
32	2	**9**	9
64	5	**13**	2
128	6	**14**	0
256	**11**	9	0
512	**13**	7	0
1024	**13**	7	0
2048	**18**	2	0
4096	**15**	5	0

Since our method scales linearly in the sequence length (as each split requires inspecting all positions; see Sect. 3), while NJ is constant in the length, we also show the run times for datasets with longer sequences. Figure 4 (right) shows that NJ is more efficient than Clus-φ for datasets with 900 nucleotides. However, as the relative run time difference between both methods decreases, we expect that Clus-φ will be faster than NJ for datasets with more than 4000 sequences[5].

We also computed the quartet distance of the trees produced by NJ and Clus-φ to the true tree for the datasets with 900 nucleotides (see Table 5). It can be noted that Clus-φ obtains better results than NJ for datasets with few sequences. However, between 128 and 256 sequences, the results change, making NJ the best method for a large number of sequences. The reason for this is related to the relatively small sequence length: in order to generate 256 sequences or more with only 900 nucleotides, each branch having on average 45 mutations, a lot of nucleotide positions are likely to have many mutations. This negatively influences Clus-φ, because it can not find the right splits anymore. How this can be dealt with (e.g. by post-processing the trees using local rearrangements) will be investigated in future work.

[5] It is infeasible to generate datasets with more than 4000 sequences of length 900 with the EvolveAGene3 program.

4.3 Comparison to PTDC

As a last experiment, we compare Clus-φ to PTDC [3]. The authors have used a single dataset to evaluate their method and report the resulting tree in their article. The dataset was prepared by Rennert et al. [17], who also report a maximum likelihood tree for the alignment, which we use as reference tree. For this experiment, both PTDC and Clus-φ calculate pairwise protein distances using the PAM1 substitution matrix [18]. Given that the reference tree in [17] is not binary (it contains a node with six branches), it is impossible to calculate the quartet distance. Therefore, for this experiment, we compare the trees using the symmetric difference measure. The symmetric difference between two trees is the number of binary node partitions that are found in one tree and not in the other. The symmetric difference to the reference tree is 11 for PTDC, and 10 for Clus-φ. We conclude that our method finds a slightly better tree.

5 Conclusion

We have proposed Clus-φ, a novel method for reconstruction of phylogenetic trees. The method differs strongly from state-of-the-art methods, both from an algorithmic point of view and from the point of view of the information it uses.

It is based on a conceptual clustering method that extends the well-known decision tree learning approach. Starting from a single cluster, it repeatedly splits the sequences into subclusters until all sequences form a different cluster, or until a sufficient level of detail is obtained to study gene or protein families. To define the best split, our method uses a criterion that is close to Neighbor Joining's optimization criterion, namely, constructing a phylogenetic tree with minimal total branch length.

Our method assumes that a split can be described by referring to particular polymorphic locations, which makes such a divisive method computationally feasible, and at the same time provides an evolutionary trace for the resulting tree topology. Moreover, by checking the polymorphic locations listed in the internal nodes, new sequences can easily be classified into subfamilies.

We have shown that the performance of Clus-φ is close to that of Neighbor Joining w.r.t. quality of the produced trees, while having a larger tendency to produce the correct tree.

We propose Clus-φ not as a substitute for Neighbor Joining or other standard methods for phylogeny reconstruction, but as a method to be used complementarily to these methods. The fact that Clus-φ behaves differently than Neighbor Joining, in terms of performance and computational cost, shows that their results can be used to complement one another.

Acknowledgments

Celine Vens is a Postdoctoral Fellow of the Research Foundation - Flanders (FWO). Eduardo Costa is supported by the Research Foundation - Flanders

(FWO) and the GOA Probabilistic Logic Learning. The authors would like to thank Etienne Danchin for providing useful suggestions to improve the text.

References

1. Saitou, N., Nei., M.: The neighbor-joining method: a new method for reconstructing phylogenetic trees. Mol. Biol. Evol. 4(4) (1987)
2. Studier, J., Keppler, K.: A note on the Neighbor-Joining algorithm of Saitou and Nei. Mol. Biol. Evol. 5(6) (1988)
3. Arslan, A.N., Bizargity, P.: Phylogeny by top down clustering using a given multiple alignment. In: Proceedings of the 7th IEEE Symposium on Bioinformatics and Biotechnology (BIBE 2007), vol. II (2007)
4. Blockeel, H., De Raedt, L., Ramon, J.: Top-down induction of clustering trees. In: Proceedings of the 15th International Conference on Machine Learning (1998)
5. Morgan, D., Kristensen, D., Mittelman, D., Lichtarge, O.: ET Viewer: an application for predicting and visualizing functional sites in protein structures. Bioinformatics 22(16) (2006)
6. Brown, D.P., Krishnamurthy, N., Sjölander, K.: Automated protein subfamily identification and classification. PLoS Computational Biology 3 (2007)
7. Gascuel, O.: A note on Sattath and Tversky's, Saitou and Nei's, and Studier and Keppler's algorithms for inferring phylogenies from evolutionary distances. Mol. Biol. Evol. 11(6) (1994)
8. Salemi, M., Vandamme, A.: The Phylogenetic Handbook: A Practical Approach to DNA and Protein Phylogeny. Cambridge University Press, Cambridge (2003)
9. Sourdis, J., Krimbas, C.: Accuracy of phylogenetic trees estimated from DNA sequence data. Molecular Biology and Evolution 4 (1987)
10. Quinlan, J.: C4.5: Programs for Machine Learning. Morgan Kaufmann series in Machine Learning. Morgan Kaufmann, San Francisco (1993)
11. Clare, A., Karwath, A., Ougham, H., King, R.D.: Functional bioinformatics for Arabidopsis thaliana. Bioinformatics 22(9) (2006)
12. Jukes, T., Cantor, C.: Evolution of protein molecules. In: Munro, H. (ed.) Mammalian Protein Metabolism, Academic Press, New York (1969)
13. Jones, D., Taylor, W., Thornton, J.: The rapid generation of mutation data matrices from protein sequences. Computer Applications in the Biosciences 8 (1992)
14. Estabrook, G., McMorris, F., Meacham, C.: Comparison of undirected phylogenetic trees based on subtrees of four evolutionary units. Systematic Zoology 34 (1985)
15. Felsenstein, J.: Phylip (phylogeny inference package) version 3.68 (2008), http://evolution.genetics.washington.edu/phylip.html
16. Hall, B.: Simulating DNA Coding Sequence Evolution with EvolveAGene3. Molecular Biology and Evolution 25(4) (2008)
17. Rennert, J., Coffman, J., Mushegian, A., Robertson, A.: The evolution of Runx genes I. A comparative study of sequences from phylogenetically diverse model organisms. BMC Evolutionary Biology 3(4) (2003)
18. Dayhoff, M., Schwartz, R., Orcutt, B.: A model of evolutionary change in proteins. Atlas of protein sequence and structure 5(suppl. 3) (1978)

A Model Free Method to Generate Human Genetics Datasets with Complex Gene-Disease Relationships

Casey S. Greene, Daniel S. Himmelstein, and Jason H. Moore

Dartmouth Medical School, Lebanon, NH 03756, USA
{Casey.S.Greene,Daniel.S.Himmelstein,Jason.H.Moore}@dartmouth.edu
http://www.epistasis.org

Abstract. A goal of human genetics is to discover genetic factors that influence individuals' susceptibility to common diseases. Most common diseases are thought to result from the joint failure of two or more interacting components instead of single component failures. This greatly complicates both the task of selecting informative genetic variations and the task of modeling interactions between them. We and others have previously developed algorithms to detect and model the relationships between these genetic factors and disease. Previously these methods have been evaluated with datasets simulated according to pre-defined genetic models. Here we develop and evaluate a model free evolution strategy to generate datasets which display a complex relationship between individual genotype and disease susceptibility. We show that this model free approach is capable of generating a diverse array of datasets with distinct gene-disease relationships for an arbitrary interaction order and sample size. We specifically generate six-hundred pareto fronts; one for each independent run of our algorithm. In each run the predictiveness of single genetic variation and pairs of genetic variations have been minimized, while the predictiveness of third, fourth, or fifth order combinations is maximized. This method and the resulting datasets will allow the capabilities of novel methods to be tested without pre-specified genetic models. This could improve our ability to evaluate which methods will succeed on human genetics problems where the model is not known in advance. We further make freely available to the community the entire pareto-optimal front of datasets from each run so that novel methods may be rigorously evaluated. These 56,600 datasets are available from http://discovery.dartmouth.edu/model_free_data/.

1 Introduction

Advances in genotyping technologies are changing the way geneticists measure genetic variation. It is now technologically feasible to measure more than one million variations from across the human genome. Here we focus on SNPs, or single nucleotide polymorphisms. A SNP is a single point in a DNA sequence that differs between individuals. A major goal in human genetics is to link the

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2010, LNCS 6023, pp. 74–85, 2010.

state of these SNPs to disease risk. The standard approach to this problem is to measure the genotypes of people with and without a disease of interest across hundreds of thousands to millions of SNPs. Each of these SNPs is then tested individually for an association with the disease of interest. The goal is to discover SNPs that reliably predict disease susceptibility across many samples [1,2]. This approach has had limited success and the discovery of robust single SNP associations has been difficult to attain [3,4,5]. Even in the cases where single SNP associations have validated in independent samples, the SNPs often cannot be combined into effective classifiers of disease risk [6]. These studies, by only examining the association of single SNPs, ignore complex interactions that may be critical to understanding disease susceptibility.

The term for complex gene-gene interactions that influence disease susceptibility is epistasis. It is now recognized that studies which ignore epistasis may neglect informative markers [7,8,9,10]. Furthermore, epistasis is thought to play a critical role in the understanding of disease because of the complexity present in cellular and biological systems [11] and because it has been well characterized for other complex traits [7,12]. Detecting and characterizing epistasis in all but small datasets is difficult. Examining gene-disease relationships in the context of epistasis requires the consideration of the joint effect of SNPs and an exhaustive analysis of all possible interactions requires the enumeration of every potential set of SNPs [13]. When datasets contain many SNPs, combinatorial methods which evaluate each such combination are not feasible [14].

In human genetics we have, therefore, been confronted by a chicken and egg problem. We believe that it is likely that complex interactions occur, but without methods to detect these interactions in large datasets, we lack the ability to find them. Without found and validated interactions that lead to disease we lack the ability to test new methods on actual genetic datasets. Thus far the problem has been approached with datasets simulated according to hypothetical genetic models as in Velez et al. [15] and Greene et al. [10] among many others. Methods are tested for their ability to find a disease model placed in the datasets. This approach is useful but limited by the diversity and representativeness of the genetic models. The work we present here uses an evolution strategy to generate datasets containing complex genetic interactions that lead to disease without imposing a specific genetic model on the datasets.

2 Evolution Strategies

Evolution Strategies are algorithms modeled after natural evolution. The combination of several key evolutionary concepts, such as natural selection, population sizes, and mutation rates, produces algorithms capable of finding sought after members of a solution space [16]. Multiple generations allow evolution strategies to direct their results towards an ideal solution by preserving beneficial mutations. One key difference between mutation driven strategies, like the one implemented in our method, and genetic algorithms is the absence of recombination [17]. Recombination, the computational equivalent of genetic crossover, consists

of creating new individuals in a population by combining the characteristics of multiple members of the previous generation. Since recombination relies on the exchange of discrete blocks of information, crossover is only appropriate when clear building blocks can be defined [18]. Here it is unclear whether a building block would be a set of individuals or a set of SNPs. Therefore, because it is unclear what the proper building blocks would be, we do not use recombination. Evolutionary algorithms that lack recombination have proven themselves equally as powerful in certain instances and remain able to solve complex problems [19]. For the purpose of our study, we are faced with the challenge of evolving a difficult problem to solve, namely, datasets that have a high-order interaction with no main or two-way effects. Previously, others have used evolutionary algorithms to create problems that are hard for a specific heuristic to solve [20,21,22]. One novelty of the present study is our use of evolutionary algorithms to find problems without assuming a specific search algorithm or model.

3 Multi-objective Optimization and Pareto Optimality

Multiobjective problems are those for which practitioners wish to maximize or minimize two or more, often competing, characteristics of solutions. Evolutionary algorithms have been used to solve multi-objective optimization problems for more than twenty years [23,24,25]. These strategies are thought to be well suited to multi-objective problems because the population can carry out a search with solutions that succeed for different objectives [26]. The drawback of this approach is that assigning a single fitness score that encompasses every objective is difficult. Effectively using linear combinations of the objective scores for each objective requires knowledge about the problem and the fitness landscape which is unlikely to be available before a thorough analysis is performed. It is possible, however, to optimize many objectives without *a priori* knowledge by considering non-dominated (i.e. Pareto optimal) solutions as highly fit individuals. A non-dominated solution is one for which there is no solution that is better in all objectives. Here we use an approach focused on Pareto optimal solutions similar to one described by Goldberg [25]. In our approach, we use all Pareto optimal solutions as parents for the next generation, which would be equivalent to using only rank 1 individuals from Goldberg's approach. With this strategy we can explore the Pareto front of solutions which optimize each of our many objectives. We can then provide a number of model free datasets which are optimal with respect to our distinct objectives from a single run of the algorithm.

4 Multifactor Dimensionality Reduction (MDR)

Multifactor Dimensionality Reduction (MDR) is a widely used and a powerful model free method to detect and model gene-gene interactions associated with disease [27,28]. At the core of the MDR approach is an attribute construction algorithm that creates a new attribute by pooling genotypes from multiple SNPs. Constructive induction using the MDR kernel is accomplished in the following

way. Given a threshold T, a multilocus genotype combination is considered high-risk if the ratio of cases (subjects with disease) to controls (healthy subjects) exceeds or equals T, otherwise it is considered low-risk. Genotype combinations considered to be high-risk are labeled G1 while those considered low-risk are labeled G0. This process constructs a new one-dimensional attribute with levels G0 and G1. Here we use MDR to evaluate both high order and low order interactions. Our goal is to minimize the accuracy of this new variable for low-order combinations and maximize the accuracy of this variable for high-order combinations. Because MDR assumes no specific model, our dataset generation method is also model free.

5 A Model Free Dataset Generation Method

The first step in our process of generating datasets is to create random datasets. For each, we initialize a specified number of people (our sample) and provide them with random genotypes at three, four, or five SNPs. This number of SNPs can be arbitrarily assigned and is the order of the predictive interaction we wish to generate. We randomly assign these a case-control status. Current genotyping platforms are targeted towards measuring bi-allelic SNPs, i.e. those with two alleles. These SNPs can exist in one of three states. Here we indicate the states as 0, 1, and 2.

Because we wish to generate datasets with high concept difficulty, one of our goals is to minimize simple genetic effects. To do this, we use MDR to evaluate the best possible low-order predictors. We then, under a Pareto strategy, select those datasets which minimize the low-order predictiveness. In the case of our work here we focus on minimizing the predictiveness of all single SNPs and all two SNP combinations. Our next goal is to maximize the predictiveness of higher order combinations. To do this we evaluate the predictiveness of all the SNPs for every individual using MDR. Because we are maximizing the predictiveness of all the SNPs in the data, we only need to test a single attribute combination (i.e. that of all SNPs) and this task is computationally simple. Using MDR in this way gives us an accuracy, which we then use a Pareto strategy to maximize. Thus in this specific case our evolution strategy exploits Pareto optimization to minimize the single locus and two-locus predictiveness while maximizing the three, four, or five locus predictiveness. Specifically we use all Pareto optimal solutions as parents for the next generation. To generate offspring these parents are duplicated and, at each SNP for each individual, the value can be changed according to the mutation rate.

For SNPs not under selective pressure in humans, these states exhibit what is called Hardy-Weinberg equilibrium (HWE). Hartl and Clark provide an excellent overview of the Hardy Weinberg principle [29]. Because most SNPs are not under selection, deviations from HWE have historically been used as a marker of genotyping error [30]. The implicit assumption is that a SNP which is not in HWE is more likely to be a genotyping error than a SNP under selection. Examinations of early genetic association studies suggested that these concerns

may be well founded [31]. As genotyping methods improve and genotyping error is reduced, it becomes more likely that these SNPs are under selective pressure and less likely that deviations from HWE are due to genotyping error, and thus it becomes less likely that geneticists will filter SNPs which deviate from HWE. Indeed new methods have been developed which use the principles of Hardy-Weinberg equilibrium to detect an association between a genotype and disease [32]. Because the field is currently in transition we provide two sets of datasets, one set where we optimize for non-significant HWE genotype frequencies and one where we do not. In both cases we have initialized the frequencies of the genotype states as under HWE but selection can alter these frequencies.

By using a test of HWE as an additional Pareto criterion we can generate datasets containing SNPs that would not be filtered by currently used quality control measures. In this way we develop datasets where there is a model free but complex relationship between genotype and disease. With a wide array of datasets we can then test the ability of novel methods to detect and characterize complex epistatic relationships without making assumptions about the underlying genetic model. Because the result of each run is a set of Pareto optimal solutions, users can pick solutions with a wide array of difficulties to use while evaluating novel methods. For the set of results where we attempted to preserve Hardy-Weinberg equilibrium we actually minimize disequilibrium. Specifically we minimize the chi-square statistic which measures deviation from HWE. Because this expands our pareto front from three dimensions to four it dramatically increases the size of the pareto front. To insure that each parent has an opportunity to generate a reasonable number of offspring, we limit the number of parents taken to the next generation to one hundred when we are also optimizing for Hardy-Weinberg equilibrium. When there are more than one hundred individuals on the front, we choose the individuals in the "elbow" of the pareto front (i.e. non-extreme individuals). This tie-breaker keeps individuals which are good in regards to more than one dimension at the cost of those which excel in a single dimension.

At the conclusion of each run we have a front of pareto optimal datasets. Because comparing entire pareto fronts is difficult we wish to provide, in addition to the front, a single member of the pareto-optimal group which can represent the run. We have done this by picking the individual dataset with the smallest euclidean distance from the best values obtained for each measure.

6 Experimental Design and Analysis

Our first task was to determine a useful mutation rate. Because this system is driven by mutation, we need to employ an effective mutation rate to evolve good solutions. We examined mutation rate by evaluating rates of 0.05, 0.04, 0.03, 0.02, 0.01, 0.008, 0.006, 0.004 and 0.002 for sample sizes of 500 and 1000 using a fixed number of generations (750) and a fixed population size (1000). We then used the four-way testing accuracy to evaluate how far the evolution had progressed. We used these results to pick an appropriate mutation rate for the number of generations and population size we use here.

Our second task was to evaluate whether our Pareto evolution strategy outperformed a random search. We generated two million random datasets and compared the resulting Pareto front to the pareto fronts generated at the end of evolution in our system. We tested the significance of the differences observed for the accuracies from the fronts from evolved runs and those from the randomly generated runs. We statistically tested these differences with Hotelling's T-test and considered the differences significant when the p-value was less than or equal to 0.05. This means that we would consider results significant only one time out of twenty when there was no significant effect.

Our final task was to generate sets of Pareto optimal datasets each exhibiting three-way interactions, four-way interactions and five-way interactions. In each case we maximized three, four, and five-way MDR accuracies respectively while minimizing one and two way accuracies. We further generated datasets both under pressure to maintain Hardy-Weinberg equilibrium and irrespective of HWE. For each parameter setting (three, four, and five-way interactions with and without HWE) we generated 100 sets of datasets for a total of 600 sets of pareto optimal datasets. In total we have generated more than 50,000 datasets with a complex gene-disease relationship and made these datasets available to researchers as described in section 8.

7 Results

Our parameter sweep of mutation rates showed that, for this problem, a mutation rate of 0.004 led to the greatest success for datasets of five hundred people and 0.002 led to the greatest success for datasets of one thousand people. Large scale parameter sweeping with the sample sizes that we wished to generate was infeasible, but because the optimal mutation rate was related to the sample size we estimated that for the situation where we wish to evolve datasets containing 3000 individuals with a complex gene-disease relationship, a mutation rate of 0.001 would work, although because of the indirectness required to perform the parameter sweep it is not necessarily optimal.

We compared the results from our evolution strategy to a random search. The results are presented in Figure 1 and Table 1. Figure 1 shows the pareto front generated during a single evolved run and the pareto front generated by a random search over the same number of datasets. It is clear that solutions from the pareto front from the evolved run are generally much better than the randomly generated datasets. As Table 1 shows, the evolution strategy consistently outperforms the random search. Furthermore, as Table 1 shows, we were able to consistently generate datasets with a complex gene-disease relationship that lack low order predictors in a model free manner. In each case the differences between the pareto front from two million random datasets and that obtained at the end of our evolution strategy was highly significantly different ($p < 0.001$) indicating that these differences are not likely to be due to chance.

Figure 2 provides some insight into the difficulty of the problem. Minimizing the one-way accuracies and two way accuracies happens relatively quickly and

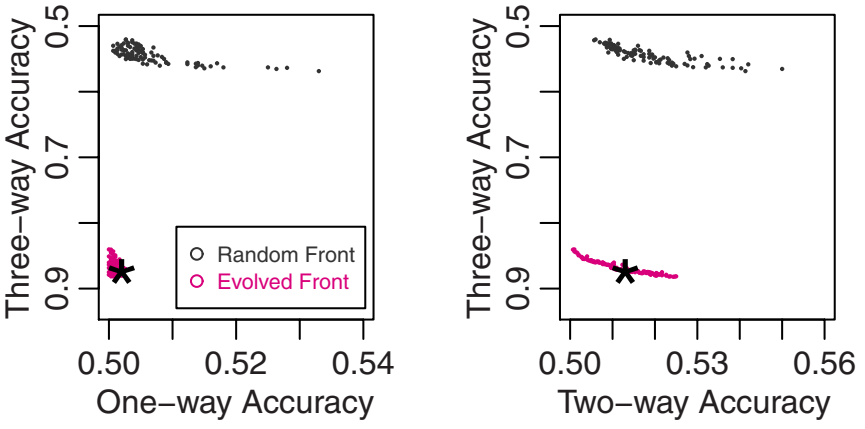


Fig. 1. This figure shows two dimensional projections of the three dimensional pareto front obtained at the end of one run of the evolution strategy and the pareto front obtained by randomly generating two million datasets. The three-way accuracy, which is maximized, is plotted against the one and two-way accuracies, which are minimized. The pareto front from the evolved run is clearly better than the pareto front from random initialization of two million datasets. The star shows the single dataset from the front chosen as the result of the run, which is used to compare across all runs in Table 1.

Table 1. A summary of the accuracies obtained for the evolution strategies and random search. HWE indicates whether or not the datasets were selected based on their conformance to Hardy-Weinberg equilibrium. The one-way and two-way accuracies are always minimized and the n-way accuracy is maximized. The accuracies are presented as mean and (standard deviation).

	Parameters			Results		
n-way	Gen.	Pop.	HWE	One-way (sd)	Two-way (sd)	n-way (sd)
Three-way	2000	1000	No	0.502 (0.001)	0.511 (0.007)	0.886 (0.023)
	2000	1000	Yes	0.504 (0.002)	0.509 (0.003)	0.680 (0.024)
	1	2000000	No	0.506 (0.006)	0.518 (0.009)	0.543 (0.012)
Four-way	2000	1000	No	0.502 (0.001)	0.510 (0.003)	0.897 (0.018)
	2000	1000	Yes	0.507 (0.003)	0.513 (0.003)	0.673 (0.009)
	1	2000000	No	0.507 (0.004)	0.519 (0.006)	0.571 (0.011)
Five-way	2000	1000	No	0.502 (0.001)	0.510 (0.002)	0.895 (0.009)
	2000	1000	Yes	0.511 (0.003)	0.518 (0.003)	0.693 (0.008)
	1	2000000	No	0.507 (0.004)	0.520 (0.005)	0.612 (0.011)

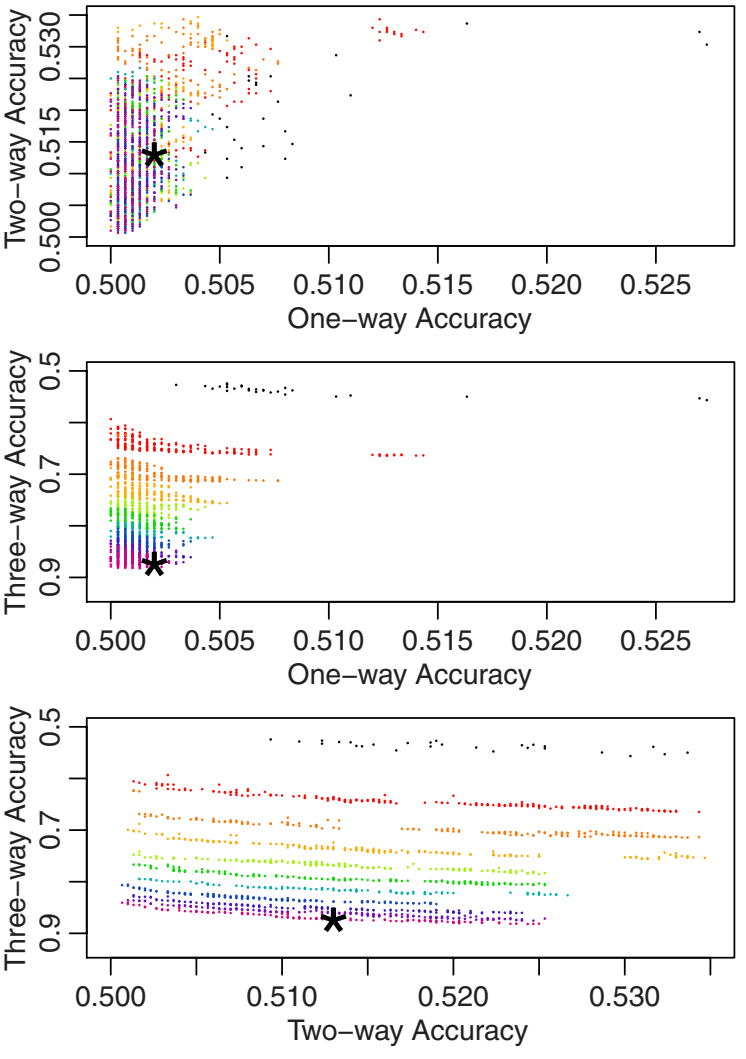


Fig. 2. The pareto front from a single run improves as the run proceeds. The pareto fronts are shown for each pairwise combination of objectives in each box at every 200 generations as shown by color. The star indicates the final solution used to evaluate the run.

within the first few hundred generations. Maximizing the higher order accuracies continues throughout the entire run and progress is still being made and the two-thousandth generation. The star indicates the dataset that was chosen from the front to represent the run. It is this one dataset that is taken from each run according to the euclidean distance strategy discussed in section 5 that is used to calculate the summary statistics in Table 1.

8 Dataset Availability

All pareto optimal datasets generated during these experiments are available from the website http://discovery.dartmouth.edu/model_free_data/. We provide a number of means of obtaining datasets. First, we provide archive files of "best of runs" obtained with and without Hardy-Weinberg equilibrium constraints. These representative datasets are obtained by choosing the dataset with the smallest euclidean distance between its own values and the optimum values obtained by all datasets on the pareto front as described in section 5. In addition to these representative datasets, for each run we provide all datasets that, at the end of the run, make up the complete Pareto front. To assist with the use of these datasets we further provide an information file for each run containing the characteristics of every dataset in the pareto-optimal front. From these files it is possible for investigators to develop suites of datasets that display certain characteristics (e.g. one and two way accuracies less than 52% and four-way accuracies of approximately 70%). Using datasets generated from our provided results, investigators can test novel methods across data exhibiting gene-disease relationships unconstrained by specific genetic models. The SNPs we provide can be combined with other noisy SNPs to represent a three, four, or five-way genetic interaction in a sea of noisy SNPs.

9 Discussion and Conclusions

Evolutionary computing has previously been used to generate epistatic (i.e. interaction based) models of a gene-disease relationship for both two-way [33] and higher order [34] interactions. Here we generate epistatic datasets in a model free manner. We also describe a novel evolution strategy for creating model free datasets and use this strategy to create 56,600 datasets with complex gene-disease relationships which we make publicly available. These datasets provide test beds for novel genetic analysis methods. By providing human genetics datasets with complex interactions that do not assume a model we hope to bypass the chicken and egg problem that has previously confronted the field. Methods tested on datasets generated in this manner may better generalize to true genetic association datasets.

Future work should focus on evaluating potential building blocks, so that crossover can improve the efficiency of the search. Potential building blocks include subsets of the individuals in each dataset, subsets of the SNPs in each dataset, or subsets of both individuals and SNPs in each dataset. It is not intuitive which approach is most useful, so these options should be fully explored. It may be that customized crossover operators are required to obtain useful genetic mixing for this problem.

Future work should also focus on making these datasets and others generated in this manner widely available. By dividing the resulting datasets into standardized testing and training datasets, it would be feasible to compare algorithms in a straightforward and objective manner. This comparison of algorithms would

provide a great deal of information about these methods to human geneticists attempting to understand the basis of common human disease. By providing open and publicly available datasets which do not assume a model but which contain a complex relationship between individual and disease, we hope to improve our understanding of commonly used methods in human genetics. We also hope to provide a framework for objectively testing future methods. Only when we can effectively judge methods across a comprehensive test suite of datasets can we develop methods likely to discover the underlying basis of common human diseases.

Acknowledgement

This work was supported by NIH grants LM009012, AI59694, HD047447, and ES007373. The authors would like to thank Peter Schmitt for his excellent technical assistance.

References

1. Chanock, S.J., Manolio, T., Boehnke, M., Boerwinkle, E., Hunter, D.J., Thomas, G., Hirschhorn, J.N., Abecasis, G., Altshuler, D., Bailey-Wilson, J.E., Brooks, L.D., Cardon, L.R., Daly, M., Donnelly, P., Fraumeni, J.F., Freimer, N.B., Gerhard, D.S., Gunter, C., Guttmacher, A.E., Guyer, M.S., Harris, E.L., Hoh, J., Hoover, R., Kong, C.A., Merikangas, K.R., Morton, C.C., Palmer, L.J., Phimister, E.G., Rice, J.P., Roberts, J., Rotimi, C., Tucker, M.A., Vogan, K.J., Wacholder, S., Wijsman, E.M., Winn, D.M., Collins, F.S.: Replicating genotype-phenotype associations. Nature 447(7145), 655–660 (2007)
2. McCarthy, M.I., Abecasis, G.R., Cardon, L.R., Goldstein, D.B., Little, J., Ioannidis, J.P.A., Hirschhorn, J.N.: Genome-wide association studies for complex traits: consensus, uncertainty and challenges. Nat. Rev. Genet. 9(5), 356–369 (2008)
3. Hirschhorn, J.N., Lohmueller, K., Byrne, E., Hirschhorn, K.: A comprehensive review of genetic association studies. Genet. Med. 4, 45–61 (2002)
4. Shriner, D., Vaughan, L.K., Padilla, M.A., Tiwari, H.K.: Problems with Genome-Wide association studies. Science 316(5833), 1840–1841 (2007)
5. Williams, S.M., Canter, J.A., Crawford, D.C., Moore, J.H., Ritchie, M.D., Haines, J.L.: Problems with Genome-Wide association studies. Science 316(5833), 1841–1842 (2007)
6. Jakobsdottir, J., Gorin, M.B., Conley, Y.P., Ferrell, R.E., Weeks, D.E.: Interpretation of genetic association studies: Markers with replicated highly significant odds ratios may be poor classifiers. PLoS Genetics 5(2), e1000337 (2009)
7. Templeton, A.: Epistasis and complex traits. In: Epistasis and the Evolutionary Process, pp. 41–57 (2000)
8. Moore, J.H.: The ubiquitous nature of epistasis in determining susceptibility to common human diseases. Human Heredity 56, 73–82 (2003)
9. Moore, J.H., Williams, S.M.: Traversing the conceptual divide between biological and statistical epistasis: systems biology and a more modern synthesis. BioEssays 27(6), 637–646 (2005)

10. Greene, C.S., Penrod, N.M., Williams, S.M., Moore, J.H.: Failure to replicate a genetic association may provide important clues about genetic architecture. PLoS ONE 4(6), e5639 (2009)
11. Tyler, A.L., Asselbergs, F.W., Williams, S.M., Moore, J.H.: Shadows of complexity: what biological networks reveal about epistasis and pleiotropy. BioEssays 31(2), 220–227 (2009)
12. Shao, H., Burrage, L.C., Sinasac, D.S., Hill, A.E., Ernest, S.R., O'Brien, W., Courtland, H., Jepsen, K.J., Kirby, A., Kulbokas, E.J., Daly, M.J., Broman, K.W., Lander, E.S., Nadeau, J.H.: Genetic architecture of complex traits: Large phenotypic effects and pervasive epistasis. Proceedings of the National Academy of Sciences 105(50), 19910–19914 (2008)
13. Freitas, A.A.: Understanding the crucial role of attribute interaction in data mining. Artif. Intell. Rev. 16(3), 177–199 (2001)
14. Moore, J.H., Ritchie, M.D.: The challenges of Whole-Genome approaches to common diseases. JAMA 291(13), 1642–1643 (2004)
15. Velez, D.R., White, B.C., Motsinger, A.A., Bush, W.S., Ritchie, M.D., Williams, S.M., Moore, J.H.: A balanced accuracy function for epistasis modeling in imbalanced datasets using multifactor dimensionality reduction. Genetic Epidemiology 31(4), 306–315 (2007)
16. Hoffmeister, F., Bäck, T.: Genetic algorithms and evolution strategies - similarities and differences. In: Schwefel, H.-P., Männer, R. (eds.) PPSN 1990. LNCS, vol. 496, pp. 455–469. Springer, Heidelberg (1991)
17. Bäck, T., Hoffmeister, F., Schwefel, H.: A survey of evolution strategies. In: Proceedings of the Fourth International Conference on Genetic Algorithms, pp. 2–9 (1991)
18. Goldberg, D.E.: The Design of Innovation: Lessons from and for Competent Genetic Algorithms. Kluwer Academic Publishers, Norwell (2002)
19. Greenwood, G., Shin, J.: On the evolutionary search for solutions to the protein folding problem. In: Fogel, G., Corne, D. (eds.) Evolutionary Computation in Bioinformatics, pp. 115–136. Elsevier Science, Amsterdam (2003)
20. van Hemert, J.I.: Property analysis of symmetric travelling salesman problem instances acquired through evolution. In: Raidl, G.R., Gottlieb, J. (eds.) EvoCOP 2005. LNCS, vol. 3448, pp. 122–131. Springer, Heidelberg (2005)
21. van Hemert, J.I.: Evolving combinatorial problem instances that are difficult to solve. Evolutionary Computation 14(4), 433–462 (2006)
22. Julstrom, B.A.: Evolving heuristically difficult instances of combinatorial problems. In: GECCO 2009: Proceedings of the 11th Annual conference on Genetic and evolutionary computation, pp. 279–286. ACM, New York (2009)
23. Schaffer, J.D.: Multiple objective optimization with vector evaluated genetic algorithms. In: Proceedings of the 1st International Conference on Genetic Algorithms, pp. 93–100. L. Erlbaum Associates Inc., Hillsdale (1985)
24. Richardson, J.T., Palmer, M.R., Liepins, G.E., Hilliard, M.: Some guidelines for genetic algorithms with penalty functions. In: Proceedings of the third international conference on Genetic algorithms, pp. 191–197. Morgan Kaufmann Publishers Inc., San Francisco (1989)
25. Goldberg, D.: Genetic algorithms in search, optimization and machine learning. Addison-Wesley Longman Publishing Co., Inc., Boston (1989)
26. Fonseca, C.M., Fleming, P.J.: An overview of evolutionary algorithms in multiobjective optimization. Evolutionary Computation 3, 1–16 (1995)

27. Ritchie, M.D., Hahn, L.W., Roodi, N., Bailey, L.R., Dupont, W.D., Parl, F.F., Moore, J.H.: Multifactor-dimensionality reduction reveals high-order interactions among estrogen-metabolism genes in sporadic breast cancer. Am. J. Hum. Genet. 69(1), 138–147 (2001)
28. Moore, J.H., Gilbert, J.C., Tsai, C.T., Chiang, F.T., Holden, T., Barney, N., White, B.C.: A flexible computational framework for detecting, characterizing, and interpreting statistical patterns of epistasis in genetic studies of human disease susceptibility. Journal of Theoretical Biology 241(2), 252–261 (2006)
29. Hartl, D.L., Clark, A.G.: Principles of Population Genetics, 3rd edn. Sinauer Associates, Sunderland (1997)
30. Hosking, L., Lumsden, S., Lewis, K., Yeo, A., McCarthy, L., Bansal, A., Riley, J., Purvis, I., Xu, C.: Detection of genotyping errors by Hardy-Weinberg equilibrium testing. Eur. J. Hum. Genet. 12(5), 395–399 (2004)
31. Xu, J., Turner, A., Little, J., Bleecker, E., Meyers, D.: Positive results in association studies are associated with departure from Hardy-Weinberg equilibrium: hint for genotyping error? Human Genetics 111(6), 573–574 (2002)
32. Ryckman, K.K., Jiang, L., Li, C., Bartlett, J., Haines, J.L., Williams, S.M.: A prevalence-based association test for case-control studies. Genetic Epidemiology 32(7), 600–605 (2008)
33. Moore, J.H., Hahn, L.W., Ritchie, M.D., Thornton, T.A., White, B.C.: Application of genetic algorithms to the discovery of complex models for simulation studies in human genetics. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 1150–1155. Morgan Kaufmann Publishers Inc., San Francisco (2002)
34. Moore, J.H., Hahn, L.W., Ritchie, M.D., Thornton, T.A., White, B.C.: Routine discovery of complex genetic models using genetic algorithms. Applied Soft Computing 4(1), 79–86 (2004)

Grammatical Evolution of Neural Networks for Discovering Epistasis among Quantitative Trait Loci

Stephen D. Turner, Scott M. Dudek, and Marylyn D. Ritchie

Center for Human Genetics Research, Department of Molecular Physiology & Biophysics, Vanderbilt University, Nashville, TN, USA
{stephen,dudek,Ritchie}@chgr.mc.vanderbilt.edu

Abstract. Growing interest and burgeoning technology for discovering genetic mechanisms that influence disease processes have ushered in a flood of genetic association studies over the last decade, yet little heritability in highly studied complex traits has been explained by genetic variation. Non-additive gene-gene interactions, which are not often explored, are thought to be one source of this "missing" heritability. Here we present our assessment of the performance of grammatical evolution to evolve neural networks (GENN) for discovering gene-gene interactions which contribute to a quantitative heritable trait. We present several modifications to the GENN procedure which result in modest improvements in performance.

Keywords: Neural networks, grammatical evolution, gene-gene interaction, quantitative traits.

1 Introduction

1.1 Genome-Wide Association Studies and Epistasis

Technological advances over the previous decade have allowed researchers in the field of human genetics to progress from coarse genomic coverage with linkage maps and candidate gene association studies, to very high resolution association analyses using single nucleotide polymorphisms (SNPs)[1]. The initial completion and ongoing development of the International HapMap Project [2;3] catalogs common human genetic variation at millions of polymorphic sites in several populations, allowing for more powerful and strategic study design of both targeted and genome wide scans. Several technologies are currently available that allow for rapid, highly accurate genotyping of >1 million common SNPs at low cost per genotype.

Despite the dizzying pace of advances in genotyping technologies that have made high-resolution genetic association studies possible, we have yet to fully explore the wealth of data generated by these studies in part because maturation of our analytical strategies for data of this scale have not kept pace. The most commonly used analytical procedures for analyzing genetic data are tests of association one SNP (single nucleotide polymorphism) at a time. This approach has been somewhat successful in identifying genetic variants associated with complex traits, but these SNPs collectively explain little of the genetic contribution (heritability) expected

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2010, LNCS 6023, pp. 86–97, 2010.

based on family and twin studies [4]. Many investigators have speculated that a portion of the missing genetic component lies in gene-gene and gene-environment interactions [4;5]. Indeed, it is well accepted that common traits are complex, and are influenced by an elaborate interplay of multiple genetic and environmental factors [6-8]. It is thought that gene-gene and gene-environment interactions are ubiquitous given the complex biomolecular interactions that are essential for regulation of gene expression and complex metabolic networks, and are likely to play a role in influencing human traits [9]. Furthermore, while recent perspectives have emphasized the fact that most true single locus genetic associations to complex traits carry a vanishingly small effect size [10;11], others have shown experimentally in model organisms that gene-gene interaction is pervasive and often carries surprisingly large effects [12].

Compelling evidence makes it clear that gene-gene interaction exists in humans and model organisms and influences human traits, yet there is no consensus on how to best investigate gene-gene interaction in genetic association data. One approach is to evaluate multi-marker combinations for potential interactive effects based on biological criteria [13]. This may include, for instance, testing for interactions between genes that share a similar structure or function, or genes in the same pathway or biological process, such as a receptor and its ligand. Using this strategy would bias the statistical analysis in favor of models with a well-established biological foundation in the literature, and novel interactions between genetic variants would be missed. Furthermore, the entire analysis depends upon the quality of the biological information used. Another approach is to select SNPs based on the strength of their independent main effects, evaluating interactions only between SNPs that meet a certain effect size or significance threshold [14]. This strategy assumes that relevant interactions may only occur between variants that independently have some major effect on the phenotype alone.

Another strategy to search for influential gene-gene or gene-environment interaction is to exhaustively evaluate the relationship between the outcome of interest and every possible combination of genetic and environmental exposures. While one may wish to fit standard regression models to every possible 2-, 3-, or n-way combination SNPs, this approach becomes problematic for several reasons. First, when interactions among multiple genetic and/or environmental components are considered, there are many combinations that are present in only a few individuals or perhaps none at all. This is known as the curse of dimensionality [15], and results in unstable estimates of population parameters from large-sample based methods. Furthermore, while the interpretation of the statistical significance of models fit using traditional methods is fairly straightforward, correction must be made for multiple testing. Tests of interactions are large in number, and are not independent, each making multiple testing corrections difficult.

Because of these limitations of using traditional statistical approaches to search for interactions in large datasets, several data-mining and machine learning approaches have been developed to search for combinations of genetic variants that influence a phenotypic outcome. A commonly used procedure is Multifactor Dimensionality Reduction (MDR) [8]. MDR evaluates all possible combinations of genotypes for a given dimensionality and tests for association to a case-control outcome by collapsing every high-dimensional multilocus genotype combination into a single risk variable,

which can then be tested for association with the case-control outcome. More recently the MDR framework was extended to allow for continuous variable exposures and outcomes [16] and for analyzing large datasets using parallel processing [17]. Other similar approaches are the subject of an extensive review in [5].

Exhaustive approaches such as the ones mentioned above are ideally suited for exploring interactions in small datasets comprised of only a few variables, such as in a candidate gene study. However, the computational resources required to exhaustively search for interactions in large genomic scale data often renders these methods impractical. For example, the number of two-way interactions that can be evaluated in a genome-wide association study (GWAS) with 500k SNPs is 1.2×10^{11}. Memory issues aside, it would take many years on a desktop computer to run this analysis. As the number of three-way interactions in such a dataset is over 2×10^{16}, searching exhaustively for higher order interactions would be infeasible even on multiprocessor computing clusters. This limitation is the motivation for developing techniques that still utilize the full dimensionality of the data without exhaustively searching all possible combinations of variables with the goal of discovering a well-fitting model that explains variance in a phenotypic outcome of interest.

1.2 Grammatical Evolution Neural Networks (GENN)

Neural networks (NNs) are a robust and flexible modeling technique that attempt to mimic the basic structure and function of biological neurons to solve complex problems. NNs have been applied to a plethora of research fields, including robotics, speech recognition, optical character recognition, task scheduling, and industrial processing among many others. NNs have also been widely applied to various problems in biological science, including microarray data analysis [18], human linkage analysis [19], genetic association studies [20], medical expert systems [21], survival analysis [22], and protein folding [23]. The traditional approach for applying NNs to a classification problem is to specify a network architecture, select which variables are included as inputs to the network, and fit network weights using a gradient-descent based approach such as backpropagation [24]. Recently, numerous evolutionary search strategies have been applied to NN classification problems to reduce the issues associated with the traditional NN approach [25]. Genetic Programming Neural Networks [26] and Grammatical Evolution Neural Networks (GENN) [27] use genetic programming [28] or grammatical evolution [29] to evolve populations of neural networks for human genetics classification problems. These populations are a heterogeneous mix of architectures, weights, and input variables which undergo mating, crossover, and recombination to ultimately identify an optimum NN solution. Recent work has shown that certain features characteristic of human genetic data may provide advantages to methods that evolve NNs to detect gene-gene interactions by transforming the fitness landscape from a “needle in a haystack” to a broader, smoother surface [30].

In many of its previous applications, however, GENN was used as a classification tool for a case-control outcome [27;31]. While case-control designs in disease gene studies have many advantages, genetic analysis of a quantitative trait that varies continuously over a range of possible values may be more optimal in many scenarios [32]. The outcome of interest in many genetic studies naturally varies on a continuous

scale. While ordinally encoding or discretizing a continuous trait can simplify analysis of the data, it can also be counterproductive as it generally comes with the cost of a substantial decrease in statistical power. Furthermore, the ability to study a continuously varying trait rather than a dichotomy will allow for the examination of genetic factors that influence the overall distribution of a phenotypic measurement in the general population rather than restricting genetic analysis to the extremes of a phenotype. With this as our motivation, we extended our implementation of GENN to allow for continuous outcomes, making it capable of modeling quantitative traits on multiple exposure variables in large datasets in a computationally feasible matter. Here we present results of a simulation study showing that our GENN algorithm has high sensitivity to detect gene-gene interactions contributing to a heritable quantitative trait with low effect sizes against a background of many unassociated variables. We then present several modifications to the GENN procedure and assess their effects on GENN's performance under different genetic models described below.

2 Methods

2.1 Genetic Data Simulation with genomeSIMLA

Simulated data where the true identity and size of the genetic or environmental effect in the population is known is a critical necessity for developing and testing novel methodology. It is also important that these true effects are embedded in a complex dataset containing many other nonfunctional polymorphisms and environmental factors, mimicking a real dataset as closely as possible. We recently developed GenomeSIMLA[33] for simulating realistic genome-wide scale data in population based case-control samples. GenomeSIMLA is capable of simulating missing data, genetic heterogeneity, phenocopy, genotyping error, and realistic patterns of linkage disequilibrium. Here we use an extension of GenomeSIMLA capable of simulating gene-gene interactions in the presence of main effects which influence a quantitative trait at a desired effect size.

The effect size of a genetic variant on a quantitative trait outcome is often expressed in terms of heritability – the proportion of variance in the trait explained by genetic variation. The narrow-sense heritability, here defined as the proportion of variance explained uniquely by a single source of genetic variation (e.g. the main effect of one member of an interacting pair of variants) is given by the semi-partial squared correlation coefficient [34]:

$$sr_i^2 = R^2_{Y.1,2,\ldots i\ldots k} - R^2_{Y.1,2,\ldots (i)\ldots k'} \quad (1)$$

Datasets are simulated using a linear regression equation where the genetic model can take a range of generally additive models, similar to the method implemented in [35] for a discrete outcome. Here we simulated a quantitative trait under additive and dominant interaction models as shown in Figure 1. In the additive model, the mean value of the simulated quantitative trait increases as a function of the number of copies of the less common allele an individual inherits both within and between the two functional genetic variants. In the dominant model, the mean value of the simulated trait is increased if individuals contain at least one or more copies of the minor allele.

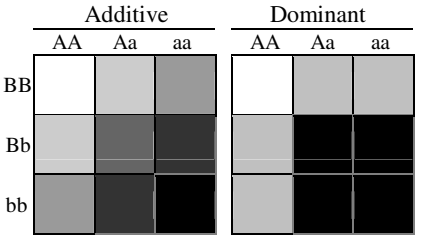


Fig. 1. Genetic model types simulated. The less common allele ("a" or "b") increases the value of a simulated quantitative trait by an amount based on type of genetic model (additive or dominant) and the number of copies of the less common allele an individual possesses. Darker cells indicate a higher mean value for the simulated trait. The genetic model type applies to both main effects and interactive effects within and between the two functional variants.

Individuals are drawn from a homoscedastic normal distribution with the mean being determined by the genotypes at the corresponding functional genetic variants.

We simulated 500 SNPs in 2000 individuals, where only two SNPs were functional and the other 498 SNPs were unassociated "noise" variables. We simulated a gene-gene interaction between these two SNPs that carried a narrow-sense heritability (h^2) of 0.05, meaning that only 5% of the variation in the quantitative trait could be explained by this gene-gene interaction. This low effect size is typical of most findings in human genetic epidemiology [10;11]. We simulated this interaction in the context of very minimal main effects at each locus (h^2=0.01). A scenario such as this where main effects explain little of the overall outcome variance represents a very difficult problem [36] for an evolutionary search procedure to model.

2.2 Grammatical Evolution Neural Networks

GENN has been implemented as previously described [27;37]. Briefly, Grammatical evolution (GE) is a variation of genetic programming (GP), an evolutionary algorithm originally proposed by Koza as a procedure to optimize NN architecture [28]. In GE, randomly initialized binary strings are transcribed into an ordered list of integers which are used to select from production rules in a Backus-Naur form grammar. Our grammar applies GE to construct neural networks, and can simultaneously select important predictor variables and optimize network weights and architecture.

Here we implemented several modifications to the GENN procedure described above. First, because the outcome of interest is now a continuously varying quantitative trait, the fitness function now being used is minimization of mean square error (MSE) rather than maximization of classification accuracy. Second, we have modified the grammar to optionally restrict the choice of arithmetic function at the activation node to addition only. Both versions of the GENN grammar are available online at http://chgr.mc.vanderbilt.edu/ritchielab/projects/ATHENA/grammars.zip. As originally implemented, production rules for the arithmetic function consisted of addition, subtraction, multiplication, and protected division. Restricting the selection of this activation function to addition only would ease subsequent local fitting of network weights by allowing use of a method such as backpropagation, which requires

addition of weighted inputs, after a global search is conducted with GE. However, it was necessary to assess any impact this reduction in variability may have on the ability of GENN to find functional variables and construct a well-fitting model. Finally, we have implemented an alternative tree-based GE crossover strategy as previously described [38]. A potential weakness of GE is the destructive single-point binary crossover (SPBXO) operator [29]. Tree-based crossover (TBXO) instead swaps functionally analogous branches by first translating the grammar into functional neural network trees, identifying branches with identical root nodes, then initializing a crossover back at the genome level which would correspond to the crossover between the whole branches. This renders GE to be much more like genetic programming (GP), while still maintaining some of the key advantages of GE.

For the simulation study presented below, performance was evaluated based on sensitivity to detect both SNPs in the gene-gene interaction that influences the outcome out of the 124,750 possible SNP-pairs in each dataset. As shown in Figure 2, GENN was run for 100, 200, and 400 generations, in runs consisting of population sizes 100, 200 and 400, in each of 20 demes (for a total population size of 2000, 4000, and 8000 respectively), using each of the two grammars on 100 datasets simulated using either an additive or dominant model. Further, we varied the number of generations where tree-based crossover was used. This could range from using single-point binary crossover for every generation (i.e. no TBXO), TBXO for the first half of the total number of generations before switching back to SPBXO, or TBXO for all generations run. This resulted in trials using 144 different combinations of GENN parameters, comprising 144,000 simulated datasets. The mean runtime per dataset was approximately 11 minutes spread across ten 1.8 GHz Opteron PCs. The respective probability of a crossover and mutation were 0.9 and 0.01, typical values for these parameters in many genetic algorithms [39].

3 Results

For the simulation study described above sensitivity was measured as the proportion of datasets out of 100 simulated datasets, where the best performing neural network model contained the two functional SNPs, with no other SNPs in the model, i.e. a perfect match. The best neural network model for each dataset was chosen using the following algorithm. First, 5-fold cross-validation (CV) was implemented, and a model was selected from each CV interval based on reducing mean square error. The fit of this model to unseen data was tested on the CV interval initially left out using the standard coefficient of determination, R^2. This process was repeated for each CV interval. At this point there are 5 models - one best model from each CV interval. The model that consistently appears most often across CV intervals is chosen as the best overall model for the entire dataset [8;40]. In case of a tie (e.g. two different models replicated across two CV intervals), the model with the higher R^2 is chosen as the overall best model. The results are summarized in Figure 2. Separate panels show the total number of generations and the size of the population in each deme. Different line and point types show different sets of activation node functions that were available and the results on different genetic models, respectively. The horizontal axis on each panel shows the proportion of the total number of generations in which TBXO was used.

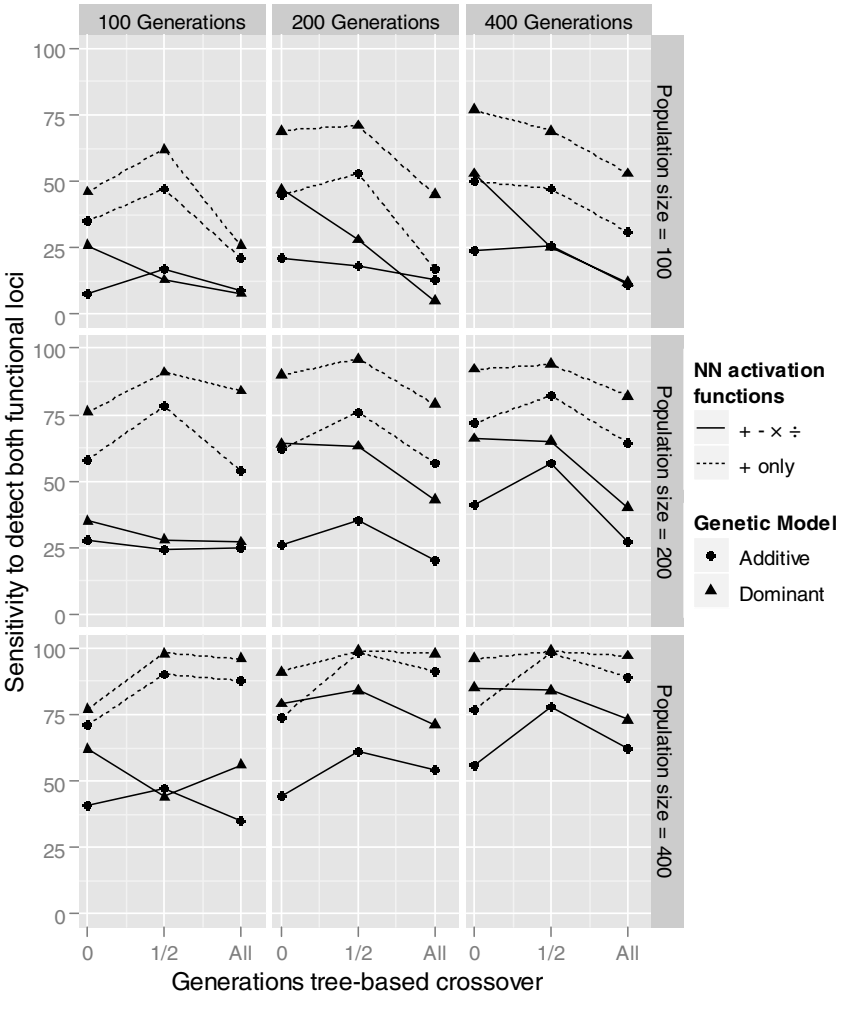


Fig. 2. Sensitivity to detect both functional loci as the best GENN model. Each individual panel shows sensitivity over the proportion of total generations (none, half, all) where tree-based crossover was used instead of binary crossover. Dotted lines show where production rules at the activation node was restricted to addition only, whereas the solid line shows sensitivity when all four functions were available. Individual panels show combinations of the total number of generations GENN was run and the population size per deme.

4 Discussion

Our results show, as expected, that sensitivity to detect and model both functional variables increases as the population size and the total number of generations increases. Rather unexpectedly, however, our results show that restricting production rules in the

grammar to allow only addition as the activation function caused an increase in sensitivity across all combinations of other parameters. It is also important to note that this was observed even when the trait was simulated based on the dominant genetic model rather than additive inheritance. This result was initially surprising because reducing the flexibility of GENN would at first thought seem to impair GENN's ability to discover complex genetic models that contribute to quantitative traits. However, it may be the case that restricting this production rule to a single choice allows for more of each individual network's genetic variability to be applied toward variable selection. When using addition as the only activation function it should be the case that the grammatical evolution of weights and architecture should be able to semantically reproduce any adaptation that the other arithmetic functions syntactically allow for.

These results also show that our implementation of TBXO yields a modest yet notable increase in sensitivity, but only when used exclusively in the early generations of training (see the center point in the line in each panel in Figure 2). This is in contrast to previous work where TBXO showed little improvement when the simulated model was an interaction contributing to a discrete trait in the complete absence of main effects [38]. This result may indicate that GE with TBXO is more efficient at variable selection, while GE with normal crossover allows more variation in building architecture and fitting weights. We postulate that TBXO is preserving "building blocks" which are functionally useful to the resultant neural network models. Our simulations contained a modest interaction effect (h^2=0.05) in the presence of very small main effects (h^2=0.01) at each of the interacting genetic variants. These small main effects may provide the building blocks upon which TBXO can capitalize. Syntactic preservation of NN genomes coding for the inclusion of these variables in NN models while allowing the full variability and broader search capability of SPBXO in the latter generations of evolution appears to be more powerful than using SPBXO or TBXO exclusively. Furthermore, recent work has shown that linkage disequilibrium (correlation between genetic variants) may provide building blocks to an evolutionary algorithm which builds neural networks when the true underlying model is an interactive effect in the complete absence of any main effect at each of the two functional variables [30]. It is expected that the crossover strategy discussed here may be optimal in this situation as well. Because our TBXO procedure mimics the function of genetic programming (GP), further studies should compare this against running GP or any hybrid GP-GE NN training algorithm.

Our results also demonstrate that sensitivity is higher to detect the two functional variables when the underlying genetic model is dominant (triangle points, ▲, in Figure 2), as compared to additive polygenic inheritance (circle points, ●, in Figure 2). This is likely due to the fact that there are more training samples with higher values for the simulated trait when the underlying model is dominant, which provides more "useful" variance in the outcome for GENN to utilize in developing a causative model. This conclusion is further supported by the observation that GENN has extremely low sensitivity to detect the functional genetic variants when the underlying model is recessive (data not shown). In the recessive model, two copies of the less common allele are required to have an increased mean value of the simulated trait. If

the frequency of each minor allele is 0.25 in the population, the probability of any one individual having two copies of the minor allele at both variants is 0.25^4=0.0039. Such individuals with these exposures will be extremely rare in the population, and even with the same effect size, such a model will be very difficult to detect in a population-based genetic association study, which assumes genetic variation contributing to a complex trait is common in the population [1]. This is the case regardless of whether one is using parametric statistics or evolutionary programming to search for gene-gene interactions.

One limitation in the current study is that these experiments make the assumption that loci involved in a gene-gene interactions contributing to a heritable trait will carry with them some small main effect at either variant. This is a reasonable assumption to make, in that there are few, if any, examples of a consistently replicating, experimentally verified gene-gene interaction in the complete absence of main effects contributing to a complex quantitative trait in humans. Perhaps the reason for this, however, is the inadequacy of our methods for finding gene-gene interactions in the absence of main effects rather than the absence of such effects altogether. Biologically, redundancy and compensatory mechanisms at other loci can mitigate the effects of a devastating mutation or polymorphism at another locus, thus rendering its effect undetectable. This is evident in the many gene knockout mouse lines that show no apparent phenotype [41-46]. Statistically, main effect components and interactions between them are mathematically independent effects [47]. Furthermore, theoretical studies have shown that traits can be influenced exclusively through the interaction of two or more genetic variants [48;49]. Finally, one group has shown that main effects at variants involved in an epistatic interaction are highly dependent on the allele frequency in different populations at each locus, which may explain the lack of replication of many gene-gene interaction studies which rely on main effects [50]. Future studies should aim to assess these and other extensions of GENN in their ability to detect and model epistatic interactions contributing to a quantitative trait in the absence of main effects.

Separately, each of the modifications here result in only modest (partial TBXO) to moderate (addition only) improvements to the ability of GENN to perform variable selection. However, combining partial TBXO with restricting the activation node function to addition only can lead to drastic increases in sensitivity (for instance, see the middle panel in Figure 2, 200 generations, population size 200 – when using TBXO for 100 generations with addition only sensitivity is increased to 76%, a nearly threefold increase over the 26% sensitivity when using all four arithmetic functions without TBXO). Future studies should aim to statistically quantify this improvement, as well as assess these and other improvements on models simulated under different genetic architectures.

Acknowledgements

This work was supported by NIH grants NS066638-01, LM010040, HG004608, and HL065962.

References

1. Risch, N., Merikangas, K.: The future of genetic studies of complex human disorders. Science 273(5281), 1516–1517 (1996)
2. International hapmap consortium; The International HapMap Project. Nature 426(6968), 789–796 (2003)
3. International hapmap consortium; A second generation human haplotype map of over 3.1 million SNPs. Nature 449(7164), 851–861 (2007)
4. Maher, B.: Personal genomes: The case of the missing heritability. Nature 456(7218), 18–21 (2008)
5. Cordell, H.J.: Genome-wide association studies: Detecting gene-gene interactions that underlie human diseases. Nat. Rev. Genet. (2009)
6. Wright, S.: The roles of mutation, inbreeding, crossbreeding, and selection in evolution. In: Proc. 6th Intl. Congress of Genetics, vol. 1, p. 356–366 (1932)
7. Moore, J.H., Williams, S.M.: Traversing the conceptual divide between biological and statistical epistasis: systems biology and a more modern synthesis. Bioessays 27(6), 637–646 (2005)
8. Ritchie, M.D., Hahn, L.W., Roodi, N., Bailey, L.R., Dupont, W.D., Parl, F.F., Moore, J.H.: Multifactor-dimensionality reduction reveals high-order interactions among estrogen-metabolism genes in sporadic breast cancer. Am. J. Hum. Genet. 69(1), 138–147 (2001)
9. Moore, J.H.: The ubiquitous nature of epistasis in determining susceptibility to common human diseases. Hum. Hered. 56(1-3), 73–82 (2003)
10. Hirschhorn, J.N.: Genomewide Association Studies – Illuminating Biologic Pathways. N. Engl. J. Med. 360(17), 1699–1701 (2009)
11. Goldstein, D.B.: Common Genetic Variation and Human Traits. N. Engl. J. Med. 360(17), 1696–1698 (2009)
12. Shao, H., Burrage, L.C., Sinasac, D.S., Hill, A.E., Ernest, S.R., O'Brien, W., Courtland, H.W., Jepsen, K.J., Kirby, A., Kulbokas, E.J., Daly, M.J., Broman, K.W., Lander, E.S., Nadeau, J.H.: Genetic architecture of complex traits: large phenotypic effects and pervasive epistasis. Proc. Natl. Acad. Sci. USA 105(50), 19910–19914 (2008)
13. Carlson, C.S., Eberle, M.A., Kruglyak, L., Nickerson, D.A.: Mapping complex disease loci in whole-genome association studies. Nature 429(6990), 446–452 (2004)
14. Kooperberg, C., Leblanc, M.: Increasing the power of identifying gene x gene interactions in genome-wide association studies. Genet. Epidemiol. 32(3), 255–263 (2008)
15. Bellman, R.: Adaptive control processes. Princeton University Press, Princeton (1961)
16. Lou, X.Y., Chen, G.B., Yan, L., Ma, J.Z., Zhu, J., Elston, R.C., Li, M.D.: A generalized combinatorial approach for detecting gene-by-gene and gene-by-environment interactions with application to nicotine dependence. Am. J. Hum. Genet. 80(6), 1125–1137 (2007)
17. Bush, W.S., Dudek, S.M., Ritchie, M.D.: Parallel multifactor dimensionality reduction: a tool for the large-scale analysis of gene-gene interactions. Bioinformatics 22(17), 2173–2174 (2006)
18. Linder, R., Richards, T., Wagner, M.: Microarray data classified by artificial neural networks. Methods Mol. Biol. 382, 345–372 (2007)
19. Lucek, P., Hanke, J., Reich, J., Solla, S.A., Ott, J.: Multi-locus nonparametric linkage analysis of complex trait loci with neural networks. Hum. Hered. 48(5), 275–284 (1998)
20. Ott, J.: Neural networks and disease association studies. American Journal of Medical Genetics (Neuropsychiatric Genetics) 105(60), 61 (2001)

21. Porter, C.R., Crawford, E.D.: Combining artificial neural networks and transrectal ultrasound in the diagnosis of prostate cancer. Oncology (Williston. Park) 17(10), 1395–1399 (2003)
22. Sato, F., Shimada, Y., Selaru, F.M., Shibata, D., Maeda, M., Watanabe, G., Mori, Y., Stass, S.A., Imamura, M., Meltzer, S.J.: Prediction of survival in patients with esophageal carcinoma using artificial neural networks. Cancer 103(8), 1596–1605 (2005)
23. Meiler, J., Baker, D.: Coupled prediction of protein secondary and tertiary structure. Proc. Natl. Acad. Sci. USA 100(21), 12105–12110 (2003)
24. Bishop, C.M.: Neural Networks for Pattern Recognition, pp. 443–482. Oxford University Press, London (1995)
25. Yao, X.: Evolving artificial neural networks. Proceedings of the IEEE 87(9), 1423–1447 (1999)
26. Ritchie, M.D., Coffey, C.S., Moore, J.H.: Genetic programming neural networks: A bioinformatics tool for human genetics. In: Deb, K., et al. (eds.) GECCO 2004. LNCS, vol. 3102, pp. 438–448. Springer, Heidelberg (2004)
27. Motsinger-Reif, A.A., Dudek, S.M., Hahn, L.W., Ritchie, M.D.: Comparison of approaches for machine-learning optimization of neural networks for detecting gene-gene interactions in genetic epidemiology. Genetic Epidemiology 32(4), 325–340 (2008)
28. Koza, J., Rice, J.: Genetic generation of both the weights and architecture for a neural network. IEEE Transactions II (1991)
29. O'Neil, M., Ryan, C.: Grammatical Evolution: Evolutionary Automatic Programming in an Arbitrary Language, 1st edn. Kluwer Academic Publishers, Norwell (2003)
30. Turner, S.D., Ritchie, M.D., Bush, W.S.: Conquering the Needle-in-a-Haystack: How Correlated Input Variables Beneficially Alter the Fitness Landscape for Neural Networks. In: Pizzuti, C., Ritchie, M.D., Giacobini, M. (eds.) EvoBIO 2009. LNCS, vol. 5483, pp. 80–91. Springer, Heidelberg (2009)
31. Ritchie, M.D., Bartlett, J., Bush, W.S., Edwards, T.L., Motsinger, A.A., Torstenson, E.S.: Exploring epistasis in candidate genes for rheumatoid arthritis. BMC Proc. 1(suppl. 1), S70 (2007)
32. Turner, S.D., Crawford, D.C., Ritchie, M.D.: Methods for optimizing statistical analyses in pharmacogenomics research. Expert Reviews in Clinical Pharmacology 2(5), 559–570 (2009)
33. Edwards, T.L., Bush, W.S., Turner, S.D., Dudek, S.M., Torstenson, E.S., Schmidt, M., Martin, E., Ritchie, M.D.: Generating Linkage Disequilibrium Patterns in Data Simulations Using genomeSIMLA. In: Kortuem, G., Finney, J., Lea, R., Sundramoorthy, V. (eds.) EuroSSC 2007. LNCS, vol. 4793, pp. 24–35. Springer, Heidelberg (2007)
34. Cohen, P., Cohen, J., West, S.G., Aiken, L.S.: Applied Multiple Regres-sion/Correlation Analysis for the Behavioral Sciences, 3rd edn. Lawrence Erlbaum, Philadelphia (2002)
35. Schmidt, M.A., Hauser, E.R., Martin, E.R., Schmidt, S.: Extension of the SIMLA Package for Generating Pedigrees with Complex Inheritance Patterns: Environmental Covariates. Gene-Gene and Gene-Environment Interaction, Statistical Applications in Genetics and Molecular Biology 4(1), Article 15, 1–21 (2005)
36. Freitas, A.: Understand the Crucial Role of Attribute Interactions in Data Mining, 16th edn., pp. 177–199 (2001)
37. Motsinger, A.A., Dudek, S.M., Hahn, L.W., Ritchie, M.D.: Comparison of Neural Network Optimization Approaches for Studies of Human Genetics. In: Rothlauf, F., Branke, J., Cagnoni, S., Costa, E., Cotta, C., Drechsler, R., Lutton, E., Machado, P., Moore, J.H., Romero, J., Smith, G.D., Squillero, G., Takagi, H. (eds.) EvoWorkshops 2006. LNCS, vol. 3907, pp. 103–114. Springer, Heidelberg (2006)

38. Motsinger, A.A., Hahn, L.W., Dudek, S.M., Ryckman, K.K., Ritchie, M.D.: Alternative cross-over strategies and selection techniques for grammatical evolution optimized neural networks. In: Proceedings of the 8th annual Genetic and Evolutionary Computation Conference (GECCO), vol. 8, pp. 947–948 (2006)
39. Poli, R., Langdon, W.B., McPhee, N.F.: A Field Guide to Genetic Programming. Lulu Enterprises, United Kingdom (2008)
40. Moore, J., Parker, J., Olsen, N., Aune, T.: Symbolic discriminant analysis of microarray data in autoimmune disease. Genet. Epidemiol. 23, 57–69 (2002)
41. Baba, T., Azuma, S., Kashiwabara, S., Toyoda, Y.: Sperm from mice carrying a targeted mutation of the acrosin gene can penetrate the oocyte zona pellucida and effect fertilization. J. Biol. Chem. 269(50), 31845–31849 (1994)
42. Colucci-Guyon, E., Portier, M.M., Dunia, I., Paulin, D., Pournin, S., Babinet, C.: Mice lacking vimentin develop and reproduce without an obvious phenotype. Cell 79(4), 679–694 (1994)
43. Gorry, P., Lufkin, T., Dierich, A., Rochette-Egly, C., Decimo, D., Dolle, P., Mark, M., Durand, B., Chambon, P.: The cellular retinoic acid binding protein I is dispensable. Proc. Natl. Acad. Sci. USA 91(19), 9032–9036 (1994)
44. Gruda, M.C., van, A.J., Rizzo, C.A., Durham, S.K., Lira, S., Bravo, R.: Expression of FosB during mouse development: normal development of FosB knockout mice. Oncogene 12(10), 2177–2185 (1996)
45. Itohara, S., Mombaerts, P., Lafaille, J., Iacomini, J., Nelson, A., Clarke, A.R., Hooper, M.L., Farr, A., Tonegawa, S.: T cell receptor delta gene mutant mice: independent generation of alpha beta T cells and programmed rearrangements of gamma delta TCR genes. Cell 72(3), 337–348 (1993)
46. Killeen, N., Stuart, S.G., Littman, D.R.: Development and function of T cells in mice with a disrupted CD2 gene. EMBO J. 11(12), 4329–4336 (1992)
47. Maxwell, S.E., Delaney, H.D.: Designing Experiments and Analyzing Data, 2nd edn. Lawrence Erlbaum Associates, Mahwah (2004)
48. Culverhouse, R., Suarez, B.K., Lin, J., Reich, T.: A perspective on epistasis: limits of models displaying no main effect. Am. J. Hum. Genet. 70(2), 461–471 (2002)
49. Moore, J., Hahn, L., Ritchie, M., Thornton, T., White, B.: Application of genetic algorithms to the discovery of complex models for simulation studies in human genetics. In: Langdon, W., Cantu-Paz, E., Mathias, K., Roy, R., Davis, D., Poli, R., Balakrishnan, K., Honavar, V., Rudolph, G., Wegener, J., Bull, L., Potter, M., Schultz, A., Miller, J., Burke, E., Jonoska, N. (eds.), pp. 1150–1155. Morgan Kaufman Publishers, San Francisco (2002)
50. Penrod, N., Greene, C., Moore, J.: Failure to replicate a genetic association may provide important clues about genetic architecture. Presented at the annual meeting of The American Society of Human Genetics, Philadelphia PA, November 14 (2008)

Grammatical Evolution Decision Trees for Detecting Gene-Gene Interactions

Sushamna Deodhar[1] and Alison Motsinger-Reif[2]

[1] Department of Computer Science, North Carolina State University,
Raleigh, NC, USA 27695
mail2sushamna@gmail.com
[2] Bioinformatics Research Center, Department of Statistics, North Carolina State University,
Raleigh, NC, USA 27695
motsinger@stat.ncsu.edu

Abstract. A fundamental goal of human genetics is the discovery of polymorphisms that predict common, complex diseases. It is hypothesized that complex diseases are due to a myriad of factors including environmental exposures and complex genetic risk models, including gene-gene interactions. Such interactive models present an important analytical challenge, requiring that methods perform both variable selection and statistical modeling to generate testable genetic model hypotheses. Decision trees are a highly successful, easily interpretable data-mining method that are typically optimized with a hierarchical model building approach, which limits their potential to identify interactive effects. To overcome this limitation, we utilize evolutionary computation, specifically grammatical evolution, to build decision trees to detect and model gene-gene interactions. Currently, we introduce the Grammatical Evolution Decision Trees (GEDT) method, and demonstrate that GEDT has power to detect interactive models in a range of simulated data, revealing GEDT to be a promising new approach for human genetics.

Keywords: Epistasis, gene-gene interactions, machine learning, decision trees, grammatical evolution, genetic epidemiology.

1 Introduction

In the last decade, the field of human genetics has experienced an unprecedented burst in technological advancement, allowing for exciting opportunities to unravel the genetic etiology of common, complex diseases [1]. As genotyping has become more reliable and cost-effective, genome-wide association studies (GWAS) have become more commonplace tools for gene mapping, where hundreds of thousands or millions of genetic variants are tested for association with disease outcomes [1]. Typically, traditional statistical approaches (i.e. χ^2 tests of association, regression analysis, etc) are used to test for univariate associations, and then those associations are evaluated for replication and validation in independent patient cohorts [2]. This traditional approach has been very successful in identifying strong single gene effects in many common diseases [3], but limitations of this traditional approach have become a focus

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2010, LNCS 6023, pp. 98–109, 2010.

as the variation explained by these singly loci do not come close to the estimates of variance explained by genetics (heritability) known for many diseases [4].

This unexplained variation is hypothesized to be due to more complex etiologies underlying complex diseases [5]. These complex mechanisms include rare variants with high penetrance, locus heterogeneity, and epistasis. In particular, the ubiquitous nature of epistasis, or gene-gene and gene-environment interactions, in the etiology of human diseases presents a difficult analytical challenge [5] . Traditional statistical approaches are limited in their ability to detect interaction models due to their reliance on hierarchical model building strategies, and concerns with high dimensional data (including the curse of dimensionality) [6]. These limitations have been previously described in detail [7]. In response to these limitations, many novel data-mining approached have been developed [8]. The majority of these methods rely on either a combinatorial search approach (such at Multifactor Dimensionality Reduction [9], Combinatorial Partitioning Method [10]) or on a hierarchical model building strategy (such as with Random Forests™[11]). The combinatorial approached are ideal for detecting purely interactive effects (with no single-locus main effects), but are too computationally intensive to detect higher order interactions in large datasets (such as GWAS). The hierarchical approaches are often computationally efficient, but are unable to detect purely epistatic effects [8]. Methods are needed that can detect pure epistatic models in the absence of main effects with realistic computation time. Additionally, as the goal of such data-mining analysis is best described as "hypothesis generation" as opposed to traditional "hypothesis testing" such methodologies need to generate understandable, interpretable models that can be evaluated in follow-up studies [12]. Both replication and functional studies are crucial for the translation of such bioinformatics models.

The use of evolutionary computation (EC) algorithms is one potential solution to these concerns, and has previously shown success in genetic association studies[8]. Several EC algorithms (including genetic algorithms (GA), genetic programming (GP), and grammatical evolution (GE)) have been used to optimize a range of classifiers (neural networks, naïve Bayes classifiers, etc.) to detect complex genotype/phenotype associations. While the success of these methods has been promising, there have been limitations in the interpretability of these models. Specifically, GE optimized neural networks (GENN) has been highly successful in a range of real and simulated data [13], but the resulting neural network models are "black box" models that are difficult to interpret, and are often passed to post hoc "white box" modeling for evaluation [13]. In overcome this problem, we propose using grammatical evolution to build "white box" models that are readily, immediately understandable. Similar approaches have been successful in other fields [14-16], strengthening the hypothesis that this approach would be successful in human genetics. Specifically, we use grammatical evolution to optimize decision trees for analysis of genetic association studies. In the current manuscript, we introduce our Grammatical Evolution Decision Tree (GEDT) approach and software. We then demonstrate the method on a range of simulated gene-gene interaction models, and show that it has high power to detect interactions in a range of effect sizes.

2 Methods

2.1 Grammatical Evolution (GE)

Grammatical Evolution (GE) is a form of evolutionary computing that allows the generation of computer programs using grammars [17]. The modularity of GE makes it flexible and easy to use. GE uses linear genomes and grammars to define populations. In GE, each individual consists of a binary string divided into codons. Mutation takes place on individual bits along this string (or chromosome) and crossover only takes place between codons. An individual phenotype is produced by translating the codons according to the grammar. The resulting individual can then be tested for fitness in the population and evolutionary operators can be applied [18].

GE is inspired by the biological process of generating a protein (phenotype) from the genetic material (DNA genotype) through the processes of *transcription* and *translation*. By using a grammar to define the phenotype, GE separates genotype from phenotype and allows greater genetic diversity within the population than other evolutionary algorithms [17]. [Analogous to the biological process, a variable-length binary string is generated as the "DNA" of the GE process, where a consecutive group of typically 8 bits is considered to be a single codon. This binary string is then *transcribed* into an integer string using the binary code with each codon representing an integer value. These integer values are then *translated* by a mapping function into an appropriate production rule from the grammar definition. An appropriate production rule is selected by the following mapping function:

```
rule = (codon integer value) MOD (Number of alternatives
for the current non-terminal)
```

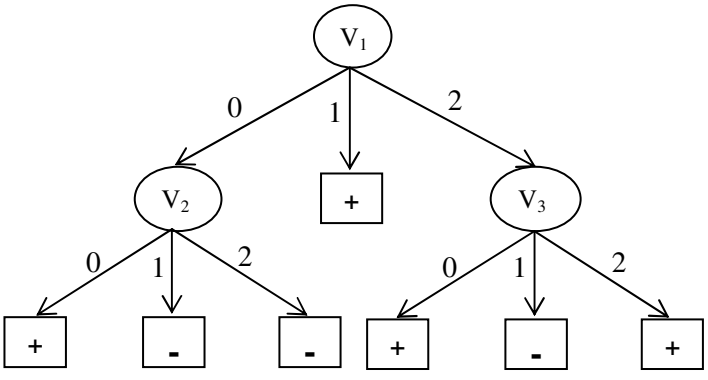
These production rules are then applied to a set of terminals to generate the terminals of the executable program. In the case that after transcribing the entire genome the production rule is not complete, wrapping is used where the genome is wrapped around like a circular list and codons reused. The grammar used for the current application is described below. Details have been previously described [18].

2.2 Decision Trees

A decision tree is a hierarchical decision-making model that consists of internal decision nodes and terminal leaf nodes [19]. Internal decision nodes represent attributes of an individual whereas leaf nodes represent the class the individual belongs to. The root node either corresponds to an initial criterion or an attribute of an individual. Root and other internal nodes are connected via directed edges so that a hierarchical structure is formed. Each outgoing edge from an internal node corresponds to the value of the attribute that the node represents.

Decision trees have been widely used in a variety of applications, such as image classification [20], and pattern recognition [21]. As a learning tool, they offer many advantages that make them ideal for application in human association studies. First, they can model data that has non-linear relationships between variables and can also handle interactions between variables. Second, they can handle large quantities of data in reasonable computation time. Thirdly, they are very easy to understand and

communicate, which is crucial in such a collaborative, interdisciplinary field such as human genetics [12]. From the output model, it is possible to determine what attributes of individuals play an important role in dividing the data in smaller parts and what decisions were made at each internal node. Finally, they are very easy to interpret. Any decision tree can be translated to IF-THEN statements or SWITCH-CASE statements, making it readily human-readable. All these characteristics of decision trees make them a "white-box" model in the sense that the way the output is derived from input variables, going through internal decision nodes, is extremely transparent. This makes them ideal for the "hypothesis generation" motivation of data-mining analysis. An example decision tree is presented in Figure 1.



Parse string: $(V_1\ 0\ (V_2\ 0 + 1 - 2\ -)\ 1 + 2\ (V_3\ 0 + 1 - 2))$

Fig. 1. An example of a decision tree generated by GEDT. It also shows the corresponding parse string for this tree, which is obtained by using the mapping process. Here, decision nodes V_1, V_2 and V_3 correspond to the SNP attributes of the data. Case and control values are represented as classes '+' and '-', respectively.

2.3 Grammatical Evolution Decision Trees

For the current study, GE has been implemented to optimize decision trees (DTs) to detect gene-gene interactions in genetic association studies. The first step of this implementation was the construction of an appropriate grammar for the mapping of DTs that conform to the problem at hand. For genetic association data, the input variables/attributes represent genotypes at specific loci, where a genotype can take one of three genotype values for a bi-allelic SNP (AA, Aa, aa), encoded as 0, 1, and 2. This encoding makes no genetic model assumptions, so this analysis is both statistically and genetically nonparametric. Additionally, while in the current study we evaluate only genetic input variable, any categorical input variables could also be evaluated to detect gene-environment interactions. The output variable (class variable) can take one of two values, either positive (for cases) or negative (control) states.

The GE process begins with the generation of a large number of randomly generated binary strings that are transcribed into integer strings, and then are translated into DTs using the following grammar.

The grammar can be represented by the tuple {N, T, P, S}, where N is the set of non-terminals, T is the set of terminals, P is a set of productions rules that maps the elements of N to T, and S is a start symbol which is a member of N. The following non-terminals were identified:

$N = \{S, pseudoV, v, val_0, val_1, val_2, class\}$

Here, S represents the start codon in the genome. The non-terminal 'pseudoV' is used to represent the tertiary structure of the tree and to keep the recursion going. Non-terminals 'val_0', 'val_1' and 'val_2' represent the possible values a genetic attribute/variable can have and finally, non-terminal 'class' represents the class an individual belongs to. The following terminals were identified:

$T = \{0, 1, 2, +, -, V_{1-n}\}$

where the set $\{V_1, V_2, \ldots, V_n\}$ represents the variable set which correspond to SNPs in the dataset. Terminals '0', '1' and '2' represent possible values these variables can hold, whereas terminals '+' and '-' represent the class values, which correspond the case/control values an individual belongs to.

The following production rules were used to define BNF grammar for GEDT:

```
(1)          S   := <pseudoV>
(2)          pseudoV     :=     <v> <val0> <pseudoV> <val1>
    <pseudoV> <val2> <pseudoV>
             | <class>
(3)          val0   :=     0
(4)          val1   :=     1
(5)          val2   :=     2
(6)          class :=     +
             |   -
(7)          v   := V1
             | Vn
```

where n is equal to the total number of potentially predictive variables/attributes in the dataset. As integer codons are read from the variable-length binary strings, these production rules are used in the mapping function to generate decision trees. The process of generating decision trees can be understood by studying the third production rule of this grammar. The 'pseudoV' non-terminal can be substituted by either a string of seven other non-terminals or 'class', where the latter represents the terminating condition (it also takes care of the cases where all individuals belong to only one class). The first alternative starts with a variable, which is the root of the tree (or sub-tree). It is followed by three values for that variable and each value corresponds to the 'pseudoV' non-terminal. This represents the recursive condition. Now, each of these 'pseudoV' terminals are again substituted in two ways and the process continues until all non-terminals are substituted.

After a tree is built using this grammar, the fitness of the DT model is measured, based on how accurately the model classifies all the individuals in the dataset. In order the make our methodology robust to class imbalance (when there is an unequal number of cases and controls in the dataset), we implemented balanced accuracy as the fitness metric [22]. Using this function, poor performance in either class will lead

to a poor overall fitness and the evolutionary process will be directed towards a solution that performs well in classifying both the sample classes correctly. The fitness function is calculated as the addition of ratio of the correctly classified case samples to the total number of case samples and ratio of correctly classified control samples to the total number of control samples. In other words, the fitness measure is equivalent to the arithmetic average of sensitivity and specificity [22]. In the case of balanced data, balanced accuracy is same as classification accuracy. The formula used is shown below:

```
Balanced accuracy = (sensitivity + specificity) / 2
                  = ½ {[TP / (TP+FN)] + [TN / (TN+FP)]}
```

where TP represents true positives, TN represents true negative, FP represents false positives, and FN represents false negatives.

This fitness metric is then used in a genetic algorithm (GA) to automatically evolve the optimal DT for the data at hand. In this GA, individuals with the highest fitness values are more likely to pass on their on their "genetic material" to the next generation. For GEDT, we use the EC process to evolve every aspect of a decision tree model – including variable selection (which attributes/variables should be included in the model) and the recursive structure of the DT. The steps of GEDT are outlined in Figure 2, and are similar to those previously described for a grammatical evolution optimized neural network strategy [13].

First, GEDT parameters must be initialized in the configuration file, including mutation rate, crossover rate, duplication rate, population size, type of selection, wrapping count, minimum and maximum chromosome size, sensible initialization depth, and number of generations. Second, the data are divided into 10 equal parts for 10-fold cross-validation. 9/10 of the data is used for training, and later the other 1/10 of the data is used to evaluate the predictive ability of the model developed during training [23]. Third, an initial population of random solutions is generated to begin the training process. Sensible initialization is used to guarantee the initial population contains only functioning DTs [17, 18]. In the sensible initialization step an expression tree is created using the grammar described above. The software assigns a minimum depth to each rule that describes the depth required for the rule to be completed. As each tree is built, the algorithm randomly selects only rules that can fit within the remaining depth of the tree. Half of the individual DTs are built to the maximum depth by only selecting recursive rules until a non-recursive rule must be chosen to complete the tree and half are generated to a random depth no greater than the maximum by selecting any rule that can fit in the remaining depth of the tree [16, 24]. The final step in initialization is to convert nodes of the tree into corresponding codons. Fourth, each individual genome is translated into a DT according to the rules of the grammar described above. Each DT is evaluated on the training set and its fitness (balanced accuracy) is recorded. Fifth, the best solutions (those with the highest balanced accuracy) are selected for crossover and reproduction using user-specified proportions. The selection can be performed in a number of ways such as rank, roulette, tournament and uniform. We have used tournament selection as it is efficient for parallel architectures and it is easy to adjust its selection pressure to fine-tune its performance [25]. During duplication, a part of the best solutions is

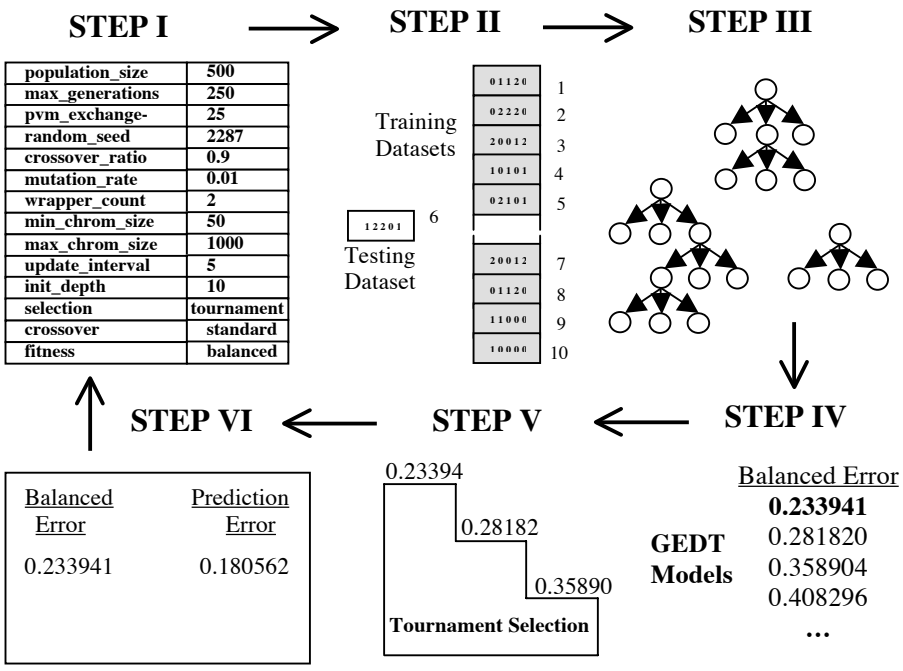


Fig. 2. An overview of the GEDT process that shows the six-step process of initialization, cross-validation, training, fitness evaluation using balanced error, natural selection (tournament) and testing – evaluating prediction error

directly duplicated (i.e. reproduced) into the new generation. During mutation, some other part of the best solutions is selected to apply mutation operator. Mutation is performed on individual bits and involves flipping of the binary value along the genome. During cross-over, some another part is selected for cross-over with other best solutions. It is performed at chromosomal level. We have used one-point cross-over operator. After these operators are applied, the new generation is formed, which is equal in size to the original population. The new generation created by a selection technique specified in the configuration file begins the cycle again. This continues until some criterion, a balanced accuracy of 100% or a user-specified limit on the number of generations, is met. An optimal solution is identified after each generation. At the end of GEDT evolution, the overall best solution is selected as the optimal DT. Sixth, this best GENN model is tested on the 1/10 of the data left out to estimate the prediction error of the model. Steps two through six are performed ten times using a different 9/10 of the data for training and 1/10 of the data for testing.

The goal of GEDT is to find a model that not only fits the data at hand, but will predict on future, unseen data. Cross validation is used in GEDT to prevent overfitting [23]. Overfitting refers to the phenomenon in which a predictive model may well describe the relationship between predictors and outcome in the sample used to develop the model, but may subsequently fail to provide valid predictions in new samples. The use of both classification balanced accuracy and prediction balanced

accuracy error within the GEDT algorithm emphasizes generalizability of the final model. As described above, for each cross-validation interval, a best model is chosen based on highest accuracy of all models evaluated for that interval – resulting in a total of 10 models (one best model for each interval). A classification accuracy and prediction accuracy are recorded for each of the models and a cross-validation consistency can be measured to determine those variables that have a strong signal in the gene-gene interaction model. Cross-validation consistency summarizes the number of times a particular variable(s) are present in the GEDT model out of the best models from the ten cross-validation data splits. The higher the cross-validation consistency, the stronger the support for the model is. The locus/loci with the highest cross-validation consistency are chosen as the final model of the GEDT analysis.

2.4 Data Simulation

For the purposes of the current study, purely epistatic genetic models were generated with varying effect sizes. These models emulate the situation where the phenotype under study cannot be predicted from the independent effects of any single gene, but is the result of combined effects of two or more genes [26]. As discussed above, such epistatic (gene-gene interaction) models are increasingly assumed to play an important role in the underlying etiology of common genetic diseases [5]. We used penetrance functions to represent epistatic genetic models, where penetrance defines the probability of disease given a particular genotype combination by modeling the relationship between genetic variations and disease risk. The genetic variations modeled are single-nucleotide polymorphisms (SNPs). For each individual, a total of 100 SNPs were simulated, where two of the SNPs are associated with the outcome, and 98 are noise SNPs. Case-control data was simulated with 250 cases and 250 controls generated for each dataset, and 100 datasets were generated for each genetic model and effect size combination (described below).

We used two well-described epistasis models exhibiting interaction effects in the absence of main effects. Models that lack main effects challenge the method to find interactions in a complex dataset. The first model, based on the nonlinear XOR function, was initially described by Li and Reich [27], and later by Moore [29]. This model generates an interaction effect in which high risk of disease is dependent on inheriting a heterozygous genotype (Aa) from one locus or a heterozygous genotype (Bb) from a second locus, but not both. The second model, called the ZZ model, was initially described by Frankel and Schork [28] and Moore [29]. In this second model, high risk of disease is dependent on inheriting exactly two high-risk alleles (A and/or B) from two different loci.

For each of the two genetic models (XOR and ZZ), three different effect sizes were used (resulting in a total of six sets of data simulations). Effect size was measured as the proportion of the trait variance that is due to genetics, or broad sense heritability. As calculated according to Culverhouse *et al* [30], heritabilities for the simulated models ranged from 5-100%, capturing a broad range of potential models. Table 1 shows the penetrance functions used for the simulations. Genotypes were generated according to Hardy-Weinberg proportions (in both models, p (the major allele frequency)=q (the minor allele frequency)=0.5). These models exhibit interaction effects in the absence of any main effects. GenomeSim software described by

Table 1. Multilocus penetrance functions used to simulate case-control data exhibiting gene-gene interactions in the absence of main effects. Penetrance is calculated as P(Disease|Genotype).

XOR Model I, Heritability=5.26%

	BB	Bb	bb
AA	0	0.1	0
Aa	0.1	0	0.1
aa	0	0.1	0

ZZ Model I, Heritability=5.13%

	BB	Bb	bb
AA	0	0	0.1
Aa	0	0.05	0
aa	0.1	0	0

XOR Model II, Heritability=33%

	BB	Bb	bb
AA	0	0.5	0
Aa	0.5	0	0.5
aa	0	0.5	0

ZZ Model II, Heritability=28.6%

	BB	Bb	bb
AA	0	0	0.5
Aa	0	0.25	0
aa	0.5	0	0

XOR Model III, Heritability=100%

	BB	Bb	bb
AA	0	1.0	0
Aa	1.0	0	1.0
aa	0	1.0	0

ZZ Model II, Heritability=100%

	BB	Bb	bb
AA	0	0	1.0
Aa	0	0.5	0
aa	1.0	0	0

Dudek *et al* [31] was used to simulate the data. Although the biological likelihood of these models is unknown, they represent the worst-case scenario for the detection method because they have minimal main effects. If a method works well with minimal main effects, seemingly the method will also work well in the presence of main effects.

2.5 Data Analysis

GEDT was used to analyze each of the 600 simulated datasets described above. The configuration parameters used for analysis were as follows: 400 generations, population size of 200 individuals, migration after every 25 generations, cross-over rate of 0.9, mutation rate of 0.1, chromosome size bounded by lower limit of 50 and upper limit of 1000, tournament type of selection, standard i.e. single-point cross-over, balanced accuracy for fitness evaluation and sensible initialization. These parameters are all defined in the configuration file. To prevent stalling in local minima in the fitness landscape, parallelization is used. The parallelization uses the island model where the best individual is passed to each of the other processes after every 25 generations[32]. GEDT is implemented in C++ and Perl, and run on quad-core Core2 Xeon processors (8 processors, each at 3 GHz and with 4GB of memory). Software and user instructions are available from the authors upon request, or linked from the following website: www4.stat.ncsu.edu/~motsinger.

Power for all analyses was estimated under each epistasis model as the number of times the algorithm correctly identified the functional loci out of each set of 100 datasets, without any false positive loci.

3 Results

The power results for each model are as shown in the Table 2. There are a few general trends that are expected for all association methods. GEDT has a minimum of 24% power to detect the lowest effect size models, and as expected, the effect size (heritability) of the models increase, the power also increases. GEDT has over 70% power to detect the higher heritability models. When the number of generations and population size were increased to 600 and 300, respectively, the highest power demonstrated by GEDT increased to 86%. When these two parameters were further increased to 1000 and 500, respectively, power as high as 98% power was achieved.

The GEDT method is also computationally efficient, making it a reasonable approach for larger scale data analysis. In the current evaluations on quad-core Core2 Xeon processors (8 processors, each at 3 GHz with 4GB of memory,) for a total of 400 generations, GEDT analysis of a single dataset consisting of 500 individuals took 5 minutes on average to complete.

Table 2. Results of GEDT analysis, No. of generations = 400

Model	XOR-I	XOR-II	XOR-III	ZZ-I	ZZ-II	ZZ-III
Power (%)	24	33	72	36	38	56

4 Discussion

In the current study, we have presented a detailed description of a new methodology to detect gene-gene interactions in genetic association studies. We propose the use of grammatical evolution to evolve every aspect of decision tree models to detect gene-gene and gene-environment interactions. We demonstrate the potential of the method on a range of simulated two-locus gene-gene interaction models. GEDT has reasonably high power to detect models of moderate to high effect sizes.

While these results are encouraging, the GEDT methodology is still in its infancy, and there are many aspects of its implementation and application that are currently under investigation. First, the parameters implemented in the configuration file are currently undergoing sweeps for a wide range of values to determine optimal settings for data analysis. For example, preliminary data (not shown) shows that as expected, as the number of generations that GEDT is run is increased, the power to detect models also increases. This trend should be further evaluated and potentially more sophisticated stopping rules should be evaluated. Other types of selection, different crossover and mutation rates, etc. should also be evaluated to maximize the power of the method. The scalability of GEDT to data with much larger number of input variables will also be crucial, as genotyping technology allows the evaluation of SNPs orders of magnitude larger than the current simulations.

Additionally, the performance of GEDT must be compared to other methods in the field. GEDT should be compared to other evolutionary computation strategies, such as Grammatical Evolution Neural Networks (GENN) to compare their relative performance in a range of genetic models. The performance of GEDT should also be

compared to other methods used in genetic epidemiology designed to detect epistasis – such as Multifactor Dimensionality Reduction [9], Grammatical Evolution Neural Networks [13], etc. Also, the performance of GEDT should be compared to other decision tree algorithms like ID3 and C4.5 [19]. No method can be considered in a vacuum – and empirical comparisons will play an important role in understanding the niche of the GEDT methodology. These comparisons should consider a variety of genetic models, including heterogeneity, the presence of phenocopy, etc.

The current results indicate the potential of this exciting new approach, but as the end goal of any methodological development is the application to real data, GEDT should be applied to real datasets in human genetics to really evaluate its potential.

Acknowledgements

We have to thank Nicholas Hardison for invaluable input throughout the project.

References

1. Altshuler, D., Daly, M.J., Lander, E.S.: Genetic mapping in human disease. Science 322, 881–888 (2008)
2. Moore, J.H., Ritchie, M.D.: STUDENTJAMA. The challenges of whole-genome approaches to common diseases. JAMA 291, 1642–1643 (2004)
3. Hirschhorn, J.N.: Genomewide association studies–illuminating biologic pathways. N. Engl. J. Med. 360, 1699–1701 (2009)
4. Goldstein, D.B.: Common genetic variation and human traits. N. Engl. J. Med. 360, 1696–1698 (2009)
5. Moore, J.H.: The ubiquitous nature of epistasis in determining susceptibility to common human diseases. Hum. Hered. 56, 73–82 (2003)
6. Bellman, R.: Adaptive Control Processes. Princeton University Press, Princeton (1961)
7. Moore, J.H., Williams, S.M.: New strategies for identifying gene-gene interactions in hypertension. Ann. Med. 34, 88–95 (2002)
8. Motsinger, A.A., Ritchie, M.D., Reif, D.M.: Novel methods for detecting epistasis in pharmacogenomics studies. Pharmacogenomics 8, 1229–1241 (2007)
9. Ritchie, M.D., Hahn, L.W., Roodi, N., Bailey, L.R., Dupont, W.D., Parl, F.F., Moore, J.H.: Multifactor-dimensionality reduction reveals high-order interactions among estrogenmetabolism genes in sporadic breast cancer. Am. J. Hum. Genet. 69, 138–147 (2001)
10. Nelson, M.R., Kardia, S.L., Ferrell, R.E., Sing, C.F.: A combinatorial partitioning method to identify multilocus genotypic partitions that predict quantitative trait variation. Genome Res. 11, 458–470 (2001)
11. Brieman, L.: Random Forests. Machine Learning 45, 27 (2001)
12. Aguilar-Ruiz, J.S., Moore, J.H., Ritchie, M.D.: Filling the gap between biology and computer science. BioData Min. 1, 1 (2008)
13. Motsinger-Reif, A.A., Dudek, S.M., Hahn, L.W., Ritchie, M.D.: Comparison of approaches for machine-learning optimization of neural networks for detecting gene-gene interactions in genetic epidemiology. Genet. Epidemiol. (2008)
14. Yao, X.: Evolutionary artificial neural networks. Int. J. Neural Syst. 4, 203–222 (1993)
15. Motsinger-Reif, A.A., Ritchie, M.D.: Neural networks for genetic epidemiology: past, present, and future. BioData Min. 1, 3 (2008)

16. Koza, J., Rice, J.P.: Genetic generation of both the weights and architecture for a neural network. IEEE Transactions 2 (1991)
17. O'Neill, M., Ryan, C.: Grammatical Evolution. Kluwer Academic Publishers, Boston (2001)
18. O'Neill, M., Ryan, C.: Grammatical Evolution: Evolutionary automatic programming in an arbitrary language. Kluwer Academic Publishers, Boston (2003)
19. Alpaydin, E.: Introduction to Machine Learning. MIT Press, Cambridge (2004)
20. Shepherd, B.A.: An appraisal of a decision-tree approach to image classification. In: Proceedings of the Eighth International Joint Conference on Artificial Intelligence, p. 2 (1983)
21. Devroye, L., Györfi, L., Lugosi, G.: A Probabilistic Theory of Pattern Recognition. Springer, New York (1996)
22. Velez, D.R., White, B.C., Motsinger, A.A., Bush, W.S., Ritchie, M.D., Williams, S.M., Moore, J.H.: A balanced accuracy function for epistasis modeling in imbalanced datasets using multifactor dimensionality reduction. Genet. Epidemiol. 31, 306–315 (2007)
23. Hastie, T.J., Tibshirani, R.J., Friedman, J.H.: The elements of statistical learning. Springer, Basel (2001)
24. Koza, J.: Genetic Programming: on the programming of computers by means of natural selection. MIT Press, Cambridge (1992)
25. Miller, B.L., Goldberg, D.E.: Genetic Algorithms, Tournament Selection and the Effects of Noise. Complex Systems 9, 193–212 (1995)
26. Cordell, H.J.: Epistasis: what it means, what it doesn't mean, and statistical methods to detect it in humans. Hum. Mol. Genet. 11, 2463–2468 (2002)
27. Li, W., Reich, J.: A complete enumeration and classification of two-locus disease models. Hum. Hered. 50, 334–349 (2000)
28. Frankel, W.N., Schork, N.J.: Who's afraid of epistasis? Nat. Genet. 14, 371–373 (1996)
29. Moore, J.H., Hahn, L.W., Ritchie, M.D., Thornton, T.A., White, B.C.: Application of genetic algorithms to the discovery of complex genetic models for simulations studies in human genetics. In: Langdon, W.B., Cantu-Paz, E., Mathias, K., Roy, R., Davis, D., Poli, R., Balakrishnan, K., Honavar, V., Rudolph, G., Wegener, J., Bull, L., Potter, M.A., Schultz, A.C., Miller, J.F., Burke, E., Jonoska, N. (eds.) Genetic and Evolutionary Algorithm Conference, pp. 1150–1155. Morgan Kaufman Publishers, San Francisco (2002)
30. Culverhouse, R., Suarez, B.K., Lin, J., Reich, T.: A perspective on epistasis: limits of models displaying no main effect. Am. J. Hum. Genet. 70, 461–471 (2002)
31. Dudek, S.M., Motsinger, A.A., Velez, D.R., Williams, S.M., Ritchie, M.D.: Data simulation software for whole-genome association and other studies in human genetics. In: Pac. Symp. Biocomput., pp. 499–510 (2006)
32. Cantu-Paz, E.: Evolving Neural Networks for the classification of galaxies. Morgan Kaufman Publishers, San Franscisco (2002)

Identification of Individualized Feature Combinations for Survival Prediction in Breast Cancer: A Comparison of Machine Learning Techniques

Leonardo Vanneschi[1], Antonella Farinaccio[1], Mario Giacobini[2,3], Giancarlo Mauri[1], Marco Antoniotti[1], and Paolo Provero[2,4]

[1] Department of Informatics, Systems and Communication (D.I.S.Co.)
University of Milano-Bicocca, Milan, Italy
[2] Computational Biology Unit, Molecular Biotechnology Center
University of Torino, Italy
[3] Department of Animal Production, Epidemiology and Ecology
Faculty of Veterinary Medicine, University of Torino, Italy
[4] Department of Genetics, Biology and Biochemistry
University of Torino, Italy

Abstract. The ability to accurately classify cancer patients into risk classes, i.e. to predict the outcome of the pathology on an individual basis, is a key ingredient in making therapeutic decisions. In recent years gene expression data have been successfully used to complement the clinical and histological criteria traditionally used in such prediction. Many "gene expression signatures" have been developed, i.e. sets of genes whose expression values in a tumor can be used to predict the outcome of the pathology. Here we investigate the use of several machine learning techniques to classify breast cancer patients using one of such signatures, the well established *70-gene signature*. We show that Genetic Programming performs significantly better than Support Vector Machines, Multilayered Perceptron and Random Forest in classifying patients from the NKI breast cancer dataset, and slightly better than the scoring-based method originally proposed by the authors of the seventy-gene signature. Furthermore, Genetic Programming is able to perform an automatic feature selection. Since the performance of Genetic Programming is likely to be improvable compared to the out-of-the-box approach used here, and given the biological insight potentially provided by the Genetic Programming solutions, we conclude that Genetic Programming methods are worth further investigation as a tool for cancer patient classification based on gene expression data.

1 Introduction

Current cancer therapies have serious side effects: ideally type and dosage of the therapy should be matched to each individual patient based on his/her risk of relapse. Therefore the classification of cancer patients into risk classes is a very active field of research, with direct clinical applications. Until recently patient classification was based on a series of clinical and histological parameters. The advent of high-throughput techniques to measure gene expression led in the last decade to a large body of research on gene

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2010, LNCS 6023, pp. 110–121, 2010.

expression in cancer, and in particular on the possibility of using gene expression data to improve patient classification. A gene signature is a set of genes whose levels of expression can be used to predict a biological state (see [25]): in the case of cancer, gene signatures have been developed both to distinguish cancerous from non-cancerous conditions and to classify cancer patients based on the aggressiveness of the tumor, as measured for example by the probability of relapsing within a given time.

While many studies have been devoted to the identification of gene signatures in various types of cancer, the question of the algorithms to be used to maximize the predictive power of a gene signature has received less attention. To investigate this issue systematically, we considered one of the best established gene signatures, the 70-gene signature for breast cancer [33], and we compared the performance of four different machine learning algorithms in using this signature to predict the survival of a cohort of breast cancer patients. The 70-gene signature is a set of microarray features selected in [33] based on correlation with survival, on which the molecular prognostic test for breast cancer "MammaPrint"^{TM}is based. While several machine learning algorithms have been used to classify cancer samples based on gene expression data (see the discussion in Section 2 and also references [8,10,11,20,26,37]), in this work we performed a systematic comparison of the performance of four machine learning algorithms using the same features to predict the same classes. In our comparison, feature selection is thus *not* explicitly performed as a pre-processing phase before executing the machine learning algorithms[1]. We considered GP, Support Vector Machines, Multilayered Perceptron and Random Forests, and we applied them to the problem of using the 70-gene signature to predict the survival of the breast cancer patients included in the NKI dataset [32]. This is considered one of the gold-standard datasets in the field, and the predictive power of the 70-gene signature on these patients was already shown in [32]. In this preliminary study we tried to use all the methods in an "out-of-the-box" version so as to obtain a preliminary evaluation as unbiased as possible of the performance of the methods.

This paper is structured as follows: Section 2 contains a review of some previous and related contributions; in Section 3 we describe the machine learning methods that we have used; Section 4 contains a description of the employed dataset; in Section 5 we report and discuss the experimental results; finally, Section 6 concludes this work.

2 Previous and Related Work

Many different machine learning methods [22] have already been applied for microarray data analysis, like k-nearest neighbors [23], hierarchical clustering [1], self-organizing maps [18], Support Vector Machines [14,15] or Bayesian networks [12]. Furthermore, in the last few years Evolutionary Algorithms (EA) [16] have been used for solving both problems of feature selection and classification in gene expression data analysis. Genetic Algorithms (GAs) [13] have been employed for building selectors where each allele of the representation corresponds to one gene and its state denotes

[1] As we will discuss later, Genetic Programming (GP) is the only method, among the ones studied in this paper, that is able to perform an automatic feature selection during the learning phase.

whether the gene is selected or not [21]. GP on the other hand has been shown to work well for recognition of structures in large data sets [24]. GP was applied to microarray data to generate programs that reliably predict the health/malignancy states of tissue, or classify different types of tissues. An intrinsic advantage of GP is that it automatically selects a small number of feature genes during the evolution [29]. The evolution of classifiers from the initial population seamlessly integrates the process of gene selection and classifier construction. In fact, in [37] GP was used on cancer expression profiling data to select potentially informative feature genes, build molecular classifiers by mathematical integration of these genes and classify tumour samples. Furthermore, GP has been shown a promising approach for discovering comprehensible rule-based classifiers from medical data [5] as well as gene expression profiling data [17]. The results presented in those contributions are encouraging and pave the way to a further investigation of GP for this kind of datasets, which is the goal of this paper.

3 Computational Methods

3.1 Genetic Programming

Genetic Programming (GP) [19,28,34] is an evolutionary approach which extends Genetic Algorithms (GAs) [16,13] to the space of programs. Like any other evolutionary algorithm, GP works by defining a goal in the form of a quality criterion (or *fitness*) and then using this criterion to *evolve* a set (also called population) of solution candidates (also called individuals) by mimic the basic principles of Darwin's theory of evolution [9]. The most common version of GP, and also the one used here, considers individuals as *abstract syntax tree* structures[2] that can be built recursively from a set of function symbols $\mathcal{F} = \{f_1, f_2, \ldots, f_n\}$ (used to label internal tree nodes) and a set of terminal symbols $\mathcal{T} = \{t_1, t_2, \ldots, t_m\}$ (used to label tree leaves). GP breeds these solutions to solve problems by executing an iterative process involving the probabilistic selection of the fittest solutions and their variation by means of a set of genetic operators, usually crossover and mutation.

We used a tree-based GP configuration inspired by boolean problems introduced in [19]: each feature in the dataset was represented as a boolean value and thus our set of terminals $\mathcal{T}$ was composed by 70 boolean variables (i.e. one for each feature of our dataset). Potential solutions (GP individuals) were built using the set of boolean functions $\mathcal{F} = \{AND, OR, NOT\}$. The fitness function is the number of incorrectly classified instances, which turns the problem into a minimization one (lower values are better)[3].

Finally no explicit feature selection strategy was employed, since we want to point out GP's ability to automatically perform an implicit feature selection. The mechanism allowing GP to perform feature selection, already pointed out for instance in [2,3,4,29], is simple: GP searches over the space of all boolean expressions of 70 variables. This

[2] Traditionally represented in Lisp notation.

[3] We are aware that, in case of minimization problems, the term "fitness" might be inappropriate, given that a fitness is usually a measure that has to be maximized. Nevertheless, we chose to use this term for simplicity.

search space includes the expressions that use *all* the 70 variables, but also the ones that use a *smaller* number of variables. In principle there is no reason why an expression using a smaller number of variables could not have a better fitness value than an expression using all the 70 variables. If expressions using smaller number of variables have a better fitness, they survive into the population, given that fitness is the only principle used by GP for selecting genes.

The parameters used in our GP experiments are reported in Table 1, together with the parameters used by the other machine learning methods we studied. There is no particular justification for the choice of those parameter values, if not the fact that they are standard for the computational tool we used, i.e. GPLab: a public domain GP system implemented in MatLab (for the GPLab software and documentation, see [31]).

3.2 Support Vector Machines

Support Vector Machines (SVM) are a set of related supervised learning methods used for classification and regression. They were originally introduced in [35]. Their aim is to devise a computationally efficient way of identifying separating hyperplanes in a high dimensional feature space. In particular, the method seeks separating hyperplanes maximizing the margin between sets of data. This should ensure a good generalization ability of the method, under the hypothesis of consistent target function between training and testing data. To calculate the margin between data belonging to two different classes, two parallel hyperplanes are constructed, one on each side of the separating hyperplane, which are "pushed up against" the two data sets. Intuitively, a good separation is achieved by the hyperplane that has the largest distance to the neighboring data points of both classes, since in general the larger the margin the lower the generalization error of the classifier. The parameters of the maximum-margin hyperplane are derived by solving large quadratic programming (QP) optimization problems. There exist several specialized algorithms for quickly solving these problems that arise from SVMs, mostly reliant on heuristics for breaking the problem down into smaller, more manageable chunks. In this work we used the implementation of John Platt's [27] sequential minimal optimization (SMO) algorithm for training the support vector classifier. SMO works by breaking the large QP problem into a series of smaller 2-dimensional subproblems that may be solved analytically, eliminating the need for numerical optimization algorithms such as conjugate gradient methods. The implementation we used is the one contained in the Weka public domain software [36]. This implementation globally replaces all missing values and transforms nominal attributes into binary ones. It also normalizes all attributes by default (in that case the coefficients in the output are based on the normalized data, not the original data and this is important for interpreting the classifier).

The main parameter values used in this work are reported in Table 1. All these parameter values correspond to the standard values offered by the Weka software [36] and they are defined for instance in [27]. We are aware that in several application domains, SVM have been shown to outperform competing techniques by using nonlinear kernels, which implicitly map the instances to very high (even infinite) dimensional spaces. Unfortunately, the implementation of SVM in Weka sets as default the polynomial kernel with degree 1. This means that we evaluate the accuracy of linear SVM (SVM in the

original feature space). Possible improvements, using more sophisticated kernel functions, are included in our future work.

3.3 Multilayered Perceptron

Multilayered Perceptron is a feed-forward artificial neural network model [30]. It is a modification of the standard linear perceptron in that it uses three or more layers of neurons (nodes) with nonlinear activation functions, and is more powerful than simple perceptron in that it can distinguish data that are not linearly separable, or separable by a hyperplane. It consists of an input and an output layer with one or more hidden layers of nonlinearly-activating nodes. Each node in one layer connects with a certain weight to every other node in the following layer. The implementation we have adopted is the one included in the Weka software distribution [36]. We used the Back-propagation learning algorithm [30] and the values used for all the parameters are reported in Table 1. As for the previously discussed machine learning methods, also in the case of Multilayered Perceptron it is important to point out that we used a parameter setting as standard as possible, without doing any fine parameter tuning for this particular application. Our goal is, in fact, to compare different computational methods under standard conditions and not to solve in the best possible way the application itself. In particular, all the values reported in Table 1 correspond to the default ones adopted by the Weka software.

3.4 Random Forests

Random Forests denotes an improved Classification and Regression Trees method [7]. It works by creating a large number of classification trees or regression trees. Every tree is built using a deterministic algorithm and the trees are different owing to two factors. First, at each node, a best split is chosen from a random subset of the predictors rather than from all of them. Secondly, every tree is built using a bootstrap sample of the observations. The out-of-bag data, approximately one-third of the observations, are then used to estimate the prediction accuracy. Unlike other tree algorithms, no pruning or trimming of the fully grown tree is involved. In this work we use the Breiman model presented in [6] and implemented in the Weka software [36]. As it can be seen from Table 1, this method, compared to the other ones, has the advantage of a smaller amount of parameter setting required. In order to allow a fair comparison with GP, we have considered random forests composed by 2500 trees (given that the GP population is composed by 500 trees and it runs for 5 generations, 2500 trees are globally inspected by GP too) and such that each node corresponds to exactly one feature (as it is for GP). All the other parameters were set to the standard values offered by the Weka software.

4 Validation Dataset

We used the NKI breast cancer dataset [32], providing gene expression and survival data for 295 consecutive breast carcinoma patients. We considered only the expression data for the genes included in the “seventy gene” signature [33].

Table 1. Parameters used in the experiments

GP Parameters	
population size	500 individuals
population initialization	ramped half and half [19]
selection method	tournament (tournament size = 10)
crossover rate	0.9
mutation rate	0.1
maximum number of generations	5
algorithm	generational tree based GP with no elitism
SVM Parameters	
complexity parameter	0.1
size of the kernel cache	10^7
epsilon value for the round-off error	10^{-12}
exponent for the polynomial kernel	1.0
tolerance parameter	0.001
Multilayered Perceptron Parameters	
learning algorithm	Back-propagation
learning rate	0.03
activation function for all the neurons in the net	sigmoid
momentum	0.2 progressively decreasing until 0.0001
hidden layers	(number of attributes + number of classes) / 2
number of epochs of training	500
Random Forest Parameters	
number of trees	2500
number of attributes per node	1

Both survival and gene expression data were transformed into binary form. For the survival data, we defined the outcome as the survival status of the patient at time $t_{end} = 10.3$ years. By choosing this particular endpoint we balanced the number of dead and alive patients: out of 148 patients for which the status at t_{end} is known, 74 were dead and 74 were alive. Binary expression data were obtained by replacing all positive logarithmic fold changes in the original dataset with 1 and all negative and missing ones with 0.

Our dataset is a matrix $H = [H_{(i,j)}]$ of binary values composed by 148 rows (instances) and 71 columns (features), where each line i represents the gene signature of a patient whose binary target ($0 =$ survived after t_{end} years, $1 =$ dead for breast cancer before t_{end} years) has been placed at position $H_{(i,71)}$. In this way, the last column of matrix H represents all the known target values. Our task is now to generate a mapping F such that $F(H_{(i,1)}, H_{(i,2)}, ..., H_{(i,70)}) = H_{(i,71)}$ for each line i in the dataset. Of course, we also want F to have a good generalization ability, i.e. to be able to assess the target value for *new* patients, that have not been used in the training phase. For this reason, we use a set of machine learning techniques, as discussed in Section 3.

Each computational method was run 50 independent times. For each run a random splitting of the dataset was performed before model construction, by partitioning it into a training and a test set: 70% of the patients are randomly selected with uniform probability and inserted into the training set, while the remaining 30% form the test set.

5 Experimental Results

Table 2 summarizes the results returned by each machine learning method on the 50 runs. The first line indicates the different methods, the second line shows the best (i.e. lowest) value of the incorrectly classified instances obtained on the test set over the 50 runs, and the third line reports the mean performances of each group of 50 runs on their test sets, together with the corresponding standard error of mean (SEM from now on).

Table 2. Experimental comparison between the number of incorrectly classified instances found on the test sets by the different machine learning methods. Each method was independently run 50 times using each time a different training/test partition of the validation dataset (see text for details). The first line indicates the method: Genetic Programming (GP), Support Vector Machine (SVM), Multilayer Perceptrons (MP), and Random Forest (RF). The second line shows the best value of the incorrectly classified instances obtained on the test set over the 50 runs, and the third line reports the average performances of each group of 50 runs on their test sets (standard error of mean is shown in parentheses).

	GP	**SVM**	**MP**	**RF**
best	10	13	10	12
average (SEM)	16.40 (0.30)	18.08 (0.37)	18.32 (0.39)	17.60 (0.35)

As Table 2 clearly shows, the best solutions were found by GP and Multilayered Perceptron and the best average result was found by GP. Moreover, statistical analysis indicates that GP consistently outperforms the other three methods. In fact, as it can be seen in Table 3, the difference between the various average results is statistically significant (P-value 0.0008 for ANOVA test on the 4 samples of solutions found by each method). Finally, pairwise 2-tailed Student t-tests comparing GP with each other method demonstrate its better performance. These statistical tests were performed since there was no evidence of deviation from normality or unequal variances.

Table 3. Statistical significance of the different performances of the methods. First line shows ANOVA test on the 4 samples of solutions found by each method, while second line depicts pairwise 2-tailed Student t-tests comparing GP with each other method.

ANOVA $P = 0.0008$		
Student t-test GP vs. SVM $P = 0.0001$	Student t-test GP vs. MP $P = 0.0009$	Student t-test GP vs. RF $P = 0.0052$

When using gene signatures to predict the survival of a cohort of breast cancer patients, one of the main goal in clinical applications is to minimize the number of false negative predictions. Table 4 summarizes the false negative predictions returned by each machine learning method on the 50 runs. The first line indicates the different methods, while the second and the third lines show the best (i.e. lowest) and mean performances (together with the corresponding SEM)values of incorrectly classified instances.

Table 4. Experimental comparison between the number of false negatives found on the test sets by the different machine learning methods. Each method was independently run 50 times using each time a different training/test partition of the validation dataset (see text for details). The first line indicates the method: Genetic Programming (GP), Support Vector Machine (SVM), Multilayer Perceptrons (MP), and Random Forest (RF). The second line shows the best value of the incorrectly classified instances obtained on the test set over the 50 runs, and the third line reports the average performances of each group of 50 runs on their test sets (standard error of mean is shown in parentheses).

	GP	SVM	MP	RF
best	2	6	5	6
average (SEM)	9.82 (0.44)	13.56 (0.53	12.88 (0.51)	13.38 (0.49)

The best solutions were found by GP, and statistical analysis indicates that GP consistently outperforms the other three methods as it can be seen in Table 5. The difference between the various average results is statistically significant (P-value 4.57×10^{-7} for ANOVA test on the 4 samples of solutions found by each method). Finally, pairwise 2-tailed Student t-tests comparing GP with each other method demonstrate its better performance.

Table 5. False negative prediction: statistical significance of the different performances of the methods. First line shows ANOVA test on the 4 samples of solutions found by each method, while second line depicts pairwise 2-tailed Student t-tests comparing GP with each other method.

ANOVA $P = 4.57 \times 10^{-7}$		
Student t-test GP vs. SVM $P = 1.36 \times 10^{-6}$	Student t-test GP vs. MP $P = 8.53 \times 10^{-6}$	Student t-test GP vs. RF $P = 2.32 \times 10^{-7}$

The solutions found by GP typically use a rather small number of features (i.e. terminals). In fact, the solutions of the 50 GP runs are functions of a number of terminal that ranges from 1 to 23, with a median value of 4, and first and third quartiles of 2 and 7 respectively. Few of these features tend to recur in several solution as it can be seen in Table 6, where the gene symbol, the gene name of each feature, together with the number of solutions where the feature occurs are shown.

The authors of Refs. [33,32] used the seventy-gene signature by assigning a coefficient to each of the features and computing a score for each patient as the scalar product of these coefficients and the patient gene expression. To compare the performance of the various machine learning algorithms with this scoring system we proceeded as follows:

- We obtained the prognostic score s of the patients (excluding the ones used to train the signature in [33]) from the Supplementary Material of [32], and classified as good prognosis the patients with $s > 0.4$ and as bad prognosis the ones with s$\leq$ 0.4. This is the cutoff used in [32].
- We generated 50 random lists of 50 patients from this set, to match the statistic used for machine learning techniques, and computed for each list the number of false predictions given by the scoring method.

Table 6. The 10 most recurring features in the solutions found by GP. The four columns show: accession ID, gene symbol, gene description, and number of solutions where that feature occurs.

Accession ID	Gene symbol	Gene description	Solutions
NM_003981	PRC1	protein regulator of cytokinesis 1	48
NM_002916	RFC4	replication factor C (activator 1) 4, 37kDa	23
AI992158	-	-	16
AI554061	-	-	10
NM_006101	NDC80	NDC80 homolog, kinetochore complex component (S. cerevisiae)	9
NM_015984	UCHL5	ubiquitin carboxyl-terminal hydrolase L5	7
NM_020188	C16orf61	chromosome 16 open reading frame 61	6
NM_016448	DTL	denticleless homolog (Drosophila)	6
NM_014791	MELK	maternal embryonic leucine zipper kinase	6
NM_004702	-	-	6

The mean number of false predictions was 16.7, with a SEM of 0.3. Therefore the scoring method appears to be superior to all machine learning algorithm other than GP, and slightly inferior to GP. The difference between the performances of GP and the scoring method are not statistically significant ($P = 0.49$, 2-tailed Student t-test).

The original scoring method of [33,32], and in particular the suggested cutoff of 0.4, was chosen in such a way as to minimize the number of false negatives. Therefore it is not surprising that in this respect the scoring method is far superior to all machine learning methods, including GP. Indeed the average number of false negatives given by the scoring method is 1.7, to be compared to the numbers reported in Table 5. The implications of these results are discussed in the next section.

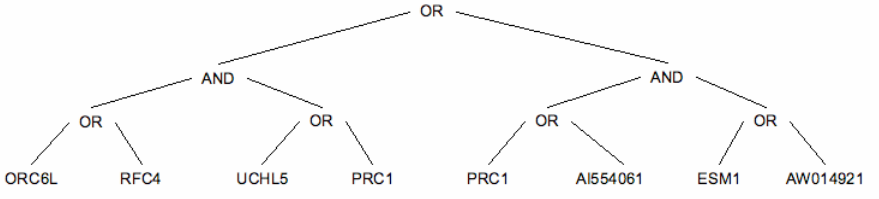
6 Discussion

The goal of our investigation was to refine the set of criteria that could lead to an individualized diagnosis of variations of Breast Cancer. To reach this goal we started from the well known "70-genes signature" and proceeded with the application of several machine learning schemes, in order to perform a comparison. We made some simplifying assumptions, preprocessed the data accordingly and ran several evaluation experiments.

Our results showed that while all the machine learning algorithms we used do have predictive power in classifying breast cancer patients into risk classes, GP clearly outperforms all other methods. Of course there is no way to do such a comparison in a completely unbiased way, as one could always argue that the levels of optimization are uneven. To minimize the possible bias, we tried to use default implementation of all the methods. The fact that all methods other than GP had very similar performances suggests that GP is indeed the most promising method.

The improvement in performance shown by GP compared to the original scoring method was rather small and not statistically significant. As expected, the scoring method was superior to all machine learning algorithms in minimizing false negatives. Nevertheless we believe our results warrant further investigation into the use of GP in this context for at least three reasons:

– As stated above, our implementation of GP was purposely not optimized, and we can expect substantial improvements in performance from further work aimed at tuning the various GP parameters.
– Maybe more importantly, GP can potentially offer biological insight and generate hypotheses for experimental work (see also [37]). Indeed an important result of our analysis is that the trees produced by GP tend to contain a limited number of features, and therefore are easily interpretable in biological terms. For example, the best-performing tree is shown in Figure 1 and includes 7 genes (features).
– Finally within the context of GP there is a natural way to tune the algorithm towards better sensitivity (specificity), simply by defining a fitness function in which false negatives (positives) are penalized more than errors of the other type.




```
(or (and (or ORC6L RFC4) (or UCHL5 PRC1))
    (and (or PRC1 AI554061) (or ESM1 AW014921)))
```

Fig. 1. Tree representation and the traditional Lisp representation of the model with the best fitness found by GP over the studied 50 independent runs

Future work along these lines should therefore focus on both improving the performance of GP and interpreting the results from the biological point of view. An obvious first step towards optimization would be to abandon the binariziation of the data (which here was used to produce trees that are easier to interpret) and build a GP based on continuous expression values. The biological interpretation might benefit from a statistical and functional analysis of the most recurring subtrees in optimal GP solutions.

In conclusion we have shown that Genetic Programming outperforms other machine learning methods as a tool to extract predictions from an established breast cancer gene signature. Given the possibility of generating biological insight and hypotheses that is intrinsic to the method, it deserves deeper investigation along the lines described above. Finally, it will be our task to test the GP approach on other features/gene sets that account for other cancers or other diseases, always with the objective of providing clinitians with more precise and individualized diagnosis criteria.

Acknowledgments. Leonardo Vanneschi gratefully acknowledges project PTDC/EIACCO/103363/2008 from Fundação para a Ciência e a Tecnologia, Portugal.

References

1. Alon, U., Barkai, N., Notterman, D., Gish, K., Ybarra, S., Mack, D., Levine, A.J.: Broad patterns of gene expression revealed by clustering analysis of tumour and normal colon tissues probed by oligonucleotide arrays. Proc. Nat. Acad. Sci. USA 96, 6745–6750 (1999)
2. Archetti, F., Lanzeni, S., Messina, E., Vanneschi, L.: Genetic programming for human oral bioavailability of drugs. In: Cattolico, M., et al. (eds.) Proceedings of the 8th annual conference on Genetic and Evolutionary Computation, Seattle, Washington, USA, pp. 255–262 (2006)
3. Archetti, F., Messina, E., Lanzeni, S., Vanneschi, L.: Genetic programming and other machine learning approaches to predict median oral lethal dose (LD50) and plasma protein binding levels (%PPB) of drugs. In: Marchiori, E., Moore, J.H., Rajapakse, J.C. (eds.) EvoBIO 2007. LNCS, vol. 4447, pp. 11–23. Springer, Heidelberg (2007)
4. Archetti, F., Messina, E., Lanzeni, S., Vanneschi, L.: Genetic programming for computational pharmacokinetics in drug discovery and development. Genetic Programming and Evolvable Machines 8(4), 17–26 (2007)
5. Bojarczuk, C.C., Lopes, H.S., Freitas, A.A.: Data mining with constrained-syntax genetic programming: applications to medical data sets. In: Proceedings Intelligent Data Analysis in Medicine and Pharmacology, vol. 1 (2001)
6. Breiman, L.: Random forests. Machine Learning 45(1), 5–32 (2001)
7. Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J.: Classification and Regression Trees. Wadsworth International Group, Belmont (1984)
8. Chu, F., Wang, L.: Applications of support vector machines to cancer classification with microarray data. Int. J. Neural Syst. 15(6), 475–484 (2005)
9. Darwin, C.: On the Origin of Species by Means of Natural Selection. John Murray (1859)
10. Deb, K., Raji Reddy, A.: Reliable classification of two-class cancer data using evolutionary algorithms. Biosystems 72(1-2), 111–129 (2003)
11. Deutsch, J.M.: Evolutionary algorithms for finding optimal gene sets in microarray prediction. Bioinformatics 19(1), 45–52 (2003)
12. Friedman, N., Linial, M., Nachmann, I., Peer, D.: Using bayesian networks to analyze expression data. J. Computational Biology 7, 601–620 (2000)
13. Goldberg, D.E.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, Reading (1989)
14. Guyon, I., Weston, J., Barnhill, S., Vapnik, V.: Gene selection for cancer classification using support vector machines. Machine Learning 46, 389–422 (2002)
15. Hernandez, J.C.H., Duval, B., Hao, J.-K.: A genetic embedded approach for gene selection and classification of microarray data. In: Marchiori, E., Moore, J.H., Rajapakse, J.C. (eds.) EvoBIO 2007. LNCS, vol. 4447, pp. 90–101. Springer, Heidelberg (2007)
16. Holland, J.H.: Adaptation in Natural and Artificial Systems. The University of Michigan Press, Ann Arbor (1975)
17. Hong, J.H., Cho, S.B.: The classification of cancer based on dna microarray data that uses diverse ensemble genetic programming. Artif. Intell. Med. 36, 43–58 (2006)
18. Hsu, A.L., Tang, S.L., Halgamuge, S.K.: An unsupervised hierarchical dynamic self-organizing approach to cancer class discovery and marker gene identification in microarray data. Bioinformatics 19(16), 2131–2140 (2003)
19. Koza, J.R.: Genetic Programming. MIT Press, Cambridge (1992)
20. Langdon, W.B., Buxton, B.F.: Genetic programming for mining dna chip data from cancer patients. Genetic Programming and Evolvable Machines 5(3), 251–257 (2004)
21. Liu, J.-J., Cutler, G., Li, W., Pan, Z., Peng, S., Hoey, T., Chen, L., Ling, X.-B.: Multiclass cancer classification and biomarker discovery using ga-based algorithms. Bioinformatics 21, 2691–2697 (2005)

22. Lu, Y., Han, J.: Cancer classification using gene expression data. Inf. Syst. 28(4), 243–268 (2003)
23. Michie, D., Spiegelhalter, D.-J., Taylor, C.-C.: Machine learning, neural and statistical classification. Prentice-Hall, Englewood Cliffs (1994)
24. Moore, J.-H., Parker, J.-S., Hahn, L.-W.: Symbolic discriminant analysis for mining gene expression patterns. In: Flach, P.A., De Raedt, L. (eds.) ECML 2001. LNCS (LNAI), vol. 2167, pp. 372–381. Springer, Heidelberg (2001)
25. Nevins, J.R., Potti, A.: Mining gene expression profiles: expression signatures as cancer phenotypes. Nat. Rev. Genet. 8(8), 601–609 (2007)
26. Paul, T.K., Iba, H.: Gene selection for classification of cancers using probabilistic model building genetic algorithm. Biosystems 82(3), 208–225 (2005)
27. Platt, J.: Fast training of support vector machines using sequential minimal optimization. In: Advances in Kernel Methods – Support Vector Learning (1998)
28. Poli, R., Langdon, W.B., McPhee, N.F.: A field guide to genetic programming (With contributions by J. R. Koza) (2008), http://lulu.com, http://www.gp-field-guide.org.uk (2008)
29. Rosskopf, M., Schmidt, H.A., Feldkamp, U., Banzhaf, W.: Genetic programming based DNA microarray analysis for classification of tumour tissues. Technical Report Technical Report 2007-03, Memorial University of Newfoundland (2007)
30. Haykin, S.: Neural Networks: a comprehensive foundation. Prentice-Hall, London (1999)
31. Silva, S.: GPLAB – a genetic programming toolbox for MATLAB, version 3.0 (2007), http://gplab.sourceforge.net
32. van de Vijver, M.J., He, Y.D., van't Veer, L.J., Dai, H., Hart, A.A.M., Voskuil, D.W., Schreiber, G.J., Peterse, J.L., Roberts, C., Marton, M.J., Parrish, M., Atsma, D., Witteveen, A., Glas, A., Delahaye, L., van der Velde, T., Bartelink, H., Rodenhuis, S., Rutgers, E.T., Friend, S.H., Bernards, R.: A gene-expression signature as a predictor of survival in breast cancer. N. Engl. J. Med. 347(25), 1999–2009 (2002)
33. van't Veer, L.J., Dai, H., van de Vijver, M.J., He, Y.D., Hart, A.A.M., Mao, M., Peterse, H.L., van der Kooy, K., Marton, M.J., Witteveen, A.T., Schreiber, G.J., Kerkhoven, R.M., Roberts, C., Linsley, P.S., Bernards, R., Friend, S.H.: Gene expression profiling predicts clinical outcome of breast cancer. Nature 415(6871), 530–536 (2002)
34. Vanneschi, L.: Theory and Practice for Efficient Genetic Programming. Ph.D. thesis, Faculty of Sciences, University of Lausanne, Switzerland (2004)
35. Vapnik, V.: Statistical Learning Theory. Wiley, New York (1998)
36. Weka. A multi-task machine learning software developed by Waikato University (2006), http://www.cs.waikato.ac.nz/ml/weka
37. Yu, J., Yu, J., Almal, A.A., Dhanasekaran, S.M., Ghosh, D., Worzel, W.P., Chinnaiyan, A.M.: Feature selection and molecular classification of cancer using genetic programming. Neoplasia 9(4), 292–303 (2007)

Correlation–Based Scatter Search for Discovering Biclusters from Gene Expression Data

Juan A. Nepomuceno[1], Alicia Troncoso[2], and Jesús S. Aguilar–Ruiz[2]

[1] Department of Computer Science, University of Sevilla, Spain
janepo@us.es
[2] Area of Computer Science, Pablo de Olavide University of Sevilla, Spain
{ali,aguilar}@upo.es

Abstract. Scatter Search is an evolutionary method that combines existing solutions to create new offspring as the well–known genetic algorithms. This paper presents a Scatter Search with the aim of finding biclusters from gene expression data. However, biclusters with certain patterns are more interesting from a biological point of view. Therefore, the proposed Scatter Search uses a measure based on linear correlations among genes to evaluate the quality of biclusters. As it is usual in Scatter Search methodology an improvement method is included which avoids to find biclusters with negatively correlated genes. Experimental results from yeast cell cycle and human B-cell lymphoma datasets are reported showing a remarkable performance of the proposed method and measure.

Keywords: Gene Expression Data, Biclustering, Scatter Search, Evolutionary Computation.

1 Introduction

Nowadays, the study of the process of how proteins are coded by genes is one of the most important research topics in Biology. This codification process is known as *Gene Expression*. DNA microarrays technology enables us to mesure the gene expression level under a specific group of conditions. Data mining techniques are needed to analyze the huge volume of all this biological information [1]. The goal of *Biclustering* techniques is to discover transcription factors which determine that a group of genes is co-expressed under a set of conditions.

Several biclustering methods have been proposed in the last few years [2]. For example, in [3] an iterative hierarchical clustering is separately applied to each dimension and biclusters are built by the combination of the obtained results for each dimension. The Cheng and Church algorithm [4] builds biclusters by adding or removing genes or conditions in order to improve the measure of quality called Mean Squared Residue (MSR). In [5], it is proposed an exhaustive biclusters enumeration by means of a bipartite graph-based model, in which nodes were added o removed in order to find subgraphs with maximum weights. The FLOC algorithm [6] improved the method presented in [4] by obtaining a set of biclusters simultaneously and by adding missing values techniques. In [7], a

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2010, LNCS 6023, pp. 122–133, 2010.

simple linear model, in which normally distributed expression level for each gene or condition was supposed, for gene expression data was applied. Evolutionary computation techniques based on the MSR measure are used in [8,9] or Simulated Annealing in [10]. But, although MSR is used in many algorithms as merit function, it is not the most appropriate measure as the MSR measure can not find scaling patterns when the variance of gene values is high in the bicluster [11]. Recently, the study of the nature of different patterns in biclusters has motivated new techniques based on the search of hyperplanes in high–dimensional data space as interesting patterns share the geometry of linear manifolds [12,13,14]. Other measures to evaluate biclusters have been proposed as fitness function for optimization methods [15].

The gene expression level under a set of conditions can be seen as the values of a discrete random variable. Thus, the linear dependency between two genes can be studied by using the correlation coefficient between two random variables. This fact has motivated the use of the proposed measure in this paper. This measure based on correlations among genes is the main term of the fitness function proposed to evaluate the quality of biclusters in the Scatter Search. Scatter Search is a population-based method that emphasizes systematic processes against random procedures. Thus, the generation of the initial population is not random but a generation method based on diversification [16] is used to generate a set of diverse initial solutions. Moreover, Scatter Search includes an improvement method with the aim of exploiting the diversity provided by the generation and combination method.

This paper is organized as follows. The proposed Scatter Search is presented and different steps such as the improvement method and the fitness function are explained in details in Section 2. Some experimental results from two real datasets are reported in Section 3. Finally, Section 4 outlines the main conclusions of the paper and future works.

2 Description of the Algorithm

Scatter Search [16] is a population-based optimization metaheuristic which has recently been applied to combinatorial and nonlinear optimization problems. Optimization algorithms based on populations are search procedures where a set of individuals that represent trial solutions evolves in order to find optimum solutions of the problem. On the opposite to other evolutionary heuristics, Scatter Search emphasizes systematic processes against random procedures. Scatter Search uses strategies to diversify and to intensify the search in order to avoid local minima and to find quality solutions.

Basically, the optimization process consists in the evolution of a set called *Reference Set*. This set is initially built with the best solutions from the population, according to the value of their fitness function, and the most scattered ones from the population regarding the previous best solutions. This set is updated by using the *Combination Method* and the *Improvement Method* until it does not change. When the *Reference Set* is stable, it is rebuilt again. That is, the

building of the *Reference Set* is based on quality and diversity, but its updating is only guided by quality. Thus, diversity is introduced in the evolutionary process when the initial population is generated and, mainly, when the reference set is rebuilt in each step. The search intensification is due to the improvement method where the solutions are improved by exploiting the knowledge of the problem.

The pseudocode of the proposed Scatter Search for biclustering is presented in Algorithm 1. The Scatter Search process is repeated $numBi$ times where $numBi$ is the number of biclusters to be found and the best solution of the reference set is stored in a set called *Results* for each iteration. Thus, the *Results* set is formed by $numBi$ biclusters and it is the output of the Algorithm 1.

2.1 Initialization Phase

Formally, a microarray is a real matrix M composed of N genes and L conditions. The element (i, j) of the matrix means the level of expression of gene i under the condition j. A bicluster B is a submatrix of the matrix M composed of $n \leq N$ rows or genes and $l \leq L$ columns or conditions. Biclusters are encoded by binary strings of length $N + L$. Each of the first N bits of the binary string is related to the genes and the remaining L bits to the conditions from microarray as it can be be seen in Fig. 1.

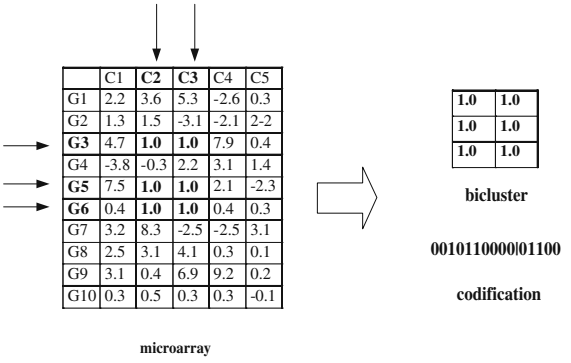


Fig. 1. Microarray and bicluster {G3,G5,G6|C2,C3} with its codification

The initial population is generated with solutions as diverse as possible. Thus, the diversification generation method [16] takes a binary string, x_i with $i = 1, \ldots, n$ where n is the number of bits, as a seed solution and generates solutions x'_i by the following rule:

$$x'_{1+kh} = 1 - x_{1+kh} \text{ for } k = 0, 1, 2, 3, \ldots, \lfloor n/h \rfloor \tag{1}$$

where $\lfloor n/h \rfloor$ is the largest integer less or equal than n/h and h is an integer less than $n/5$. All remaining bits of x' are equal to that of x.

Algorithm 1. Scatter Search Algorithm for Biclustering

INPUT microarray M, number of biclusters to be found $numBi$, penalization factors M_1 and M_2, maximum number of iterations $numIter$, size of the initial population and size S of the reference set.

OUTPUT Set $Results$ with $numBi$ biclusters.

```
begin
  num ← 0, Results ← ∅
  while (num < numBi) do
    Initialize population P
    P ← Improvement Method (P)
    //Building Reference Set
    R1 ← S/2 best biclusters from P (according to the fitness function)
    R2 ← S/2 most scattered biclusters, regarding R1, from P \ R1 (according to a
    distance).
    RefSet ← (R1 ∪ R2)
    P ← P \ RefSet
    //Initialization
    stable ← FALSE, i ← 0
    while (i < numIter) do
      while (NOT stable) do
        A ← RefSet
        B ← Combination Method(RefSet)
        B ← Improvement Method(B)
        RefSet ← S best biclusters from RefSet ∪ B
        if (A = RefSet) then
          stable ← TRUE
        end if
      end while
      //Rebuilding Reference Set
      R1 ← S/2 best biclusters from RefSet
      R2 ← S/2 most scattered biclusters from P \ R1
      RefSet ← (R1 ∪ R2)
      P ← P \ RefSet
      i ← i + 1
    end while
    //Storage in Results
    Results ← the best one from RefSet
    num ← num + 1
  end while
end
```

After generating all posible solutions with that seed, if more solutions are necessary, the diversification generation method is applied again by using the last solution as new seed.

2.2 Biclusters Evaluation: Fitness Function

In this work, biclusters with shifting and scaling patterns are desired. A group of genes has a *shifting pattern* when the expression values vary in the addition

of a fixed value for all the genes. A group of genes has a *scaling pattern* when the expression values vary in the multiplication of a fixed value for all the genes. Two genes show a shifting and scaling pattern if they are described from (2).

$$g_Y = \alpha g_X + \beta \quad \alpha, \beta \in \mathbb{R} \tag{2}$$

Consequently, two genes with shifting and scaling patterns are linearly dependent.

The correlation coefficient between two variables X and Y measures the grade of linear dependence between them. It is defined by:

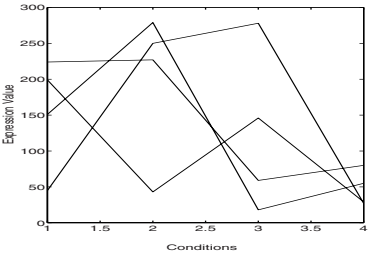
$$\rho(X,Y) = \frac{cov(X,Y)}{\sigma_X \sigma_Y} = \frac{\sum_i^n (x_i - \overline{x})(y_i - \overline{y})}{n\sigma_X \sigma_Y} \tag{3}$$

where $cov(X,Y)$ is the covariance of the variables X and Y, $\overline{x}$ and $\overline{y}$ are the average of the values of the variables X and Y and σ_X and σ_Y are the standard deviations of X and Y, respectively. The values for the correlation coefficient vary between -1 and 1. If $\rho(X,Y) = 0$, the variables X and Y are linearly independent, and if $\rho(X,Y) = \pm 1$ the variables are linearly dependent. When the correlation value is equal to -1, the variables X and Y are dependent with negative correlation, that is, when the values of the variable X increase the values of the variable Y decrease linearly.

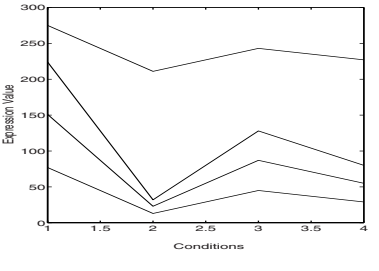
Given a bicluster B composed of N genes, $B = [g_1, \ldots, g_N]$, the average correlation of B, $\rho(B)$, is defined as follows,

$$\rho(B) = \frac{1}{\binom{N}{2}} \sum_{i=1}^{N} \sum_{j=i+1}^{N} \rho_{g_i g_j} \tag{4}$$

where $\rho_{g_i g_j}$ is the correlation coefficient between the gene i and the gene j. Note that $\rho_{g_i g_j} = \rho_{g_j g_i}$, therefore, only $\binom{N}{2} = \frac{N(N-1)}{2}$ elements have been considered in the aforementioned sum.



$$\begin{bmatrix} 151 & 279 & 18 & 55 \\ 199 & 43 & 146 & 29 \\ 224 & 227 & 59 & 80 \\ 45 & 250 & 278 & 27 \end{bmatrix} \Rightarrow \rho(B) = 0.17$$



$$\begin{bmatrix} 151 & 23 & 87 & 55 \\ 77 & 13 & 45 & 29 \\ 224 & 32 & 128 & 80 \\ 275 & 211 & 243 & 227 \end{bmatrix} \Rightarrow \rho(B) = 1$$

Fig. 2. Biclusters with lowly–correlated and highly–correlated genes

Fig. 2 presents two biclusters along with their average correlations. It can be observed that the bicluster with perfect shifting and scaling patterns has an average correlation of 1 while that the bicluster without patterns has an average correlation close to 0.

In this work, biclusters with highly–correlated genes and high volume are preferred. Therefore, the fitness function used to evaluate the quality of biclusters is defined by:

$$f(B) = (1 - \rho(B)) + \sigma_\rho + M_1 \left(\frac{1}{nG}\right) + M_2 \left(\frac{1}{nC}\right) \tag{5}$$

where nG and nC are the number of genes and conditions of the bicluster B, respectively, M_1 and M_2 are penalization factors to control the volume of the bicluster B and σ_ρ is the standard deviation of the values ρ_{g_i,g_j} from (4). The standard deviation is included in order to avoid that the value of the average correlation can be high for a bicluster and this bicluster can contain several non–correlated genes with the remaining ones of the bicluster. Best biclusters are the ones with the lowest value for the fitness function. Thus, it has been considered $(1 - \rho(B))$ to evaluate biclusters with highly–correlated genes as good biclusters.

2.3 Improvement Method

Scatter Search uses improvement methods when the solutions have to fulfill some constraints or simply to improve them in order to intensify the search process. This method depends on the problem under study and usually it consists in classical local searches for continuous optimization problems.

The goal of this work is to find biclusters with shifting and scaling patterns. Thus, biclusters with positively–correlated genes are only searched for. Therefore, the proposed improvement method aims at extracting positively–correlated genes either from biclusters of the initial population or from biclusters obtained by the combination method. The pseudocode of the improvement method is presented in the Algorithm 2.

2.4 Building and Rebuilding Method of the Reference Set

Reference set is initially built with the best solutions, according to the value of their fitness function, and the most scattered ones from the population regarding the previous best solutions. The *Hamming* distance is used to measure the distance among biclusters in this work. After getting the stability of reference set in the updating process, it is rebuilt to introduce diversity in the search process. Thus, the reference set is rebuilt with the best biclusters from the updated reference set, according to the fitness function, and the most distant from the population regarding the previously chosen best solutions.

The initial population has to be updated too in the evolutionary process by removing solutions which have already been considered in the building or rebuilding of the reference set. When the initial population is empty, a new

Algorithm 2. IMPROVEMENT METHOD

INPUT Bicluster $B = [g_1, \ldots, g_N]$
OUTPUT Bicluster $B' \subseteq B$ such that $\rho_{g_i,g_j} \geq 0 \quad \forall g_i, g_j \in B'$

```
begin
  i ← 1, B' ← {g_i}, R ← {}
  while (i ≤ N) do
    j ← i + 1
    while (j ≤ N) do
      if (ρ_{g_i,g_j} > 0) then
        if (g_j ∉ R) then
          B' ← B' ∪ {g_j}
        end if
      else
        R ← R ∪ {g_j}
        B' ← B' \ {g_j}
      end if
      j ← j + 1
    end while
    i ← i + 1
  end while
end
```

population is created by using the diversification generation method previously explained.

2.5 Combination Method and Reference Set Updating

New solutions are introduced in the search process by the combination method. Two solutions are combined by using a uniform crossover operator and a new one is generated. All pairs of biclusters in the reference set are combined, generating thus, $S * (S-1)/2$ new biclusters where S is the size of the reference set. This crossover operator generates randomly a mask and the child is composed of values from the first parent when there is a 1 in the mask, and from the second parent when there is a 0.

After combining all pairs of biclusters, the best solutions from the joining of the previous reference set and the new solutions are chosen. Hence, best solutions according to the value of their fitness function remain in the reference set.

3 Experimental Results

Two well known data sets have been used to show the performance of the proposed algorithm. The first data set is the *human B-cells lymphoma* expression data with 4026 genes and 96 conditions [17]. The second one is the *yeast Saccharomyces cerevisiae* cell cycle expression with 2884 genes and 17 experimental conditions [18]. Both data sets were used in [4] where original data were processed by replacing missing values with random values.

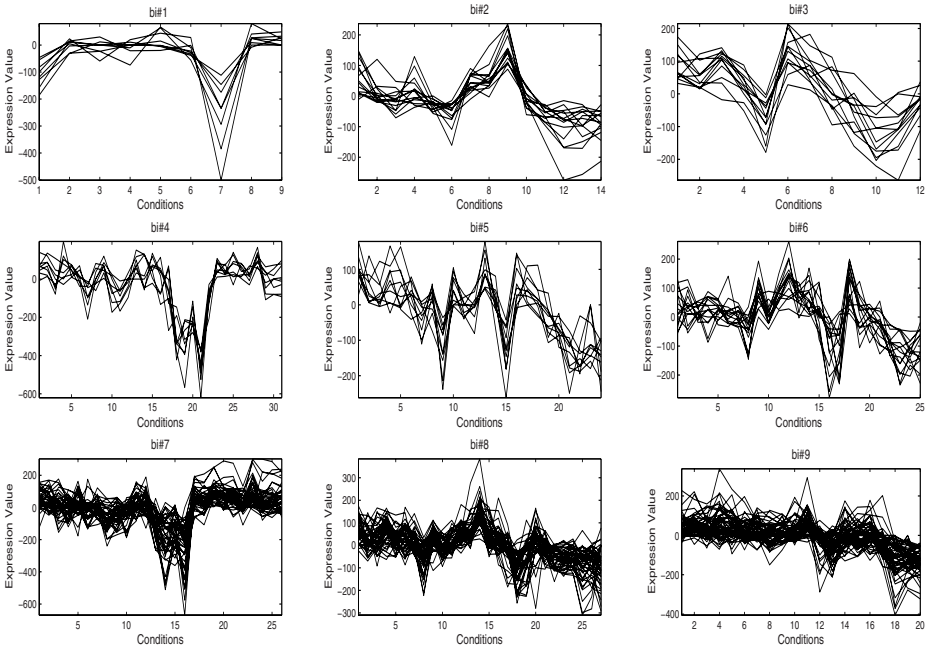


Fig. 3. Biclusters found by Algorithm 1 from lymphoma data set

The main parameters of the Algorithm 1 are as follows: 20 for the maximum number of iterations of the Scatter Search, 10 for the size of the reference set, 200 for the number of solutions of the initial population and 10 for the number of biclusters to be found in each execution. M_1 and M_2 are weights in order to drive the search depending on the required size of biclusters. High values of M_1 and M_2 may be used when biclusters with a lot of genes and conditions are desired.

Fig. 3 presents several biclusters obtained by the application of the Scatter Search to the Lymphoma dataset. These biclusters have been obtained with different values for the weights M_1 and M_2 in order to test their influence. The biclusters $bi\#1$, $bi\#2$ and $bi\#3$ have been found with $M_1 = 1$ and $M_2 = 1$, $bi\#4$, $bi\#5$ and $bi\#6$ with $M_1 = 1$ and $M_2 = 10$ and $bi\#7$, $bi\#8$ and $bi\#9$ with $M_1 = 10$ and $M_2 = 10$. It can be observed how these weights determine the number of genes and conditions of the biclusters. Note that shifting and scaling patterns can clearly be appreciated in all biclusters.

Table 1 shows the information about the biclusters represented in Fig. 3. For each bicluster an identifier of the bicluster, the number of genes, the number of conditions, the volume, the average correlation, the standard deviation, the MSR measure and the variance of gene values are presented. The variance of gene values measures how different the values for the gene expression level are and if this value is high then the MSR measure is high too [11]. It can be observed that

Table 1. Information about biclusters found by Algorithm 1 from lymphoma dataset

id.	Genes	Conditions	Volume	$\rho(B)$	$\sigma_\rho(B)$	MSR	Genes Variance
bi#1	8	9	72	0.92	0.5	2124.6	9742.3
bi#2	14	14	196	0.85	0.4	1657.4	6468.9
bi#3	12	12	144	0.84	0.4	1608.7	8184.2
bi#4	8	31	248	0.82	0.5	3337.4	20042.2
bi#5	10	24	240	0.77	0.4	1963.3	8007.5
bi#6	15	25	375	0.70	0.4	2389.5	7620.4
bi#7	44	26	1144	0.62	0.4	4800.0	11558.8
bi#8	47	27	1269	0.60	0.3	2779.3	6219.8
bi#9	58	20	1160	0.59	0.3	2878.0	6157.4

the higher volume for biclusters, the smaller value for the average correlation. However, all biclusters present a high value for the average correlation which indicates that such biclusters present shifting and scaling patterns. Moreover, the standard deviation is low, that is, the correlation coefficients of each pair of genes have similar values and they are close to the average correlation of the bicluster. Therefore, all biclusters with a high average correlation do not contain non–correlated genes as it can be observed in Fig. 3. Most of papers published in the literature present algorithms based on the MSR measure and a bicluster is considered a high—quality bicluster for the Lymphoma dataset if the value

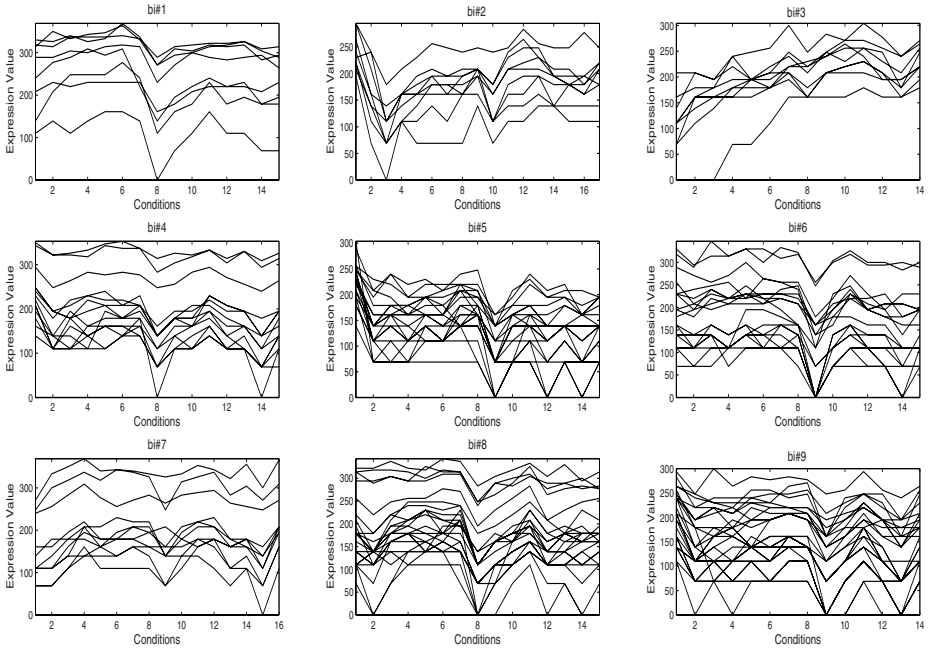


Fig. 4. Biclusters found by Algorithm 1 from yeast dataset

of its MSR is less than 1200 [4,8]. It can be noticed that the bicluster $bi\#4$ has a value of MSR much greater than 1200 since its variance of gene values is high. However, it can be considered a high–quality bicluster because it presents shifting and scaling patterns as its average correlation is 0.82.

In Fig. 4 several biclusters found by the proposed Scatter Search from Yeast dataset are shown. These biclusters have been obtained with values $M_1 = 1$ and $M_2 = 10$ in order to obtain biclusters with a moderate volume. From a geometrical point of view, it can be noticed that the genes present a similar behavior under a set of conditions. Information about these biclusters is presented in Table 2. In the literature, a bicluster is considered a high—quality bicluster for the Yeast dataset if the value of its MSR is less than 300 [4,8]. However, it can be appreciated how several biclusters that have values of MSR greater than 300 present high average correlations, and therefore, shifting and scaling patterns.

Table 2. Information about biclusters found by Algorithm 1 from yeast dataset

id.	Genes	Conditions	Volume	$\rho(B)$	$\sigma_\rho(B)$	MSR	Genes variance
bi#1	8	15	120	0.83	0.3	273.4	1028.5
bi#2	9	17	153	0.74	0.4	306.6	1230.9
bi#3	9	14	126	0.84	0.4	367.8	1740.7
bi#4	14	15	210	0.76	0.4	257.7	854.0
bi#5	20	15	300	0.75	0.3	437.9	1367.5
bi#6	19	15	285	0.75	0.4	342.9	1119.4
bi#7	11	16	176	0.70	0.4	245.7	842.2
bi#8	23	15	345	0.68	0.4	369.4	991.7
bi#9	27	14	378	0.72	0.3	332.2	1038.0

Figs. 5 and 6 show the performance of the Scatter Search with and without improvement method from Lymphoma dataset (Figures. 5a) and 5b), respectively) and Yeast dataset (Figures. 6a) and 6b), respectively). The evolution of the average fitness function, average correlation and average volume for all biclusters of the reference set throughout the iterations is presented. It can be observed how the average correlation increases when the fitness function decreases during the evolutionary process. When the improvement method is not included in the Scatter Search scheme, the convergence of the fitness function to an optimum solution is very slow and it is necessary more iterations. Obviously, the improvement method leads to lower volumes for both datasets due to this method just selects the positively–correlated genes by removing the negatively–correlated genes.

Table 3 presents the average correlation and average volume for ten biclusters obtained by the Algorithm 1, ignoring the improvement method and considering it in order to show the importance of such method in the proposed Scatter Search. Notice that the average correlation is lower when the improvement method is not included due to the negatively–correlated genes can be considered.

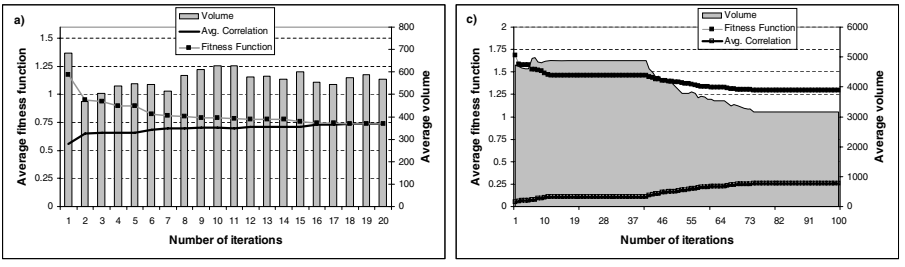


Fig. 5. Evolutionary process for Lymphoma dataset: a)improvement method considered; b) no improvement method considered

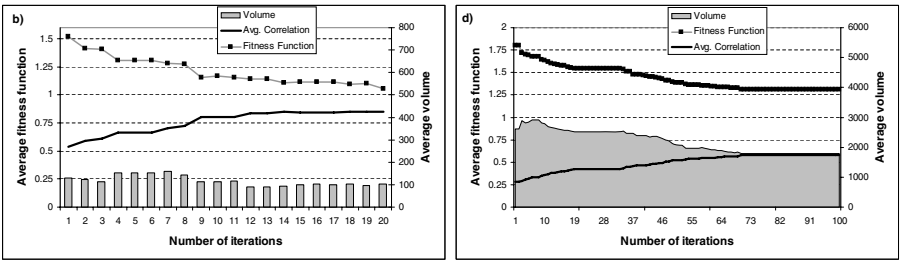


Fig. 6. Evolutionary process for Yeast dataset: a)improvement method considered; b) no improvement method considered

Table 3. Optimal solution with and without improvement method

	With Improvement Method		Without Improvement Method	
	Correlation	Volume	Correlation	Volume
lymphoma	0.74	600	0.02	14110
yeast	0.80	154	0.16	7065

4 Conclusions

In this paper, a Scatter Search for finding biclusters from gene expression data has been presented. The proposed Scatter Search has used as merit function to evaluate biclusters a new measure based on correlations among genes with the aim of obtaining biclusters with shifting and scaling patterns. Moreover an improvement method, which consist in removing negatively–correlated genes from biclusters, has been incorporated to intensify the search. Experimental results from human B-cell lymphoma data set and yeast cell cycle data set have been reported revealing the good convergence and remarkable performance of the proposed method and measure.

Future works will focused on some improvements for the proposed algorithm with regard to the overlapping among genes and the comparison with other biclustering techniques using Gene Ontology Database [14].

Acknowledgments

The financial support given by the Spanish Ministry of Science and Technology, project TIN2007-68084-C00 and by the Junta de Andalucía, project P07-TIC-02611 is acknowledged.

References

1. Larranaga, P., et al.: Machine learning in bioinformatics. Briefings in Bioinformatics 7(1), 86–112 (2006)
2. Busygin, S., Prokopyev, O., Pardalos, P.: Biclustering in data mining. Computers and Operations Research 35(9), 2964–2987 (2008)
3. Getz, G., Levine, E., Domany, E.: Couple two-way clustering analysis of gene microarray data. In: Proceedings of the National Academy of Sciences (PNAS) of the USA, pp. 12079–12084 (2000)
4. Cheng, Y., Church, G.: Biclustering of Expression Data. In: Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology, vol. 8, pp. 93–103 (2000)
5. Tanay, A., Sharan, R., Shamir, R.: Discovering statistically significant biclusters in gene expression data. Bioinformatics 18(90001), 136–144 (2002)
6. Yang, J., Wang, H., Wang, W., Yu, P.: Enhanced biclustering on expression data. In: 3rd IEEE Simposium on Bioinformatics and Bioengeneering, pp. 321–327 (2003)
7. Bergmann, S., Ihmels, J., Barkai, N.: Iterative signature algorithm for the analysis of large-scale gene expression data. Physical Review E 67(031902) (2003)
8. Divina, F., Aguilar-Ruiz, J.: Biclustering of Expression Data with Evolutionary Computation. IEEE Transactions on Knowledge and Data Engineering 18(5), 590–602 (2006)
9. Mitra, S., Banka, H.: Multi-objective evolutionary biclustering of gene expression data. Pattern Recognition 39(12), 2464–2477 (2006)
10. Bryan, K.: Biclustering of Expression Data Using Simulated Annealing. In: Proceedings of the 18th IEEE International Symposium on Computer-Based Medical Systems, USA, pp. 383–388 (2005)
11. Aguilar-Ruiz, J.: Shifting and scaling patterns from gene expression data. Bioinformatics 21(20), 3840–3845 (2005)
12. Harpaz, R., Haralick, R.: Mining Subspace Correlations. In: IEEE Symposium on Computational Intelligence and Data Mining, pp. 335–342 (2007)
13. Zhao, H., Liew, A., Xie, X., Yan, H.: A new geometric biclustering algorithm based on the Hough transform for analysis of large-scale microarray data. Journal of Theoretical Biology 251(2), 264–274 (2008)
14. Gan, X., Liew, A., Yan, H.: Discovering biclusters in gene expression data based on high-dimensional linear geometries. BMC Bioinformatics 9(209), 1–15 (2008)
15. Nepomuceno, J.A., Troncoso Lora, A., Aguilar-Ruiz, J.S., García-Gutiérrez, J.: Biclusters Evaluation Based on Shifting and Scaling Patterns. In: Yin, H., Tino, P., Corchado, E., Byrne, W., Yao, X. (eds.) IDEAL 2007. LNCS, vol. 4881, pp. 840–849. Springer, Heidelberg (2007)
16. Marti, R., Laguna, M.: Scatter Search. In: Methodology and Implementation in C. Kluwer Academic Publishers, Boston (2003)
17. Alizadeh, A., et al.: Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. Nature 403, 503–511 (2000)
18. Cho, R., et al.: A Genome-Wide Transcriptional Analysis of the Mitotic Cell Cycle. Molecular Cell 2(1), 65–73 (1998)

A Local Search Appproach for Transmembrane Segment and Signal Peptide Discrimination

Sami Laroum[1,2], Dominique Tessier[1], Béatrice Duval[2], and Jin-Kao Hao[2]

[1] UR1268 Biopolymères Interactions Assemblages,
INRA, 44300 Nantes, France
{slaroum,tessier}@nantes.inra.fr
[2] LERIA, 2 Boulevard Lavoisier, 49045 Angers, France
{bd,hao}@info.univ-angers.fr

Abstract. Discriminating between secreted and membrane proteins is a challenging task. This is particularly true for discriminating between transmembrane segments and signal peptides because they have common biochemical properties. In this paper, we introduce a new predictive method called LSTranslocon (Local Search Translocon) based on a Local Search methodology. The method takes advantage of the latest knowledge in the field to model the biological behaviors of proteins with the aim of ensuring good prediction. The LS Prediction approach is assessed on a constructed data set from Swiss-Prot database and compared with one of the best methods from the literature.

Keywords: Subcellular localization, amino acid position, transmembrane segment insertion, local search.

1 Introduction

Subcellular localization of proteins is important for the understanding of gene/protein function. Most eukaryotic protein cells are synthesized in the cytosol and are translocated to various subcellular compartments. Recent studies have led to a better understanding of the transport mechanisms of protein entering the secretory pathway and a complete review can be found in [1]. During biosynthesis, newly synthesized proteins that contain a targeting signal are directed towards the endoplasmic reticulum (ER). The targeting signals are either N-terminal signal sequences called signal peptides (SP) or, in the case of many membrane proteins that lack discrete signal peptides, the first transmembrane sequence called a signal anchor (SA). Then, proteins are translocated across the ER through the translocon channel. If the segment of amino acids inside the channel contains the "right key", the translocon opens sideways and the protein fits in the membrane. Otherwise, the protein is fully translocated across the ER membrane and released into the ER lumen. This phenomena induces an additional difficulty for discrimination between SP and SA and requires special techniques in order to obtain reliable predictive results.

Many prediction methods for this difficult protein subcellular localization problem have been developed over the years to localize proteins with signal

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2010, LNCS 6023, pp. 134–145, 2010.

peptides ([2,3]), or to localize protein with transmembrane segments ([4,5]). However, in spite of their high prediction performance, in certain cases they still yield false predictions. Discrimination between secreted and transmembrane proteins thus remains a challenging task [6]. The frequent false classifications are mainly a consequence of the fact that these proteins contain both a hydrophobic stretch that can be interpreted either as the hydrophobic core region of a signal peptide - the targeting N-terminal signal of secreted proteins - or as a transmembrane segment (TM segment).

Based on an analysis of known soluble and transmembrane sequences, the purpose of the present study was to design a predictive method to define the rules for membrane insertion during the crossing of the translocon. We looked for the code that enables the opening of the translocon knowing that it has to distinguish peptides that share certain biochemical properties: signal peptides, signal anchors and helical transmembrane segments. By sliding a window of about 19 amino acids along the sequence of a protein - the width of the ER membrane -, if we know the code for opening the translocon, we will be able to predict if the protein is a soluble protein or if it is integrated in the membrane.

The method presented in this paper is based on a local search approach and takes advantage of the latest biological knowledge. For this purpose, we consider subcellular localization prediction as a combinatorial search problem and devise a local search based procedure to determine near-optimal solutions. One notices that this is the first time that such an approach is applied to this difficult prediction problem.

The paper is organized as follows: in Section 2, we present some state-of-the-art prediction methods. In Section 3, we describe our approach for membrane protein recognition. In Section 4, we show computational results and comparisons with one best performing method. Finally, in Section 5, we conclude the paper by giving a summary and the future work.

2 Prediction Methods

Many predictive algorithms dedicated to signal peptide or transmembrane segments are available. The first prediction methods were based on the evaluation of the hydrophobicity of each amino acid. The method used a "sliding window" with fixed width (19 residues) to identify transmembrane segments along the protein sequence and the hydrophobicity average was calculated for amino acids within the window [10]. At the moment, the most popular predictors implemented as web servers for both signal peptide or transmembrane predictions, are usually built upon learning systems such as neural networks (NN) [11], Hidden Markov Model (HMM) [16] and Support Vector Machines (SVM) ([17,18]).

Programs dedicated to signal peptide prediction try to localize precisely the N-terminal signal sequence, the signal sequence cleavage site, or a combination of both features. The current most widely used methods are PrediSi [12] and SignalP3.0 [13].

Membrane topology[1] prediction programs evaluate if a protein is likely to be a membrane protein or not and how many membrane protein domains it has. Sometimes, more precise information is added with the orientation and the boundaries of the TM domains. Some popular transmembrane predictors are TOPPred [15] and HMMTOP [14]. Description and evaluation of the main methods can be found in ([19,20]).

However, few methods focus on the difficulty of a correct discrimination between transmembrane segments and signal peptides. One of the best methods dedicated to this difficult prediction problem is Phobius [7] published in 2004. The Phobius method is based on a HMM model which combines a TM helix submodel with a SP submodel. Transition between states are arranged such that the position of SP is located at the N-terminal of the sequence -the hydrophobic region of a SP is seldom located after the first 30 amino acids- whereas TM segments can be found at any position in the sequence. An evolution of the Phobius method is proposed with Philius [8] which implements a topology prediction by dynamic Bayesian networks. The state transition topology of Philius exactly mimics that of Phobius, and performances of Philius are close to those obtained with Phobius. The last method, SPOCTOPUS [9] combines a neural network method with a hidden Markov model. SPOCTOPUS first performs a homology search to create a sequence profile. This method reports a very high accuracy, but it is difficult to link these results with a biological interpretation.

3 A New Approach for Recognition of TM Proteins

3.1 New Biological Knowledge

Recent studies expand understanding of the recognition and the insertion of TM segments by the translocon ([21,22]). Briefly, the experiments describe the insertion of the membrane proteins into the ER membrane by the evaluation of the amino acid contribution during the insertion process. These studies suggest that insertion or not of helical transmembrane segment depends mainly on the local contribution of each amino acid.

First, in their experiments Hessa *et al.* assess the contribution of each amino acid in different positions along the membrane. The key observation is that the amino acid position plays a determining role during targeting by the translocon. The insertion would be mainly related to an interaction energy between the sequence of amino acids committed in the translocon and the membrane. To calculate this energy of interaction, Hessa *et al.* suggest a hydrophobic scale called "biological hydrophobicity scale". For each amino acid a curve determines its influence according to its position in the translocon.

The experimental studies show that (1) each amino acid has a different hydrophobic index for different sequence positions [21] and (2) the scales are symmetric across sequence positions [23]. For example, Figure 1 displays the curves for two amino acids.

[1] Membrane topology describes which regions of the polypeptide chain span the membrane.

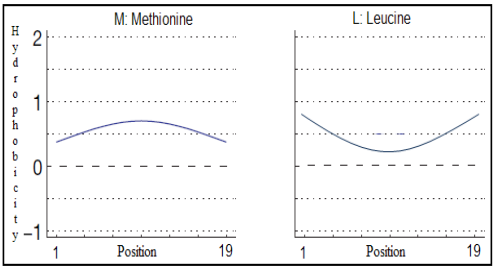


Fig. 1. Contribution scales of two amino acids Methionine and Leucine. The curves describe the contribution according to the position on a 19-residue segment.

Unfortunately, the experiments leading to these curves are very complex to realize, expensive and time-consuming and predictive systems issued from this work such as SCAMPI [24] do not allow a good distinction between SP and TM segments.

3.2 Our Proposal: *In Silico* Fine-Tuning of the Curves

Considering the difficulty of biological experiments that determine the scale of position-specific amino-acid contributions, we propose in this work, to determine *in silico* the scale curves. Our goal is to determine curves that enable a good discrimination between SP and TM segments.

Prediction System. We shall denote in the following by a one of the 20 amino acids that will be represented by a one-letter abbreviation, i.e. $a \in \{$A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, Y$\}$.

For each amino acid a, we have to determine a curve noted $C[a]$, defined on the interval $[1, 19]$ that represents the 19 positions in a segment that we consider as relevant for membrane insertion. Let us note that this position parameter may vary and for more generality we denote it by $l = 19$ in the following. Therefore, we use $C[a, j] = C[a](j)$ to denote the value of curve $C[a]$ for an integer j where j represents a position, $j \in [1, 19]$.

When we consider a sequence Seq of amino acids of length $l = 19$, we use the notation $Seq = < a^{(1)}a^{(2)} \dots a^{(l)} >$, where $a^{(j)}$ is the index of the amino acid in position j in the segment $a^{(j)}$ is therefore a letter in $\{$A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, Y$\}$). The hydrophobicity of sequence Seq is defined as follows as the average of $C[a^{(j)}, j]$ for j varying from 1 to l:

$$E(Seq) = \frac{\sum_{j=1}^{l} C[a^{(j)}, j]}{l}$$

Now given a longer sequence $Seq = < a^{(1)}a^{(2)} \dots a^{(n)} >$ of size $n > l$, a sliding window of fixed length l is scanned on the sequence and we define the hydrophobicity of the sequence as the maximum hydrophobicity value of a sub-sequence of length l denoted by:

$$E(Seq) = \max_{1 \leq k \leq n-l+1} \{E(Seq_k)\}$$

where $Seq_k = < a^{(k)} a^{(k+1)} \dots a^{(k+l-1)} >$.

The distinction between a SP and a TM segment is given by the value of $E(Seq)$ and a threshold τ. Seq is classified as a signal peptide when $E(Seq) < \tau$, and Seq is classified as a transmembrane segment when $E(Seq) > \tau$. So a set of curves $\left(C[a^i]\right)_{i=A}^{i=Y}$ determines a classification system for discrimination of SP and TM segments. Our goal is to optimize the 20 curves in order to obtain a good classification accuracy of SP and TM segments.

The quality of such a classification system can be evaluated by the *Area Under the ROC Curve* (AUC) [27]. A ROC[2] curve is obtained by selecting a series of thresholds τ and plotting sensitivity on the Y axis versus specificity on the X axis. The AUC gives the probability that a classifier will rank a randomly selected positive example higher than a randomly selected negative example.

Our problem is therefore an original and difficult optimization problem that can be solved by local search.

3.3 Local Search for Determination of the Curves $C[a]$

Local search approach is a metaheuristic method which is known to be an effective technique for solving computationally hard optimization problems [25]. Our local search algorithm moves in the space of candidate solutions to optimize the AUC of the associated classification system until a solution deemed optimal is found.

Representation of a Solution. In our problem, a candidate solution S is a set of 20 curves $\left(C[a^i]\right)_{i=A}^{i=Y}$ where each curve $C[a]$ is defined on $[1, 19]$. We suppose that each curve is symmetric, so for each amino acid a, $C[a, j] = C[a, 20 - j]$ and we decide to approximate each curve by an amino acid contribution function of the following polynomial form:

$$Y = \alpha x^2 + \beta \tag{1}$$

This contribution function is defined by two parameters: H_{middle}, the Y extremum of the function obtained for $j = 10$, and $H_{extremity}$ the value of the function for $j = 1$ and $j = 19$. So a curve is entirely fixed by a pair of values $(H_{extremity}, H_{middle})$. Figure 2 displays an example of a curve of the amino acid contribution function.

Initialization. To start the local search process, we consider curves based on the hydrophobic values given by the Kyte & Doolittle scales [10], which are known to be one of the best hydrophobic indexes [26]. Each scale considers that the index is independent of the position of the amino acid in the sequence and so in our formalization a constant curve h_i is associated to each amino acid a, so our initial solution is $S_0 = (C[a^i])_{i=A}^{i=Y}$ such that $\forall j \in [1, 19]$, $C[a^i, j] = h_i$.

[2] The term ROC comes from signal detection theory and means *Receiver Operating Characteristic*.

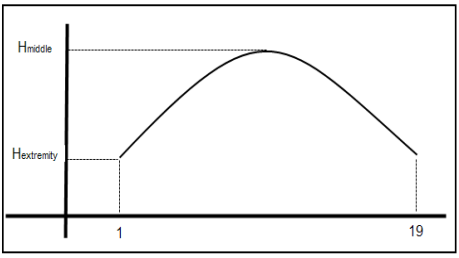


Fig. 2. Curve of contribution function defined by two parameters

Neighborhood. The local search process moves from solution S to an improving neighboring solution S' of the current solution S. To obtain a neighboring solution, a curve from S is randomly selected and then modified to generate a new curve. More precisely, let C denote our selected curve. As explained previously, C is entirely defined by a couple $(H_{extremity}, H_{middle})$. A new curve C' is generated by a new couple $(H_{extremity} + \epsilon, H_{middle} + \delta)$, with the constraints that $(\epsilon, \delta) \in \{-0.5, 0, 0.5\}^2$ or $(\epsilon, \delta) \in \{-0.3, 0, 0.3\}^2$. This leads to a total of 18 combinations. Removing the two trivial cases with $(\epsilon, \delta) = (0, 0)$, one can generate 16 new curves from an existing curve. In other words, each solution S has 16 neighboring solutions (for a chosen curve). These neighbor solutions are examined by the local search procedure at each iteration.

Evaluation Function. At each local search iteration, all candidate neighbors of the current solution S are evaluated. According to the principle of *steepest descent*, the best solution is chosen to replace the current solution S and the local search process is iterated from this new solution. The quality of a neighbor solution S' is assessed by the evaluation function AUC(S') that estimates the ability of the solution to obtain a suitable discrimination between SP and TM segments, using the classification system based on the curves of S'. Let us note that it is sufficient to compute the AUC of the classification system associated to a set of curves to evaluate a solution. The determination of a specific threshold τ is not necessary at this moment.

LSTranslocon Procedure. The general LSTranslocon procedure is composed of two main phases: the modeling phase and τ-calibration phase. The modeling phase consists in the determination of the best curves for the discrimination between signal peptide and transmembrane segments, as just explained. The stopping condition of this phase is obtained by evaluation of the proposed solutions on a validation dataset (see next Section). When the classification between SP and TM segments becomes stable - the AUC value remains unchanged - the search process ends and returns the best solution.

After the modeling phase, the τ-calibration phase is performed to determine the most appropriate threshold τ that permits to classify the protein sequences

as transmembrane sequences or signal peptides. This threshold corresponds to the best classifier of the ROC curve.

4 Experimentations and Discussion

4.1 Benchmark

The literature does not offer suitable ready-to-use dataset for our protein localization problem. Consequently, to assess the performance of our proposed method, we built a high quality benchmark database where the desired constituent sequences were extracted from the most recent version of Swiss-Prot database 57.8 (released on 22 September 2009) according to the following four steps: (1) The selected proteins are only those that are marked in the OC (organism classification) line as "eukaryota or eukaryotic", the eukaryotic proteins differ from prokaryotic proteins in particular in the addressing in the cell. (2) For the proteins obtained from the above step, we extract those which were marked as "signal peptide" and "transmen" in the FT (Feature Table) line, (3) We removed those which were annotated with uncertain terms for their signal peptide or transmem, such as "potential", "probable", or "by similarity" (4) For the resulting data set, the sequence identity is checked and analyzed by using the program CD-HIT [28], which produces a non-redundant dataset at the 50% sequence identity level.

By strictly following the above steps, we finally obtained a benchmark database for eukaryotic proteins. The database contains 5469 sequences with signal peptide and 798 transmembrane protein segments.

In order to equalize the sizes of the database between SP and TM segments, we randomly selected 6% from the dataset with SP. Thus, the final benchmark database contains 900 SP and 798 TM segments.

Now, from the selected data with signal peptide, we extracted the first 55 amino acids (because the maximum length of SP is 55 amino acids). Note that in our study we consider a SA as TM segment, for this reason we selected only the first TM or SA segment according to its annotations in Swiss-Prot. In the case where the selected segment has a length inferior to 19, we expanded the selected window to obtain a segment superior or equal to 19 amino acids.

The statistical distribution of amino acids in the benchmark database is presented in Figure 3. The figure shows that some amino acids are less represented than others in the data sequences.

This benchmark database is used to evaluate our approach by a process of cross-validation that involves 10 experiments. For each experiment, we build from the initial benchmark database three types of datasets: a *training* set, a *validation* set and a *test* set. The training set is used for the modeling phase (optimization of the curves) and the determination of the threshold τ. The validation set is used to determine the stopping condition and to avoid overfitting. And the test set is used to evaluate the classification accuracy of our prediction model. Each of these sets is constituted by randomly selecting from our initial

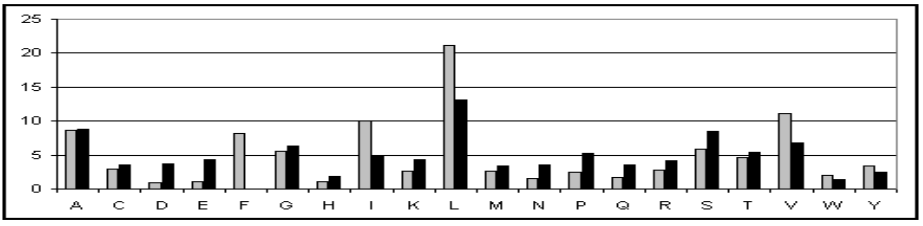


Fig. 3. Statistical distribution of each amino-acid in the dataset. The Y axis gives the frequency of each amino acid. The dark bars indicate the distribution for the data with SP segments while the grey bars show the distribution with TM segments (or SA).

database according to the following proportions: 60% for training data, 10% for validation data and 30% for test data.

The classification accuracy reported in the next Section is the averaged value calculated only on the *test* sets of the ten experiments.

4.2 Results and Discussion

This section reports the experimental results that assess our approach. The main focus of our approach is to determine a new hydrophobic scale that enables biological understanding of the insertion phenomena inside the translocon. Therefore, we first present a set of curves obtained by LSTranslocon and discuss them. Then, we demonstrate that our approach can lead to an effective predictor for the discrimination between SP and TM segments.

Figure 4 shows the 20 curves (one curve per amino acid) obtained when we apply our method to the benchmark database described in 4.1. We can remark that the shapes of the curves are quite different. These shapes suggest that some amino acids like Proline (P) or Methionine (M) facilitate more the insertion when they are embedded inside the membrane in the middle of the curve whereas other amino acids like Leucine (L) or Serine (S) prefer the interface positions at the extremities of the curves. The experimental curves given by Hessa et al [23] were obtained *in vitro*. The procedure is complex and time consuming. Our computational approach allows us to obtain similar curves much more rapidly. However, we observe that for some amino acids and especially those which are less frequent in our database, our curves have a different shape with respect to those suggested before. One possible explanation of this observation is that these amino acids do not have a great influence in the insertion process, which explains their lack of representation in the benchmark database and the difficulty to properly adjust their insertion curve. Now we turn our attention to the predictor system that results from our local search approach. We compare our prediction rates with those obtained by Phobius, one of the best performing prediction tools (see section 2). As these two programs do not completely share the same objectives, a fair comparison is difficult. For all that, we performed two experiments to evaluate the capabilities of these programs.

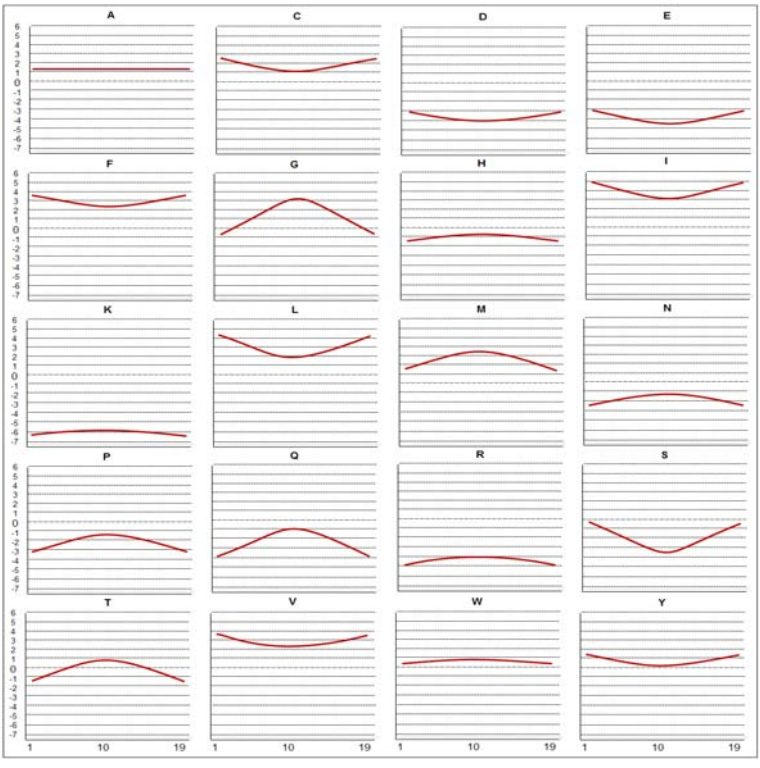


Fig. 4. Curves of insertion for each amino acid arbitrarily selected from one experiment. The curves describe the contribution according to the position on a 19-residue segment. The X axis shows the positions and the Y axis indicates the insertion index.

In the two experiments, LSTranslocon is trained according to the experimental protocol described in Section 4.1 and the Phobius predictor available on the web is used. The two programs are evaluated 10 times on the same test sets and the results are summarized in Table 1.

In the first experiment, the 10 test sets are generated as explained in Section 4.1. On these datasets, Phobius achieves about 84% average accuracy whereas LSTranslocon is limited to 80% (see results "Experiment 1" in Table 1). The Phobius web server accepts the whole protein sequence as input data whereas LSTranslocon requires only the TM or SP segments (see Section 4.1). The Phobius model takes advantage of the supplementary information. In fact, in its model, the transition between states are arranged such that the position of SP are located at the N-terminal region whereas TM segments are more often found later in the sequence.

To confirm this observation, we conduct a second experiment. This time, Phobius and LSTranslocon are tested on 10 restricted test sets : we only keep the TM segments located in the N-terminal regions from the previous 10 test

Table 1. Results of two experiments that compare our method LSTranslocon and Phobius. Each cell of the table indicates the *best classification accuracy* achieved by the corresponding classifier on the given test set.

Experiment 1			Experiment 2		
Data set	LSTranslocon	Phobius	Data set	LSTranslocon	Phobius
Test 1	0.79	0.85	Test 1'	0.80	0.82
Test 2	0.81	0.86	Test 2'	0.83	0.81
Test 3	0.82	0.84	Test 3'	0.81	0.81
Test 4	0.81	0.87	Test 4'	0.81	0.83
Test 5	0.79	0.85	Test 5'	0.81	0.81
Test 6	0.81	0.84	Test 6'	0.80	0.81
Test 7	0.81	0.87	Test 7'	0.82	0.84
Test 8	0.80	0.85	Test 8'	0.82	0.82
Test 9	0.80	0.84	Test 9'	0.81	0.81
Test 10	0.79	0.82	Test 10'	0.80	0.79
Average	0.80	0.84	Average	0.81	0.81

sets. We consider that a TM segment is not in the N-terminal region, if the start position of the TM segment is beyond the 30^{th} amino acid. The results "Experiment 2" show that the accuracy of Phobius decreases and is equal to the accuracy obtained with LSTranslocon. As we look for the curves which explain the insertion phenomena inside the translocon, we do not want to exploit the statistical bias concerning the TM and SP positions inside the protein sequence. Under this strong constraint, we observe that the results reported in this paper are as good as those obtained by Phobius. This demonstrates that our approach, which is quite new, has an interesting potential that we hope to improve in the near future.

5 Conclusion

In this paper, we have presented a new method based on the Local Search Approach for the discrimination of transmembrane segments and signal peptides. The method integrates the latest knowledge acquired in the biological field and presents the insertion curves in the membrane for each amino acid. LSTranslocon is evaluated with the ability to maximize the distinction between TM segments and SPs. These two characteristics ensure that our method constitutes a complete approach and gives a good explanation of insertion machinery.

The main advantage of such a predictive method is the chemically interpretable rules that will enable experts to understand biological phenomena. Furthermore, the proposed method can be applied to very large data, even whole proteome datasets.

Several improvements to the proposed method can be envisaged. One immediate possibility would be to study alternative functions to optimize the curves and to introduce more biological knowledge to provide more effective guidance

of the local search process. Another natural extension would be to reinforce the basic local search procedure by more powerful metaheuristics. Moreover, a next step will be to apply our approach on full sequences with the aim to localize the TM segment positions.

Acknowledgements

This research was partially supported by the region Pays de la Loire (France) with its "Bioinformatics Program" (2007-2009). The authors are grateful to the reviewers for their useful comments.

References

1. Mandon, E.C., Trueman, S.F., Gilmore, R.: Translocation of proteins through the Sec61 and SecYEG channels. Current Opinion in Cell Biology 21, 501–507 (2009)
2. Emanuelsson, O., Nielsen, H., Brunak, S., von Heijne, G.: Predicting subcellular localization of proteins based on their N-terminal amino acid sequence. J. Mol. Bio. 300, 1005–1016 (2000)
3. Horton, P., Park, K.-J., Obayashi, T., Fujita, N., Harada, H., Adams-Collier, C.J., Nakai, K.: WoLF PSORT: protein localization predictor. Nucleic Acids Research 35, W585–W587 (2007)
4. Viklund, H., Granseth, E., Elofsson, A.: Structural classification and prediction of reentrant regions in alpha-helical transmembrane proteins: Application to complete genomes. J. Mol. Biol. 361, 591–603 (2006)
5. Nugent, T., Jones, D.T.: Transmembrane protein topology prediction using support vector machines. BMC Bioinformatics 10 (2009)
6. Kaell, L., Krogh, A., Sonnhammer, E.L.L.: Advantages of combined transmembrane topology and signal peptide prediction - the Phobius web server. Nucleic Acids Research 35, W429–W432(2007)
7. Kall, L., Krogh, A., Sonnhammer, E.L.L.: A combined transmembrane topology and signal peptide prediction method. J. Mol. Biol. 338, 1027–1036 (2004)
8. Reynolds, S.M., Käll, L., Riffle, M.E., Bilmes, J.A., Noble, W.S.: Transmembrane topology and signal peptide prediction using dynamic bayesian networks. PLoS Comput. Biol. 4 (2008)
9. Viklund, H., Bernsel, A., Skwark, M., Elofsson, A.: SPOCTOPUS: a combined predictor of signal peptides and membrane protein topology. Bioinformatics 2, 2928–2929 (2008)
10. Kyte, J., Doolittle, R.F.: A simple method for displaying the hydropathic character of a protein. J. Mol. Biol. 157, 105–132 (1982)
11. Small, I., Peeters, N., Legeai, F., Lurin, C.: Predotar: A tool for rapidly screening proteomes for N-terminal targeting sequences. Proteomics 4, 1581–1590 (2004)
12. Hiller, K., Grote, A., Scheer, M., Munch, R., Jahn, D.: PrediSi: prediction of signal peptides and their cleavage positions. Nucleic Acids Research 32, W375–W379 (2004)
13. Bendtsen, J.D., Nielsen, H., von Heijne, G., Brunak, S.: Improved prediction of signal peptides: SignalP 3.0. J. Mol. Bio. 340, 783–795 (2004)
14. Tusnady, G.E., Simon, I.: The HMMTOP transmembrane topology prediction server. Bioinformatics 17 (2001)

15. Von Heijne, G.: Membrane protein structure prediction: Hydrophobicity analysis and the positive-inside rule. J. Mol. Bio. 225, 487–494 (1992)
16. Bagos, P.G., Liakopoulos, T.D., Hamodrakas, S.J.: Algorithms for incorporating prior topological information in HMMs: application to transmembrane proteins. BMC Bioinformatics 7 (2006)
17. Hua, S., Sun, Z.: Support vector machine approach for protein subcellular localization prediction. Bioinformatics 17, 721–728 (2001)
18. Garg, A., Bhasin, M., Raghava, G.P.S.: Support vector machine-based method for subcellular localization of human proteins using amino acid compositions, their order, and similarity search. J. Biol. Chem. 280, 14427–14432 (2005)
19. Klee, E.W., Sosa, C.P.: Computational classification of classically secreted proteins. Drug Discovery Today 12, 234–240 (2007)
20. Klee, E.W., Ellis, L.B.M.: Evaluating eukaryotic secreted protein prediction. BMC Bioinformatics 6 (2005)
21. Hessa, T., Kim, H., Bihlmaier, K., Lundin, C., Boekel, J., Andersson, H., Nilsson, I., White, S.H., von Heijne, G.: Recognition of transmembrane helices by the endoplasmic reticulum translocon. Nature 433, 377–381 (2005)
22. Hessa, T., White, S., von Heijne, G.: Membrane insertion of a potassium-channel voltage sensor. Science 307, 1427 (2005)
23. Hessa, T., Meindl-Beinker, N.M., Bernsel, A., Kim, H., Sato, Y., Lerch-Bader, M., Nilsson, I., White, S.H., von Heijne, G.: Molecular code for transmembrane-helix recognition by the Sec61 translocon. Nature 450, 1026-1030 (2007)
24. Bernsel, A., Viklund, H., Falk, J., Lindahl, E., von Heijne, G., Elofsson, A.: Prediction of membrane-protein topology from first principles. Proc. Natl. Acad. Sci. USA 105, 7177–7181 (2008)
25. Hoos, H., Stützle, T.: Stochastic Local Search: Foundations & Applications. Morgan kaufmam Publishers Inc., San Francisco (2004)
26. Bannai, H., Tamada, Y., Maruyama, O., Nakai, K., Miyano, S.: Extensive feature detection of N-terminal protein sorting signals. Bioinformatics 18, 298–305 (2002)
27. Fawcett, T.: ROC Graphs: Notes and Practical Considerations for Researchers. Technical report, HP Labs (2004)
28. Weizhong, L., Godzik, A.: Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences. Bioinformatics 22, 1658–1659 (2006)

A Replica Exchange Monte Carlo Algorithm for the Optimization of Secondary Structure Packing in Proteins*

Leonidas Kapsokalivas and Kathleen Steinhöfel

Department of Computer Science
King's College London
The Strand, London WC2R 2LS, United Kingdom

Abstract. We approach the problem of packing secondary structure fragments into low energy conformations with a local search optimization algorithm. Protein conformations are represented in a simplified off-lattice model. In that model we propose a move set that transforms a protein conformation into another in order to enable the use of local search algorithms for protein folding simulations and conformational search. The energy minimization problem behind protein folding is adapted to our model. Special care has been taken so that amino acids in a conformation do not overlap. The constraint of producing an overlap-free conformation can be seen as a objective that conflicts with the energy minimization. Thus, we approach protein folding as a two-objective problem. We employ a replica exchange Monte Carlo algorithm in combination to the proposed move set. The algorithm deals with the energy minimization problem while maintaining overlap-free conformations. Initial conformations incorporate experimentally determined secondary structure, which is preserved throughout the execution of local search. Our method produced conformations with a minimum RMSD of alpha-carbon atoms in the range of 4.71$\AA$ to 6.82$\AA$ for all benchmarks apart from one for which the value was 9.68$\AA$.

1 Introduction

Research on protein structure and folding has a history of over fifty years [19,23], but the problem of how proteins fold to their native state still remains unsolved. There is a continuous stream of both new theories and experimental work aiming at a final answer to this fundamental question, see [25,22,27], and [21] for a recent overview.

The research on structural proteomics goes along with attempts to solve the tertiary structure prediction problem through protein folding simulations. In accordance with Anfinsen's thermodynamic hypothesis that proteins settle into conformations of minimum energy, protein folding simulations aim at sampling the space of protein conformations in search of the global minimum. In order to speed up simulations, a variety of coarse-grained models has been introduced. In one of the most simplified models, namely the HP (Hydrophobic-Polar) model [9], the amino acid chain of a protein is embedded into a two- or three-dimensional rectangular lattice. In the HP model, the

* Research partially supported by EPSRC Grant No. EP/D062012/1.

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2010, LNCS 6023, pp. 146–157, 2010.

quality of an embedding is measured by the number of neighboring pairs (contacts) of hydrophobic amino acids in the lattice.

Simplified lattice models for protein folding, such as the HP model, have several disadvantages due to the prescribed positions of atoms in a lattice. Specifically, it is not possible to represent proteins accurately. Moreover, the energetic interactions are highly dictated by the geometry of the lattice. Subsequently, the energy functions employed in lattice models, can only consider a limited set of interactions. Off-lattice models overcome these limitations, since the atoms can be placed at points in a continuous space. Nevertheless, the extra degrees of freedom result in an enormous space of protein conformations, thus motivating the study of efficient ways to sample it.

Stochastic local search methods have been successfully applied to lattice protein folding simulations in order to sample the space of conformations in search of the energy minimum. Examples include tabu search [15,4], simulated annealing [1], replica exchange Monte Carlo [26] and population-based local search [14]. In order to employ a local search method, it is necessary to devise a neighborhood relation or a move set, namely a set of rules that transform a conformation into another. There are various kinds of move sets in off-lattice models. A move set usually alters the torsion angles of a protein's backbone. In [24], for instance, the authors employ a move set which performs a concurrent rotation of dihedral pair angles ϕ/ψ of two, four or six successive amino acids, while the rest of the angles remain unchanged. The latter move set is also used in [5] along with a knowledge-based move set which favors the sampling of specific dihedral pair values, observed in real proteins. Another move set operating on the torsion angles of a protein's backbone consists of the biased Gaussian steps in [11]. More complex move sets aim at rearranging a section of a molecular chain and then trying to bridge it back to the remainder of the chain. Such a move set is the ConRot algorithm [10] and its variants [17] as well as algorithms dealing with the loop-closure problem like in [6].

Solving protein folding based on the assembly of secondary structure fragments has been proposed in [7], where the authors employ constraint logic programming strategies. A similar idea has been proposed in [12], where the authors employ a simulated annealing-based local search algorithm. Exploiting secondary structure in practice is useful, provided that it can be predicted accurately and efficiently from the sequence information. The state-of-the-art methods for secondary structure prediction using sequence information, achieve an accuracy between 70% and 80%. For a comparative analysis of various prediction methods, the reader is refered to [20].

In this paper we first describe a coarse-grained off-lattice model which we then use to predict the tertiary structure of proteins starting from conformations with fixed secondary structure fragments. The fixed secondary structure fragments in our approach consist of α-helices and/or β-sheets only. The aim is to optimize the packing of those fixed secondary structure fragments via an energy minimization procedure. Towards this aim, we employ a replica exchange Monte Carlo sampling algorithm in combination to a move set especially developed for our off-lattice model. The moves are applicable only in the non-secondary structure parts of the conformation. Moreover, in order to avoid overlaps between the amino acids we introduce a penalty function, which prevents them from collapsing into each other such that the conformation remains

self-avoiding. In this way, one can realize protein folding as a multiobjective problem, since maintaining overlap-free conformations is a goal opposite to the energy optimization that leads to compact conformations as a result of the attractive nature of energetic interactions. Given the nature of energy function employed in our approach, we adapt our optimization strategy so as to handle two objectives. Our methods are tested for a set of α/β proteins and the results are presented in terms of RMSD (Root Mean Square Deviation) to the native structure.

Although the secondary structure remains unaffected throughout the local search, in a similar fashion to [7], our optimization strategy and model are fundamentally different from those in [7]. Also, our method differs from the one in [12], where moves are allowed in the secondary structure part and moreover the energy function is optimized for their specific off-lattice model. The aim of our paper is to demonstrate the idea of secondary structure assembly in a simplified model as a two-objective optimization problem.

2 The Off-Lattice Model

We consider a coarse-grained off-lattice model in which the backbone of a protein is represented as a trace of alpha-carbon atoms. The side chain groups are omitted. Specifically, each amino acid of the protein is represented as a sphere of radius b, that corresponds to the alpha-carbon atom. The centers of those spheres are connected into a chain with bonds of fixed length $l = 3.8\mathring{A}$. Let three successive amino acids $i-1$, i and $i+1$, defining a plane $\mathcal{A}$. The angle formed by those amino acids on the plane $\mathcal{A}$, can take any value in the interval $[\frac{74*\pi}{180}, \ldots, 2\pi - \frac{74*\pi}{180}]$. This implies that $i-1$ and $i+1$ cannot come closer than $c = 2*l*\sin(\frac{37*\pi}{180}) = 4.57\mathring{A}$. We consider half of this value as the radius $b = 2.28\mathring{A}$, mentioned above. The rotation angle of the plane $\mathcal{A}$ around the axis crossing the $i-1$-th and the i-th amino acid (or the i-th and the $i+1$-th) can take any value in the interval $[0, \ldots, 2\pi]$. We should note that this model could be extended to include side chains as separate spheres connected to the backbone with a flexible bond. Bond lengths have been normalized to a unit length, in order to simplify numerical calculations in the experiments (i.e. smaller numbers yield more stable numerical calculations). Thus, $l = 3.8\mathring{A} \rightarrow l = 1$ and $c = 4.57\mathring{A} \rightarrow c = \frac{4.57}{3.8} = 1.2$. Secondary structure is represented in the same fashion as the remainder of the chain, namely as a trace of alpha-carbon atoms. In Section 4, we further explain how to produce initial conformations which incorporate secondary structure from PDB (Protein Data Bank) files.

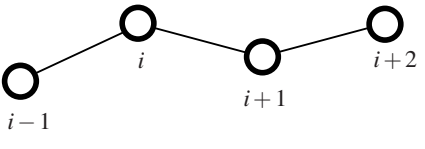


Fig. 1. A conformation without side chains

2.1 Problem Formulation

In this section we formulate the protein structure prediction problem in the off-lattice model introduced above. The aim of our approach is to fold the secondary structure fragments of a protein into structures of minimum energy, while preserving the connectivity of the chain in an overlap-free conformation. Let us now explain how those two constraints are expressed and how the energy function works in our off-lattice model.

The energy function we employ is a matrix $\mathcal{M}$ of all pairwise interactions among the 20 amino acids. The choice for matrix $\mathcal{M}$ is the empirical contact potential in [2]. According to [18] the cut-off threshold d to form a non bonded contact between the centers of two amino acids is 6.5Å. For a conformation ϕ, the following function has to be minimized:

$$E(\phi) = \sum_{i=0}^{n} \sum_{j=i}^{n} \mathcal{M}(\phi(i), \phi(j)) \cdot \Delta(i,j), \tag{1}$$

where n is the length of the protein, $\phi(i)$ returns a number from 1 to 20 according to the type of the i-th amino acid and $\Delta(i,j) = 1$ if $||r_i - r_j|| \leq d$ meaning that the euclidean distance between the centers of the i-th and the j-th amino acid is less than or equal to d and $\Delta(i,j) = 0$ otherwise. The minimization is subjected to the constraint of connectivity, thus $||r_i - r_{i+1}|| = l$,$i = 0, \ldots, n-2$. It is also subjected to the constraint of an overlap-free conformation, which prevents the chain from crossing itself as well as any overlaps among the spheres representing the amino acids, thus we have $||r_i - r_j|| \geq c$ for any pair i, j where $|i-j| \geq 2$. These means that no overlaps are considered between two successive amino acids i and $i+1$ since a choice for radius $c > \frac{l}{2}$ would always violate the constraint. We should stress the fact that the assigned secondary structure fragments are also a set of constraints that could be expressed as a series of torsion angles. In our case, those constraints are met by excluding secondary structure fragments from being modified by the move set, as we explain in Section 3.1.

3 Methods

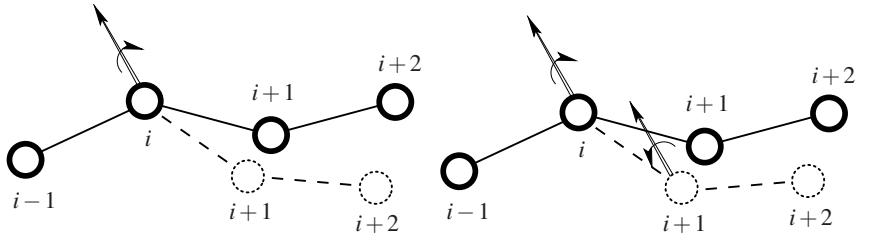
In this section we outline the optimization method we used to tackle the problem described in Section 2.1. Our method is based on a Monte Carlo simulation which has been adapted so as to satisfy the constraints we described. In order to facilitate the application of a Monte Carlo algorithm or any other local search method, we devised a move set for our off-lattice model.

3.1 The Move Set

The move set consists of two types of moves, namely the *single* and the *double rotation*. In both types of moves, an amino acid i is chosen uniformly randomly among all candidate amino acids. Then, an axis is chosen such that it crosses the conformation at the center of amino acid i. The direction of the axis is calculated as follows: Two angles θ and ϕ are uniformly randomly picked in the interval $[0, \cdots, 2\pi]$. The choice of one angle is independent from the choice of the other angle. The direction of the axis is determined by the vector $\alpha = (x, y, z)$

$$x = sin(\theta) * cos(\phi)\ y = sin(\theta) * sin(\phi)\ z = cos(\theta), \tag{2}$$

namely the translation of polar into cartesian coordinates.



(a) The single rotation move. (b) The double rotation move.

Fig. 2. The solid line represents the initial conformation. The dotted line represents the final one.

The *single rotation* consists of a rigid rotation by an angle ρ around vector α. The rigid rotation affects the position of amino acids $(i+1,\cdots,n)$, while the rest of the conformation remains the same. See Figure 2(a) for an illustration. The angle ρ is drawn from a Gaussian distribution of zero mean and variance equal to $\beta * \pi/180$ radians, $\beta = 10$ for our experiments. Positive values of the angle ρ denote a clockwise rotation around α, while negative values an anti-clockwise one.

The *double rotation* consists of two successive single rotations on the i-th and the $i+1$-th amino acid respectively and aims at improving the locality of the move set. The first rotation works in the same fashion as shown in Figure 2(a). The second rotation operates on the next amino acid, namely $i+1$, and performs a rigid rotation around the same vector α by an angle $-\rho/2$. The direction of the second rotation is opposite to the direction of the first one, namely if the first rotation is clockwise, then the second is anti-clockwise and vice-versa. See Figure 2(b) for an illustration.

It is evident that both types of rotation preserve the connectivity of the conformation. In Section 3.2 we will examine our strategy of handling the constraint of an overlap-free conformation. Secondary structure constraints are imposed by restricting the available choices of amino acids where moves can be applied.

Let $\mathcal{F} = f_1, \ldots f_k$ be a set of k secondary structure fragments. For each fragment f_r consisting of amino acids $[i_r, \ldots, j_r]$, a *single* or a *double* rotation can only be applied at either the i_r-th or the j_r-th amino acid. In case a double rotation is applied at the i_r-th amino acid, then the first rotation is applied at the i_r-th and the second rotation at the j_r-th amino acid. All other amino acids outside set $\mathcal{F}$ can be chosen for either a single or a double rotation.

3.2 The Monte Carlo-Based Optimization Strategy

The pairwise energy functions we employ in our approach consist mostly of attractive and rather few repulsive interactions. Therefore, low energy conformations will be compact. On the other hand the constraint of an overlap-free conformation, as described in Section 2.1, prevents the centers of amino acids from being packed closer than a certain threshold. Thus, this constraint can be viewed as a separate objective which conflicts with the energy minimization.

We have examined two conventional Monte Carlo-based approaches for this problem. The first approach optimizes the energy and only allows transitions to conformations which meet the constraints. Every time a move is applied, the algorithm checks the amino acid overlap constraints and rejects the resulting conformation in case the constraints are not met. However, this results in the algorithm getting trapped in conformations, for which few transitions are possible to other overlap-free conformations. Thus, only a limited amount of conformations is sampled and a lot of computational effort is spent on searching for transitions to overlap-free conformations, rather than lower energy ones.

The second approach optimizes a combined fitness function $H(\phi) = w * E(\phi) + (1 - w) * Z(\phi)$, i.e. a weighted sum of a penalty function $Z(\phi)$ for overlaps and the energy function $E(\phi)$. The energy function is taken from Equation 1. The penalty function is defined in Equation 3.

$$Z(\phi) = \sum_{i=0}^{n} \sum_{j=i}^{n} [(\mathcal{D}(i,j) - c) \cdot \Delta'(i,j)]^2, \tag{3}$$

In Equation 3 we consider $\Delta'(i,j) = 1$ if $||r_i - r_j|| \leq c$ and $|i - j| \geq 2$. Otherwise $\Delta'(i,j) = 0$. Also, $\mathcal{D}(i,j)$ is the euclidean distance $||r_i - r_j||$ between the i-th and the j-th amino acid. Intuitively, the penalty function $Z(\phi)$ expresses the square error of a conformation with overlaps from a slightly modified overlap-free conformation. The optimal value of penalty is zero. We observed that the second Monte Carlo approach minimizes only one of the objectives, depending on the value of weight w, while the other objective increases arbitrarily. Moreover, it is hard to accurately express the interdependence of the energy and the penalty as a linear combination of the two objectives. Therefore, we devised a Monte Carlo-based optimization strategy to optimize each objective separately. The interplay of repulsive and attractive interactions in a single objective could be expressed with a more elaborate function, such as a force-field.

Our optimization strategy is outlined in Algorithm 1. This strategy performs a Monte Carlo simulation in order to sample minimum energy conformations for which the penalty function is bounded by a threshold $\mathcal{P}_{thres}$. Transitions to conformations with a penalty value above this threshold are rejected. Consequently, the penalty function is treated as a constraint. Setting $\mathcal{P}_{thres} = 0$ is equivalent to checking the overlapping constraints and rejecting the resulting conformation, i.e. the first Monte Carlo approach we described above. Setting $\mathcal{P}_{thres}$ to a non-zero value, allows the Monte Carlo algorithm to sample a larger portion of the search space. Relaxing the penalty constraint, though, results in sampling conformations that are not overlap-free anymore. We deal with this problem by replacing the overlap threshold distance c in the penalty function (Equation 3) with $c' > c$, thus increasing the contribution of overlaps to the penalty.

To sum up, in our optimization strategy the constraint of an overlap-free conformation is realized as a separate penalty objective. The penalty objective is then treated as a constraint. This constraint is relaxed and on the same time penalty values are assigned in a more strict fashion. From the viewpoint point of fitness landscapes, allowing $\mathcal{P}_{thres} > 0$ results in a smoother fitness landscape, since a transition between two overlap-free conformation is interpolated by transitions to overlapping conformations.

Algorithm 1. Monte Carlo-based Optimization

```
Start with the initial conformation ϕ and i=1
while i = 1 ≤ MaxIter do
  Generate a random real number r ∈ [0,...,1]
  if r ≥ 0.5 then
    Apply a single rotation to obtain a neighbor ϕ' of ϕ
  else
    Apply a double rotation to obtain a neighbor ϕ' of ϕ
  end if
  Calculate the penalty Z(ϕ')
  if Z(ϕ') ≤ P_thres then
    i=i+1
    Calculate the contact energy E(ϕ')
    if E(ϕ') ≤ E(ϕ) then
      Accept the transition from to ϕ to ϕ' and set ϕ = ϕ'
      Update the best conformation seen in case E(ϕ') ≤ E(ϕ_best)
    else
      Generate a random number q ∈ [0,...,1]
      if q < min(1, e^((E(ϕ')−E(ϕ))/T)) then
        Accept the transition from to ϕ to ϕ' and set ϕ = ϕ'
      else
        Reject the transition from to ϕ to ϕ'
      end if
    end if
  end if
end while
```

3.3 Replica Exchange Monte Carlo

Replica exchange Monte Carlo has been proposed by various researchers as an extension to the conventional Monte Carlo approach (see [13] for a review). The replica exchange Monte Carlo algorithm maintains an array of K conformations (replicas) and a range of temperatures $T_1 \leq T_2 \leq, \ldots T_K$, where T_i corresponds to the i-th position of the array. For each conformation ϕ_i, a separate Monte Carlo simulation is performed at temperature T_i and is terminated after a number of steps, $MCsteps$. The Monte Carlo simulation consists of Algorithm 1. At the end of $MCsteps$, neighboring conformations ϕ_i and ϕ_{i+1} exchange positions in the array with a probability $P_{ex} = min\{1, exp((1/T_i - 1/T_{i+1}) * (E(\phi_i) - E(\phi_{i+1})))\}$. The replica exchange algorithm is terminated after $REMCsteps$ rounds of Monte Carlo simulations each of which is followed by an exchange of conformations.

4 Experiments

In our experiments with the Monte Carlo-based optimization algorithm we have selected the following benchmarks: 1CTF and 1R69 from [12], 1ENH, 1YPA and 2IGD from [7] and 1RHX and 1S12 from the CASP6 experiment.

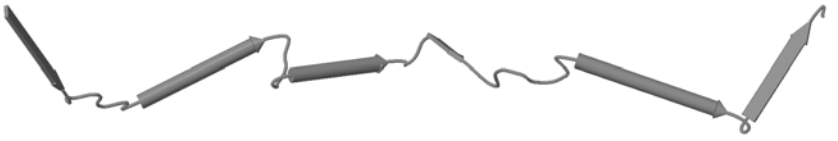


Fig. 3. Initial conformation for 1CTF

The aim of the experiments is to show that the move set described in Section 3.1 combined with the replica exchange Monte Carlo optimization algorithm is able to efficiently sample the conformational space and produce conformations of sufficiently low RMSD values. The experimental protocol involves 10 independent runs of replica exchange Monte Carlo for each benchmark, starting from an unfolded conformation. An unfolded conformation for each benchmark is produced by arranging the secondary structure fragments in a chain that closely resembles a straight line conformation. The secondary structures are extracted from the PDB file that contains the native conformation for a particular benchmark. The assigned secondary structure fragments include only α-helices and β-sheets. Note that the non-secondary fragments of a conformation are unfolded but not arranged in a perfect straight line. See Figure 3 for an example of an initial conformation. The number of Monte Carlo steps was set to $MCsteps = 1000$ for all benchmarks. The number of rounds $REMCsteps$ along with the number of replicas vary for a particular benchmark and they are presented in Table 1.

The maximum temperature T_{max} is the same for all benchmarks. The rest of the temperatures are exponentially distributed according to the following formula $T_i = T_{max} * exp(i/K - 1)$, such that $T_K = T_{max}$ where K is the number of replicas.

The choice of $\mathcal{P}_{thres}$ value is calibrated through long experiments of $1,000,000$ iterations of Algorithm 1 for a benchmark of 68 amino acids (1CTF). Since the length l and the overlap threshold c are normalized, the values of $\mathcal{P}_{thres}$ in the experiments are normalized as well. As mentioned in Section 3.2 we use a larger overlap threshold c' in Equation 3; that was set to $c' = 1.42$. Starting from a relatively high value for $\mathcal{P}_{thres}$, we monitor the average distance between overlapping amino acids d_{av} in the energetically best conformation at the end of an experiment. Then, we decrease $\mathcal{P}_{thres}$ and perform another experiment. We stop at a $\mathcal{P}_{thres}$ value where $d_{av} \approx c$. The normalized value found by this procedure was $\mathcal{P}_{thres} = 3$.

Long experiments of a 68 amino acid benchmark, showed that in the folded conformation with penalty value $Z = 3$ for $c' = 1.42$, there are 82 overlaps but the average distance between overlapping amino acids is $d_{av} = 1.28$. Only 20 of these overlaps correspond to distances smaller than the desired $c = 1.2$ which yields a total penalty value of 0.5. Although the penalty threshold should be adapted according to the length of the protein, we used $\mathcal{P}_{thres} = 3$ for our set of benchmarks.

4.1 Results

In Table 1 we summarize the replica exchange Monte Carlo based results without an optimization of the penalty function, i.e. the penalty value of the resulting conformations is 3. The energetically best conformation is collected from each of the 10 independent

Table 1. Summary of the Monte Carlo-based optimization results

PDB id.	Length	Best conformation RMSD		Number of replicas	REMCsteps
		Atoms	RMSD(Å)	Total RMSD(Å)	
1CTF	68	62	4.71	7.2	8	200
1R69	63	59	6.82	7.72	8	200
		58	6.63			
1ENH	54	54	5.11	5.115	8	150
1YPA	64	59	6.73	8.14	8	200
2IGD	61	56	5.77	7.38	8	200
1RHX	87	85	9.31	9.68	10	250
1S12	94	81	6.06	14.66	10	250
		80	5.89			

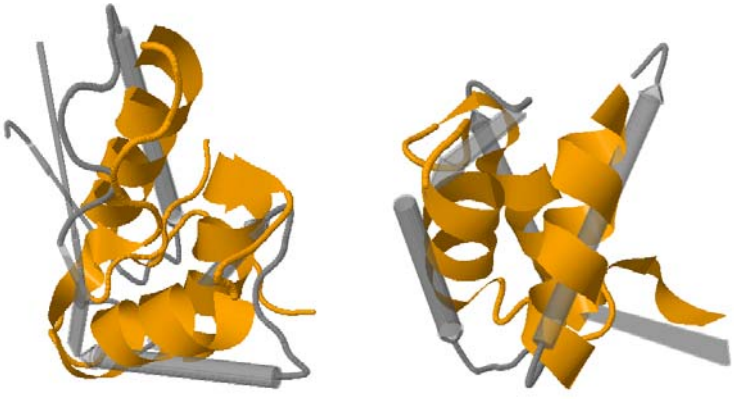
*Atoms stands for the number of C-alpha carbon atoms in the alignment.

runs. For a particular benchmark and a specific energy function, all 10 conformations produced, are structurally aligned to the native conformation, such that their RMSD is minimized. The RMSD, namely the root mean square deviation, is a measure of similarity between the predicted and the native conformation and it is given by the following formula: $\sqrt{\frac{1}{N}\sum_{i=0}^{N}\delta_i^2}$, where δ_i is the distance between the corresponding alpha-carbons in the two conformations. The alignment is performed by using the PyMOL software [8]. We should note that PyMOL allows the exclusion of some amino acids from being aligned, in case they contribute to an increased RMSD. Thus, in Table 1 we also present the alignment for subset of amino acids, that yield a better RMSD value than the alignment of the whole conformation.

The results show that, apart from 1RHX, for every other benchmark at least one out of 10 conformations produced, has a similarity between 4.71Å and 6.82Å. Although the total similarity score for that particular conformation is outside that range, only a small portion of alpha-carbon atoms is excluded from the low RMSD alignments. Thus, the subset of atoms having low RMSD values, define a conformation which captures the correct packing of secondary structure. See Figures 4(a) and 4(b). Instead of performing an alignment, more complex energy functions can be employed to evaluate the conformations produced and to rule out the ones which do not capture the correct packing of secondary structure fragments.

In Table 2 we present a comparison of our method to the methods in [7] and [12]. Our method produces better conformations for all benchmarks, compared to the best conformations presented in [7]. A comparison to the method in [12] is not directly possible, since the conformations produced in the latter approach include all backbone atoms, while our conformations include only the alpha-carbon atoms. Hence, the alignment scores to the native conformation are not entirely compatible. It is evident though, that the method in [12] produced better conformations than our method.

Better accuracy in the results from [12] is due to a more elaborate model of representating conformations, where all backbone atoms are taken into account. Another reason is that the energy function used in [12] has been optimized through a perceptron



(a) 1CTF (b) 1ENH

Fig. 4. The transparent is the native and the solid is the superimposition of the predicted conformation with the best RMSD

Table 2. Comparison of our method to the method in Dal Palu et al. [7] in terms of the best conformations obtained with respect to RMSD.

	1ENH	1YPA	2IGD
Our Method	5.11 (all alpha-carbons)	6.73 (59 atoms) 8.14 (all alpha-carbons)	5.77 (56 atoms) 7.38 (all alpha-carbons)
Dal Palu et al.	8.6 (45 atoms) 9.9 (all alpha-carbons)	9.8 (41 atoms) 12.9 (all alpha-carbons)	15.0 (54 atoms) 16.9 (all alpha-carbons)
	1CTF	1R69	
Our Method	4.71 (62 atoms) 7.2 (all alpha-carbons)	6.63 (58 atoms) 7.72 (all alpha-carbons)	
Hoang et al.	2.94 (all backbone atoms)	4.21 (all backbone atoms)	

*Atoms stands for the number of C-alpha carbon atoms in the alignment.

learning procedure. Benchmark 1CTF happens to be in the training set for perceptron and this might favour the quality of prediction. Finally, our method generated 10 candidate conformations (decoys) for each benchmark, while the authors in [12] generated 50. The advantage of our method over the method in [12] is that the simplicity of our model for representating protein conformations, allows cheaper calculations of the move set's operations and steric clashes.

A single run of our method with the parameters given in Section 4 requires around 1.2h for benchmarks of length 54-68 and 1.8 h for the two longest benchmarks on an Intel© Xeon E5440 machine and a GNU Octave© ver. 3.0.5 implementation. The method in [7] requires 10h for each of the following 1ENH, 1YPA and 2IGD compared to around 12h for our method. The runtime of the method in [12] is not provided.

5 Conclusions

In this paper we approach the problem of optimizing the packing of secondary structure fragments in proteins. We showed how this problem can be realized as a two-objective optimization problem and we have devised a stochastic local search method to deal with it. Our method combines a novel move set and a replica exchange Monte Carlo algorithm. Experimental results show that our method produces conformations which capture the topology of secondary structure in the native conformation. Those conformations can be used to reconstruct the backbone of a protein, using tools such as SABBAC [16]. Subsequently, conformations can be extended to all atom representations by placing side chains with a tool such as [28]. All atom conformations can be further refined to increase the similarity to the native conformation. Hence, our method can serve as the first part of a hierarchical approach to protein folding, that starts from a coarse-grain model and progresses to more elaborate models.

Future research will focus on applying the current method to an extended version of the current off-lattice model that incorporates side chains. Another extension could consider the packing of other structural fragments, such as elements of a structural alphabet (see [3]). Extending the method to longer proteins is another challenging task, since the number of possible topology packings would increase drastically with an increasing number of fragments. In that case, we need to limit the number of different topologies explored or employ a population-based search algorithm, such as a genetic algorithm. The current model and move set could be a testing ground for heuristic search methods in general.

References

1. Albrecht, A.A., Skaliotis, A., Steinhöfel, K.: Stochastic protein folding simulation in the three-dimensional HP-model. Computational Biology and Chemistry 32(4), 248–255 (2008)
2. Berrera, M., Molinari, H., Fogolari, F.: Amino acid empirical contact energy definitions for fold recognition in the space of contact maps. BMC Bioinformatics 4(1), 8 (2003)
3. Camproux, A.C., Gautier, R., Tuffry, P.: A Hidden Markov Model Derived Structural Alphabet for Proteins. Journal of Molecular Biology 339(3), 591–605 (2004)
4. Cebrián, M., Dotú, I., Hentenryck, P., Clote, P.: Protein Structure Prediction on the Face Centered Cubic Lattice by Local Search. In: AAAI, pp. 241–246. AAAI Press, Menlo Park (2008)
5. Chen, W.W., Shick Yang, J., Shakhnovich, E.I.: A knowledge-based move set for protein folding. Proteins: Structure, Function, and Bioinformatics 66(3), 682–688 (2007)
6. Coutsias, E., Seok, C., Jacobson, M., Dill, K.A.: A kinematic view of loop closure. Journal of Computational Chemistry 25(4), 510–528 (2004)
7. Dal Palù, A., Dovier, A., Fogolari, F.: Constraint Logic Programming approach to protein structure prediction. BMC Bioinformatics 5, 186 (2004)
8. DeLano, W.L.: The PyMOL Molecular Graphics System. DeLano Scientific, Palo Alto (2002)
9. Dill, K.A., Bromberg, S., Yue, K., Chan, H.S., et al.: Principles of protein folding-a perspective from simple exact models. Protein Science 4(4) (1995)
10. Dodd, L.R., Boone, T.D., Theodorou, D.N.: A concerted rotation algorithm for atomistic Monte Carlo simulation of polymer melts and glasses. Molecular Physics: An International Journal at the Interface Between Chemistry and Physics 78(4), 961–996 (1993)

11. Favrin, G., Irback, A., Sjunnesson, F.: Monte Carlo update for chain molecules: Biased gaussian steps in torsional space. The Journal of Chemical Physics 114(18), 8154–8158 (2001)
12. Hoang, T.X., Seno, F., Banavar, J.R., et al.: Assembly of protein tertiary structures from secondary structures using optimized potentials. Proteins: Structure, Function, and Genetics 52(2), 155–165 (2003)
13. Iba, Y.: Extended Ensemble Monte Carlo. International Journal of Modern Physics C 12, 623 (2001)
14. Kapsokalivas, L., Gan, X., Albrecht, A.A., Steinhöfel, K.: Two Local Search Methods for Protein Folding Simulation in the HP and the MJ Lattice Models. In: Elloumi, M., Küng, J., Linial, M., Murphy, R.F., Schneider, K., Toma, C. (eds.) BIRD 2008. CCIS, vol. 13, pp. 167–179. Springer, Heidelberg (2008)
15. Lesh, N., Mitzenmacher, M., Whitesides, S.: A complete and effective move set for simplified protein folding. In: Proceedings of RECOMB 2003, pp. 188–195. ACM, New York (2003)
16. Maupetit, J., Gautier, R., Tuffery, P.: SABBAC: online Structural Alphabet-based protein BackBone reconstruction from Alpha-Carbon trace. Nucl. Acids Res. 34(2), W147–W151 (2006)
17. Mavrantzas, V., Boone, T., Zervopoulou, E., Theodorou, D.: End-Bridging Monte Carlo: A Fast Algorithm for Atomistic Simulation of Condensed Phases of Long Polymer Chains. Macromolecules 32(15) (1999)
18. Miyazawa, S., Jernigan, R.L.: Residue - Residue Potentials with a Favorable Contact Pair Term and an Unfavorable High Packing Density Term, for Simulation and Threading. Journal of Molecular Biology 256(3), 623–644 (1996)
19. Pauling, L., Corey, R.B.: Stable Configurations of Polypeptide Chains. Proceedings of the Royal Society of London. Series B, Biological Sciences 141(902), 21–33 (1953)
20. Robles, V., Larraaga, P., Pea, J.M., Menasalvas, E., et al.: Bayesian network multi-classifiers for protein secondary structure prediction. Artificial Intelligence in Medicine 31(2), 117–136 (2004)
21. Rose, G., Fleming, P., Banavar, J., Maritan, A.: A backbone-based theory of protein folding. PNAS 103(45), 16623–16633 (2006)
22. Saibil, H.R., Ranson, N.A.: The chaperonin folding machine. Trends in Biochemical Sciences 27(12), 627–632 (2002)
23. Scheraga, H.A.: Protein Structure. Academic Press, New York (1961)
24. Shimada, J., Kussell, E., Shakhnovich, E.I.: The folding thermodynamics and kinetics of crambin using an all-atom Monte Carlo simulation. Journal of Molecular Biology 308(1), 79–95 (2001)
25. Sridhar, G., Goldstein, R.A.: On the thermodynamic hypothesis of protein folding. PNAS 95(10), 5545–5549 (1998)
26. Thachuk, C., Shmygelska, A., Hoos, H.: A replica exchange Monte Carlo algorithm for protein folding in the HP model. BMC Bioinformatics 8(1), 342 (2007)
27. Weinkam, P., Zong, C., Wolynes, P.G.: A funneled energy landscape for cytochrome-c directly predicts the sequential folding route inferred from hydrogen exchange experiments. PNAS 102(35), 12401–12406 (2005)
28. Xu, J., Berger, B.: Fast and accurate algorithms for protein side-chain packing. J. ACM 53(4), 533–557 (2006)

Improving Multi-Relief for Detecting Specificity Residues from Multiple Sequence Alignments

Elena Marchiori

Radboud University Nijmegen, The Netherlands
elenam@cs.ru.nl

Abstract. A challenging problem in bioinformatics is the detection of residues that account for protein function specificity, not only in order to gain deeper insight in the nature of functional specificity but also to guide protein engineering experiments aimed at switching the specificity of an enzyme, regulator or transporter. The majority of the state-of-the art algorithms for this task use multiple sequence alignments (MSA's) to identify residue positions conserved within- and divergent between- protein subfamilies. In this study, we focus on a recent method based on this approach called multi-RELIEF. We analyze and modify the two core parts of the method in order to improve its predictive performance. A parametric generalization of the popular RELIEF machine learning algorithm for weighting residues is introduced and incorporated in multi-RELIEF. The ensemble criterion of multi-RELIEF for merging the weights of multiple runs is simplified. Finally, the method used by multi-RELIEF for exploiting tertiary structure information is modified by incorporating prior information describing the confidence of the original scores assigned to residues. Extensive computational experiments on six real-life datasets show improvement of both robustness and detection capability of the new multi-RELIEF over the original method.

1 Introduction

Many homologous protein families have a common biological function but different specificity towards substrates, ligands, effectors, proteins and other interacting molecules. All these interactions require a certain specificity. Identifying crucial residues for this specificity is important for understanding the nature of functional specificity, for planning experiments on functional analysis or protein redesign, and for guiding point mutations aimed at switching the specificity of an enzyme, regulator or transporter.

In order to detect specificity residues, advanced computational techniques are used, because of a great variety of functional specificities observed in nature and the vast amount of protein sequence data.

Many algorithms have been proposed in recent years for this task (e.g., [1,2,4,5,6,8,10,13,22,23]). Most of them employ information-entropy related scoring functions [18] to rank residue positions according to the association with the subfamilies (see for instance the overview contained in [21]). Many methods

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2010, LNCS 6023, pp. 158–169, 2010.

require either a predefined subdivision of the MSA into classes, while unsupervised methods induce also a grouping during execution. The SDPpred method [10] uses mutual information to identify residue positions in which amino acid distributions correlate with the sub-family grouping [14]. In [9] the authors extended this approach to the problem of predicting protein functional sites. The Two-entropies analysis algorithm (TEA) [23] creates a 2-dimensional plot of residue conservation in terms of Shannon entropy at both superfamily and subfamily level. Recently, another method based on correlated mutation analysis [12] has been introduced to determine networks of functionally related residues in enzymes that upon mutation influence enzyme specificity and/or activity. The TreeDet approach [4] contains three algorithms for detecting so-called tree-determinant residues from an un-partitioned MSA. The Sequence Harmony (*SH*) method [5,16] scores compositional overlap between two user-specified groups. In a recent work [3], state-of-the-art methods for specificity residue detection, including *MR*, have been experimentally compared. An ensemble approach that combines predictions of the three best performing methods was used to identify new potential specificity determining sites.

In this paper we focus on a recent method for this task is multi-RELIEF (*MR*) [22], which identifies specificity residues from a given MSA and predefined multiple classes using 'local' conservation properties. *MR* employs an ensemble approach based on a machine learning algorithm for feature weighting, called RELIEF [11], that it applied multiple times to pairs of classes. Weights resulting from multiple runs are then merged.

The merging criterion assigns equal (best) score to those residues yielding perfect discrimination of at least two classes, that is, having complete within-class conservation and between-class divergence. This is undesirable, because residues discriminating well many pairs of classes should be considered more relevant than those discriminating only one pair of classes. Furthermore, RELIEF assigns equal (zero) score to residues that are either fully conserved or fully divergent. However, fully conserved residues are more relevant than fully divergent ones with respect to protein functionality. We tackle these two drawbacks of multi-RELIEF by introducing a parametric generalization RELIEF, that incorporates multiplicative factors in the formula used by RELIEF to weight residues. These factors are used to bias the weight of each residue, depending on its conservation composition in the protein sub-families considered when computing its weight. Furthermore, we replace the criterion used for merging weights with the simpler one that ensembles weights by means of their average. We call the resulting method *new-MR*.

Next, we refine the criterion used in *MR*, here called *3D*, for boosting the scores assigned to residues using information on tertiary structure, when available. Specifically, we incorporate prior information, in the form of a scaling factor associated to the original scores, describing the confidence assigned by the user to these scores. Then the new score is computed as the mean of the scaled original score and the average of the original scores of the 3D neighbors. We call the resulting criterion *new-3D*.

To assess the effectiveness of the resulting method thoroughly, six experimentally determined benchmark sets are considered, taken from five widely studied protein families: G protein-coupled receptors (GPCRs), the LacI family of bacterial transcription factor, the Ras-superfamily of small GTP-ases, the MIP-family of integral membrane transporters and the Smad family of transcription factors. We compare experimentally *MR*, and its modifications obtained by using either *new-MR* or *new-3D*, or both. Using ROC curves we show that *new-MR* identifies specificity residues as good as or better than *MR*. Moreover, when using *new-3D* overall robustness and improved predictive performance is achieved.

The rest of the paper is organized as follows. Section 2 describes the multi-RELIEF approach. In Section 3 we introduce the new methods. Section 4 contains the experimental analysis. Finally, in Section 5 we briefly summarize the results and point to future work.

2 Setting the Stage: The Multi-RELIEF Approach

Multi-RELIEF uses as core procedure RELIEF [11], a successful two-class feature weighting algorithm, and an ensemble approach for handling multiple classes, based on random sub-sampling pairs of classes. Random sampling of pairs of classes is mainly employed for efficiency reasons, while random sub-sampling of sequences is applied for handling unbalanced classes as well as for gaining efficiency.

Multi-RELIEF [22] is illustrated below in pseudo-code, where nr_positions denotes the total number of positions in the considered MSA. The algorithm works as follows. Multiple runs (*nr_iter*) of RELIEF are performed. At each run i, first two classes are randomly selected. Next, *nr_sample* sequences from each class are randomly selected. Finally, RELIEF is applied to the resulting two classes, yielding an output vector W_i.

```
Multi-RELIEF
input:  X1,..,Xm (m classes of an MSA), nr_iter, nr_sample
output: multi_W (weights assigned to residues)
for i=1: nr_iter
    select randomly two classes Xj, Xk
    X = select randomly nr_sample sequences from Xj and from Xk
    W_i = apply RELIEF to X
end;
for s=1: nr_positions
multi_W(s) = (see formula below);
end;
return multi_W
```

When the multiple runs are completed, the weight $multi\_W(s)$ of a residue s is computed using the formula below, where $N^+ = |\{W_i(s) > 0 \ \ \forall \ \ i\}|$ and $N^- = |\{W_i(s) < 0 \ \ \forall \ \ i\}|$.

$$multi\_W(s) = \begin{cases} \frac{1}{N^+} \sum_i \{W_i(s) > 0 \ \ \forall \ \ i\} & \text{for} \ \ N^+ > 0 \\ \frac{1}{N^-} \sum_i \{W_i(s) < 0 \ \ \forall \ \ i\} & \text{for} \ \ N^+ = 0 \ \wedge \ N^- > 0 \\ 0 & \text{for} \ \ N^+ = 0 \ \wedge \ N^- = 0 \end{cases}$$

When computing multi_W(s), all runs yielding zero weight for s are discarded. If this results in an empty set, then s is assigned weight equal to zero. Otherwise, $multi\_W(s)$ is set to the average over only the positive weights assigned to s. If there are only zero or negative weights, then $multi\_W(s)$ is the average of the negative weights assigned to s.

The core part of multi-RELIEF is RELIEF, considered one of the most successful multivariate feature weighting algorithms [7], due to its simplicity and effectiveness [11]. Recent bioinformatics applications employing (modifications of) RELIEF include, for instance, genome-wide genetic analysis [15] and gene selection [24]. RELIEF constructs a vector of weights, one for each position, by means of an iterative procedure, illustrated in pseudo-code below for two classes having the same number of sequences.

```
RELIEF
input:  X (samples of aligned proteins from two classes)
output: W (weights assigned to residues)
W = zero vector of size nr_positions
for each seq in X
     W = W + (seq - miss(seq)) - (seq - hit(seq))
end;
return W
```

The weights vector W is initialized to zero. At each iteration, one sequence seq is selected. The weights vector is updated by adding the 'difference' between seq and its nearest neighbor computed across sequences of the other class, called by $miss(seq)$, and subtracting the difference between seq and its nearest neighbor computed across sequences of the same class, called $hit(seq)$. This procedure is iterated over all sequences of the dataset X. The difference between two sequences $seq1 - seq2$ is a vector representing matches (0) and mismatches (1) between residues. For instance, ALM − VLM = 100.

2.1 Exploiting Structural Information: 3D

Multi-RELIEF can optionally include tertiary structure information, if available. It does this by employing the following heuristic based on the assumption that a specificity residue does not evolve in isolation, but within a functional cluster in the protein structure [22]. This means that a residue would be more likely to be a specificity residue if its neighboring residues are also specific.

We use 3D neighbors, that is, residues that share surface with a given residue as calculated by the web server at http://ligin.weizmann.ac.il/cma/ [19].

Alternatively, the tools available at http://www.infobiotics.org/ could be used. For instance, the PSP server could be used to either calculate, if the structure is known, or predict when the structure is unknown, many topological and geometrical 3D features. Moreover, the ProCKSI server could be used to calculate multiple versions of contact maps, and compare contact maps of the structures of the MSA sequences.

The weight of a residue is adjusted by adding the average weight of its 3D neighbors. In this way the score of a residue is boosted if its neighbors have a high average score.

3 Improving Multi-RELIEF

In multi-RELIEF, $multi\_W(s)$ assigns a high score to position s discriminating at least two classes. In particular, a maximum weight is assigned if s fully discriminates two specific classes but does not differentiate (i.e. weight less than or equal to zero) any other pair of classes. Furthermore, RELIEF assigns equal (zero) score to residues that are either fully conserved or fully divergent. However, fully conserved residues are more relevant than fully divergent ones with respect to protein functionality. Example of these two cases is shown in Table 1, where residues b and c have equal maximum score, and a and d have both score equal to zero. We propose the following approach for overcoming these two drawbacks of multi-RELIEF.

3.1 New-Multi-RELIEF

We introduce a parametric formula for computing weights assigned to position s. Let $cl(seq)$ denotes the class label (protein sub-family) of seq, and $c1, c2$ the labels of the two classes. Denote the element of a sequence seq in position s by seq_s. Let

$$W_{miss}(s) = \sum_{seq \in X}(seq_s - miss(seq)_s),$$
$$W_{hit,min}(s) = \min(W_{c1}(s), W_{c2}(s)),$$
$$W_{hit,max}(s) = \max(W_{c1}(s), W_{c2}(s))$$

and

$$W_{c1}(s) = \sum_{seq \in X, cl(seq)=c1}(seq_s - hit(seq)_s,$$
$$W_{c2}(s) = \sum_{seq \in X, cl(seq)=c2}(seq_s - hit(seq)_s).$$

Then we define

$$W_{new}(s) = \alpha_0 W_{miss}(s) - \alpha_1 W_{hit,min}(s) - \alpha_2 W_{hit,max}(s),$$

with $\alpha_0, \alpha_1, \alpha_2$ in $(0, 1]$ We call $W_{hit,min}$ and $W_{hit,max}$ *within-one-class weights*, and the parameters α *factors*, which can be viewed as multiplicative factors measuring the relevance given by the user to each term of the sum. Their values can be chosen depending on the specific type of application.

One can easily check that, for $\alpha_0 = \alpha_1 = \alpha_2 = 1$ one obtains the formula used in RELIEF to compute weights.

For each residue s, we assign more relevance (that is, high α_1) to the highest within-one-class conservation, that is, to the minimum within-one-class weight of s. Moreover, we account for uncertainty stemming, for instance, from the incompleteness of the considered protein sub-families, the quality of the alignments, and the possible presence of class noise (that is, proteins assigned to incorrect sub-families). We address uncertainty by choosing α_1 smaller than 1 (in our experimental analysis, the value 0.8 is selected). The remaining two terms of $W_{new}(s)$ are considered to have half the relevance of the minimum within-one-class weight (in our experiments $\alpha_0 = \alpha_2 = 0.4$).

Using W_{new} instead of W in Multi-RELIEF and by averaging the resulting weights for computing multi_W(s), we obtain the new-RELIEF algorithm shown below in pseudo-code.

```
new-Multi-RELIEF
input:  X1,..,Xm (m classes of an MSA), nr_iter, nr_sample
output: multi_W (weights assigned to positions)
for i=1: nr_iter
    select randomly two classes
    X = select randomly nr_sample sequences from each selected class
    W_i = W_new computed using X
end;
for s=1: nr_positions
new_multi_W(s) = average of W_i(s);
end;
return new_multi_W
```

The example given in Table 1, slightly adapted from [22], shows the (normalized) weights generated by multi-RELIEF and by new-multi-RELIEF. The MSA of four sub-families, $C_1, \ldots, C_4$, is considered, where each protein sequence has six positions $a, \ldots, f$.

The two feature weighting methods induce different rankings of the residues. Indeed, multi-RELIEF ranks both b and c as most relevant positions, followed by f, then both a and d which are considered equally relevant, followed by e. Instead new-multi-RELIEF generates a different ranking, namely c, b, a, f, d, e. In particular, new-multi-RELIEF considers residue c more relevant than b and the fully conserved residue a more relevant than d.

3.2 New-3D

In the boosting procedure of multi-RELIEF $multi\_W(s)$ is adjusted by adding the average weight of its 3D neighbors. Here we use 3D neighbors to change the weight of position s by associating the scaling factor β for the original weights, which can be viewed as prior confidence of these weights. Then the new weights

Table 1. Weights computed by multi-RELIEF applied to a toy example

	a	b	c	d	e	f
C_1	R	F	T	I	T	T
	R	F	T	Q	F	F
	R	F	T	N	V	V
	R	F	T	A	D	D
C_2	R	F	Y	S	T	N
	R	F	Y	F	F	N
	R	F	Y	D	V	N
	R	F	Y	V	D	N
C_3	R	Y	D	E	T	T
	R	Y	D	V	F	F
	R	Y	D	W	V	V
	R	Y	D	G	D	D
C_4	R	Y	H	H	T	T
	R	Y	H	P	F	F
	R	Y	H	Y	V	V
	R	Y	H	C	D	D
Multi-RELIEF weights	0	1	1	0	-1	0.5
new Multi-RELIEF weights	0	0.3	0.4	-0.8	-1.2	-0.6

are obtained by computing the mean of the original weight scaled by its prior importance, and the average of the original weights of its 3D neighbors:

$$new\_3D\_W(s) = \frac{1}{2}(\beta multi\_W(s) + \frac{\sum_{s' \in 3Dnn(s)} multi\_W(s')}{|3Dnn(s)|}),$$

where $3Dnn(s)$ are the positions of the 3D neighbors of s.

In the experimental analysis conducted in the sequel we use $\beta = 3$, that is, we assign high confidence to the original weights. (In general, higher values of β lead to similar results.)

4 Experimental Analysis

To assess the performance of the proposed modifications of the multi-RELIEF method, we considered the following six algorithms:

1. *MR*: the original multi-RELIEF algorithm from [22];
2. *new-MR*: the new multi-RELIEF algorithm;
3. *MR 3D*: multi-RELIEF with 3D weight boosting procedure;
4. *MR new-3D*: multi-RELIEF with the new 3D weight boosting procedure;
5. *new-MR 3D*: new multi-RELIEF with the 3D weight boosting procedure;
6. *new-MR new-3D*: new multi-RELIEF with the new 3D weight boosting procedure.

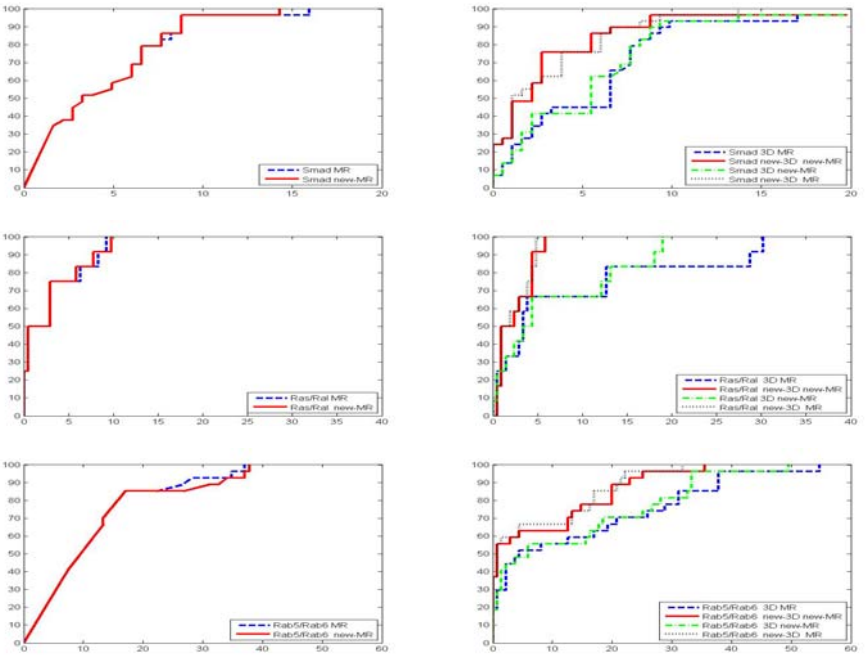


Fig. 1. ROC curves of the algorithms. From top to bottom: dataset Smad, Ras/Ral and Rab5/Rab6.

We conducted experiments datasets previously used in [22], containing different protein families with various associated functional specificity properties. Properties of these datasets are summarized in Table 2; they are thoroughly described in [22].

The following parameter values were used: $\alpha_1 = 0.8, \alpha_2 = \alpha_0 = 0.4, \beta = 3$, $nr\_iter = 2000$ and $nr\_samples = 5$. In general, a high value of nr_iter and a reasonably small value of $nr\_samples$ can be used. Ties were broken by sorting residue positions with equal score in increasing sequence position.

The Receiver-operator characteristic (ROC) curve is used for testing the capability of an algorithm to separate true and false positives [20,17]. Known functional specificity residues are considered true positives, the other ones true negatives. The weight values are used as threshold for generating the ROC curve. For each weight value v the set of residues with weight higher than or equal to v is considered: the true positive percentage is reported on the y-axis (sensitivity, or coverage), and the false positive percentage (1−specificity, or error) on the x-axis. The ROC curve thus describes the goodness of a method in giving higher ranking to *known* functionally specific residues.

Table 2. Properties of the datasets used for testing the algorithms

dataset	nr of classes	avg class size	standard deviation	max class size	min class size	sequence length	formation
GPCR	77	26.8	34	189	3	214	ligand
LacI	15	3.6	2.5	12	2	339	ligand and DNA
Ras/Ral	2	44.5	24.5	69	20	218	protein
Rab5/Rab6	2	5.0	1	4	6	163	protein
MIP	2	30.0	18	48	12	430	protein
Smad	2	10.0	2	12	8	211	protein

Results of experiments are given in Figures 1, 2. They can be summarized as follows:

- Significant improvement of new-*MR* is achieved on the GPCR dataset, which contains many classes, and on the MIP one. On the other datasets results of new-*MR* and *MR* do not differ significantly (see left column of Figures 1, 2).
- When also tertiary structure information is used, results improve on all datasets except LacI (see right column of Figures 1, 2). Specifically, on Smad, Ras/Ral and Rab5/Rab6 new-3D new-*MR* outperforms significantly all other *MR* variants employing 3D information. On the other three datasets performances of the *MR* variants do not significantly differ from each other, except for GPCR, where both 3D new-*MR*and new-3D new-*MR* ROC curves significantly dominate the other ones. On the LacI dataset, new-3D new-*MR* show slightly worse performance than 3D *MR*. However, classes in this dataset are rather small, with one class containing 12 elements and all other classes having very few elements (≤ 4). We can take into account this characteristic of the dataset and set the parameter values in such a way that importance of the maximum within-one-class conservation is strengthen (smaller values for α_0, α_2), and the confidence of the original weights is decreased (smaller value for β). For instance, improvement is achieved by choosing $\alpha_1 = 0.8, \alpha_0 = \alpha_2 = 0.2, \beta = 0.5$ (see Figure 2, bottom plots).
- *new-3D new-MR* shows best overall results across all datasets. Performance of *3D* new-*MR* is also very good on the GPCR, MIP and LaCI datasets (see Figure 1), but is significantly worse than the one of the other algorithms on the Smad, Ras/Ral and Rab5/Rab6 datasets (see Figure 2).

In general, the results substantiate the effectiveness of the proposed new method for detecting of specificity residues from multiple sequence alignment. Indeed, new-3D new-*MR* achieves best overall performance, in particular significantly improving robustness and performance of the state-of-the-art method multi-RELIEF when tertiary structure information is used.

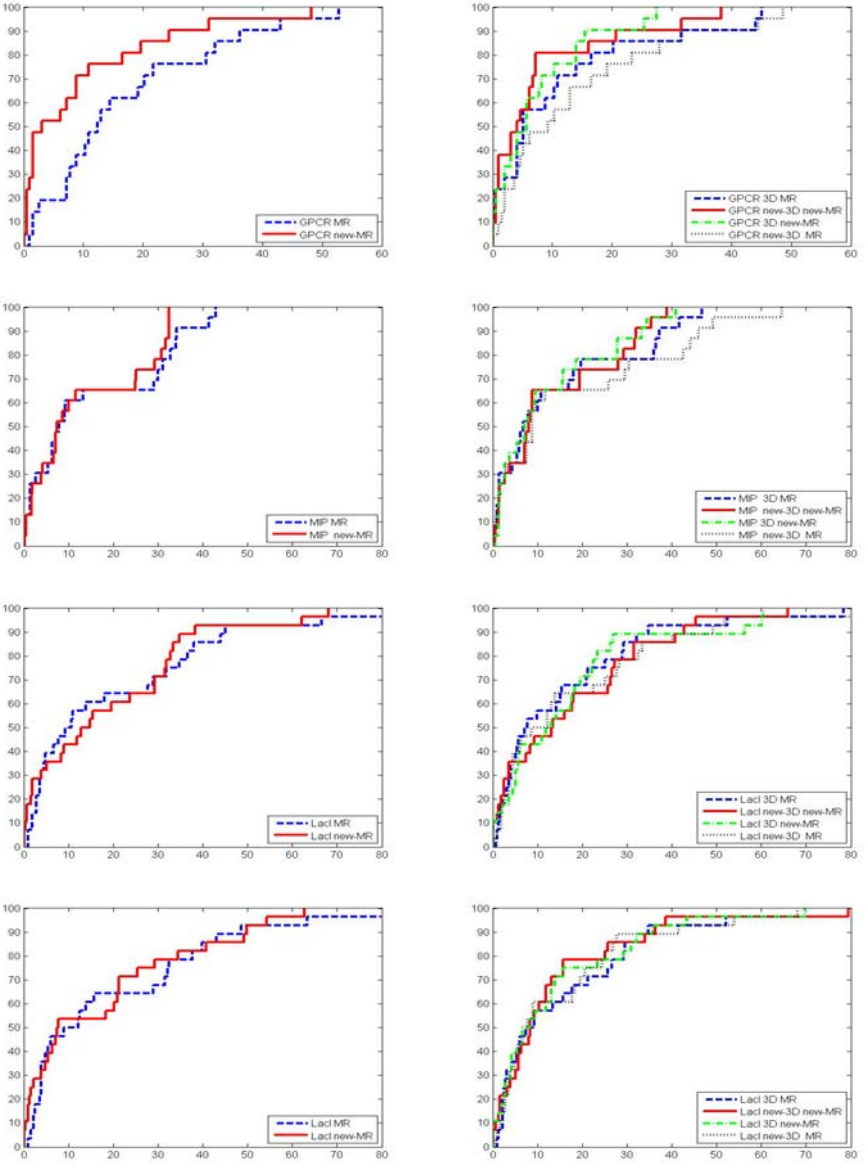


Fig. 2. ROC curves of the algorithms. From top to bottom: dataset GPCR, MIP, LacI, and LacI with $\alpha_1 = 0.8$, $\alpha_0 = \alpha_2 = 0.2$, $\beta = 0.5$.

5 Conclusion

We proposed a new algorithm for detecting specificity residues from multiple sequence alignments, which is obtained by modifying the core parts of the recent state-of-the-art method for this task multi-RELIEF. Results of extensive experiments showed improved overall performance and robustness of the new method, especially when tertiary structure information is used. In future work, we want to extend the proposed method to predict functional sites.

Acknowledgements

Thanks to Leonid A. Mirny and Mikhail S. Gelfand for making their LacI dataset available, and to Anton Feenstra, Walter Pirovano and Kai Ye for the other datasets and for the useful past discussions on the subject of this paper. This work was partially supported by the NWO project 639.023.604.

References

1. Bickel, P.J., Kechris, K.J., Spector, P.C., Wedemayer, G.J., Glazer, A.N.: Finding important sites in protein sequences. Proc. Natl. Acad. Sci. USA 99, 14764–14771 (2002)
2. Carro, A., Tress, M., de Juan, D., Pazos, F., Lopez-Romero, P., Del Sol, A., Valencia, A., Rojas, A.M.: Treedet: a web server to explore sequence space. Nucleic Acids Res. 35(web server issue), 99 (2006)
3. Chakrabarti, S., Panchenko, A.R.: Ensemble approach to predict specificity determinants: benchmarking and validation. BMC Bioinformatics 10, 207 (2009)
4. Del Sol Mesa, A., Pazos, F., Valencia, A.: Automatic methods for predicting functionally important residues. J. Mol. Biol. 326(4), 1289–1302 (2003)
5. Feenstra, K.A., Pirovano, W., Krab, K., Heringa, J.: Sequence harmony: detecting functional specificity from alignments. Nucleic Acids Res. 35(web server issue), W495–W498 (2007)
6. Gu, X.: A simple statistical method for estimating type-ii (cluster-specific) functional divergence of protein sequence. Mol. Biol. Evol. 23, 1937–1945 (2006)
7. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. Journal of Machine Learning Research 3, 1157–1182 (2003)
8. Hannenhalli, S.S., Russell, R.B.: Analysis and prediction of functional sub-types from protein sequence alignments. J. Mol. Biol. 303(1), 61–76 (2000)
9. Kalinina, O.V., Gelfand, M.S., Russell, R.B.: Combining specificity determining and conserved residues improves functional site prediction. BMC Bioinformatics (2009)
10. Kalinina, O.V., Novichkov, P.S., Mironov, A.A., Gelfand, M.S., Rakhmaninova, A.B.: SDPpred: a tool for prediction of amino acid residues that determine differences in functional specificity of homologous proteins. Nucleic Acids Res. 32(web server issue), W424–W428 (2004)
11. Kononenko, I.: Estimating attributes: Analysis and extensions of relief. In: Bergadano, F., De Raedt, L. (eds.) ECML 1994. LNCS, vol. 784, pp. 171–182. Springer, Heidelberg (1994)

12. Kuipers, R.K., Joosten, H.-J.J., Verwiel, E., Paans, S., Akerboom, J., van der Oost, J., Leferink, N.G., van Berkel, W.J., Vriend, G., Schaap, P.J.: Correlated mutation analyses on super-family alignments reveal functionally important residues. Proteins 76(3), 608–616 (2009)
13. Mihalek, I., Res, I., Lichtarge, O.: A family of evolution-entropy hybrid methods for ranking protein residues by importance. J. Mol. Biol. 336(5), 1265–1282 (2004)
14. Mirny, L.A., Gelfand, M.S.: Using orthologous and paralogous proteins to identify specificity-determining residues in bacterial transcription factors. J. Mol. Biol. 321(1), 7–20 (2002)
15. Moore, J.H., White, B.C.: Tuning relieff for genome-wide genetic analysis. In: Marchiori, E., Moore, J.H., Rajapakse, J.C. (eds.) EvoBIO 2007. LNCS, vol. 4447, pp. 166–175. Springer, Heidelberg (2007)
16. Pirovano, W., Feenstra, K.A., Heringa, J.: Sequence comparison by sequence harmony identifies subtype specific functional sites. Nucleic Acids Res. 34, 6540–6548 (2006)
17. Provost, F., Kohavi, R.: Guest editors' introduction: On applied research in machine learning. Machine Learning 30, 127–132 (1998)
18. Shenkin, P.S., Erman, B., Mastrandrea, L.D.: Information-theoretical entropy as a measure of sequence variability. Proteins 11(4), 297–313 (1991)
19. Sobolev, V., Sorokine, A., Prilusky, J., Abola, E.E., Edelman, M.: Automated analysis of interatomic contacts in proteins. Bioinformatics 15, 327–332 (1999)
20. Swets, J.A.: Measuring the accuracy of diagnostic systems. Science 240, 1285–1293 (1988)
21. Whisstock, J.C., Lesk, A.M.: Prediction of protein function from protein sequence and structure. Quart. Rev. Biophys. 36(3), 307–340 (2003)
22. Ye, K., Feenstra, K.A., Heringa, J., IJzerman, A.P., Marchiori, E.: Multi-relief: a method to recognize specificity determining residues from multiple sequence alignments using a machine-learning approach for feature weighting. Bioinformatics 24(1), 18–25 (2008)
23. Ye, K., Lameijer, E.W., Beukers, M.W., IJzerman, A.P.: A two-entropies analysis to identify functional positions in the transmembrane region of class a g protein-coupled receptors. Proteins 63, 1018–1030 (2006)
24. Zhang, Y., Ding, C., Li, T.: Gene selection algorithm by combining relieff and mrmr. BMC Genomics 9(suppl. 2) (2008)

Using Probabilistic Dependencies Improves the Search of Conductance-Based Compartmental Neuron Models

Roberto Santana, Concha Bielza, and Pedro Larrañaga

Departmento de Inteligencia Artificial, Universidad Politécnica de Madrid
28660 Boadilla del Monte, Madrid, Spain
roberto.santana@upm.es, pedro.larranaga@fi.upm.es, mcbielza@fi.upm.es

Abstract. Conductance-based compartmental neuron models are traditionally used to investigate the electrophysiological properties of neurons. These models require a number of parameters to be adjusted to biological experimental data and this question can be posed as an optimization problem. In this paper we investigate the behavior of different estimation of distribution algorithms (EDAs) for this problem. We focus on studying the influence that the interactions between the neuron model conductances have in the complexity of the optimization problem. We support evidence that the use of these interactions during the optimization process can improve the EDA behavior.

Keywords: Conductance-based compartmental neuron models, estimation of distribution algorithm, probabilistic models.

1 Introduction

The intrinsic electrophysiological properties of neurons condition the activity of neuronal circuits and ultimately determine biological responses to multiple stimuli. Conductance-based compartmental neuron models [6,7] have been very useful to study different aspects of neuronal dynamics. The electrical activity of this type of neuron model is mostly influenced by its ionic current conductances. Single compartmental models are particularly suitable for investigating the way in which different ionic currents act on the neuronal subthreshold behavior and spike generation.

A common method used in the creation of conductance-based compartmental neuron models is to record the *in vitro* response of the neuron to a set of simple current stimuli and then attempt to replicate the response on a detailed compartmental model of that cell [2]. Usually, the general form of the model (determined by a set of differential equations) is known but it is necessary to find a choice of the model parameters that guarantees a good match between the model behavior and the experimental data. This identification process can be posed as the problem of finding the optimal (given a predefined measure) values for the set of parameters.

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2010, LNCS 6023, pp. 170–181, 2010.

However, the parameter optimization problem is not straightforward. Disparate parameter combinations can lead to similar neuron electrophysiological properties, interactions between the parameters can be highly nonlinear and the simulation of the models can be very costly. Therefore, it is an important issue to conceive optimization algorithms than can deal with this type of problems. We use EDAs, [9,10], a class of evolutionary algorithms that construct probabilistic models of the set of selected solutions. Our analysis focuses on the influence that the interactions between the neuron model parameters have in the behavior of the different EDAs. We support evidence that the use of these interactions during the optimization process can help to obtain better sets of neuron parameters.

As a case study we use a database of about 1.7 million model neurons that were generated by independently varying the 8 maximal conductances of a realistic conductance-based model [13]. This information allows us to know the function landscape and optima, permitting us to accurately evaluate the performance of different variants of the optimization algorithms. It has already been shown [5,13,19] that databases of model neurons are particularly convenient to study the way in which the coordinated regulations (corregulations) of ionic currents influence different aspects of the neurons dynamics. We speculate that corregulations are translated into interactions between the variables of the optimization problem. Therefore, the databases can also be useful to investigate the way the search is affected by the strong interactions between the conductance parameters and to design more appropriate optimization methods for these problems.

2 Neurons and Models of Neurons

Neurons can display different types of spontaneous electrical activity. A neuron is *silent* when no electrical activity is displayed. On the contrary, *tonically active neurons* display different patterns of electrical activity and can be classified accordingly based on these patterns. *Spiking neurons* are those that display narrow spikes while *bursting neurons* have a broad shoulder after each spike. Neuron can also show an *irregular* activity, where none of the previous classifications can be given. The same classification is used to describe the behavior of neurons under electrical stimulation, i.e. when a current is input to the neurons. In general, different input current stimuli may determine changes in the electrical activity of the neuron.

To characterize the electrophysiological activity of neurons, measures that describe their electrical behavior have been defined. The *burst duration* is the interval in which there are a significantly higher number of spikes as compared to other intervals in the spike train. The *burst period* is the burst duration plus the largest interspike interval. The *number of maxima per period* is the number of maxima comprised in a bursting period. The *after-hyperpolarization (AHP)* potential is the trough voltage between bursts.

For tonically active and bursting neurons, the *spike amplitude* is the difference between the maximum value of the membrane potential and the value at the onset of the action potential. The average *inter-maximum interval (IMI)* is the

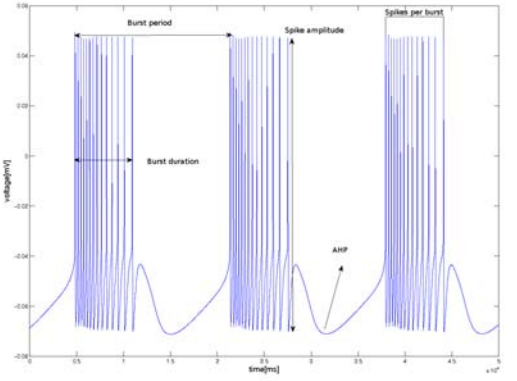


Fig. 1. Different measures that characterize the electrophysiological activity of neurons

average of the distances between two subsequent voltage maxima. The *discharge frequency* is the inverse of the IMI. Figure 1 illustrates some of these measures.

Neuron models are indispensable tools for understanding the neuron structural organization and testing different hypotheses about their behavior. A neuron model should display an appropriate balance between its accuracy to reproduce the neuron behavior and its computational complexity and tractability. We use a single compartmental model. This type of model neglects the neuron's spatial structure and focuses on how its various ionic currents contribute to subthreshold behavior and spike generation.

3 Model Description

The model was constructed based on experimental data obtained from lobster stomatogastric neurons [13,21]. The 8 currents in the single-compartmental model are based on those of the lobster stomatogastric ganglion neurons (STG) [21,14] and include a Na^+ current, I_{Na}; two Ca^{2+} currents, I_{CaT} and I_{CaS}; a transient K^+ current, I_A; a Ca^{2+} dependent K^+ current, I_{KCa}; a delayed rectifier K^+ current, I_{Kd}; a hyperpolarization-activated inward current, I_H; and a leak current, I_{leak}. In what follows we present the basic details about the model. More details can be found in [14].

Each of the model membrane currents is described by

$$I_i = \bar{g}_i (m_i)^p h_i (V - E_i) A \tag{1}$$

where E_i is the reversal potential, $A = 0.628 \times 10^{-3}$ cm^2 is the membrane area and $\bar{g}_i$ is the maximal conductance. The database of models was constructed varying the maximal conductances of all 8 currents independently. Information

about the way in which the reversal potentials E_i where computed, as well the equations for the activation and inactivation variables m_i and h_i can be obtained from [14].

I_{input}, being a given input current, the neuron potential V is modeled by

$$C\frac{dV}{dt} = -\sum_i I_i - I_{input} \tag{2}$$

where $C = 0.628$ nF is the membrane capacity. In [14], some experiments were conducted to observe the model electrical activity with different input currents. For current step simulations, the input current was stepped from zero to a depolarizing DC current of 3 or 6 nA.

Different assignments of the model maximal conductance parameters will influence the model electrical activity. Therefore, it is possible to investigate which are the particular characteristics of a set of models determined by different parameter combinations. In [14], the database of $1,679,626$ models was generated by varying the 8 parameters of the model previously described over 6 equidistant values.

Let X_i and x_i respectively represent a discrete random variable and a possible assignment to X_i. Similarly, we use $\mathbf{X} = (X_1, \ldots, X_n)$ to represent an n-dimensional random variable and $\mathbf{x} = (x_1, \ldots, x_n)$ to represent one of its possible values. In our problem representation, each variable will represent one of the neuron model parameters, $n = 8$ and $x_i \in \{0, \ldots, 5\}$. Table 1 shows the variable codification of the neuron model parameters and the conductance values assigned to each of the membrane currents which were used in the construction of the database.

Each set of conductances represented by a vector $\mathbf{x} = (X_1, \ldots, X_8)$ defines a "model neuron" or simply a *neuron*. The spontaneous activity of each neuron was simulated and classified into 4 categories: silent, tonically active, bursting, and non-periodic. Several features were extracted from the neurons. Different descriptors of the electrical activity were computed.

Table 1. Parameter representation and Conductance values assigned to each of the membrane currents and used in the construction of the database

Var	$Current$	0	1	2	3	4	5
X_1	I_{Na}	0	100	200	300	400	500
X_2	I_{CaT}	0	2.5	5.0	7.5	10.0	12.5
X_3	I_{CaS}	0	2.0	4.0	6.0	8.0	10.0
X_4	I_A	0	10.0	20.0	30.0	40.0	50.0
X_5	I_{KCa}	0	5.0	10.0	15.0	20.0	25.0
X_6	I_{Kd}	0	25.0	50.0	75.0	100.0	125.0
X_7	I_H	0	0.01	0.02	0.03	0.04	0.05
X_8	I_{leak}	0	0.01	0.02	0.03	0.04	0.05

4 Optimization Approach

Our goal is to find, from a large set of candidates, a neuron model that resembles the electrical activity of a given target neuron as described by recorded experimental data. One simplification is to use a search of candidate solutions amenable for tractable exhaustive enumeration (neuron database). Since the neuron model database provides a description of all the neurons, we can carefully design a fitness function based on this information and know a priori the function values for all the solutions of the search space.

For each optimization problem, we will use as a target neuron, one neuron model selected from the database according to some predefined criterion related to its electrophysiological activity. This choice of the target neuron guarantees that there is at least a solution of the search space that optimizes the fitness function. However, the question of how to measure the similarity between the dynamics of the target neuron and any other neuron has to be solved.

There are three main types of functions used to find optimal neuron model parameters [22]: feature-based functions, point-by-point comparison of voltage traces, and multi-objective functions. In this paper we use the first approach and the frequency of voltage maxima as the feature that will characterize the dynamics of the neuron models. The spontaneous frequency of voltage maxima contains information about the neurons dynamics but this information does not support much details about the neuron behavior. Therefore, in addition to the spontaneous frequency we use the steady state maximal frequencies as computed during 3 nA and 6 nA current injections.

Let $\mathbf{x}^*$ and $\mathbf{x}$ respectively be the target neuron and any other neuron of the search space. The fitness function $f(\mathbf{x})$ is defined as:

$$f(\mathbf{x}) = -\left((fq_s(\mathbf{x}) - fq_s(\mathbf{x}^*))^2 + \left(fq_3(\mathbf{x}) - fq_3(\mathbf{x}^*)\right)^2 + (fq_6(\mathbf{x}) - fq_6(\mathbf{x}^*))^2\right)^{\frac{1}{2}} \tag{3}$$

where fq_s, fq_3 and fq_6 respectively represent the spontaneous frequency and the steady state maximal frequencies as computed during during 3 nA and 6 nA current injections. When $\mathbf{x}^* = \mathbf{x}$, $f(\mathbf{x}) = 0$ which is the maximum of the function. Therefore, we transform the search for a neuron model with similar electrical activity to the target neuron in the maximization of function $f(\mathbf{x})$. Notice that the fitness landscape of this function will depend on the choice of $\mathbf{x}^*$.

5 Estimation of Distribution Algorithms

EDAs [9,10] are optimization algorithms that can learn and exploit the search space regularities in the form of probabilistic dependencies. They are very similar to genetic algorithms, but instead of using genetic operators, they construct an explicit probability model of a set of selected solutions. The model is used to generate new promising solutions. One characteristic that serves to distinguish different types of EDAs is the probabilistic model used by the algorithm. The

models may differ in the order and number of the probabilistic dependencies that they represent.

Let $p(\mathbf{x})$ denote a positive probability distribution. In this paper, we use three different types of models: A univariate marginal model, a tree model and a Bayesian network model. In a univariate model, the joint probability distribution can be factorized as the product of the univariate probabilities of the variables, i.e. $p(\mathbf{x}) = \prod_i p(x_i)$. This is the model used by the univariate marginal distribution algorithms (UMDA) [10].

A probability distribution $p_{Tree}(\mathbf{x})$ that conforms to a tree is defined as $p_{Tree}(\mathbf{x}) = \prod_{i=1}^{n} p(x_i|pa(x_i))$ where $Pa(X_i)$ is the parent of X_i in the tree, and $p(x_i|pa(x_i)) = p(x_i)$ when $Pa(X_i) = \emptyset$, i.e. X_i is the root of the tree. Probabilistic trees are represented by acyclic connected graphs. In this paper we use the Tree-EDA [18], an EDA that uses trees to represent the probability distributions. A Bayesian network can be seen as a generalization of a tree where each variable can have multiple parents. In this paper we use the estimation of Bayesian networks algorithm (EBNA) [3], one of the EDAs based on the use of Bayesian networks.

Pseudocode for EBNA is shown in Algorithm 1. The algorithm was implemented in Matlab using the MATEDA-2.0 software [15]. The implementations of the UMDA and Tree-EDA follow the same scheme but the learning and sampling steps are modified accordingly. These were implemented using the Matlab Bayes Net (BNT) toolbox [11]. The scoring metric used for the Bayesian network was the Bayesian metric with uniform priors, and each node was allowed to have a maximum number of 5 parents. The truncation parameter was $T = 0.5$. Best elitism, in which the selected population is passed to the next population, was used.

Algorithm 1. **EBNA**

```
1  Generate an initial population D_0 of individuals and evaluate them
2  t ← 1
3  do {
4     D_{t-1}^{Se} ← Select N individuals from D_{t-1} using truncation selection
5     Using D_{t-1}^{Se} as the data set, apply local search to find one BN structure
      that optimizes the scoring metric
6     Calculate the parameters of the BN using D_{t-1}^{Se} as the data set
7     D_t ← Sample M individuals from the BN and evaluate them
8  } until Stopping criterion is met
```

6 Related Work

There is considerable work on the application of optimization methods to neuron model parameter optimization. Usually, single objective functions that measure a particular aspect of the model performance given the parameters are used.

Among the most commonly employed optimization methods are [12,22]: hand tuning, gradient descent, evolutionary algorithms, bifurcation analysis and hybrid methods.

We review some of the work that seems to corroborate the convenience of modeling the interactions between the conductances in the search of neuron models that exhibit a desired electrophysiological behavior pattern. In the next section we present empirical results that show that this is indeed the case.

Achard and De Schutter [1] use an evolutionary strategy to obtain a set of different good models of the cerebellar *Purkinje* cells. The authors investigate different hypotheses that could explain the large diversity models with similar good conductance density values. Although probabilistic models of the neuron model parameter space are not constructed, the correlations between pairs of parameters are computed and used to investigate the relationship between the parameters.

In [4], Golowasch et al. generated a set of neuron models by randomly sampling the maximal conductance of the STG neuron [21]. The models were used to identify sets of maximal conductances that generate one-spike bursters. A model using the means of the maximal conductance of this set was constructed. It turned out that the model was not itself a one-spike burster. The authors concluded that averages over multiple samples can fail to characterize a system whose behavior depends on interactions involving a number of highly variable components.

In [5], a neuron database is used to investigate spiking variability in *Globus pallidus* neurons. The authors acknowledge that the effect of each conductance in the neuron electrophysiological properties was highly dependent on the background context of other present conductances. The fact that every conductance in the model could show opposite effects on spike rate when it was increased depending on the background of other present conductances [5] suggests that fitness functions that use the spike rate as a feature may have malign interactions between the variables [8]. Malign interactions can deceive evolutionary algorithms that do not take interactions into account.

In [19], Smolinski and Prinz analyze a different database of neuron models. From the analysis of the subset of models that resemble the electrophysiological behavior of natural neurons the authors identify three types of corregulations: 1) No significant interactions. 2) Expression of co-preference for specific ranges of values and 3) Corregulation, expressed by a characteristic "ridge" in the plot of pair-by-pair co-occurrence of specific parameter values. We could expect that variables with no significant interactions are less likely to appear together in the graphical model structures learned by EDAs.

7 Experiments

The main objective of our experiments is to investigate whether the use of interactions, represented by the graphical models used by EDAs, improves the results achieved when no interactions are taken into account. We assume that,

Table 2. Target neuron models selected from the database and a number of properties and measures determined from their simulation

$Index$	I_{Na}	I_{CaT}	I_{CaS}	I_A	I_{KCa}	I_{Kd}	I_H	I_{leak}	T_s	T_3	T_6	fq_s	fq_3	fq_6
720973	100	7.5	4	40	5	125	0	0.01	2	1	1	1.1632	54.6364	60.1949
1522117	500	5.0	6	40	10	125	0	0.01	2	1	1	1.1559	37.7609	42.2354
833389	100	12.5	10	10	0	25	0.04	0.01	2	0	0	215.0538	0	0
965338	300	5.0	8	0	25	0	0.05	0.04	2	1	1	4.5382	5.8246	7.8225
436821	100	7.5	4	10	0	25	0.05	0.03	2	0	0	68.2173	0	0
1071411	300	10.0	10	40	20	25	0.02	0.03	2	2	2	3.9518	23.1826	27.9265
83317	0	2.5	8	40	5	100	0.02	0.01	2	0	0	2.4422	0	0
882103	300	0	10	20	15	100	0.05	0.01	2	1	1	18.0810	37.0142	42.6758
300566	100	0	4	30	25	75	0	0.02	2	1	1	4.8984	26.9808	35.5637
1374808	400	12.5	4	40	20	125	0	0.04	2	2	2	8.0523	35.9712	47.1328

if EDAs that represent probabilistic dependencies, i.e. Tree-EDA and EBNA, outperform those that do not represent such dependencies, i.e. UMDA, then the problem exhibits interactions between the variables and these interactions are important to solve the problems.

To test the algorithms, we select 10 functions defined by selecting different target neuron models from the data sets. The target neurons, shown in Table 2, are all bursting neurons during the spontaneous activity and have been selected trying to cover different patterns of electrical activity of the neurons. Under spontaneous activity, the first five neurons have the following relevant characteristics: for 720973: a high burst period and small burst duration; for 1522117: a high burst period and high burst duration; for 833389: the smallest burst period; for 965338: one of the smallest burst durations; for 436821: one of the highest number of maxima. The rest of the target neurons have been randomly selected from the set of bursting neurons.

In the table, $Index$ is the index of the neuron in the database, T_s, T_3 and T_6 are the types of electrical activity under the different experimental conditions, coded as silent neuron (0), spiking neuron (1), bursting neuron (2). Similarly fq_s, fq_3 and fq_6 represent the frequencies under the different experimental conditions.

We conducted 30 experiments of UMDA, Tree-EDA and EBNA for each of the 10 functions and with two different settings: I) Population size is 500 and 30 generations; II) Population size is 1000 and 60 generations are conducted. These settings were determined aiming to keep the number of function evaluations relatively small, and after preliminary experiments were conducted. The number of times each algorithm found the optimum for all the functions are shown in Table 3. It can be seen in the table that only one of the problems is relatively easier to solve by the methods. Instance 1071411 is particularly complex for all the methods. However, the best results are clearly achieved by EBNA that outperforms both the Tree-EDA and UMDA.

To determine whether differences between the algorithms are statistically significant, we have used the Kruskal-Wallis test to accept or reject the null hypothesis that the samples have been generated from the same probability distribution. The test significance level was 0.01. For all instances, except instances 833389 and 436821, significant statistical differences have been found between

Table 3. Number of times each algorithm found the optimum for each of the 10 functions

Alg.	720973		1522117		833389		965338		436821		1071411		83317		882103		300566		1374808		*Tot.*
Setting	*I*	*II*	*I*	*II*	*I*	*II*	*I*	*II*	*I*	*II*	*I*	*II*	*I*	*II*	*I*	*II*	*I*	*II*	*I*	*II*	
UMDA	0	0	0	0	6	27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	33
TREE − EDA	0	0	0	0	13	19	0	0	0	4	0	0	0	0	0	0	0	0	0	0	36
EBNA	0	2	1	0	30	30	2	4	6	12	0	0	0	1	2	1	3	8	18	25	129

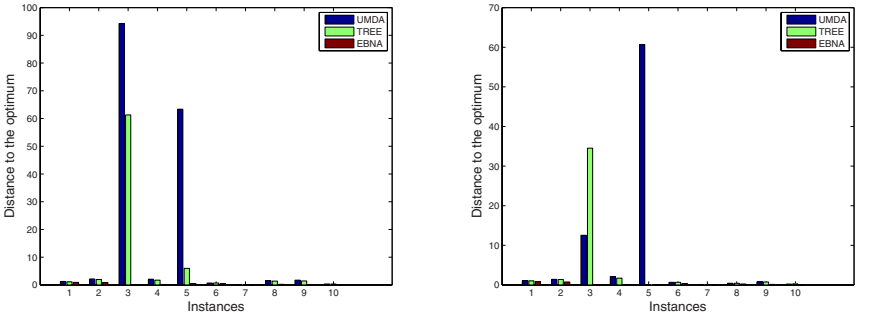


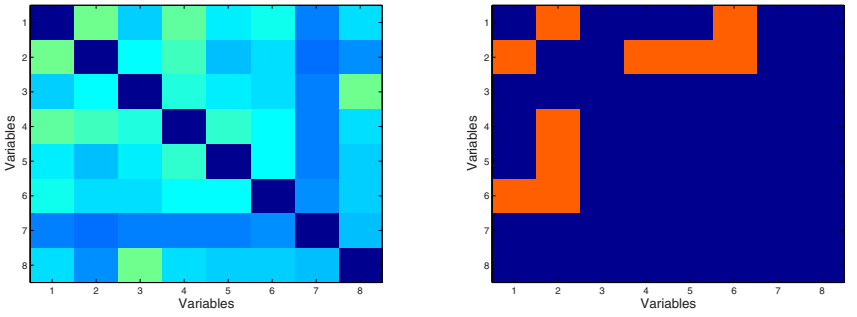
Fig. 2. Distance of the solutions found by the different EDAs to the best solutions for: a) EDA setting $(500 - 30)$. b) EDA setting $(1000, 60)$.

the EBNA and the other two EDAs for the two settings. There were not significant statistical differences between EBNA and Tree-EDA for instance 436821 for both EDA settings, and between EBNA and the other two EDAs for instance 833389, setting II.

It is interesting to note that the use of bivariate dependencies are not sufficient to improve the EDA results. To investigate the quality of the average solutions found by the algorithms, we also compute the average fitness distance of the best solutions found by the different EDAs to the optimal solutions. This information is shown in Figure 2. It can be appreciated that the increment in the number of allowed function evaluations, due to more individuals in the population and more generations, does not significantly improve the results of UMDA and Tree-EDA.

Finally, we investigate the structure of the dependencies learned by EBNA. For each problem, we computed the frequency of the arcs learned by the Bayesian network. The arc frequencies were calculated from the set of 900 Bayesian networks learned using the first EDA experimental setting. Figure 3 a) shows the frequency matrix learned for instance 1071411. Lighter colors represent stronger dependencies between the variables. Note that not all the dependencies appear with the same frequency.

For each problem, frequency matrices were thresholded to leave those arcs that were in at least 550 of the 900 Bayesian networks. We then computed the thresholded dependencies that were present in at least 5 of the 10 instances. These dependencies are shown in Figure 3 b). We hypothesize that some of them



a) b)

Fig. 3. Structure of the interactions captured by the Bayesian networks learned by EDAs. Lighter colors represent stronger dependencies between the variables. a) Most frequent structures learned for instance 1071411. b) Dependencies that were frequent for at least five of the 10 functions.

are due to the existence of important corregulations between the conductances of bursting neurons. To validate this hypothesis, we investigate previous studies of the conductance structure [20].

In [20], the construction of a dimensional stack image that visualizes the relationship between the spontaneous neuron activity and the conductance values allowed to make some conclusions about the structure of the conductance space. Of the four statements made about the "layout" of neuron models in the conductance space, only two refer to the interactions between conductances: 1) Many one-maxima bursters that have nonzero $\bar{g}_{Na}$ have zero delayed-rectifier potassium conductance $\bar{g}_{Kd}$. 2) There seems to be a regular gradation of bursters from few maxima per burst to many maxima per burst as one increases $\bar{g}_{Kd}$ and decreases $\bar{g}_{CaT}$. The interactions between the conductances pairs ($\bar{g}_{Na}$, $\bar{g}_{Kd}$) and ($\bar{g}_{Kd}$, $\bar{g}_{CaT}$) are both captured in the matrix shown in Figure 3 b). Further analysis, for instance the inspection of the corresponding marginal tables, should reveal the type of relationship between these pairs of parameter conductances as captured by the Bayesian networks. This remains as a subject of further work.

8 Conclusions

In this paper we have shown that interaction should be taken into account in the search of neuron models that have a predefined physiological activity. This is the first time, to our knowledge, that EDAs have been applied in the context of neuron modeling[1]. In contrast with previous EDA applications to problems from the biological domain [17], where bivariate interactions are sufficient to

[1] This is not the case in the field of artificial neural networks where several applications of EDAs have been reported.

remarkably improve the results achieved by univariate models, for the problems addressed in this paper, higher order interactions are needed.

Elucidating the relationship between the distribution of their intrinsic properties and dynamic activity of neurons is a crucial step in understanding larger-scale phenomena such as network oscillations and inter-nuclei synchronization. We speculate that further application of intelligent data analysis techniques [15,16] to the data generated by the EDAs can unveil additional information about the structure of the conductance space.

Acknowledgements

We thank to the Prinz laboratory, particularly to Amber Hudson and Tomasz Smolinski for providing useful information on neuron model databases, including the database used in this paper. This research is part of the CajalBlueBrain project. It has been partially supported by TIN-2008-06815-C02-02, TIN2007-62626 and Consolider Ingenio 2010 - CSD2007-00018 projects (Spanish Ministry of Science and Innovation).

References

1. Achard, P., De Schutter, E.: Complex parameter landscape for a complex neuron model. PLoS Computational Biology 2(7), 794–804 (2006)
2. Druckmann, S., Banitt, Y., Gidon, A., Schuermann, F., Markram, H., Segev, I.: A novel multiple objective optimization framework for constraining conductance-based neuron models by experimental data. Frontiers in Neuroinformatics 1(1) (2007)
3. Etxeberria, R., Larrañaga, P.: Global optimization using Bayesian networks. In: Ochoa, A., Soto, M.R., Santana, R. (eds.) Proceedings of the Second Symposium on Artificial Intelligence (CIMAF 1999), Havana, Cuba, pp. 151–173 (1999)
4. Golowasch, J., Goldman, M.S., Abbott, L.F., Marder, E.: Failure of averaging in the construction of a conductance-based neuron model. Journal of Neurophysiology 87, 1129–1131 (2002)
5. Günay, C., Edgerton, J.R., Jaeger, D.: Channel density distributions explain spiking variability in the globus pallidus: A combined physiology and computer simulation database approach. Journal of Neuroscience 28(30), 7476–7491 (2008)
6. Herz, A.V.M., Gollisch, T., Machens, C.K., Jaeger, D.: Modeling single-neuron dynamics and computations: A balance of detail and abstraction. Science 314(5796), 80–85 (2006)
7. Hodgkin, A.L., Huxley, A.F.: A quantitative description of ion currents and its applications to conduction and excitation in nerve membranes. Journal of Physiology 117, 500–544 (1952)
8. Kallel, L., Naudts, B., Reeves, R.: Properties of fitness functions and search landscapes. In: Kallel, L., Naudts, B., Rogers, A. (eds.) Theoretical Aspects of Evolutionary Computing, pp. 177–208. Springer, Heidelberg (2000)
9. Larrañaga, P., Lozano, J.A. (eds.): Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation. Kluwer Academic Publishers, Dordrecht (2002)

10. Mühlenbein, H., Paaß, G.: From recombination of genes to the estimation of distributions I. Binary parameters. In: Ebeling, W., Rechenberg, I., Voigt, H.-M., Schwefel, H.-P. (eds.) PPSN 1996. LNCS, vol. 1141, pp. 178–187. Springer, Heidelberg (1996)
11. Murphy, K.: The BayesNet toolbox for Matlab. Computer Science and Statistics: Proceedings of Interface 33 (2001)
12. Prinz, A.A.: Neuronal parameter optimization. Scholarpedia 1(7), 1903 (2007)
13. Prinz, A.A., Billimoria, C.P., Marder, E.: Alternative to hand-tuning conductance-based models: Construction and analysis of databases of model neurons. Journal of Neurophysiology 90(6), 3998–4015 (2003)
14. Prinz, A.A., Thirumalai, V., Marder, E.: The functional consequences of changes in the strength and duration of synaptic inputs to oscillatory neurons. The Journal of Neuroscience 23(3), 943–954 (2003)
15. Santana, R., Bielza, C., Larrañaga, P., Lozano, J.A., Echegoyen, C., Mendiburu, A., Armañanzas, R., Shakya, S.: MATEDA: A Matlab package for the implementation and analysis of estimation of distribution algorithms. Journal of Statistical Software (2010) (accepted for publication)
16. Santana, R., Bielza, C., Lozano, J.A., Larrañaga, P.: Mining probabilistic models learned by EDAs in the optimization of multi-objective problems. In: Proceedings of the 11th Annual Genetic and Evolutionary Computation Conference GECCO 2009, pp. 445–452. ACM, New York (2009)
17. Santana, R., Larrañaga, P., Lozano, J.A.: Adding probabilistic dependencies to the search of protein side chain configurations using EDAs. In: Rudolph, G., Jansen, T., Lucas, S., Poloni, C., Beume, N. (eds.) PPSN 2008. LNCS, vol. 5199, pp. 1120–1129. Springer, Heidelberg (2008)
18. Santana, R., Ochoa, A., Soto, M.R.: The mixture of trees factorized distribution algorithm. In: Proceedings of the Genetic and Evolutionary Computation Conference GECCO 2001, pp. 543–550. Morgan Kaufmann Publishers, San Francisco (2001)
19. Smolinski, T.G., Prinz, A.: Computational intelligence in modeling of biological neurons: A case study of an invertebrate pacemaker neuron. In: Proceedings of the International Joint Conference on Neural Networks, Atlanta, Georgia, pp. 2964–2970. IEEE Computer Society Press, Los Alamitos (2009)
20. Taylor, A.L., Hickey, T.J., Prinz, A.A., Marder, E.: Structure and visualization of high-dimensional conductance spaces. Journal of Neurophysiology 96, 891–905 (2006)
21. Turrigiano, G.G., LeMason, G., Marder, E.: Selective regulation of current densities underlies spontaneous changes in the activity of cultured neurons. Journal of Neuroscience 15, 1129–1131 (1995)
22. Van Geit, W., De Schutter, E., Achard, P.: Automated neuron model optimization techniques: A review. Biological Cybernetics 99, 241–251 (2007)

The Informative Extremes: Using Both Nearest and Farthest Individuals Can Improve Relief Algorithms in the Domain of Human Genetics

Casey S. Greene, Daniel S. Himmelstein, Jeff Kiralis, and Jason H. Moore

Dartmouth Medical School, Lebanon, NH 03756, USA
Jason.H.Moore@dartmouth.edu
http://www.epistasis.org

Abstract. A primary goal of human genetics is the discovery of genetic factors that influence individual susceptibility to common human diseases. This problem is difficult because common diseases are likely the result of joint failure of two or more interacting components instead of single component failures. Efficient algorithms that can detect interacting attributes are needed. The Relief family of machine learning algorithms, which use nearest neighbors to weight attributes, are a promising approach. Recently an improved Relief algorithm called Spatially Uniform ReliefF (SURF) has been developed that significantly increases the ability of these algorithms to detect interacting attributes. Here we introduce an algorithm called SURF* which uses distant instances along with the usual nearby ones to weight attributes. The weighting depends on whether the instances are are nearby or distant. We show this new algorithm significantly outperforms both ReliefF and SURF for genetic analysis in the presence of attribute interactions. We make SURF* freely available in the open source MDR software package. MDR is a cross-platform Java application which features a user friendly graphical interface.

1 Introduction

New genotyping technologies are allowing human geneticists to routinely measure individual genetic variation on a vast "genome-wide" scale [1,2,3]. It is now feasible to measure more than one million variations from across the human genome. Here we focus on a particular type of variation, the single nucleotide polymorphism or SNP. Each SNP is a single point in a DNA sequence that differs between individuals. A major goal of human genetics is to link these genetic variations to disease risk [4]. Currently this problem is approached as a set of independent steps. The first step in the process is to discover SNPs that reliably predict disease susceptibility across many samples [5], but discovery of these robust single predictors has proven difficult [6,7,8]. Furthermore even the reliable and robust disease-associated SNPs that have been discovered often cannot be combined into effective classifiers of disease risk [9]. These association studies, by their nature, ignore complex interactions that may lead to disease susceptibility.

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2010, LNCS 6023, pp. 182–193, 2010.

The term for complex gene-gene interactions that influence a trait such as disease susceptibility is epistasis. It is becoming apparent that studies ignoring epistasis are also likely to be neglecting informative markers [10,11]. Because of the complexity present in cellular and biological systems, epistasis is thought to be fundamental to an individual's risk for common human diseases [12]. This knowledge, combined with the inability of single-marker approaches to offer predictive models of individual disease risk, suggests that researchers should also carefully examine gene-gene interactions for associations with disease. Unfortunately examining the joint effect of these polymorphisms is difficult because commonly used methods are combinatorial.

Relief algorithms [13], which use nearest neighbors, have successfully detected gene-gene interactions in genetic association studies [14]. Here we introduce a novel Relief algorithm called SURF*. SURF* is better able than other Relief algorithms to detect SNPs which participate in epistatic interactions that relate to disease risk. The novel feature of SURF* is that it uses distant individuals, as well as the usual near ones, to adjust the scores of SNPs. Using these distant individuals has the effect of increasing sample size considerably.

This paper is organized as follows. Section 1.1 discusses approaches used in genetic association studies. Section 2 discusses intuitively how Relief algorithms can, in linear time with respect to the number of SNPs, detect epistatic interactions. Section 2.1 examines how SURF specifically is able to detect these interactions. This is important because we improve SURF with a novel approach, SURF*. A theoretical assessment of the improvement provided by SURF* is described in Section 3. We evaluate the new SURF* method empirically using a study design described in Section 4. This framework allows us to directly assess the success rate of the method. The results of the simulation are discussed in Section 5 and we discuss their implications in Section 6.

1.1 Related Work

The state of the art in this field still relies on only the analysis of single SNPs as in a recent large study of 17,000 individuals and seven common diseases from the Wellcome Trust Case Control Consortium [15]. While some approaches do consider complexity, these often condition on the effect of single SNPs or require combinatorial methods to exhaustively examine all potential interactions [16,17]. In the first case, these have the potential to miss interactions without main effects. In the second, the time to analyze large datasets becomes prohibitive because this type of analysis requires the consideration of the joint effect of attributes, here SNPs, a combinatorial challenge which has been previously described [18,19,20]. When datasets contain many SNPs, such combinatorial methods are infeasible.

2 Relief Algorithms

Cordell [21] provides a recent and thorough review of current analysis methods, including Relief algorithms, for these studies as well as the potential benefits and

drawbacks of each. To this point the use of Relief algorithms in this field has been relatively limited [14,22,23], probably because previous small scale studies have not required these types of algorithms, and large scale studies have, thus far, often ignored epistasis. Given the difficulty of detecting predictive interacting SNPs, our novel and more effective Relief algorithm could greatly enhance the state of the field.

Relief algorithms, the first of which was developed by Kira and Rendell [13], are a natural fit for large scale genetic association studies designed to detect epistasis. They are fast and scale linearly with the number of SNPs and quadratically with the number of individuals. Furthermore these algorithms are able to detect interacting pairs of attributes that contribute to disease susceptibility. We have previously discussed how Relief algorithms do this from a mathematical point of view [24]. In summary, the Relief algorithm returns a weight for each SNP. Higher scores indicate that a SNP is more likely to be predictive of disease status. The adjustment of these scores is performed using the genetically most similar individuals. Here the inter-individual distance is the number of SNPs with differing genotypes between two individuals. Therefore, nearest individuals share the greatest number of genotypes. Relief works on the assumption that the SNPs of nearby individuals with different genotypes are most useful for assessing the predictiveness of the SNP. The algorithm adjusts the scores of these SNPs– upward if the two individuals have different disease status, and downward by the same amount if they have the same status. More precisely, for each individual I_i, SNP scores are adjusted using its nearest hit (the individual which is closest to I_i and in the same class as I_i) and its nearest miss, (the individual which is closest to I_i and in the other class from I_i). ReliefF [25] differs from Relief largely because it uses multiple neighbors for weighting instead of only the single nearest neighbor.

Table 1. Penetrance values for an example epistasis model with a heritability of 0.2.

		SNP_1		
		AA (0.36)	Aa (0.48)	aa (0.16)
SNP_2	BB (0.36)	0.393	0.764	0.664
	Bb (0.48)	0.850	0.398	0.733
	bb (0.16)	0.406	0.927	0.147

2.1 Spatially Uniform ReliefF (SURF)

Spatially Uniform ReliefF (SURF), developed by Greene et al. [24], detects attribute interactions in the same manner as Relief and ReliefF. SURF, like ReliefF, uses multiple nearest neighbors, but, instead of using a fixed number of nearest neighbors, SURF uses all neighbors within a specific similarity threshold, T. Instances may not be uniformly distributed in space and some instances may have more informative neighbors than other instances. SURF uses all neighbors more similar than the threshold, T, for weighting, while Relief and ReliefF may use either more or fewer neighbors. This can cause ReliefF to potentially include

uninformative neighbors or to neglect informative ones. This swaps the number-of-neighbors used by Relief for the similarity-threshold used by SURF. For this we use the mean of the distances between all pairs of individuals, which can be easily computed from the data [24].

Here we will briefly outline how SURF is capable of detecting interacting pairs of attributes. This is thoroughly discussed in the appendix to Greene et al. [24] but here we highlight the parts necessary to understand how SURF* improves on SURF and adjust the notation to accommodate both the nearest and furthest individuals. To understand these algorithms, it is first necessary to understand the problem. We illustrate the situation of interacting pairs of SNPs using the penetrance table given in Table 1. According to this example, if an individual has genotype BB, the probability she is sick is $.36 \cdot .393 + .48 \cdot .764 + .16 \cdot .664 \approx .614$. If she has genotype Bb, this probability is the same, and likewise if she has genotype bb. Thus just SNP 2's genotype is not predictive of disease status. Similarly if SNP 1's genotype is known, but not SNP 2's, the probability she is sick is as before, $.36 \cdot .764 + .48 \cdot .398 + .16 \cdot .927 \approx .614$. Thus the genotypes of SNPs 1 and 2 are together predictive of disease status, but neither is individually. This is what makes SNPs 1 and 2 an epistatic pair of SNPs. In our study we employ 9000 datasets from 30 of these genetic models. In all models there are pairs of SNPs which are jointly predictive but no singly informative SNPs. Detecting these epistatic pairs is much more difficult then detecting SNPs which alone have an effect.

A basic fact we will use about epistatic pairs is that

$$|H_{2\Delta}| - |M_{2\Delta}| = \frac{1}{2}(|M_{1\Delta}| - |H_{1\Delta}|) = |H_{0\Delta}| - |M_{0\Delta}|. \tag{1}$$

This is discussed in sections 1 and 2 of the appendix in the paper first describing SURF [24].

Now let I_i be a random, but fixed, individual and let T be the threshold distance. Then each miss with distance less than T from I_i is in one of the three sets $M_{0\Delta}$, $M_{1\Delta}$ or $M_{2\Delta}$. For $k = 0$, 1 and 2, let $CM_{k\Delta}$ be the subset of $M_{k\Delta}$ consisting of those individuals with distance $< T$ from I_i. The notation $CM_{k\Delta}$ might be read as "close misses involving k changes of the relevant SNPs". Using analogous notation for hits with H in place of M, the contribution of individual I_i to the (SURF) score of a relevant SNP is

$$\begin{aligned} S_i^C &= \frac{1}{2}(|CM_{1\Delta}| - |CH_{1\Delta}|) + (|CM_{2\Delta}| - |CH_{2\Delta}|) \\ &= \frac{1}{2}(|CM_{1\Delta}| - |CH_{1\Delta}|) - (|CH_{2\Delta}| - |CM_{2\Delta}|). \end{aligned} \tag{2}$$

The $\frac{1}{2}$ is here since each individual in $CM_{1\Delta}$ and $CH_{1\Delta}$ changes the score of a relevant SNP by $\frac{1}{2}$, on the average. The total SURF score of a relevant SNP is the sum of the S_i^C over all individuals.

It follows from equation (1) that if arbitrary neighbors are used, rather than nearest ones, the expected score of a relevant SNP would be 0 since

$$\frac{1}{2}(|M_{1\Delta}| - |H_{1\Delta}|) = |H_{2\Delta}| - |M_{2\Delta}|.$$

The score S_i^C tends to be positive though because close neighbors are more apt to lie in the sets $M_{1\Delta}$ and $H_{1\Delta}$ rather than in $M_{2\Delta}$ and $H_{2\Delta}$, making

$$\frac{1}{2}(|CM_{1\Delta}| - |CH_{1\Delta}|) > |CH_{2\Delta}| - |CM_{2\Delta}|.$$

The reason close neighbors are more apt to lie in the 1Δ-sets than in the 2Δ ones is that relevant SNPs of individuals in the 1Δ-sets contribute one to the distance from I_i, while those in the 2Δ-sets contribute two.

3 The Value of Both Nearest and Farthest

It is clear that the assumption made by Relief algorithms such as SURF, that the SNPs of nearby individuals with different genotypes are useful for assessing the predictiveness of the SNP, is correct as these algorithms are successful. It is not clear that distant individuals are not also useful. Our analysis suggests that using the states of genotypes for these most distant individuals can substantially improve the success rates of these algorithms. Using this information effectively increases the sample size available to SURF greatly improving its ability to detect epistatic SNPs when sample sizes are limited. We call the algorithm SURF* because using distant individuals is the opposite of SURF and because, in mathematics, * indicates opposite. Strictly speaking, the SURF* that we discuss includes both SURF and this additional opposite component.

We outline how this approach using both closest and farthest individuals outperforms the nearest neighbor approaches. The SURF* algorithm we introduce uses nearby neighbors in the same way the SURF algorithm does. The new part of the SURF* algorithm using distant individuals identifies those SNPs of distant individuals in different states and adjusts their scores–downward by one if the two individuals have different disease status, and upward by the same amount if they have the same status. (This is the same as with Relief algorithms, but upward and downward have been interchanged.) Specifically, we define subsets $DM_{k\Delta}$ made up of distant misses of $M_{k\Delta}$ consisting of those misses with distance $> T$ from I_i. Subsets $DH_{k\Delta}$ made up of distant hits of $H_{k\Delta}$ consist of those hits with distance $> T$ from I_i Then the contribution of individual I_i to the (distant individuals) score of a relevant SNP is

$$\begin{aligned} S_i^D &= -\frac{1}{2}(|DM_{1\Delta}| - |DH_{1\Delta}|) - (|DM_{2\Delta}| - |DH_{2\Delta}|) \\ &= -\frac{1}{2}(|DM_{1\Delta}| - |DH_{1\Delta}|) + (|DH_{2\Delta}| - |DM_{2\Delta}|). \end{aligned} \tag{3}$$

The mean of this is positive for essentially the same reason that the mean of S_i^C is. Namely, individuals in the 2Δ-group tend to be one farther from I_i than those in the 1Δ-group.

The means of S_i^C and S_i^D are the same, or nearly so. So the mean of the overall score $\Sigma_i S_i^C + \Sigma_i S_i^D$ of a relevant SNP is doubled by using distant individuals along with the usual close ones. We suspect that $\Sigma_i S_i^C$ and $\Sigma_i S_i^D$ are not independent. If so, with V denoting variance, we have

$$V(\Sigma_i S_i^C + \Sigma_i S_i^D) > V(\Sigma_i S_i^C) + V(\Sigma_i S_i^D).$$

Thus using distant individuals does not quite have the effect of doubling the sample size, but it does substantially increase the success rate. This improvement in success rate indicates that these methods are more likely to detect interacting relevant SNPs in these genetic association studies.

4 Experimental Design

Here we evaluate these methods in the context of a simulation study. The goal of our simulation study is to generate artificial datasets with high concept difficulty to evaluate these methods in the domain of human genetics. Our dataset characteristics were chosen to closely match common genetic association study designs from human genetics. We first develop 30 different penetrance functions (i.e. genetic models) which determine the relationship between genotype and phenotype in our simulated data. These functions determine the probability that an individual has the studied disease given his or her genotype. This probability depends only on the genotypes of the two interacting SNPs, not on the genotype of any one SNP. This case where there are no single SNP effects is thought to be the most difficult. Single SNP effects are easily found with other methods. The 30 penetrance functions consist of six groups of five with heritabilities of 0.025, 0.05, 0.1, 0.2, 0.3, or 0.4. Each of the six heritabilities is realized by all five models in one group. These heritabilities range from very small to large genetic effect sizes and thus test the algorithms across a broad swathe of scenarios.

SNPs are chosen for genotyping such that each SNP has two alleles due to technological constraints and such that these alleles are both common in the population. Here each model contains SNPs with two alleles which have frequencies of 0.4 and 0.6. Each of the models is used to generate 100 datasets with sample sizes of 800, 1600, and 3200. Studies with 800 individuals would be considered small relative to other genetic association studies while studies with 3200 individuals would be considered large. Each consists of an equal number of case and control subjects because genetic association studies are frequently designed to be balanced. Each pair of relevant SNPs is added to a set of 998 irrelevant SNPs for a total of 1000 attributes. This is similar to the size seen in association studies using custom SNP arrays to perform genotyping. A total of 9,000 datasets are generated and analyzed. This large number of datasets and study design allows us to rigorously evaluate and compare these methods across situations likely to be encountered. Due to the difficulty of detecting and characterizing epistasis,

well studied real datasets of these sizes where epistatic interactions have been validated are not widely available.

By performing a simulation study it is possible to determine the success rate of a method. This is possible because the relevant SNPs are known before the algorithm is applied to the data. The success rate is the percentage of time that a method scores both relevant SNPs above a given threshold. To estimate it, we use all 100 datasets for each of the 30 models. Specifically, the percentage of datasets in which a method ranks the two relevant SNPs above the N^{th} percentile of all SNPs is the estimate of the method's success rate. We examine the 95^{th} percentile because this is likely to be useful in practice and because ReliefF has been used in the genetic analysis of complex diseases in this fashion [14]. This represents the situation where the method filters a dataset with 1000 SNPs to 50 SNPs before a combinatorial analysis is performed on this manageable subset.

It is also important to understand whether differences observed between the estimates of success rates for the various methods are due to chance or are due to a true performance difference. To determine whether differences between success rates at these thresholds are likely due to chance, we apply Fisher's exact test to assess the significance of these differences. Fisher's exact test is a significance test appropriate for categorical count data such as success rate [26]. The resulting p-value for this test can be interpreted as the likelihood of seeing a difference of the size observed among success rates when the methods do not differ. We consider results statistically significant when $p \leq 0.05$. Additionally, we graphically show results for filtering to each percentile from the 99^{th} to the 50^{th}. Highly significant results indicate that the observed differences are unlikely to be due to chance.

We test each method using parameter settings from Greene et al. [24]. ReliefF requires that a number of neighbors be specified. In 2003 Robnik-Sikonja and Kononenko [27] performed a comprehensive analysis and determined that ten neighbors was an appropriate number for ReliefF, so we use ten neighbors here. Similarly, SURF requires a distance threshold. Greene et al. [24] suggest that the mean distance can be used as an acceptable threshold and thus we use the mean distance in this situation. To facilitate comparison between these methods we do not use a distance decay, although in future studies altering this parameter could allow for further improvement in success rate because the distance decay increases the influence of the most extreme individuals.

5 Empirical Results

The novel method, SURF*, which uses both near and far individuals for weighting, significantly outperforms both the SURF and ReliefF methods that use only nearby individuals. Figure 1 shows an example plot for a specific sample size (1600) and heritability (0.2) combination. This figure summarizes the success rate estimated from analysis of 500 simulated independent datasets with this heritability and sample size. The arrows on the right side of the graph indicate whether the methods varied significantly in their abilities to successfully filter a dataset to the 95th percentile (i.e. filter a dataset of 1000 SNPs to 50 SNPs without removing either relevant SNP). In this case the differences between all three

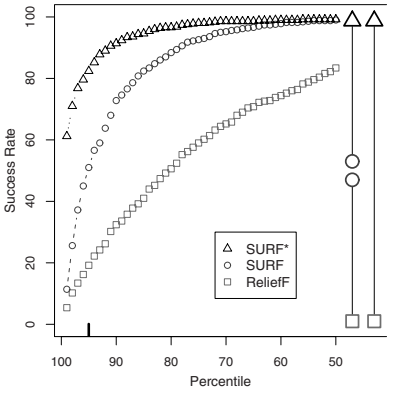


Fig. 1. This figure shows success rate analysis results for an example heritability (0.2) and sample size (1600). The arrows on the right side of the graph indicate whether the methods varied significantly in their abilities to successfully filter a dataset to the 95th percentile (shown by the tick mark above the x-axis between the 100^{th} and 90^{th} percentiles). The caps of the arrows illustrate which two methods are being compared, and the line connecting these caps indicates the level of significance of the differences between this pair of methods (no line represents $p \geq 0.05$, a dotted line represents $0.01 \leq p < 0.05$, a dashed line represents $0.001 \leq p < 0.01$, and a solid line represents $p < 0.001$. In this case the differences between all three methods were highly significant.

methods were highly significant. These results indicate clear differences between these methods for this heritability and sample size. Furthermore the differences observed were highly significant ($p \leq 0.001$) indicating that differences of this magnitude are likely to be observed by chance less than one time out of 1000. While this figure shows clear differences in success rate at this heritability and sample size, it is most informative to consider an algorithm's performance a wide range of potential use cases.

Figure 2 shows results as small multiples of the example shown in Figure 1 across all tested sample sizes and heritabilities. Each plot represents results for 500 datasets with the specified sample size and heritability. None of the methods perform particularly well at the lowest sample sizes and heritabilities. That is, when the genetic effect is smallest, a larger study would be needed to discover the relevant SNPs. This is well known in genetics and, fortunately, studies aiming to detect smaller effects are designed to contain more individuals. Also as expected, at the highest sample sizes and heritabilties all of the methods perform well.

The range where results are most similar to what would be seen in practice, the simulations with 1600 individuals and modest heritabilities, 800 individuals and high heritabilities, and 3200 individuals with lower heritabilities are also the areas where SURF* outperforms other methods by the widest margin. In these bands the differences between SURF* and the other methods are highly statistically significant. These results indicate that SURF* greatly improves upon currently used approaches. SURF*'s consistently high performance indicates that

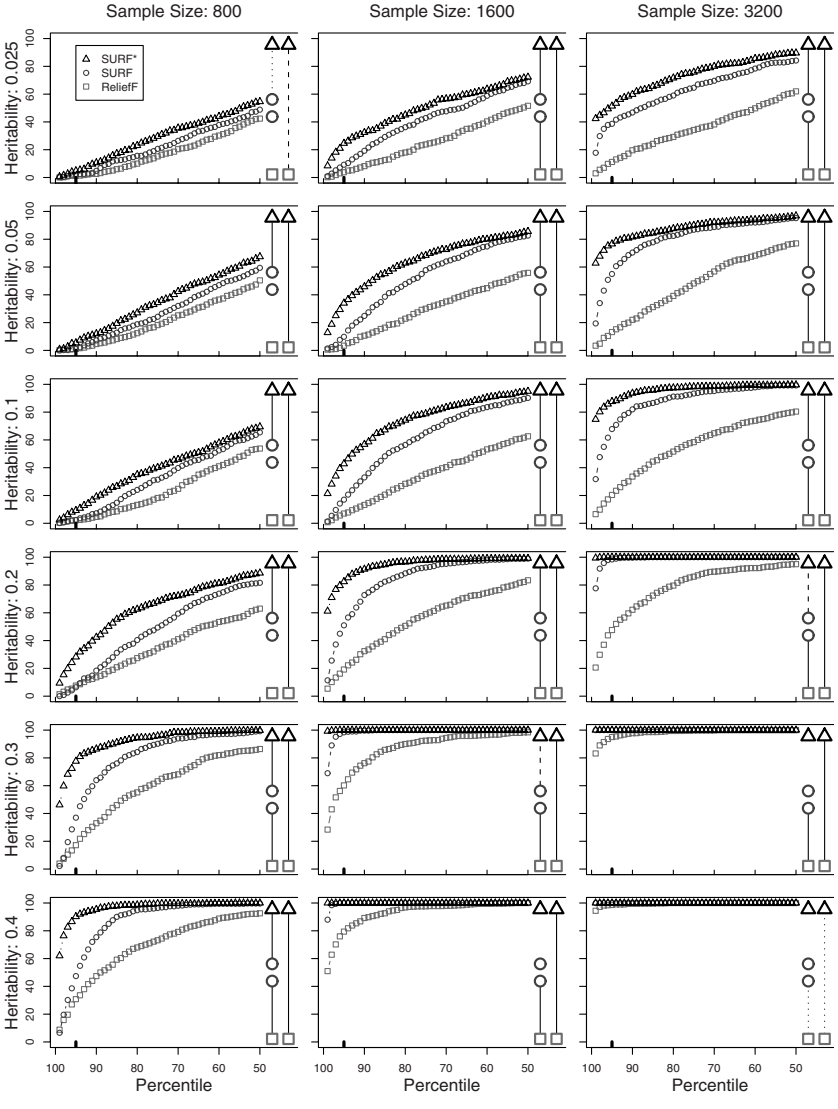


Fig. 2. This is a summary of success rates as shown in Figure 1 across a wide range of sample sizes and heritabilities. The x-axis for each plot corresponds to the percentiles as in Figure 1. The y-axis corresponds to the success rate. Significance is shown with arrows as described in Figure 1. Across these situations, SURF* outperforms both existing methods.

it should be used in place of SURF when the goal is to detect SNPs predictive of disease through epistatic interactions. While here we are most interested in the ability to filter a dataset of 1000 SNPs to a smaller dataset of 50 SNPs which can be combinatorially analyzed, it is important to note that across the

entire range of percentiles examined, SURF* outperforms currently used methods. This indicates that when used for more or less stringent filtering, SURF* is still more effective than currently existing methods. Using both the nearest and farthest individuals greatly and significantly improves SURF's ability to detect SNPs which interact to predict disease.

6 Discussion and Conclusions

Epistatic interactions have often been shown to affect complex traits in model organisms, and thus it would be prudent to consider the potential role of epistasis on the complex traits of common human disease susceptibilities [28,29]. Unfortunately epistasis is not often considered because an exhaustive analysis is computationally intractable [20]. Machine learning methods such as SURF offer promise but these approaches must be modified to cope with the small sample sizes and large number of attributes present in high throughput genetic datasets. Our theoretical work in Section 3 suggests that SURF*, which uses a greater number of individuals for attribute weighting than SURF, will be a more powerful way to approach this problem. We observe this effect in our empirical results (Section 5). Using the farthest individuals in addition to the nearest ones greatly increases the success rates of these methods at moderate sample sizes and heritabilities. Additionally, these improvements may generalize to other Relief algorithms and could increase their ability to detect interactions.

Here we examine the role of these Relief algorithms in isolation, but it is important to note that these can be used in conjunction with other information sources as well during a genetic analysis [23]. Improved Relief algorithms should offer an immediate increase in success rate to detect interactions when they are used in place of current algorithms as information sources for these methods. SURF* does perform more weighting due to the increased number of individuals that are used, but with SURF* it is no longer necessary to find the nearest individuals so the computational costs remain relatively similar. A method which provides a significant increase in success rate is likely to improve our understanding of common human diseases.

Future work should focus on effective and efficient methods to assess the significance of discovered SNPs. Relief methods return scores which are a measure of SNP quality but which are not easily converted to statistical significance. Additionally, work should be done to develop powerful Relief methods capable of detecting interactions between discrete and continuous variables and endpoints. Genetic association studies often include SNPs, which are discrete, in addition to measures of the environment, which are continuous. Methods capable of detecting gene-gene, gene-environment, and environment-environment interactions will therefore be useful. Relief methods capable of examining continuous data exist [30,27], but they should be rigorously evaluated for their ability to detect interactions between discrete and continuous attributes. The impact of including farthest individuals on the success of those algorithms should also be examined.

7 Method Availability

SURF* is freely available in the open source MDR software package from http://sourceforge.net/projects/mdr/. MDR is a cross-platform Java application which features a user friendly graphical interface.

Acknowledgement

This work was supported by NIH grants LM009012, AI59694, HD047447, and ES007373.

References

1. Gunderson, K.L., Steemers, F.J., Lee, G., Mendoza, L.G., Chee, M.S.: A genome-wide scalable SNP genotyping assay using microarray technology. Nat. Genet. 37(5), 549–554 (2005)
2. Steemers, F.J., Gunderson, K.L.: Whole genome genotyping technologies on the BeadArray platform. Biotechnology Journal 2(1), 41–49 (2007)
3. Thomas, D.C., Haile, R.W., Duggan, D.: Recent developments in genomewide association scans: A workshop summary and review. Am. J. Hum. Genet. 77(3), 337–345 (2005)
4. Chanock, S., Taylor, J.G.: Using genetic variation to study immunomodulation. Current Opinion in Pharmacology 2(4), 463–469 (2002)
5. McCarthy, M.I., Abecasis, G.R., Cardon, L.R., Goldstein, D.B., Little, J., Ioannidis, J.P.A., Hirschhorn, J.N.: Genome-wide association studies for complex traits: consensus, uncertainty and challenges. Nat. Rev. Genet. 9(5), 356–369 (2008)
6. Hirschhorn, J.N., Lohmueller, K., Byrne, E., Hirschhorn, K.: A comprehensive review of genetic association studies. Genet. Med. 4, 45–61 (2002)
7. Shriner, D., Vaughan, L.K., Padilla, M.A., Tiwari, H.K.: Problems with Genome-Wide association studies. Science 316(5833), 1840–1841 (2007)
8. Williams, S.M., Canter, J.A., Crawford, D.C., Moore, J.H., Ritchie, M.D., Haines, J.L.: Problems with Genome-Wide association studies. Science 316(5833), 1841–1842 (2007)
9. Jakobsdottir, J., Gorin, M.B., Conley, Y.P., Ferrell, R.E., Weeks, D.E.: Interpretation of genetic association studies: Markers with replicated highly significant odds ratios may be poor classifiers. PLoS Genetics 5(2), e1000337 (2009)
10. Moore, J.H.: The ubiquitous nature of epistasis in determining susceptibility to common human diseases. Human Heredity 56, 73–82 (2003)
11. Phillips, P.C.: Epistasis – the essential role of gene interactions in the structure and evolution of genetic systems. Nat. Rev. Genet. 9(11), 855–867 (2008)
12. Tyler, A.L., Asselbergs, F.W., Williams, S.M., Moore, J.H.: Shadows of complexity: what biological networks reveal about epistasis and pleiotropy. BioEssays 31(2), 220–227 (2009)
13. Kira, K., Rendell, L.A.: A practical approach to feature selection, pp. 249–256 (1992)
14. Beretta, L., Cappiello, F., Moore, J.H., Barili, M., Greene, C.S., Scorza, R.: Ability of epistatic interactions of cytokine single-nucleotide polymorphisms to predict susceptibility to disease subsets in systemic sclerosis patients. Arthritis and Rheumatism 59(7), 974–983 (2008)

15. The Wellcome Trust Case Control Consortium: Genome-wide association study of 14,000 cases of seven common diseases and 3,000 shared controls. Nature 447(7145), 661–678 (2007)
16. Gayan, J., Gonzalez-Perez, A., Bermudo, F., Saez, M., Royo, J., Quintas, A., Galan, J., Moron, F., Ramirez-Lorca, R., Real, L., Ruiz, A.: A method for detecting epistasis in genome-wide studies using case-control multi-locus association analysis. BMC Genomics 9(1), 360 (2008)
17. Ritchie, M.D., Hahn, L.W., Roodi, N., Bailey, L.R., Dupont, W.D., Parl, F.F., Moore, J.H.: Multifactor-dimensionality reduction reveals high-order interactions among estrogen-metabolism genes in sporadic breast cancer. Am. J. Hum. Genet. 69(1), 138–147 (2001)
18. Cordell, H.J.: Epistasis: what it means, what it doesn't mean, and statistical methods to detect it in humans. Hum. Mol. Genet. 11(20), 2463–2468 (2002)
19. Freitas, A.A.: Understanding the crucial role of attribute interaction in data mining. Artif. Intell. Rev. 16(3), 177–199 (2001)
20. Moore, J.H., Ritchie, M.D.: The challenges of Whole-Genome approaches to common diseases. JAMA 291(13), 1642–1643 (2004)
21. Cordell, H.: Detecting gene-gene interactions that underlie human diseases. Nature Reviews Genetics 10(6), 392–404 (2009)
22. McKinney, B., Reif, D., White, B., Crowe, J., Moore, J.: Evaporative cooling feature selection for genotypic data involving interactions. Bioinformatics 23(16), 2113–2120 (2007)
23. McKinney, B.A., Crowe, J.E., Guo, J., Tian, D.: Capturing the spectrum of interaction effects in genetic association studies by simulated evaporative cooling network analysis. PLoS Genet. 5(3), e1000432 (2009)
24. Greene, C.S., Penrod, N.M., Kiralis, J., Moore, J.H.: Spatially Uniform ReliefF (SURF) for Computationally-efficient Filtering of Gene-gene Interactions. BioData Mining 2, 5 (2009)
25. Kononenko, I.: Estimating attributes: Analysis and extensions of RELIEF. In: Bergadano, F., De Raedt, L. (eds.) ECML 1994. LNCS, vol. 784, pp. 171–182. Springer, Heidelberg (1994)
26. Sokal, R.R., Rohlf, F.J.: Biometry: the principles and practice of statistics in biological research, 3rd edn. W. H. Freeman and Co., New York (1995)
27. Robnik-Sikonja, M., Kononenko, I.: Theoretical and empirical analysis of relieff and rrelieff. Mach. Learn. 53, 23–69 (2003)
28. Kroymann, J., Mitchell-Olds, T.: Epistasis and balanced polymorphism influencing complex trait variation. Nature 435(7038), 95–98 (2005)
29. Shao, H., Burrage, L.C., Sinasac, D.S., Hill, A.E., Ernest, S.R., O'Brien, W., Courtland, H., Jepsen, K.J., Kirby, A., Kulbokas, E.J., Daly, M.J., Broman, K.W., Lander, E.S., Nadeau, J.H.: Genetic architecture of complex traits: Large phenotypic effects and pervasive epistasis. Proc. Nat. Acad. Sci. 105(50), 19910–19914 (2008)
30. Robnik-Sikonja, M., Kononenko, I.: An adaptation of relief for attribute estimation in regression. In: ICML 1997: Proceedings of the Fourteenth International Conference on Machine Learning, pp. 296–304 (1997)

Artificial Immune Systems for Epistasis Analysis in Human Genetics

Nadia M. Penrod, Casey S. Greene, Delaney Granizo-MacKenzie, and Jason H. Moore

Dartmouth Medical School, Lebanon, NH 03756, USA
Jason.H.Moore@dartmouth.edu
http://www.epistasis.org

Abstract. Modern genotyping techniques have allowed the field of human genetics to generate vast amounts of data, but analysis methodologies have not been able to keep pace with this increase. In order to allow personal genomics to play a vital role in modern health care, analysis methods capable of discovering high order interactions that contribute to an individual's risk of disease must be developed. An artificial immune system (AIS) is a method which maps well to this problem and has a number of appealing properties. By considering many attributes simultaneously, it may be able to effectively and efficiently detect epistasis, that is non-additive gene-gene interactions. This situation of interacting genes is currently very difficult to detect without biological insight or statistical heuristics. Even with these approaches, at low heritability (i.e. where there is only a small genetic signal), these approaches have trouble distinguishing genetic signal from noise. The AIS also has a compact solution representation which can be rapidly evaluated. Finally the AIS approach, by iteratively developing an antibody which ignores irrelevant genotypes, may be better able to differentiate signal from noise than machine learning approaches like ReliefF which struggle at small heritabilities. Here we develop a basic AIS and evaluate it on very low heritability datasets. We find that the basic AIS is not robust to parameter settings but that, at some parameter settings, it performs very effectively. We use the settings where the strategy succeeds to suggest a path towards a robust AIS for human genetics. Developing an AIS which succeeds across many parameter settings will be critical to prepare this method for widespread use.

1 Introduction

Personal genomics is expected to play a central role in the realization of personalized medicine. The definition of personal genomics includes not only the unbiased collection of genetic information from individuals but also the bioinformatic analyses used to extract meaningful knowledge regarding disease status or risk from this genetic information. The common approach is to design genome-wide association studies which use statistics to correlate genetic variation, in the form of single nucleotide polymorphisms (SNPs), with disease. While this method has

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2010, LNCS 6023, pp. 194–204, 2010.

produced some promising results the majority of genetic markers detected make only minor contributions to an individual's disease risk. One explanation is that underlying most common diseases is a complex genetic architecture defined by non-linear gene-gene interactions (epistasis) that complicate genotype to phenotype mapping [1]. In order to capture a more accurate picture of the health implications stored within the genome we need to develop algorithms capable of finding these non-linear interactions in genome-wide data.

The term epistasis has been used to describe a variety of genetic phenomena from biological interactions at the molecular level to statistical interactions at the population level [2,3]. Here we focus on statistical epistasis meaning the detection of patterns, of genetic variation, that collectively give rise to a phenotype. We are interested in developing algorithms capable of mining large case-control data sets to uncover combinations of SNPs that best differentiate the disease status of individuals. Our view of a complex genetic architecture, based on gene-gene interactions, has been substantiated in the recent literature across many common disease models including Crohn's Disease, bipolar disorder, hypertension and rheumatoid arthritis [4]. These reports describe two-way interactions, however, with appropriate methods and enough computational power it is likely that higher-order interactions can be found. We have begun working with an artificial immune system (AIS) algorithm as a means of finding these high-order interactions which may serve as SNP signatures of disease.

AIS algorithms are adaptive systems metaphorically based on the vertebrate immune system [5]. AISs mimic the biological functions of learning and memory from the adaptive immune response displaying many features desirable for problem solving including pattern recognition, noise tolerance and robustness making them applicable across a variety of problem domains. This work assesses the potential of taking an AIS approach to finding high-order epistatic patterns in simulated genome-wide data. In this paper, we show that the most basic AIS method cannot outperform a random search for this problem. However, our results clearly indicate that with the appropriate modifications, the AIS approach has the potential to detect epistasis.

We discuss related work in human genetics in Section 1.1. The implementation and evaluation of the AIS is outlined in Section 2. Specifically Section 2.1 discusses the AIS. Section 2.2 discusses the random search algorithm that was used for comparison. Sections 2.3 and 2.4 lay out the experimental design and the statistical analysis performed. Experimental results are presented in Section 3. The discussion and concluding remarks are imparted in Section 4.

1.1 Related Work

From a methodological standpoint the AIS approach is probably best compared and contrasted with two previous evolutionary computing approaches. One previous approach, Genetic Programming Neural Networks (GPNN), was initially introduced by Ritchie et al. [6]. In GPNN the initial solutions include many attributes representing genetic variation and can include all attributes as inputs. This allows the algorithm to effectively sift through attribute combinations

to find epistatic relationships between genes that associate with disease. Motsinger-Reif and Ritchie provide a recent review of GPNN [7]. The other previous approach we compare to is ant colony optimization with multifactor dimensionality reduction [8]. In this approach single SNPs are combined via an ant colony metaheuristic. This approach uses single-SNP weights to probabilistically link SNPs. This type of approach is unable to outperform random search without heuristic information, although with the addition of heuristic information it does succeed.

Like GPNN, the AIS strategy can consider all attributes concurrently. The expectation is that this property will allow the algorithm to effectively discover accurate models of individual disease risk. Like the ant colony approach, the solution representation is relatively simple. By potentially considering all attributes while preserving a simple but flexible solution representation, the AIS strategy has the potential to effectively discover interpretable models of disease risk without requiring expert knowledge. Here we are comparing these approaches qualitatively. Once a robust AIS is developed for this problem, a quantitative comparison should be performed and the results rigorously analyzed.

Finally, the AIS should be compared to the ReliefF algorithm [9] which has proven useful in addressing questions in human genetics [10,11,12]. The affinity measure of the AIS approach is somewhat similar to the distance measure used in Relief algorithms [13]. We are not the first to draw this parallel between Relief algorithms and the AIS strategy. Bereta and Burczynski [14] have previously acknowledged the similarity and point out that the hybridization of the classification strategy of the AIS with the feature selection strategy of ReliefF may be advantageous. The benefit of combining these strategies may be most apparent when the genetic effect size is small. Additional improvements may be made to increase the power of the AIS algorithm by incorporating some of the characteristics that allow Relief to succeed. This could include steps such as training on neighborhoods of individuals.

2 Artificial Immune Systems

Artificial immune systems are a class of nature-inspired algorithms modeled after known biological phenomena. One goal of the physical immune system is to eliminate non-self molecules from the body through a sophisticated recogition and proliferation process known as adaptive immunity. These non-self molecules are called antigens. Molecules produced by the immune system, called antibodies, are able to bind antigens based on complementary matching. To produce an effective response, antibodies are activated upon antigen recognition and enter a period of clonal expansion with hypermutation, a process that increases the affinity of the antibodies to the corresponding antigen. Through an iterative process these antibodies go through selection based on their evolving affinity to the antigen. It is on these general principles that AIS algorithms are built.

2.1 Basic AIS Implementation

To establish the baseline performance of a traditional AIS for detecting epistasis we have implemented a basic version of the algorithm. The antibodies are generated to represent three genotype coding schemes, 0 indicates homozygous for the minor allele, 1 indicates heterozygous and 2 indicates homozygous for the major allele. The antibodies may also include neutral match ($*$) characters. The algorithm uses affinity scores, calculated for each antibody, by counting the number of positional matches relative to the antigen and subtracting the number of positional mismatches. Neutral matches do not affect the affinity score. Affinity maturation is done by clonal selection. The antibodies scoring at or above a set threshold survive and may be chosen for proliferation with mutation. Upon repeated exposure the to same antigens the surviving antibodies, over a number of generations, evolve to increase their affinity scores. The following steps outline this process:

1. Data simulation to generate the antigen population, Ag (vectors comprised of 0,1,2; each antigen vector represents an individual with a case or control classification).
2. Randomly generate the initial population of N antibodies, Ab (vectors comprised of 0,1,2,$*$).
3. Calculate affinity between each antibody Ab_i and each antigen Ag_i classified as a control.
4. Remove a percentage, determined by $(1 - \mathit{survival\ rate})$, of the highest ranked antibody vectors.
5. Calculate affinity between remaining antibodies and each antigen classified as a case.
6. Remove a percentage, determined by $(1 - \mathit{survival\ rate})$, of the lowest ranked antibody vectors.
7. Calculate the number of antibodies, n, necessary to restore the initial population size ($n = N \cdot \mathit{survival\ rate}^2$).
8. From remaining antibody population select 1 antibody at random and clone $\frac{n}{2}$ times with mutation and add to antibody pool.
9. Randomly generate $\frac{n}{2}$ new antibodies and add to antibody pool.
10. Repeat steps 3-9 to set number of generations.

The user defined parameters are number of antibodies, neutral match percentage, survival rate, mutation rate and number of generations. The rank is calculated as the average affinity over all tested antigens regardless of case-control status. Point mutations are used. Examples of an antibody vector and an affinity score calculation are included in Figure 1.

2.2 Random Search

To perform a random search a population of antibodies was generated at $N = \mathit{number\ of\ antibodies} \cdot \mathit{number\ of\ generations}$ where number of antibodies

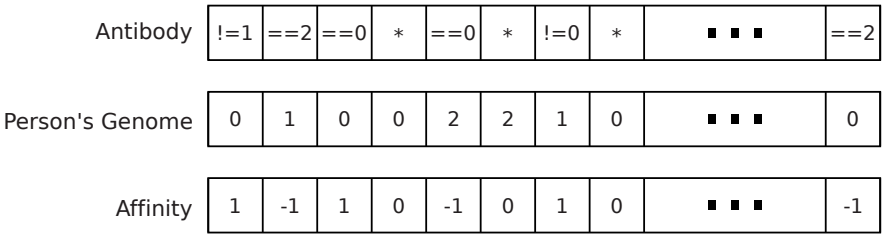


Fig. 1. A representation of an antibody vector in our system showing the affinity scoring method

and generations are taken from the basic AIS parameters to which it is being compared. It is of note that this affords the random search more power than the AIS. The only time AIS would have this much power is when the $survival\ rate = 0$, and the actual number of new random antibodies considered by the AIS is $N = number\ of\ antibodies \cdot number\ of\ generations \cdot (1 - survival\ rate)$. This is reasonable because the AIS gains power through the evolution process which is absent in the random search. This initial population of antibodies is then scored by the same affinity measures as the basic AIS following steps 3-6 as outlined above. The pool of antibodies that remain are subjected to the same statistical analyses as the pool of antibodies in the last generation of the basic AIS. This preserves the solution representation and initialization of the AIS while eliminating the selection and mutation components of an AIS.

2.3 Data Simulation

We evaluate the AIS and compare its performance to random search using a simulation study. The goal of this study is to gauge the performance of these methods. We obtained the datasets from `http://discovery.dartmouth.edu/epistatic_data/`. We wanted to use data characteristic of candidate-gene type study designs. We chose the 20 attribute datasets with the lowest heritability generated by Velez et al. [**?**]. These datasets cover five different penetrance functions. It is these penetrance functions that determine the relationship between an individuals genes (genotype) and whether or not they have a disease (phenotype). In these penetrance functions, an individual's probability of disease depends only on the genotypes at two interacting SNPs but not on the genotypes at any single SNP. The case where there are no single SNP effects is considered the most difficult to solve. The heritability for these penetrance functions is 0.01, which is extremely low, and difficult to detect with existing methods. In each dataset there are only 400 individuals evenly divided into cases and controls. These low heritability datasets with few individuals provide the most challenging cases of all the datasets available from our repository.

The evaluation process for gauging the performance of these methods is power. It is possible to determine the power of each method because we are using simulated data which allows tracking of the success rate. We can do this because,

in simulated datasets, the relevants SNPs are known before the algorithm is applied. We then calculate the power as the percentage of datasets that produced a best antibody comprised exclusively of the two relevant attributes. Using this power metric it is possible to directly compare between the ability of the AIS and random search to find the true answer. Furthermore it is possible to compare between AIS parameter settings to provide guidance about parameter settings for future studies.

2.4 Statistical Analysis

We used logistic regression to examine the significance of each parameter in our basic AIS and the random search method. The significance level was set to $p = 0.05$. Performance of the basic AIS was directly compared in a power analysis to the random search. Power was calculated by tracking the number of times the correct genetic model was found over 500 data sets and is displayed as a percentage.

3 Results

We were first interested in assessing the impact of various parameters on the ability of both the random search and the basic AIS to solve the genetic model in our simulated data. The results of a logistic regression analysis are presented in Table 1. The number of antibodies and the neutral match percentage were statistically significant for success of a random search. Because there is not an evolution step in the random search method we would not expect the survival rate or the mutation rate to impact the performance of the algorithm. In comparison, the basic AIS shows that each parameter is statistically significant. Owing to the selection and evolutionary processes, it follows that all of the individual parameters should influence the performance of the algorithm, as expected, this is what we observe.

To establish the baseline performance of a basic AIS we used power analyses to evaluate the competency of the algorithm in detecting epistatic interactions in our simulated data. For comparison we did corresponding power analyses for the random search method. Power analysis results are presented as boxplots representing the aggregate findings over 500 data sets. Each plot shows the power at various settings of the parameter on the x-axis while all other parameters are kept constant.

The basic AIS has minimal power for solving the genetic model in our simulated data as shown in Figure 2. Interestingly, these plots consistently show variable dispersion between parameter settings and include many outliers at increasingly higher powers. This indicates that there are a number of instances where the algorithm performs satisfactorily in this task.

In contrast Figure 3 shows the results for the random search method. The number of antibodies in random search reflect 50, 100 or 500 antibodies

Table 1. The significance of each parameter was tested using logistic regression

Algorithm	Parameter	p-value
Random Search	Number of Antibodies	$\ll .0001$
	Number of Generations	NA
	Survival Rate	0.928
	Neutral Match Percentage	$\ll .0001$
	Mutation Rate	0.096
Basic AIS	Number of Antibodies	$\ll .0001$
	Number of Generations	$\ll .0001$
	Survival Rate	$\ll .0001$
	Neutral Match Percentage	$\ll .0001$
	Mutation Rate	$\ll .0001$

multiplied by 50, 100 or 500 representing generations to get antibody pools comparable to those in the AIS. While we see less variability in powers, the spread of the data is smaller and we do not see the same pattern of outliers. This indicates that we have reached the ability limit of the random search algorithm for this task without attaining adequate power.

We show that the median powers across the parameters are very similar between the two methods. The median powers of the AIS show slight increases with a lower survival rate, a higher number of generations, a higher mutation rate and a higher neutral match percentage. The powers of the AIS also show a slight decrease with the population size. This seems to be related to the very strict definition of power which requires the best antibody to retain only the two relevant attributes. The median powers of the random search are slightly elevated at a higher number of antibodies and a lower neutral match percentage. This is consistent with the results of the logistic regression analysis.

If we examine the parameter set that leads to the observed high power outliers, we can gain some insight into what guides the success of the algorithm. The optimum setting for survival rate is 0.2. The optimum percentage of neutral matches was 0.8. The optimum mutation rate was 0.8. The optimum population size was 50 which, as discussed earlier, seems to be a side effect of the stringent selection strategy and strict definition of power. When we look at the powers of an AIS over 50, 100 and 500 generations using these parameters, it is apparent that the AIS strategy can be very successful as it approaches 100% power. The highest value for the mutation rate was optimum and the lowest value for the survival rate was optimum. This pair of parameters leads to the broadest possible search. In light of this observation, it might be expected that the random search approach, which covers an even more expansive search space, is better able to discover the relevant SNPs. As our results show, the random search strategy never approaches these powers.

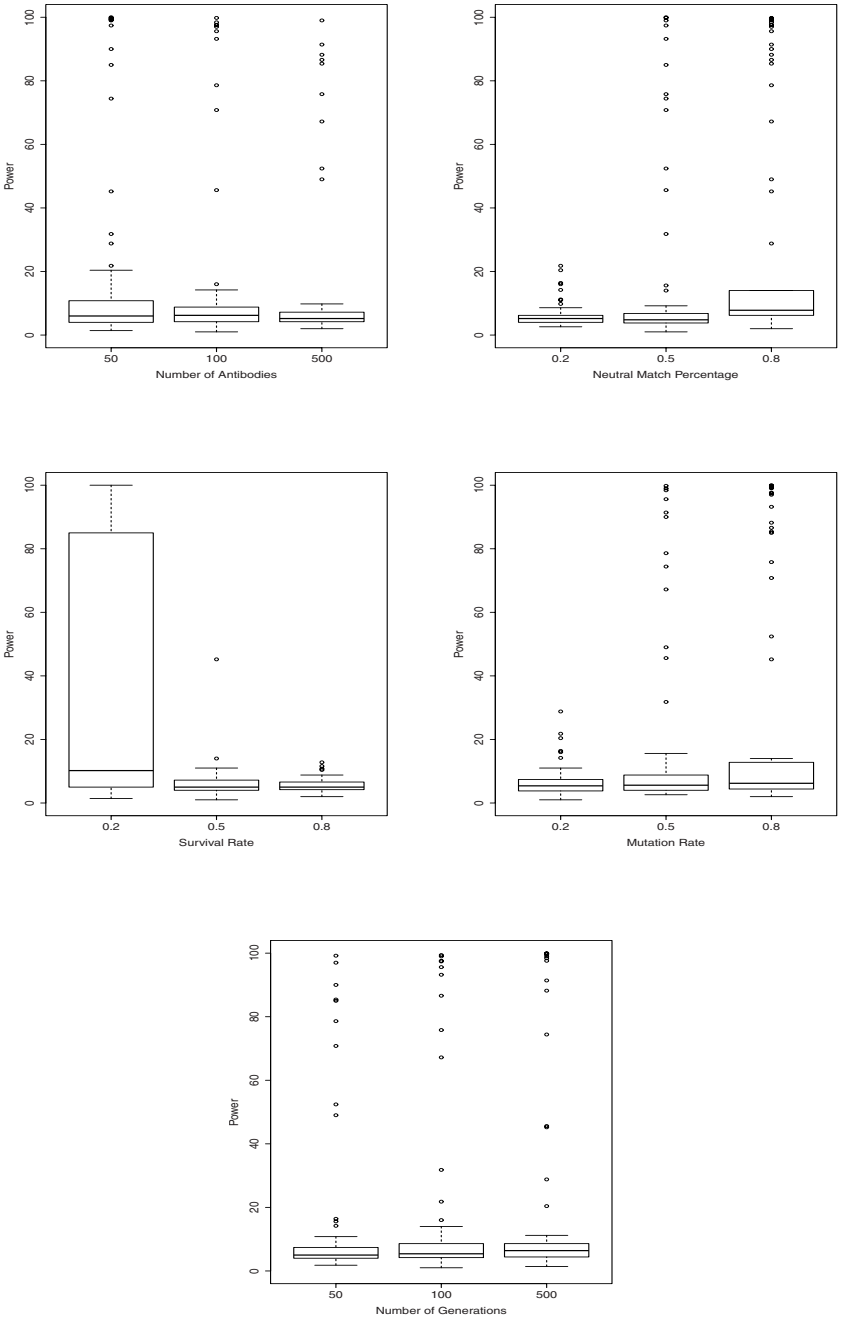


Fig. 2. Summary results for basic AIS. Boxplots showing the power of each parameter to successfully detect the interacting pair of SNPs in simulated data by the basic AIS.

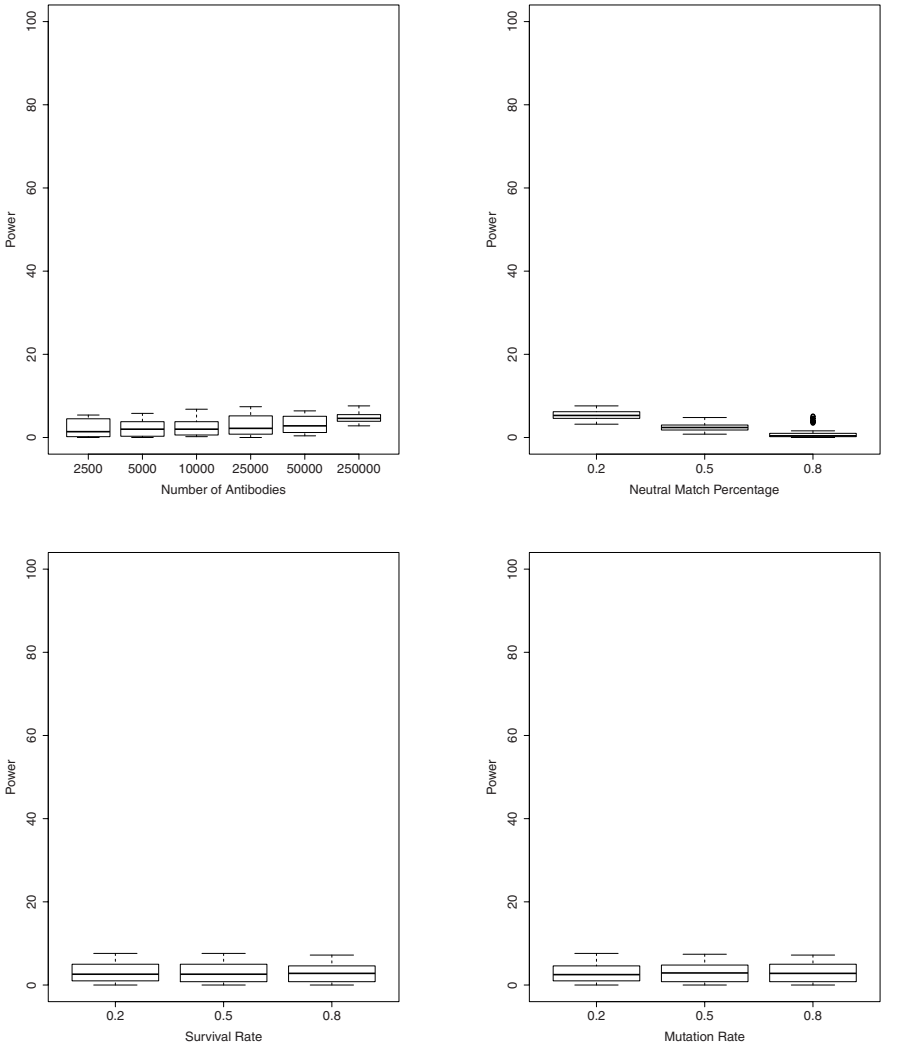


Fig. 3. Summary results for random search. Boxplots showing the power of relevant parameters to successfully detect the interacting pair of SNPs in simulated data by the random search. Number of antibodies reflects the $number\ of\ antibodies \cdot number\ of\ generations$ of the basic AIS.

4 Discussion and Conclusions

We believe that a complex genetic architecture, built on non-linear, gene-gene interactions, underlies most common diseases. In order to use genetic variation to create disease specific signatures it is necessary to develop methods that are able to detect such interactions. AIS algorithms provide a means of identifying

patterns of genetic variation on a large scale. Here we have taken a basic version of the algorithm and tested it against a random search strategy to establish the baseline performance of an AIS for finding epistatic interactions in case-control data.

Evaluation of the AIS was done by power analysis and comparison to random search (Figures 2 and 3). The median powers are very similar between the two methods across all parameters. This suggests that the basic AIS does not provide an obvious advantage, however, the potential of this algorithm is clearly illustrated by the number of outliers we observe corresponding with adequate statistical power. Each power calculated represents a specific parameter combination. This means that some parameter combinations, i.e. these outliers, are able to dramatically outperform random search and obtain powers competitive with exhaustive methods.

Under conditions of parameter optimization, we see powers approaching 100% for the AIS method. These powers are unmatched by the random search method. It seems likely, then, that the stringent selection strategy heavily exploits good solutions while requiring a permissive search strategy to effectively explore the search space. While it is encouraging that the algorithm can succeed with a specific set of parameters, it is telling that the performance drops dramatically at sub-optimal parameters. For real datasets the correct answer is not known and thus a parameter sweep to determine the optimal parameter set is more difficult. For this reason modifications to the algorithm itself and to the selection function should be explored to determine if a more robust AIS can be developed for this problem.

The slight decrease in power observed for increasing population sizes in the AIS was unexpected. It appears that these larger antibody populations, combined with the selection against antibodies which incorrectly recognize control individuals, may be leading the algorithm to overfit. It is possible that the traditional AIS's selection function is not optimal for this problem where the genetic effect size is small because some individuals with a high risk genotype do not have disease. A more tolerant selection strategy may prevent this over-fitting and make the algorithm more robust.

We have shown that the basic AIS approach can outperform random search on this problem but that detailed parameter tuning is currently required. This suggests that if the correct adjustments are made to the algorithm we may be able to increase the power of this method to detect gene-gene interactions. Future work should focus on developing a robust AIS which is much less parameter dependent. This may require a re-evaluation of the objective function. It is possible that the selection function used in the traditional AIS setting is ineffective in the case of human genetics where the environment plays a critical role and genetic factors cannot entirely explain an individual's disease state. The individuals whose disease states are not explained by genetics may confound the algorithm and lead to the removal of reasonable antibodies. The AIS representation can be advantageous for this problem, but the selection function may need

to be dramatically altered to cope with the inherent noise in human genetics data if we are to develop an AIS which is more robust to parameter settings.

In conclusion, the AIS is a promising approach with a desireable solution representation but it will require more development and evaluation to improve the robustness of the algorithm before it becomes a common tool used by modern human geneticists.

References

1. Moore, J., Williams, S.: Epistasis and Its Implications for Personal Genetics. The American Journal of Human Genetics 85(3), 309–320 (2009)
2. Bateson, W.: Mendels principles of heredity. Molecular and General Genetics MGG 3(1), 108–109 (1910)
3. Fisher, R.: The correlation between relatives on the supposition of Mendelian inheritance. Transactions of the Royal Society of Edinburgh 52(2), 399–433 (1918)
4. Emily, M., Mailund, T., Hein, J., Schauser, L., Schierup, M.: Using biological networks to search for interacting loci in genome-wide association studies. European Journal of Human Genetics 1, 10 (2009)
5. De Castro, L., Timmis, J.: Artificial immune systems: a new computational intelligence approach. Springer, Heidelberg (2002)
6. Ritchie, M., White, B., Parker, J., Hahn, L., Moore, J.: Optimization of neural network architecture using genetic programming improves detection and modeling of gene-gene interactions in studies of human diseases. BMC Bioinformatics 4(1), 28 (2003)
7. Motsinger-Reif, A., Ritchie, M.: Neural networks for genetic epidemiology: past, present, and future. BioData Mining 1(1), 3 (2008)
8. Greene, C., White, B., Moore, J.: Ant colony optimization for Genome-Wide genetic analysis. In: Dorigo, M., Birattari, M., Blum, C., Clerc, M., Stützle, T., Winfield, A.F.T. (eds.) ANTS 2008. LNCS, vol. 5217, pp. 37–47. Springer, Heidelberg (2008)
9. Kononenko, I.: Estimating attributes: Analysis and extensions of RELIEF. In: Bergadano, F., De Raedt, L. (eds.) ECML 1994. LNCS, vol. 784, pp. 171–182. Springer, Heidelberg (1994)
10. McKinney, B.A., Reif, D.M., White, B.C., Crowe, J.E., Moore, J.H.: Evaporative cooling feature selection for genotypic data involving interactions. Bioinformatics 23(16), 2113–2120 (2007); PMID: 17586549
11. Beretta, L., Cappiello, F., Moore, J.H., Barili, M., Greene, C.S., Scorza, R.: Ability of epistatic interactions of cytokine single-nucleotide polymorphisms to predict susceptibility to disease subsets in systemic sclerosis patients. Arthritis and Rheumatism 59(7), 974–983 (2008); PMID: 18576303
12. McKinney, B.A., Crowe, J.E., Guo, J., Tian, D.: Capturing the spectrum of interaction effects in genetic association studies by simulated evaporative cooling network analysis. PLoS Genetics 5(3), e1000432 (2009); PMID: 19300503
13. Kira, K., Rendell, L.A.: A practical approach to feature selection. In: ML 1992: Proceedings of the ninth international workshop on Machine learning, pp. 249–256. Morgan Kaufmann Publishers Inc., San Francisco (1992)
14. Bereta, M., Burczynski, T.: Comparing binary and real-valued coding in hybrid immune algorithm for feature selection and classification of ECG signals. Engineering Applications of Artificial Intelligence 20(5), 571–585 (2007)

Metaheuristics for Strain Optimization Using Transcriptional Information Enriched Metabolic Models

Paulo Vilaça[1,2], Paulo Maia[1,2], Isabel Rocha[2], and Miguel Rocha[1]

[1] Department of Informatics / CCTC, University of Minho
{paulo.maia,mrocha}@di.uminho.pt
[2] IBB, Institute for Biotechnology and Bioengineering
Centre of Biological Engineering, University of Minho
Campus de Gualtar, 4710-057 Braga, Portugal
{pvilaca,irocha}@deb.uminho.pt

Abstract. The identification of a set of genetic manipulations that result in a microbial strain with improved production capabilities of a metabolite with industrial interest is a big challenge in Metabolic Engineering. Evolutionary Algorithms and Simulated Annealing have been used in this task to identify sets of reaction deletions, towards the maximization of a desired objective function. To simulate the cell phenotype for each mutant strain, the Flux Balance Analysis approach is used, assuming organisms have maximized their growth along evolution.

In this work, transcriptional information is added to the models using gene-reaction rules. The aim is to find the (near-)optimal set of gene knockouts necessary to reach a given productivity goal. The results obtained are compared with the ones reached using the deletion of reactions, showing that we obtain solutions with similar quality levels and number of knockouts, but biologically more feasible. Indeed, we show that several of the previous solutions are not viable using the provided rules.

Keywords: Metabolic Engineering, Strain Optimization, Flux-Balance Analysis, Transcriptional Models, Set based representations.

1 Introduction

Over the last few years, the combined efforts of Metabolic Engineering and Systems Biology have allowed the development of some genome-scale metabolic models for several microorganisms, with an industrial interest in Biotechnology. These have been used to predict cellular phenotypes under some simplifying assumptions, aiding in the effort of finding appropriate genetic modifications to make the microorganism fit to comply with industrial purposes, i.e. to be able to synthesize some desired compounds in significant amounts, rather then to follow their natural aims (e.g. the maximization of growth) [14] [8].

The most popular approach considers the cell to be in a steady-state, i.e., the concentrations of all intracellular compounds are assumed to remain constant throughout time. Together with the known stoichiometry and reversibility

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2010, LNCS 6023, pp. 205–216, 2010.

or irreversebility of the reactions, this assumption is used in a constraint-based framework to restrict the set of possible values for the fluxes of the reactions contained in the metabolic model. Therefore, cellular behavior can be predicted by addressing the underlying optimization problems, given a biologically plausible objective function. The Flux Balance Analysis approach [6] follows this path, maximizing a particular flux, typically for biomass production, using linear programming [5]. Solving this problem allows to reach the values for all the reaction fluxes.

Using this approach or others recently proposed for the same purpose (e.g. MOMA [12], ROOM [13]), it is possible to predict the behavior of a microorganism under distinct environmental and genetic conditions (such as gene deletions). Indeed, both can be represented by adding/ changing constraints under the previous framework. Therefore, both wild type and mutant strains can be simulated. This has allowed the definition of a bi-level strain optimization problem, adding a layer that searches for the best mutant that can be obtained by applying a set of selected genetic modifications. In previous work, this has been restricted to the possibility of removing reactions from the original model. The idea is to force the microorganisms to synthesize a desired product, while keeping it viable. The optimization task consists in reaching an optimal subset of reaction deletions to optimize an objective function related with the production of a given compound.

A first approach to this problem was the *OptKnock* algorithm [1], where mixed integer linear programming methods are used to reach a guaranteed optimum solution. However, this algorithm does not allow to consider nonlinear objective functions and a considerable computation time is required. An alternative was proposed by the *OptGene* algorithm [9], that uses Evolutionary Algorithms (EAs). EAs are capable of providing near optimal solutions in a reasonable amount of time and also allow the optimization of nonlinear objective functions. Extending this work, the authors [11] proposed a new encoding scheme for the problem, consisting in variable-sized sets, allowing the automatic determination of the ideal number of reactions to eliminate, since solutions with distinct cardinalities compete within the search space. Also, a Simulated Annealing (SA) based algorithm was put forward for the same task. Both algorithms were tested with four case studies and the SA presented some advantage over the EA.

A common limitation of these approaches is the fact that they rely on determining sets of reactions to be eliminated from the metabolic model, while the real purpose is to determine a set of genes to knockout. Therefore, to create the desired mutants in the lab there is the need to determine which set of genes can lead to the elimination of a given set of reactions. This would not be a problem if the rule 1 gene - 1 enzyme - 1 reaction was universal. However, this is not the case, since there are many exceptions, due to iso-enzymes, protein complexes, enzymes that catalyze several reactions or reactions that can be catalyzed by several enzymes.

The solution is, therefore, to use transcriptional information in association with the genome-scale metabolic model. This approach is mostly limited by the lack of information available, since in most cases there is no comprehensive

model of transcriptional information available. However, this situation is gradually changing and some metabolic models with transcriptional information of well known microorganisms are appearing [10].

In this work, we propose phenotype simulation and strain optimization methods that are able to take advantage on this transcriptional information. The optimization methods will be able to suggest sets of genes to knockout replacing the reaction list usually provided. Two case studies related to the production of succinate and lactate using the bacterium *Escherichia coli* will be presented to evaluate the approach. We will also study in detail the major differences between the reaction and gene based approaches and compare the results obtained.

2 Simulation Algorithms for the Prediction of Metabolic Behavior

2.1 Flux Balance Analysis

The Flux Balance Analysis (FBA) [6] approach is based on a steady state approximation to the concentrations of internal metabolites, which reduces the corresponding mass balances to a set of linear homogeneous equations. For a network of M metabolites and N reactions, this is expressed as:

$$\sum_{j=1}^{N} S_{ij} v_j = 0 \tag{1}$$

where S_{ij} is the stoichiometric coefficient for metabolite i in reaction j and v_j is the flux over the reaction j. The maximum/minimum values of the fluxes can be set by additional constraints in the form $\alpha_j \leq v_j \leq \beta_j$, usually used to specify both thermodynamic and environmental conditions (e.g. availability of nutrients).

For most metabolic networks, since the number of fluxes is greater than the number of metabolites, the set of linear equations obtained from the application of Eq. 1 to the M metabolites usually leads to an under-determined system, for which there exists an infinite number of feasible flux distributions that satisfy the constraints. However, if a given linear function over the fluxes is chosen to be maximized, it is possible to obtain a single solution by applying standard algorithms (e.g. *simplex*) for linear programming problems.

The combination of this technique with the existence of validated genome-scale stoichiometric models [2] allows to simulate the phenotypic behavior of a microorganism, under defined environmental conditions, without performing any experiments. The most common flux chosen for maximization is the biomass, based on the premise that microorganisms have maximized their growth along natural evolution, a premise that has been validated experimentally for some situations [5].

2.2 Integrating Transcriptional Information

Recently, some studies attempted to improve the characterization of organisms by inserting a transcriptional layer into the metabolic models [10,3,15]. Thus, adding this level of information about the behavior of biological systems, raises the metabolic models into the genetic level, where the metabolic processes depend on the genes that encode enzymes which catalyse metabolic reactions.

To create this transcriptional layer it is necessary to define the cascade of interactions between genes, proteins, peptides and reactions of a given system. These are not easy to find due to the complexity of the different types of interactions between biological entities:

- the genes encode the information that leads to the creation of peptides through the processes of transcription and translation;
- proteins can be constructed from one or more peptides;
- proteins can bind to create protein complexes;
- the reactions are catalyzed by enzymes (proteins or protein complexes);
- more than one protein can catalyze the same reaction (iso-enzymes);
- a single protein can catalyze more than one reaction.

In this work, all available transcriptional information will be transformed into gene-reaction rules. Gene-reaction rules are based on boolean logic representation. For each reaction (dependent variable), there is a boolean expression, where the independent variables are the encoding genes; their interactions are defined using logical operations (AND, OR). In Figure 1, some examples of different associations between genes, peptides, proteins and reactions are shown, as well as their simplification for gene-reaction rules.

3 Strain Optimization

3.1 Problem Definition, Solution Encoding and Evaluation

The problem addressed in this work consists in selecting, from a set of genes in a microbe's genome-scale model, a subset to be deleted to maximize a given objective function. The encoding of a solution is achieved by a variable size set-based representation, where only gene deletions are represented. Each solution consists of a set of integer values representing the genes that will be deleted. Therefore, if the value i is in the set, this means the i-th gene in the model is removed. Each value in the set is an integer with a value between 1 and G, where G is the number of genes in the model.

The first step is to take the genes indexed by the solution and then calculate which reactions will be removed as a consequence of knocking out these genes, using the transcriptional information. For all reactions involved, the flux will be constrained to 0, therefore disabling that reaction in the metabolic model. The process proceeds with the simulation of the mutant using FBA. The output is the set of values for the fluxes of all reactions, that are then used to compute

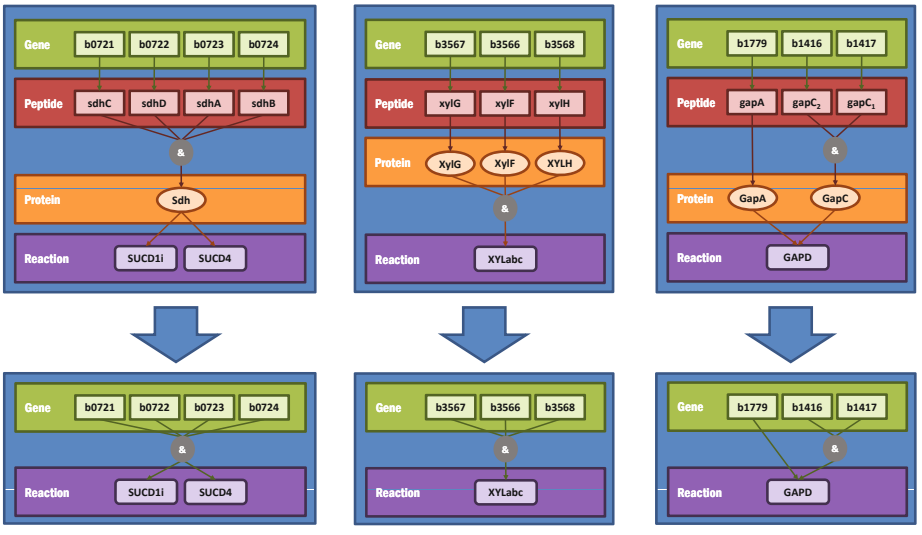


Fig. 1. Schematic representation of the transcriptional information included in the metabolic models

the fitness value, given by an appropriate objective function. The used objective function is the Biomass-Product Coupled Yield (BPCY) [9], given by:

$$BPCY = \frac{PG}{S} \tag{2}$$

where P stands for the flux representing the excretion of the desired product; G for the organism's growth rate (biomass flux) and S for the substrate intake flux. Besides optimizing for the production of the desired product, this function also allows to select for mutants that exhibit high growth rates. The complete process of decoding and evaluation is depicted in Figure 2.

3.2 Evolutionary Algorithms

To address the previous task, we will use Evolutionary Algorithms (EAs) with a set-based representation, previously proposed in [11]. This EA uses four reproduction operators: one crossover and three mutation operators. The crossover operator is inspired on traditional uniform crossover operators and works as follows: the genes that are present in both parent sets are kept in both offspring; the genes that are present in only one of the parents are sent to one of the offspring, selected randomly with equal probabilities.

A random mutation operator is used that replaces a gene by a random value in the allowed range, avoiding duplicates in the set. Two additional mutation operators are defined to be able to create solutions with a distinct size:

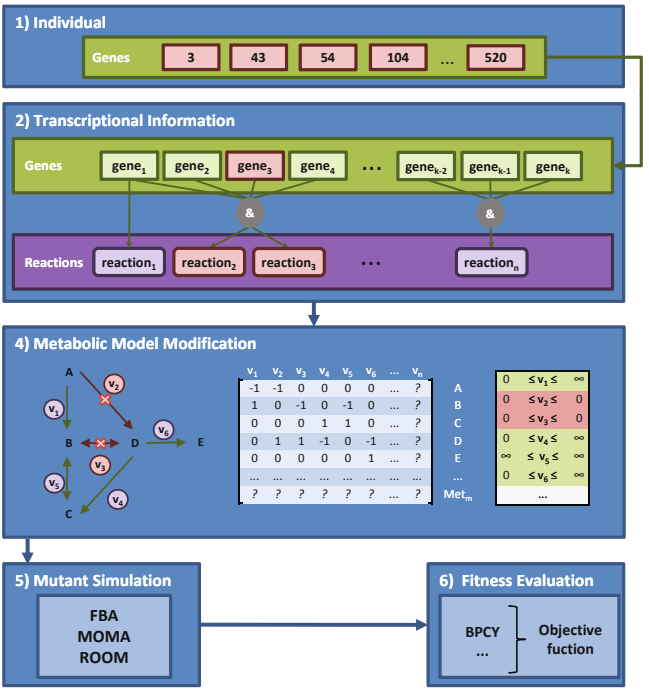


Fig. 2. Scheme of the phenotypic simulation methods using transcriptional information and their transformation into gene-reaction rules

- *Grow*: consists in the introduction of a new gene into the solution, whose value is randomly generated in the available range (avoiding duplicates in the set).
- *Shrink*: a randomly selected gene is removed from the genome.

The Grow and Shrink mutation operators are each used with a probability of 5% each. The remaining operators are used with equal probabilities. The EA uses a selection procedure that consists in converting the fitness value into a linear ranking of the individuals in the population, and then applying a roulette wheel scheme. In each generation, 50% of the individuals are kept from the previous generation, and 50% are bred by the application of the reproduction operators. An initial population is randomly created and the termination criterion is based on a fixed number of solution evaluations.

3.3 Simulated Annealing

Also, Simulated Annealing (SA) was used to address the optimization task and compare the results. As before, the SA is also similar to the one proposed by the authors in [11]. The SA makes use of the same set-based representation used in

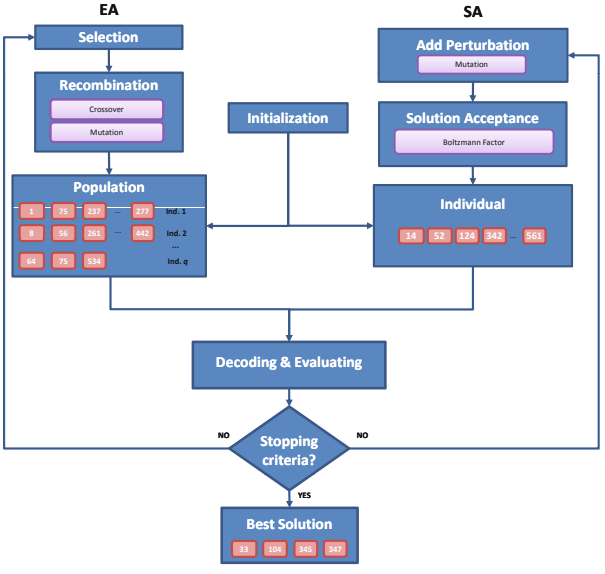


Fig. 3. Illustration of the structure of the strain optimization algorithms

the EA, also keeping the mutation operators presented before. An illustration of the structure of both algorithms is given in Figure 3.

The cooling schedule used is exponential, decreasing the temperature T according to: $T_{n+1} = \alpha T_n$, where $0 < \alpha \leq 1$. As the choice of initial (T_0) and final temperatures (T_f) is problem dependent, it was decided to use the following configuration parameters:

ΔE_0 – The difference in energy that corresponds to an acceptance probability of 50% of worse solutions at the beginning of the run;
ΔE_f – The difference in energy that corresponds to an acceptance probability of 50% of worse solutions at the end of the run;
trials – The number of iterations per temperature;
NFEs – The number of function evaluations.

Using these parameters, the initial temperature, the final temperature and the scale parameter were computed using the following equations:

$$T_0 = -\frac{\Delta E_0}{\log 0.5} \tag{3}$$

$$T_f = -\frac{\Delta E_f}{\log 0.5} \tag{4}$$

$$\alpha = \exp\left(\frac{\log T_f - \log T_0}{\left[\frac{\text{NFEs}}{\text{trials}}\right]}\right) \tag{5}$$

The advantage of using ΔE_0 and ΔE_f is that it allows the user who knows the fitness landscape of the optimization problem to automatically define the temperatures by reasoning over the values of the objective function. Supplying the number of function evaluations instead of the scale parameter α allows the user to accurately define the number of function evaluations the optimization algorithm will use, enabling a simpler comparison with other approaches.

In the SA, the *Grow* and *Shrink* mutations are each used with a probability of 25% each, meaning that half of the new individuals are created in this way. The remaining are created by the aforementioned random mutation operator.

3.4 Pre-processing and Post-processing

In genome-scale models the number of variables (genes/ reactions) is in the order of hundreds or a few thousands and therefore the search space is very hard to address. Thus, every operation that gives a contribution to reduce this number, greatly improves the convergence of the algorithms. In this work, two operations were implemented to reduce the search space:

- Removal of reactions that, given the constraints of the linear programming problem, cannot exhibit flux values different from 0. All genes only encoding those reactions are also removed.
- Discovery of essential genes that can not be deleted from the model since their removal leads to non growth (biomass flux value of zero). As these genes should not be considered as targets for deletion, the search space for optimization is reduced.

Also, the best solution in each run goes through a simplification process, by identifying all gene deletions that contribute to the fitness of the solution, and removing all deletions that keep the objective function unaltered. The aim is to keep only the necessary knockouts.

3.5 Implementation Issues

The implementation of the proposed algorithms was performed by the authors in the *Java* programming language. In the implementation of FBA, the *GNU linear programming package (GLPK)*[1] was used to run the *simplex* algorithm. An user interface was also built within the OptFlux framework[2], a Metabolic Engineering open-source software platform.

4 Experiments

4.1 Experimental Setup

Two case studies were used to test the algorithms, both considering the microorganism *Escherichia coli*. The aim is to produce succinate and lactate with

[1] http://www.gnu.org/software/glpk/
[2] http://www.optflux.org

glucose as the limiting substrate. The genome-scale model used in the simulations was developed by Reed et al [10]. This model considers the metabolic network of *E. coli*, including a total of $N = 1075$ fluxes, $M = 761$ metabolites, $G = 904$ genes and 873 gene-reaction rules. After the pre-processing stages, the simplified model remains with $N = 610$, $M = 383$ metabolites, 617 genes and 562 gene-reaction rules. Furthermore, 115 essential genes are identified, which leaves 502 variables to be considered by the optimization algorithms.

In the EA the population size was set to 100. The SA used $\Delta E_0 = 0.005$, $\Delta E_f = 5E^{-5}$ and $trials = 50$. In both cases, the termination criterion was defined based on 50000 fitness evaluations. For each configuration, the process was repeated for 30 runs and the mean and standard deviation were calculated.

4.2 Case Studies

Succinate is one of the key intermediates in cellular metabolism and therefore an important case study for metabolic engineering [7]. The knockout solutions that lead to an improved phenotype regarding its production are not straightforward to identify since they involve a large number of interacting reactions. Succinicate and its derivatives have been used to synthesize polymers, as additives and flavoring agents in foods, supplements for pharmaceuticals, or surfactants. Currently, it is mostly produced through petrochemical processes that can be expensive and have significant environmental impacts.

Lactate and its derivatives have been used in a wide range of food-processing and industrial applications like meat preservation, cosmetics, oral and health care products. Additionally, and because lactate can be easily converted to readily biodegradable polyesters, it is emerging as a potential material for producing environmentally friendly plastics from sugars [4]. Several microorganisms have been used to produce lactate, such as *Lactobacillus* strains. However, those bacteria have undesirable traits, such as a requirement for complex nutrients which complicates acid recovery. *E. coli* has many advantageous characteristics, such as rapid growth and simple nutritional requirements.

4.3 Results

In Tables 1 and 2 we show the results for both case studies, taking the BPCY as the objective function. In both cases, we show the results for our current approach using transcriptional information, compared to the results using the previous method based on reaction deletions [11]. It should be emphasized that all the setup is the same for both cases. The first two columns show the optimization target (genes or reactions) and the algorithm used (EA or SA). In the third and fourth columns, we show the mean of the BPCY and of the number of knockouts over the 30 runs, also showing the standard deviation (surrounded by parentheses). Finally, the last column shows the BPCY and the number of knockouts of the best solution obtained over the 30 runs.

Also, we investigated how the solutions obtained for reaction based optimization can be converted into a gene knockout set. So, we analyzed the best solution obtained in each of the 30 runs according to the following: (i) we took the set

Table 1. Results for the succinate case study

Optimization Type	Algorithm	Fitness (BPCY)	Number Knockouts	Best Solution
Reactions	EA	0.35345 (0.01405)	11.7 (2.6)	0.35785 (15)
Reactions	SA	0.35766 (0.00015)	9.7 (1.0)	0.35781 (11)
Genes	EA	0.23188 (0.09945)	10.6 (1.9)	0.34429 (7)
Genes	SA	0.30636 (0.07713)	10.4 (4.0)	0.34429 (10)

Table 2. Results for the lactate case study

Optimization Type	Algorithm	Fitness (BPCY)	Number Knockouts	Best Solution
Reactions	EA	0.21387 (0.09180)	9.7 (7.3)	0.34786 (5)
Reactions	SA	0.27654 (0.05713)	16.8 (8.9)	0.34843 (26)
Genes	EA	0.25447 (0.05215)	10.3 (2.9)	0.34786 (5)
Genes	SA	0.25428 (0.05039)	12.1 (4.3)	0.29328 (8)

of reactions to delete and calculated the minimum set of genes that had to be removed in order to inactivate those reactions; (ii) we checked if there were other reactions that would be inactivated as a result of those gene deletions; (iii) finally, we simulated the resulting mutant strain and calculated the BPCY.

The results are given in Table 3 where we show, for each case, the number of solutions (over the best solutions in each run) where the BPCY is still larger than zero (third column), the number of solutions that keep the same BPCY (fourth column) and also the mean number of knockouts that were added in step (ii) of the previous process.

Table 3. Conversion of reaction deletion based solutions to gene deletion based solutions

Case study	Algorithm	$BPCY > 0$	same $BPCY$	Additional Knockouts
Succinate	EA	0/30	0/30	12.5
Succinate	SA	0/30	0/30	6.8
Lactate	EA	8/30	5/30	8.0
Lactate	SA	8/30	3/30	15.7

4.4 Discussion

The first conclusion to retain is that the overall objective function results, when optimizing gene deletions, are quite near the ones obtained before by deleting reactions. Although in most cases the BPCY is slightly lower, the differences are generally not statistically significant. Also, the number of knockouts does not increase, even decreasing in most cases (again differences are not significant).

The differences in performance, when they exist, are small when compared to the gains obtained considering that models with transcriptional information

characterize better the behavior of the organism and the simulation using this level of information is closer to the biological reality and thus more reliable. Also, the implementation of the solutions in the lab will be based on a gene list, which makes these results easier to implement.

Studying the results in more depth we see that the succinate case study seems to have larger differences between the two approaches. Looking at Table 3 we can understand the reasons, since we see that all the best solutions obtained are unfeasible at the level of genes (reporting a value of 0 for the BPCY) and therefore impossible to implement in the lab. From that table, we also conclude that in the lactate case study most of the solutions (around 70%) also have a BPCY of 0 and some of the others deteriorate the fitness value. This shows that, in general, it seems unlikely that solutions reached with reaction deletion based optimization are biologically feasible (i.e. can be implemented through gene knockouts). Comparing both meta-heuristics for optimization, we observe that the SA and EA shown very similar performances, but the SA confirms a slight advantage, already reported in [11].

5 Conclusions and Further Work

In this work, we have studied the effects of using transcriptional information to complement the knowledge contained in metabolic models, on the results of strain optimization algorithms such as EA and SA. The main conclusion of this analysis indicates that most solutions obtained previously, considering reaction deletion optimization, are impossible to translate to gene knockouts and therefore to implement in the lab.

We proposed improved algorithms for the tasks of phenotype simulation and strain optimization that can take advantage on the transcriptional information. The results obtained by those methods reveal an overall solution quality very similar to the previous methods and the number of suggested knockouts does not increase, also an important result considering the feasibility of the solutions.

Since these solutions are biologically more feasible, we believe that an important step has been made towards the use of these methods in Biotechnology. Also with this aim, we have implemented these methods under *OptFlux*, an open-source software platform. This allows the methods to be used freely by the Metabolic Engineering community.

As future work, we aim to apply these methods to other relevant case studies in Metabolic Engineering, considering other target compounds, as well as other organisms and models. Also, the integration of regulatory information with these models, in the form of new constraints, is a promising path.

Acknowledgements

This work was partially funded by Portuguese FCT through the AspectGrid project and also through project MIT-PT/BS-BB/0082/2008.

References

1. Burgard, A.P., Pharya, P., Maranas, C.D.: Optknock: A bilevel programming framework for identifying gene knockout strategies for microbial strain optimization. Biotechnol. Bioeng. 84, 647–657 (2003)
2. Covert, M.W., Schilling, C.H., Famili, I., Edwards, J.S., Goryanin, I.I., Selkov, E., Palsson, B.O.: Metabolic modeling of microbial strains *in silico*. Trends in Biochemical Sciences 26(3), 179–186 (2001)
3. Duarte, N.C., Herrgård, M.J., Palsson, B.Ø.: Reconstruction and validation of *Saccharomyces cerevisiae* ind750, a fully compartmentalized genome-scale metabolic model. Genome Res. 14(7), 1298–1309 (2004)
4. Hofvendahl, K., Hahn-Hagerdal, B.: Factors affecting the fermentative lactic acid production from renewable resources. Enzyme Microbial Technology 26, 87–107 (2000)
5. Ibarra, R.U., Edwards, J.S., Palsson, B.G.: *Escherichia coli* k-12 undergoes adaptive evolution to achieve *in silico* predicted optimal growth. Nature 420, 186–189 (2002)
6. Kauffman, K.J., Prakash, P., Edwards, J.S.: Advances in flux balance analysis. Curr. Opin. Biotechnol. 14, 491–496 (2003)
7. Lee, S.Y., Hong, S.H., Moon, S.Y.: *In Silico* metabolic pathway analysis and design: succinic acid production by metabolically engineered *Escherichia coli* as an example. Genome Informatics 13, 214–223 (2002)
8. Nielsen, J.: Metabolic engineering. Appl. Microbiol. Biotechnol. 55, 263–283 (2001)
9. Patil, K., Rocha, I., Forster, J., Nielsen, J.: Evolutionary programming as a platform for *in silico* metabolic engineering. BMC Bioinformatics 6(308) (2005)
10. Reed, J.L., Vo, T.D., Schilling, C.H., Palsson, B.O.: An expanded genome-scale model of *Escherichia coli* k-12 (ijr904 gsm/gpr). Genome Biology 4(9), R54.1–R54.12 (2003)
11. Rocha, M., Maia, P., Mendes, R., Pinto, J.P., Ferreira, E.C., Nielsen, J., Patil, K.R., Rocha, I.: Natural computation meta-heuristics for the *in silico* optimization of microbial strains. BMC Bioinformatics 9 (2008)
12. Segre, D., Vitkup, D., Church, G.M.: Analysis of optimality in natural and perturbed metabolic networks. PNAS 99, 15112–15117 (2002)
13. Shlomi, T., Berkman, O., Ruppin, E.: Regulatory on/off minimization of metabolic flux changes after genetic perturbations. PNAS 102(21), 7695–7700 (2005)
14. Stephanopoulos, G., Aristidou, A.A., Nielsen, J.: Metabolic engineering principles and methodologies. Academic Press, San Diego (1998)
15. Thiele, I., Vo, T.D., Price, N.D., Palsson, B.Ø.: Expanded metabolic reconstruction of Helicobacter pylori (iit341 gsm/gpr): an in silico genome-scale characterization of single- and double-deletion mutants. J. Bacteriol. 187(16), 5818–5830 (2005)

Using Rotation Forest for Protein Fold Prediction Problem: An Empirical Study

Abdollah Dehzangi, Somnuk Phon-Amnuaisuk, Mahmoud Manafi, and Soodabeh Safa

Center of Artificial Intelligence and Intelligent computing, Faculty of Information Technology, Multi Media University, Cyberjaya, Selangor, Malaysia
abdollah.dehzangi07@mmu.edu.my, somnuk.amnuaisuk@mmu.edu.my
mahmoud.manafi09@mmu.edu.my, soodabeh.safa07@mmu.edu.my

Abstract. Recent advancement in the pattern recognition field has driven many classification algorithms being implemented to tackle protein fold prediction problem. In this paper, a newly introduced method called Rotation Forest for building ensemble of classifiers based on bootstrap sampling and feature extraction is implemented and applied to challenge this problem. The Rotation Forest is a straight forward extension of bagging algorithms which aims to promote diversity within the ensemble through feature extraction by using Principle Component Analysis (PCA). We compare the performance of the employed method with other Meta classifiers that are based on boosting and bagging algorithms, such as: AdaBoost.M1, LogitBoost, Bagging and Random Forest. Experimental results show that the Rotation Forest enhanced the protein folding prediction accuracy better than the other applied Meta classifiers, as well as the previous works found in the literature.

Keywords: Protein Fold Prediction Problem, Ensemble Classifier, Rotation Forest, Feature Extraction, Principal Component Analysis, Decision Tree, C4.5, Random Forest, AdaBoost.M1, LogotBoost, Bagging.

1 Background

Prediction of the tertiary structure of a protein from its primary structure is a challenging task in bioinformatics and biological science. Recently, due to tremendous advancements in pattern recognition, many classifiers have been implemented and applied to challenge this task. In this paper, Rotation Forest, as a newly proposed method for building an ensemble of classifiers employed to tackle the protein fold prediction problem.

The Rotation Forest, by Rodriguez and his co-workers [1], is based on bagging algorithm [2] that aims to build a more accurate and diverse classifier. Rotation Forest uses bootstrap samples of training dataset to train a group of decision trees as well as bagging, but dissimilar to bagging, to reinforce diversity within classifiers ensemble; it splits the feature set to randomly K subset, then runs *Principal Component Analysis (PCA)* on each of them, and finally rebuilds the feature set of N linear extracted features by combining all principle components. In this way it transforms

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2010, LNCS 6023, pp. 217–227, 2010.

the feature set to the new $M(where M \leq N)$ dimensional feature spaces. This process is repeated to extract new set of features to train each of the base learners in parallel. At last, Rotation Forest combines the results of all base classifiers using majority voting.

Rotation Forest has been widely applied for different benchmarks and in many cases outperformed other Meta Classifiers such as Adaboost.M1 [3], or other classifiers such as Support Vector Machine which is considered as the state-of-the-art in machine learning ([1], [4], and [5]). To the best of our knowledge, Rotation Forest has never been applied to deal with the protein folding task. Experimental results demonstrated that the Rotation Forest enhanced the prediction accuracy as well as reducing time consumption of the classification task better than the previous related works found in the literature.

One of the most important factors that affect the performance of the Rotation Forest is the number of base classifiers. Therefore, in this paper, to study the sensitivity of the Rotation Forest to the number of base classifiers for the protein fold prediction problem, six different numbers of base classifiers in the range between 10 and 200 were employed (10, 20, 50, 100, 150, and 200). Finally, the Rotation Forest compared with the best-of-the-shelf Meta classifiers that based on boosting and bagging methods, namely: *Multi Class Adaptive Boosting (AdaBoost.M1), LogitBoost, Random Forest (RF) and Bagging* which demonstrated better results compared to other similar methods ([1], [4], [6] and [7]).

Recently, many efforts have been made to challenge the protein fold prediction problem ([8], [9], [10], [11], [12], and [13]). Most of the classification methods, used for this task were based on Artificial Neural Network (ANN) ([14], [15], [16], and [17]) and Support Vector Machine (SVM) ([18], [19], [20], and [21]). In 2001, Ding and Dubchak used three SVM based multi-class classification methods *(one-versus-others (OvO), unique one-versus others (uOvO), and all-versus-all (AvA)*), with six feature groups named: *Composition of amino acids (C), Predicted secondary structure (X), polarity (P), polarizability (V), hydrophobicity (H) and van der vaals volume (V)* [23]. They reported 56% prediction accuracy using the AvA SVM.

Motivated by the work of Ding and Dubchak [22], Bologna and Appel [14] used ensemble of four-layer *Discretized Interpretable Mulri Layer Perceptron (DIMLP)* trained with the dataset produced by Ding and Dubchak. Different to Ding and Dubchak, in their work, each classifier learned all folds simultaneously. To the best of our knowledge, they reported the highest prediction accuracy (61.1%) using same set of features introduced by Dubchak and her co-workers [23].

NNs and SVMs classifiers used again by Chung and his co-workers as basic building blocks of two-level classifier for the protein folding task. In their work, each NN or SVM was a multi-class classifier [24]; hence, the number of classifiers that they used compared to other works had been greatly reduced. In their work, the common and most popular NN based models with a single hidden layer name: *Multi Layer Perceptron (MLP), Radial Basis Function Network (RBFN), and General Regression Neural Network (GRNN*) were used. However, in Chung and his co-workers and also in their previous works, it was observed that the model constructed by using neural networks and SVMs, perform badly due to the imbalanced proportion of the data which caused high rate of false positive error.

To address this problem, Nanni used non-density-based Fisher's linear classifier (FLC) and an ensemble of Hyper-plane K-Nearest Neighbor classifier (HKNN) [12]. FLC were used to find the linear discrimination function between the classes in the dataset by minimizing the errors in the least-squares sense, and HKNN were used to find a decision surface, by separating different classes of the data, in input space. However, HKNN as a kind of K-Nearest Neighbor (KNN) (Instance Based Learner) based method, suffers from curse of dimensionality while dealing with small dataset contains high amount of features [25].

To conquer inefficiencies of the mentioned methods, and also to reduce computational complexity of the protein fold classification task, Krishnaraj and Reddy employed Boosting approaches as kind of Meta classifiers to tackle the protein fold prediction problem [26]. They employed the AdaBoost.M1 [3] and the LogitBoost [6] to tackle this task. Boosting approaches and generally bootstrap sampling based classifiers avoid false positive error and build robust prediction rules by combining weak learners [3]. They reported comparable prediction accuracy in dramatically lower time complexity (60.3% compared to 61.1% achieved by Bologna and Appel [14]) with other works have been conducted in the literature. Despite all the advantages of the boosting algorithms, they suffer from over-fitting problem while dealing with noisy and high dimensional datasets [27].

Inspired by Krishnaraj and Reddy and in order to exploit the merits of Meta classifiers, we employed the Rotation Forest which illustrated better performance for different benchmarks compared to the other Meta classifiers ([1], [4], and [28]). As like as the Random Forest [7], the Rotation Forest overcome the over-fitting problem by providing a proper method to approximate missing data when dealing with noisy data or in case which large numbers of data are missing ([1] and [7]). Results showed that the Rotation Forest outperformed previous methods developed in the literature for the protein fold prediction problem.

The rest of this paper is organized as follows: in section (2), we introduced the Rotation Forest, how it works and tools which were used in this experiment. In section (3), we introduced the dataset and the features that used in this study. Section (4), concerned about the results and discussion achieved and finally followed by section (5), where the conclusions and future works were explained.

2 Rotation Forest

The Rotation Forest is a recently proposed method based on bootstrap sampling and Principal Component Analysis (PCA) [29]. It builds a group of independent trained decision trees to build an ensemble of classifiers in a parallel manner [1]. Rotation Forest is formulated based on the Random Forest idea [7]. The base classifiers independently built decision trees, but instead of using decision trees for random set of features, each tree in the Rotation Forest is trained on the whole set of dataset in a rotated feature space. It splits feature set (total N features) randomly into K (K is the parameter of the algorithm) subsets and then applied principal component analysis separately to each subset. Finally, based on all principal components the data is transformed linearly into new feature space and make new set of ($M \leq N$ in case

where some of Eigen Values are zero [1]) linear feature set by combining all K transformed feature subsets [4].

In Rotation Forest as in the bagging algorithm, bootstrap samples are taken as the training set for the individual classifiers [30]. It performs transformation of feature set for each of the base classifiers, trains each classifier with a boot strap sample of train dataset and transformed feature set, and finally combines all independent base classifiers by using majority voting. In the Rotation Forest classifier, diversity within the classifier ensemble and individual prediction accuracy of the base learners are considered, simultaneously. In this method, diversity is enhanced through feature extraction for each base classifier better than the Random Forest which just uses feature selection to encourage diversity within ensemble classifier [7]; and individual accuracy is also pursued by maintaining all principal components and also using whole dataset to train each base classifier [4].

One of the useful characteristics of the Rotation Forest is that it can be used with almost any base classifier which makes it more flexible than the Random Forest which is capable to be used with Decision Trees as base classifier [7]. Therefore, a lot of possible improvements and modifications can be considered in the Rotation Forest [1]. However, in this paper, decision trees were chosen because of its sensitivity to the rotation of the feature axe.

Data mining toolkit WEKA (Waikato Environment for Knowledge Analysis) version 3.6.0 is used for the classification. WEKA is an open source toolkit and it consists of a collection of machine learning algorithms for solving data mining problems [30]. In this experiment, J48 (WEKA's own version of C4.5 [31]) decision tree algorithm is used as a base classifier. C4.5 is an algorithm used to generate a decision tree.

C4.5 uses the fact that each attribute of the data can be used to make a decision that splits the data into smaller subsets. C4.5 examines the normalized information gain (difference in entropy) that results from choosing a feature for splitting the data [31].

$$SplitInfo_x T = -\sum_{i=1}^{n} \frac{T_i}{T} \log_2 \frac{T_i}{T} \tag{1}$$

$$GainRatio_x(T) = \frac{Gain_x(T)}{SplitInfo_x T} \tag{2}$$

Where $SplitInfo_x T$ represents the potential information provided by dividing dataset, T, into n partition corresponding to the outputs of attributes x, and $Gain_x(T)$ is how much gain would achieve by branching on x.

3 Dataset and Features

To compare our results with the previous work have done by the literature, we used the dataset introduced by Ding and Dubchak [22]. This dataset contains a train and a

test dataset. The training dataset comprises of 313 protein belong to the 27 most populated protein folds in *Structural Classification of Protein (SCOP)* protein databank [32], [33]. Each fold contains seven or more proteins. The dataset represents all major structural classes (α, β, α/β, and α + β). The original test dataset is based on the *Protein Data Bank (PDB)* protein databank [34]; it is also developed by the authors of the SCOP database. This dataset contains 385 proteins. Over 90% of our data in the test set has less than 20% sequential similarity with the proteins in test set. Among these proteins, two proteins (2SCMC and 2GPS) in the training dataset and two proteins (2YHX_1 and 2YHX_2) in the testing dataset excluded due to insufficient sequence information. As a result, there are 311 and 383 proteins remain respectively in training and testing dataset.

In this paper, six feature groups were introduced by Dubchak and her co-workers were used [23]. These feature groups were extracted from the proteins amino acid-sequence based on physical, chemical and physiochemical properties of amino acids, named: *amino acids composition (C), predicted secondary structure based on normalized frequency of a-helix residue (S), hydrophobicity (H), normalized Van Der Waals volume (V), polarity (P), and polarizability (Z).* In particular, the first feature represents a vector of the percentage composition of the 20 amino acids in the sequence. The other feature vectors properties are based on three descriptors: composition, percent composition of three constituents (polar, neutral and hydrophobic residues); transition, the transition frequencies (polar to neutral, neutral to hydrophobic, etc.): and distribution, the distribution pattern of constituents (where the first residue of a given constituent is located, and where 25%, 50%, 75%, and 100% of that constituent are contained). Therefore, there are 20 features in composition feature vector and 21 features for other feature vectors. More detail can be found in the literature ([11], [22], and [35]). The length of the amino acid plays an important role in the protein folding task ([14], [35], and [36]. Thus, it is included in every combination of feature groups for experiments.

4 Results and Discussion

The proposed method was evaluated for eleven different combinations of feature groups compared to six combinations of the feature groups used by Ding and Dubchak [22], and Krishnaraj and Reddy [26]. New combinations of the feature groups were applied to find proper combination of features and also investigate the effectiveness of each feature group to the achieved prediction accuracy. In addition, the length of proteins was also added to the all combinations of the feature groups due to its discriminatory information ([11], and [14]).

To study the sensitivity of the Rotation Forest to the number of base classifiers, the employed method with six different numbers of base classifiers in range between 10 and 200 were used for the applied dataset (10, 20, 50, 100, 150, and 200). As shown

in Table.1, the best result was achieved by applying the Rotation Forest with 100 base classifiers to the combination of the all feature groups, and the lowest prediction accuracy was obtained by using 10 base classifiers. As we can see in Table.1, by raising the number of base classifiers from 10 to 100, the prediction accuracy of the Rotation Forest also increased significantly, but differences in prediction accuracy between 100, 150 and 200 was nontrivial. Therefore, using 100 base classifiers can be addressed to the future works as an appropriate number of the base classifiers for the applied dataset or more generally for the protein fold prediction task (in similar cases).

According to the results, by using the Rotation Forest with 100 base classifiers, we achieved a 62.4% prediction accuracy which is 1.3% higher than the result reported by Bologna and Appel [14] and 2.1% higher than Nanni [12] and Krishnaraj and Reddy [26] (Table.2). We also achieved a 56.9% prediction accuracy, using the Rotation Forest with 50 base classifiers by employing the composition of amino acid feature group (20-dimensional feature group) which is slightly better than the result achieved by Ding and Dubchak [22] using the AvA SVM and the combination of four feature groups (*Amino Acids Composition, Predicted Secondary Structure, Hydrophobicity, Polarity*).

As shown in Table.1, the Rotation Forest classifier was capable of achieving to the high prediction accuracy depends on the using the appropriate number of base classifiers. The computational complexity of this method was also crucially depended on the number of base classifiers. Therefore, using the Rotation Forest classifier with appropriate number of the base classifiers can achieve to the high prediction accuracy as well as reducing the computational complexity compared to the SVM or ANN based classifiers ([1] and [4]). Despite using PCA algorithms as a feature extraction approach, having parallel structure and using simple and fast base learner (C4.5); made the Rotation Forest classifier as fast as the other Meta classifiers that were based on boosting algorithm (AdaBoos.M1 and LogitBoost). In this paper, the highest result achieved by using 100 base classifiers for Rotation Forest in a comparable computational complexity to the AdaBoost.M1 using the same number of base classifiers ([26]).

Table 1. Comparison of the results achieved (in percentage) by using the Rotation Forest with six different numbers of base classifiers for eleven combinations of the feature groups

Number of Base Classifiers	C	CS	CSV	CSZ	CSP	CSH	CSH V	CSH P	CSH PV	CSH PZ	CSHP ZV
10	50.1	54.3	54.8	55.9	53.5	54.0	53.0	54.8	52.0	54.3	50.1
20	54.3	57.2	55.1	55.4	56.9	56.9	57.2	55.9	56.9	59.0	58.0
50	56.9	59.3	**60.1**	60.1	59.5	60.1	60.6	60.6	60.0	60.6	60.0
100	56.7	58.0	60.8	60.3	59.3	60.6	58.8	60.8	58.7	61.4	**62.4**
150	56.7	60.1	61.1	60.3	**62.1**	60.6	59.5	61.4	61.4	60.3	59.8
200	56.7	60.6	59.8	57.4	61.6	60.8	59.8	60.3	60.8	61.1	**62.3**

The other remarkable result achieved by applying the Rotation Forest for the combination of three feature groups (*Composition of Amino Acid, Predicted Secondary Structure and Polarity feature groups in addition to the length feature*). We achieved 62.1% prediction accuracy, which was 1% higher than the result reported by Bologna and Appel [14] for the independent test dataset.

Table 2. Results achieved by using the Rotation Forest (in percentage) compared to the results achieved by the related works found in the literature for the protein fold prediction problem

[22]	OvO (SVM)	C+S+H	45.2
[22]	Unique OvO (SVM)	C+S+H	51.1
[22]	AvA(SVM)	C+S+H+P+Z+V	56.4
[24]	MLP-Based HLA	C+S	48.6
[24]	RBFN-Based HLA	C+S+H+P+Z+V	56.4
[24]	SVM-Based HLA	C+S+H+P+Z+V	53.2
[26]	AdaBoost.M1	C+S+H	58.2
This Paper	Rotation Forest (150 Decision Trees)	C+S+H	62.1
[26]	LogitBoost	C+S+H+P+V	60.3
[14]	DIMLP	C+S+H+P+Z+V	61.1
[10]	HKNN	C+R+H+P+Z+V	57.4
[12]	RS1_HKNN_K125	C+S+H+P+Z+V	60.0
[12]	RS1_KHNN_K25	C+S+H+P+Z+V	60.3
[11]	BAYESPROT	C+S+H+P+Z+V	58.8
[9]	MOFASA	C+S+H+P+Z+V	60.0
[8]	ALH	C+S+H+P+Z+V	60.8
[38]	RBF Majority voting Fuse	C+S+H+P+Z+V	49.7
[38]	RBF Bayesian Fuse	C+S+H+P+Z+V	59.0
This Paper	Rotation Forest (100 Decision Trees)	C+S+H+P+Z+V	***62.4***

In a different task, the employed method compared to the other Meta classifiers, such as the AdaBoost.M1 that is declared to be the best-of-the-shelf Meta classifier [1], [27], the Logitboost that has been successfully applied for different tasks [6], the Bagging as one of the most popular Meta classifiers has been applied for different machine learning tasks, and the Random Forest, recent modified version of the bagging that showed remarkable results compared to the other meta classifiers ([7], and [37]). Each Meta classifier was tested with the combination of the all feature groups.

For all of Meta classifiers default parameters applied except for the base classifiers and the number of base classifiers. In this paper, the J4.8 and the Decision Stump were respectively employed as the base learners for the AdaBoost.M1 and the LogitBoost based on the experiment were conducted by Krishnaraj and Ready [26]. The J4.8 was also used as the base classifier for the bagging to compare how the modifications made in the Rotation Forest would affect its performance compared to bagging by using the same base learner. The numbers of base learners for all cases were set to 100 as well as the Rotation Forest.

Table 3. Results achieved by using the Rotation Forest with 100 base classifiers for each individual fold, compared to the AdaBoost.M1, LogitBoost, Bagging and the Random Forest as best-of-the-shelf Meta classifiers based on boosting and bagging approaches. For most of the fold, the Rotation Forest demonstrated better results compared to the other Meta classifiers

Index	Fold	N-test	Rotation Forest	AdaBoost.M1	LogitBoost	Bagging	Random Forest
	α						
1	Globin-like	6	83.3%	83.3%	83.3%	83.3%	83.3%
3	Cytochrome c	9	100.0%	77.8%	55.6%	77.8%	88.9%
4	DNA-Binding 3-Helical	20	85.0%	70.0%	55.0%	55.0%	60.0%
7	4-helical up-and-down bundle	8	25.0%	75.0%	37.5%	62.5%	25.0%
9	4-helical cytokines	9	100.0%	88.9%	77.8%	88.9%	100.0%
11	Alpha; EF-hand	9	33.3%	22.2%	22.2%	11.1%	33.3%
	B						
20	Immunoglobulin-like	44	72.7%	70.5%	77.3%	63.6%	84.1%
23	Cuperdoxins	12	16.7%	41.7%	16.7%	16.7%	25.0%
26	Viral coat and capsid	13	69.2%	76.9%	76.9%	76.9%	76.9%
30	ConA-like lectins/glucanases	6	33.3%	33.3%	33.3%	33.3%	33.3%
31	SH3-like barrel	8	75.0%	75.0%	62.5%	75.0%	62.5%
32	OB-fold	19	26.3%	21.1%	36.8%	21.1%	31.6%
33	Trefoil	4	50.0%	75.0%	50.0%	75.0%	50.0%
35	Trypsin-like serine proteases	4	25.0%	25.0%	50.0%	25.0%	25.0%
39	Lipocalins	7	42.9%	28.6%	57.1%	28.6%	57.1%
	α/β						
46	(TIM)-barrel	48	87.5%	81.3%	81.3%	75.0%	91.7%
47	FAD (also NAD)	12	58.3%	58.3%	41.7%	50.0%	58.3%
48	Flavodoxin-like	13	61.5%	46.2%	46.2%	46.2%	46.2%
51	NAD(P)-binding Rossmann fold	27	40.7%	33.3%	51.9%	33.3%	25.9%
54	P-loop containing nucleotide	12	58.3%	33.3%	33.3%	41.7%	33.3%
57	Thioredoxin-like	8	62.5%	37.5%	50.0%	50.0%	50.0%
59	Ribonuclease H-like motif	12	66.7%	50.0%	58.3%	66.7%	58.3%
62	Hydrolases	7	71.4%	28.6%	57.1%	57.1%	28.6%
69	Periplasmic binding	4	25.0%	0.0%	25.0%	0.0%	25.0%
	α+ β						
72	β-grasp	8	37.5%	25.0%	37.5%	25.0%	25.0%
87	Ferredoxin-like	27	29.6%	48.1%	55.6%	37.0%	37.0%
110	small inhibitors	27	100.0%	100.0%	85.2%	100.0%	96.3%
	TOTAL	383	**62.4%**	58.5%	59.0%	55.4%	59.8%

The overall results were shown in Table.3. Based on the results (Table.3), the Rotation Forest achieved at least, more than 2% higher prediction accuracy compared to other Meta classifiers based on boosting and bagging. It also achieved to the highest prediction accuracy for 16 folds compared to the other employed classifiers, which shows the ability to enhance the prediction accuracy of the Rotation Forest for each fold separately (Table.3). It outperformed Random Forest, the other modification of the bagging classifier by more than 2% of the prediction accuracy, as well as the AdaBoost.M1 classifier by more than 3% of prediction accuracy. Our experimental

results showed that the Rotation Forest outperformed the methods which have been used for the protein fold prediction task as well as other Meta classifiers based on boosting and bagging algorithms (AdaBoost.M1, LogitBoost, Bagging, and Random Forest).

5 Conclusion and Future Works

In this paper, an empirical study on the performance and advantages of using the Rotation Forest to solve the protein fold recognition problem were conducted. We also studied the sensitivity of the Rotation Forest to the number of base classifiers by using six different numbers of base classifiers in range between 10 and 200. Finally, employed method compared to the other Meta classifiers based on boosting and bagging approaches which have showed remarkable results on different benchmarks ([1], [2], [3], [6], and [7]).

The ensemble classifier built using the Rotation Forest with 100 base classifiers achieved better results compared to the previous works found in the literature as well as the other best-of-the-shelf Meta classifiers, namely: the AdaBoost.M1, LogitBoost, Bagging and the Random Forest. The proposed method achieved a 62.4% prediction accuracy which is 1.3% higher than the result achieved by Bologna and Appel [14] who used an ensemble of DIMLP, more than 2% better than Nanni [12] who used ensemble of HKNN with FCA and Krishnaraj and Reddy [26] who used the AdaBoost.M1 and the LogitBoost with the same number of base classifiers.

High prediction performance as well as low computational complexity and time consumption of the Rotation Forest shows the potential of this method for further researches. The Rotation Forest is capable to be used by any classifier as a base classifier, being used in hierarchical structure or as part of an ensemble of heterogeneous classifiers to achieve better results for the protein fold prediction problem and other classification tasks.

Acknowledgements

We would like to thank professor Ding and professor Dubchak for letting us use their datasets. We also would like to thank Omid Dehzangi, Kien-Huei Chan, and Keng-Hoong Ng for helpful discussions.

References

1. Rodríguez, J.J., Kuncheva, L.I., Alonso, C.J.: Rotation forest: A new classifier ensemble method. IEEE Transactions on Pattern Analysis and Machine Intelligence 28(10), 1619–1630 (2006)
2. Breiman, L.: Bagging Predictors. Machine Learning 24(2), 123–140 (1996)
3. Freund, Y., Schapier, R.E.: A Short Introduction to Boosting. Journal of Japanese Society for Artificial Intelligence 14(5), 771–780 (1997)

4. Kuncheva, L.I., Rodríguez, J.J.: An Experimental Study on Rotation Forest Ensembles. In: Haindl, M., Kittler, J., Roli, F. (eds.) MCS 2007. LNCS, vol. 4472, pp. 459–468. Springer, Heidelberg (2007)
5. Stiglic, G., Kokol, P.: Effectiveness of Rotation Forest in Meta-learning Based Gene Expression Classification. In: Proceedings of Twentieth IEEE International Symposium on Computer-Based Medical Systems (2007) ISBN: 0-7695-2905-4
6. Friedman, J., Hastie, T., Tibshirani, R.: (Published version) Additive Logistic Regression: a Statistical View of Boosting Annals of Statistics 28(2), 337–407 (2001)
7. Breiman, L.: Random Forest. Machine learning. Kluwer Academic Publishers, Dordrecht (2001) ISSN: 0885-6125
8. Kecman, V., Yang, T.: Protein Fold Recognition with Adaptive Local Hyper plane Algorithm. In: IEEE Symposium Computational Intelligence in Bioinformatics and Computational Biology, CIBCB 2009 (2009); 4925710
9. Stanley, Y., Shi, M.P., Suganthan, N.: Multiclass protein fold recognition using multiobjective evolutionary algorithms. Computational Intelligence in Bioinformatics and Computational Biology (2004); 0-7803-8728-7
10. Okun, O.G.: Protein Fold Recognition with K-local Hyperplane Distance Nearest Neighbor Algorithm. In: Proceedings in the Second European Workshop on Data Mining and Text Mining in Bioinformatics, Pisa, Italy, pp. 51–57 (2004)
11. Chinnasamy, A., Sung, W.K., Mittal, A.: Protein structure and fold prediction using tree-augmented naive Bayesian classifier. Pacific Symposium on Biocomputing. In: Pacific Symposium on Biocomputing, vol. 9, pp. 387–398 (2004)
12. Nanni, L.: Ensemble of classifiers for protein fold recognition. In: New Issues in Neurocomputing: 13th European Symposium on Artificial Neural Networks, vol. 69, pp. 850–853 (2006)
13. Karplus, K.: SAM-T08, HMM-based protein structure prediction. Nucleic Acids Research 37(suppl. 2), W492–W497 (2009)
14. Bologna, G., Appel, R.D.: A comparison study on protein fold recognition. In: Proceedings of the Ninth International Conference on Neural Information Processing, November 2002, vol. 5, pp. 2492–2496 (2002)
15. Huang, C., Lin, C., Pal, N.: Hierarchical learning architecture with automatic feature selection for multi class protein fold classification. IEEE Transactions on Nano Bioscience 2(4), 221–232 (2003)
16. Lin, K.L., Li, C.Y., Huang, C.D., Chang, H.M., Yang, C.Y., Lin, C.T., Tang, C.Y., Hsu, D.F.: Feature Selection and Combination Criteria for Improving Accuracy in Protein Structure Prediction. IEEE Transactions on Nano Bioscience 6(2) (2008)
17. Lin, K.L., Lin, C.Y., Huang, C.D., Chang, H.M., Yang, C.Y., Lin, C.T., Tang, C.Y., Hsu, D.F.: Feature Selection and Combination Criteria for Improving Accuracy in Protein Structure Prediction. IEEE Transactions on Nano Bioscience (2007) ISSN: 1536–1241
18. Chen, C., Zhou, X.B., Tian, Y.X., Zou, X.Y., Cai, P.X.: Predicting protein structural class with pseudo-amino acid composition and support vector machine fusion network. Anal. Biochem. 357, 116–121 (2006)
19. Lewis, D.P., Jebara, T., Noble, W.S.: Support vector machine learning from heterogeneous data: an empirical analysis using protein sequence and structure. Bioinformatics 22, 2753–2760 (2006)
20. Zhang, S.W., Pan, Q., Zhang, H.C., Shao, Z.C., Shi, J.Y.: Prediction protein homo-oligomer types by pseudo amino acid composition: approached with an improved feature extraction and naive Bayes feature fusion. Amino Acids 30, 461–468 (2006)

21. Zhou, X.B., Chen, C., Li, Z.C., Zou, X.Y.: Improved prediction of subcellular location for apoptosis proteins by the dual-layer support vector machine. Amino Acids 35, 383–388 (2008)
22. Ding, C., Dubchak, I.: Multi-class protein fold recognition using support vector machines and neural networks. Bioinformatics 17(4), 349–358 (2001)
23. Dubchak, I., Muchnik, I., Kim, S.K.: Protein folding class predictor for SCOP: approach based on global descriptors. In: 5th International Conference on Intelligent Systems for Molecular Biology, vol. 5, pp. 104–107 (1997)
24. Chung, I.F., Huang, C.D., Shen, Y.H., Lin, C.T.: Recognition of structure classification of protein folding by NN and SVM hierarchical learning architecture. In: Artificial Neural Networks and Neural Information Processing, pp. 1159–1167 (2003)
25. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer, Heidelberg (2006) ISBN-13: 978-0387-31073-2
26. Krishnaraj, Y., Reddy, C.K.: Boosting methods for Protein Fold Recognition: An Empirical Comparison. In: IEEE International Conference on Bioinformatics and Biomedicine (2008) ISBN: 978-0-7695-3452-7
27. Cai, Y.D., Feng, K.Y., Lu, W.C., Chou, K.C.: Using LogitBoost classifier to predict protein structural classes. Journal of Theoretical Biology 238, 172–176 (2006)
28. Zhang, C.X., Zhang, J.S., Wang, J.W.: An empirical study of using Rotation Forest to improve regressors. Applied Mathematics and Computation 195, 618–629 (2007)
29. Scholkopf, B., Smola, A., Muller, K.R.: Nonlinear component analysis as a kernel eigenvalue problem. Neural Computation 10(5), 1299–1319 (1998)
30. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques, 2nd edn. Morgan Kaufmann, San Francisco (2005)
31. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, San Francisco (1993)
32. Lo Conte, L., Ailey, B., Hubbard, T.J.P., Braner, S.E., Murzin, A.G., Chothia, C.: SCOP a structural classification of proteins database 28(1), 257–259 (2000)
33. Murzin, A.G., Brenner, S.E., Hubbard, T., Chothia, C.: SCOP: a structural classification of proteins database for the investigation of sequences and structures. Journal of Molecular Biology 247, 536–540 (1995)
34. Hobohm, U., Scharf, M., Schneider, R., Sander, C.: Selection of a representative set of structure from the Brookhaven Protein Bank protein. Science 1, 409–417 (1992)
35. Duwairi, R., Kassawneh, A.: A Framework for Predicting Proteins 3D Structures. In: Computer Systems and Applications, AICCSA 2008 (2008); 978-1-4244-1968
36. Chou, K.C., Zhang, C.T.: Prediction of protein structural classes, Critical Review. Biochem. Mol. Biol. 30(4), 275–349 (1995)
37. Livingston, F.: Implementation of Breiman's Random Forest Machine Learning Algorithm. ECE591Q Machine Learning Journal Paper (2005)
38. Hashemi, H.B., Shakery, A., Naeini, M.P.: Protein Fold Pattern Recognition Using Bayesian nsemble of RBF Neural Networks. In: International Conference of Soft Computing and Pattern Recognition, pp. 436–441 (2009)

Towards Automatic Detecting of Overlapping Genes - Clustered BLAST Analysis of Viral Genomes

Klaus Neuhaus[1], Daniela Oelke[2], David Fürst[3]
Siegfried Scherer[1], and Daniel A. Keim[2]

[1] Chair of Microbial Ecology, Technische Universität München, Weihenstephaner Berg 3, 85354 Freising, Germany
[2] Chair of Data Analysis and Visualization, Universität Konstanz, Universitätsstr. 10, 78457 Konstanz, Germany
[3] Chair of Data Management and Data Exploration, Rheinisch-Westfälische Technische Hochschule Aachen, Informatik 9, 52056 Aachen, Germany

Abstract. Overlapping genes (encoded on the same DNA locus but in different frames) are thought to be rare and, therefore, were largely neglected in the past. In a test set of 800 viruses we found more than 350 potential overlapping open reading frames of >500 bp which generate BLAST hits, indicating a possible biological function. Interestingly, five overlaps with more than 2000 bp were found, the largest may even contain triple overlaps. In order to perform the vast amount of BLAST searches required to test all detected open reading frames, we compared two clustering strategies (BLASTCLUST and k-means) and queried the database with one representative only. Our results show that this approach achieves a significant speed-up while retaining a high quality of the results (>99% precision compared to single queries) for both clustering methods. Future wet lab experiments are needed to show whether the detected overlapping reading frames are biologically functional.

Keywords: overlapping genes, clustering, BLAST analysis.

1 Introduction

1.1 Overlapping Reading Frames and Overlapping Genes

DNA consists of two complementary strands, uses a triplet code and, consequently, open reading frames (ORFs), which may code for proteins, are possible in six reading frames overlapping in different phases. A protein coding reading frame on a given DNA sequence is, by convention, phase +1, the next frames are +2 and +3, and, on the other strand -1, -2, and -3 (Fig. 1). This construction results in a considerable theoretical coding density.

The term "overlapping gene" has been used in the literature for several related biological phenomena. In order to avoid confusion, we introduce a distinction

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2010, LNCS 6023, pp. 228–239, 2010.

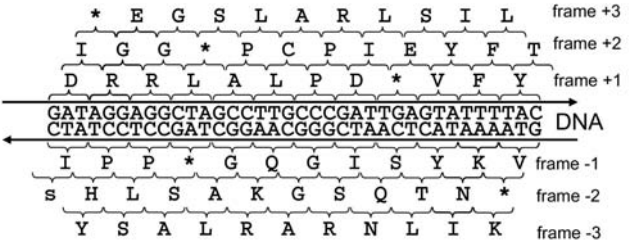


Fig. 1. A double strand of DNA with the different possible reading frames. Encoded amino acids are shown in the single letter code according to the standard genetic code. Stop codons, which do not code for an amino acid, are depicted by a star.

between trivial and embedded overlaps. In case of short, trivial overlaps, the overlapping sequence has no important function at the protein level, but may play a role in the regulation of gene expression, e.g., in transitional coupling [1,2]. The process of gain and loss of trivial overlapping genes has been modeled by mutational events, which displace the start or stop codons [3,4,5,6,7,8,9]. The focus of this project, however, is on embedded genes which encode two completely different functional amino acid chains (proteins) in different phases of the same DNA locus. Here, a protein coding reading frame is largely or entirely superposed on an annotated reading frame (Fig. 2), and is therefore termed "embedded ORF" or, if a function of the encoded protein has been demonstrated, "embedded gene".

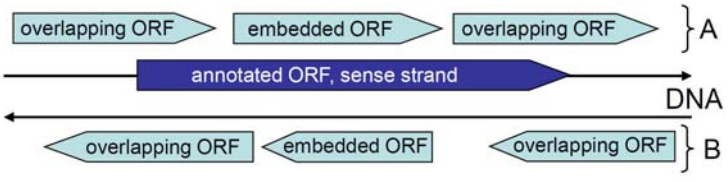


Fig. 2. Types of ORFs to be investigated. A: sense strand ORFs, either partially or completely overlapping. They can be in phase +2 or +3, since the annotated gene is by definition in phase +1. B: antisense strand ORFs, either partially or completely overlapping. They can be in phase -1, -2 or -3.

Usually, genome annotation programs consider embedded ORFs as being non-functional. The rationale may be an intuitive one: It rests on the presumed improbability of embedded genes to originate by chance [10,11]. Furthermore, overlaps pose severe restrictions on the evolution of both ORFs involved [7] since many mutations in one phase directly affect the amino acid sequence encoded in the other phase [12,13]. Nevertheless, the first fully sequenced organism, bacteriophage ΦX174, contains several embedded genes [14]. Subsequently, a number

of embedded genes have been discovered in viruses [15,16,17,18] and only recently, the existence of several overlapping genes in other organisms has been acknowledged [19,20]. Okamura et al. [21] suggested that amino acid chains encoded in alternative reading frames are a hidden coding reserve serving as novelty pool. Once a frame shift mutation occurs, such formerly hidden ORFs become exposed, which means that they are translated. Along those lines we wanted to examine how many of the (overlapping) reading frames currently not annotated have sufficient similarity to annotated genes to generate a BLAST hit in GenBank.

1.2 Detection of Embedded Genes by Computational Methods

An ORF, by definition, starts with a start codon and ends with a stop codon. Clearly, not all ORFs are genes. To identify the genes among the many ORFs in a genome during the annotation process is one of the central tasks of bioinformatics. Numerous algorithms have been developed by many bioinformatic groups, such as GeneMark, Glimmer, ZCURVE, BLASTX, FASTA, ORPHEUS or EasyGene (for an overview see [22]). After identification, such ORFs which are likely to be true genes are labeled "annotated" and recorded in genome databases. The aim of this work is to examine genomes by using the implicit state-of-the-art knowledge recorded in databases in terms of annotated genes to see, if "hidden" overlapping reading frames can be discovered. In this feasibility study, we restricted ourselves to viral genomes sequenced until May 2008. However, the main problem is that the number of ORFs encoded in genomes is huge and, therefore, even searches which use a locally installed copy of these databases would take months to complete. An efficient strategy to perform these searches is therefore imperative. One important method to reduce the number of database queries is to use clustering algorithms to meaningfully group the ORFs (see [23,24] for a review of common clustering algorithms) and then only perform one query per cluster. This allows to query databases such as GenBank using BLAST with only a fraction of the ORFs and to transfer the query results to the rest of the cluster without risking that the introduced error is too large. In later stages, received hits will be analyzed with further bioinformatic methods, e. g. promoter predictions and alike.

2 Computational Methods for the Detection of Overlapping Genes

2.1 The Data

For our analysis, we downloaded all available viral nucleotide sequences from [25] on May 26, 2008 (nearly 3,000 viruses). All open reading frames (ORFs in six reading frames) were extracted from the viral sequences using *getorf* [26]. ORFs with less than 150 base pairs have not been considered, since smaller ORFs rarely encode for a functional protein [27]. In total, about 229,000 ORFs with at least 150 base pairs were extracted. To find out if the ORFs extracted eventually

encode functional proteins a comparison with the NCBI-Protein-Database *nr* [28] is conducted. At the day of the download (May 26, 2008) the database contained 6,544,368 protein sequences, totalling 5.33 GB of data. To query the database the ORFs are translated to the corresponding amino acid sequence. This poses a lesser constraint in finding potentially functional sequences in the ORFs not originally annotated [22].

2.2 Querying the Database

For querying the database we use the Basic Local Alignment Search Tool (BLAST) that comprises a set of similarity search programs that were designed to find regions of similarity between biological sequences [29]. BLAST allows searching large databases for optimal local alignments. A list of the sequences with the best local alignments is returned including similarity scores for each sequence. In order to efficiently access the database, the collection of BLAST algorithms was installed locally instead of using the web based version located on the NCBI server.

For an arbitrarily chosen subset of 76,928 ORFs the above mentioned NCBI-Protein-Database *nr* was queried using BLASTP, the algorithm of the BLAST suite for querying protein databases. We used default BLASTP parameters except the cut-off for the e-value has been set to ≤ 0.1. About 43% of the sequences generated a hit which includes the already annotated genes. In average, it took about 47.5 seconds per query. In total, we needed more than 42 days to process the test dataset. Processing all 229,000 ORFs would have taken about 4 month. Thus, the approach is clearly not efficient for future studies comprising more sequence data.

2.3 Clustering for Speed-Up

To cut runtime, we first cluster the sequences according to their similarity and subsequently query the database with only one representative.

We compared the results of two different clustering algorithms: BLASTCLUST [30] from NCBI [25] and k-means [24]. BLASTCLUST provides hierarchical clustering based on the single linkage approach. Basically, it implements the BLAST-algorithms, which take evolutionary relatedness into account. The advantages are to use end-to-end the same algorithm, and that two sequences can be directly compared without transforming them into an information reduced vector. In contrast to this, applying k-means as a partitioning-based clustering algorithm requires to transform each sequence into a point in Euclidean space. A histogram is formed by counting the occurrence of each amino acid in an ORF. The result is expressed as a 20-dimensional feature vector and similar sequences are assumed to locate at a similar position in this feature space.

Running BLASTCLUST with the default settings resulted in a set of 181,015 clusters. By subsequently relaxing the similarity requirements for sequences that are placed in the same cluster, the number of clusters was reduced to 164,593 (score density threshold S=1.0), 160,915 (S=0.5), 156,009 (S=0.001),

and 121,774 (S=5, length covered L=0.1)[1]. The least stringent clustering reduced the dataset approximately half to the starting number. In order to get comparable results the exact same amount of clusters were generated using k-means. Our analysis for both methods showed that there are many clusters that contain only a single sequence and only few containing 20 or more sequences. The reason for this is that the applied thresholds using BLASTCLUST are quite strict to ensure that the whole data set is represented well. Regarding the speed of the clustering process the k-means algorithm turned out to be about five times faster than BLASTCLUST (approximately 7.5h compared to 38h, respectively). However, this difference becomes insignificant if we look at the time required for the total analysis. Figure 3 shows the total runtimes for clustering and data base queries. For the smallest number of clusters, the approach saved up to 45% of the runtime compared to querying each single sequence which in case of about 229,000 candidate sequences accounts for a saving of 58 days. Since BLAST is a computationally demanding algorithm, this achievement is significant. Further runtime reductions using BLAST necessitates special computer hardware [31].

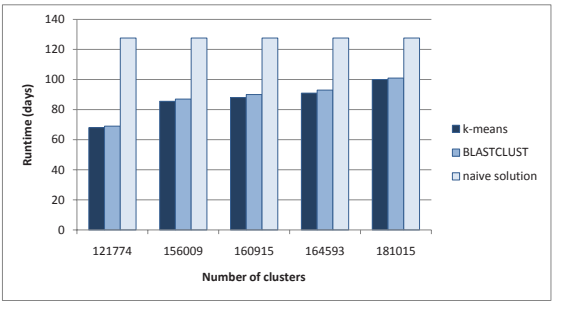


Fig. 3. Comparison of the total runtimes needed as a function of the number of clusters. For comparison the time that would be needed when querying each single sequence is given as well.

2.4 Effectiveness of the Approach

Our experiments presented in the former section clearly show that the process can be significantly accelerated by applying clustering techniques before the database is queried. However, this approach can only be considered as useful if the quality of the results remains acceptable.

In order to evaluate this, a measure for effectiveness has to be found. Key for the effectiveness of our approach is to get clusters with a high purity. A cluster is considered as "pure" if it contains only sequences that generate the same BLAST

[1] The *S*-value denotes the score density which is calculated by dividing the BLAST score by the length of the alignment. The *L* value specifies the percentage of the length of the sequence that must be covered. If not specified, *L* is set to 1.0 (=100%) in our experiments.

hit (or none) if tested one by one. If this is the case, then our assumption holds that the result that we get for one sequence can be transferred to all the other sequences in the same cluster. We measure this purity by calculating a precision score for each cluster. The precision is thereby defined as:

$$Precision(C) = max\{Precision_f(C), Precision_{nf}(C)\} \tag{1}$$

where

$$Precision_f(C) = \frac{O_f}{O_a},$$

$$Precision_{nf}(C) = \frac{O_{nf}}{O_a}.$$

with
O_f = number of functional ORFs
O_{nf} = number of non-functional ORFs
O_a = number of all analyzed ORFs

In the experiment that is described in section 2.2, we queried the database using a subset of 76,928 sequences. This dataset serves now as basis for evaluation. Since we could not include all sequences in the experiment due to time constraints, the calculated numbers can only be considered as an approximation of the precision.

While equation 1 gives us a precision value for a single cluster, we would need a value that measures the quality of the complete clustering. To take the significant differences in cluster sizes into account, we use a weighted average to calculate the cluster precision (equation 2; k = number of clusters, t_i = number of sequences in cluster C_i).

$$Precision = \frac{\sum_{i=1}^{k}(Precision(C_i) \cdot t_i)}{\sum_{i=1}^{k} t_i} \tag{2}$$

Both clustering algorithms, BLASTCLUST and k-means, were evaluated by calculating the precision values for the clustering structure which we retrieved from section 2.3. Figure 4 shows the results. Less cluster lower the precision, which is expected since a smaller number of clusters corresponds to a lesser stringency. Despite the fact that the k-means clustering places the sequences in a 20-dimensional Euclidean space without any consideration of biological significance, the average performance of both clustering algorithms is approximately, and quite surprisingly, the same. It somehow appears that the proteins composition is quite able to circumscribe a cluster. The precision values are convincingly high which confirms our assumption that the speed-up that we gain from using clustering algorithms does not significantly decrease the quality of the results. Using our method, with a loss of at most 0.1% of the precision (for k-means 0.5 percent) we were able to get a speed-up of approximately 33%. If we are willing

to accept a loss in the precision of about 1% (2.5% for BLASTCLUST), the acceleration was even higher with savings of about 45%.

Despite the fact that many "cluster" contain only one sequence, the precision drops faster for BLASTCLUST when relaxing stringency (Fig. 4). This is due to the fact that BLASTCLUST tends to cluster different (in the sense of gaining different BLAST hits) sequences in larger clusters, resulting in a dramatic drop in precision for cluster $\geq$12 sequences of around 90%.

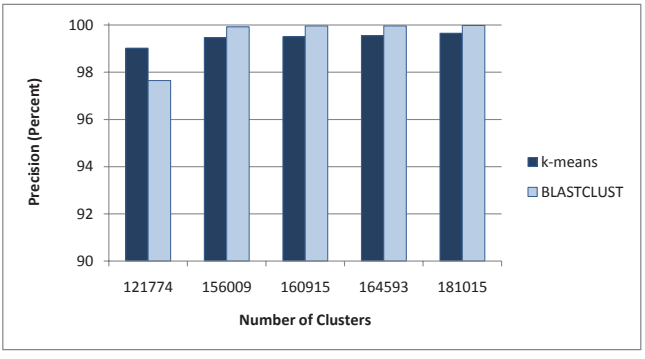


Fig. 4. Comparison of the precision of the clustering BLASTCLUST and k-means for different numbers of clusters

2.5 Detection of Presumed Overlapping Genes

After recording all ORFs with BLAST hits we determined those which overlap. The analysis of occurring overlaps was conducted on 800 arbitrarily chosen viruses. Their genomes were sequentially examined to find ORFs which overlap in different reading frames. To distinguish between trivial (short) and non-trivial cases, lengths of overlap were recorded and are shown in Figure 5.

3 New Overlapping Genes in Viruses

In total, about 800 viruses were analyzed for overlapping gene sequences which generated a BLAST hit. From those, 62% of the genomes contained at least trivial overlaps, whereas in 44% of the viral genomes overlaps of 100% could be observed. Since non-trivially overlapping genes are considered to be unlikely, one reading frame is usually dismissed in favor of the other one. For instance, Yooseph et al. [10] dismiss overlapping encoded genes if their orthologous cluster is smaller than the cluster of the corresponding gene. However, current genome databases implicitly reflect our state-of-the-art knowledge about which ORF might be (or is) a gene and which one is not. Therefore, a BLAST hit can be used as first approximation for a possible biological function.

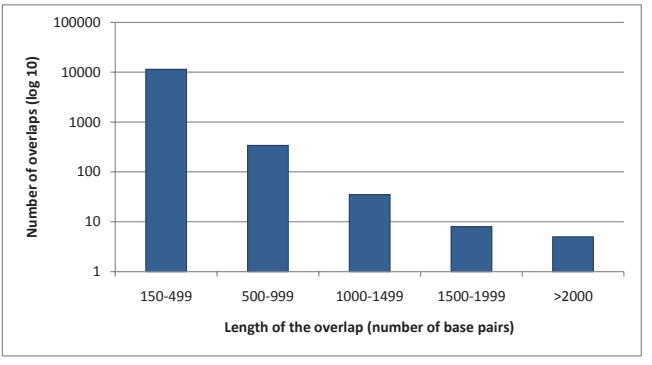


Fig. 5. Histogram of the recorded lengths of the overlaps

Especially, longer overlaps are of biological interest. Figure 5 shows a histogram of the length of the detected overlapping open reading frames which generated a BLAST hit. As expected, there are many short ORFs with less than 500 base pairs. However, there is a considerable number of longer overlapping open reading frames (>350 cases for 500 and more bp) and even five presumed genes with ORFs of more than 2000 base pairs. Since only 800 viruses have been examined, we expect further 1000 of such cases in the rest of the 2200 viral genomes. Indeed, several recent publications about viral genome analysis revealed new overlapping genes. However, those searches included evidence of positive selection (see [32] and references therein).

In the past, viral overlapping genes have been considered to be a "specialty" of these organisms. Most often, viruses have only a limited genome size due to capsid size constraints, with some notable exceptions like *Acanthamoeba polyphaga* mimivirus (genome length ≈1.2Mbp). Indeed, in viruses the number of overlapping genes is inversely correlated with genome length [33]. However, in bacteria, the number of overlapping genes corresponds with genome size and as a rough approximation it can be said that 10-30% of genes overlap [34,7]. But most of those overlaps are trivial, which means the overlapping encoded amino acid chain is not functional at the protein's level. Biologically more interesting are nested genes in which both protein chains assume a function. A recent textbook like example might be the gene pair *dmdR1* and *adm* from *Streptomyces*. Both genes are antiparallel to each other. DmdR1 regulates iron metabolism and Adm is a regulator for antibiotic production. Quite interestingly, both ORFs were recorded in databases with at that time unknown functions [35].

The longest overlapping ORFs we could find in our analysis is from *Mycobacterium* phage PBI1 and is a very interesting case (Figure 6). The largest ORF, per definition +1, has been annotated as protein g27 (accession no. YP_655223), but no function has been assigned to it [36]. This reading frame encodes 1590 amino acids and starts according to the GenBank entry with an unusual GTG start codon. However, this start might be questionable, since another CTG start codon can be found upstream and an ATG start codon downstream of the

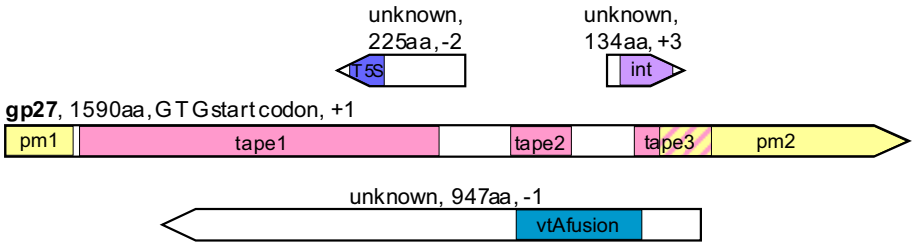


Fig. 6. The genetic locus of the *Mycobacterium* phage PBI1 in which the protein gp27 is encoded (longest arrow; locus tag PBI1p27). This ORF contains three further embedded genes, shown above or below (smaller arrows). For each ORF the length in amino acids (aa), as well as the phase of the reading frame is given. Marked with boxes in different colors are areas generating BLAST hits. int, integron integrase; pm1-2, putative membrane protein of phage origin; T5S, type V secretory pathway protein; tape1-3, tape measure protein; vtAfusion, viral A-type inclusion protein. For further details see text.

annotated GTG. This reading frame contains two annotated conserved domain fragments, *flhB* from the flagellar biosynthesis protein FlhB and *tra*, which encodes a transglycosylase-like domain. If BLASTed, this ORF will generate three hits with identical ORFs from very similar phages. The e-value of those hits is 0. The next four hits are from different but still related *Mycobacterium* phages, the e-values are in a range of $6 \cdot 10^{-61}$ to $1 \cdot 10^{-51}$ [36]. The next BLAST hit reveals a putative membrane protein of phage origin in *Mycobacterium marinum* strain M; the e-value being $9 \cdot 10^{-51}$ (indicated in Figure 6 with pm1 and pm2). The next hit further down the list, with an e-value of $9 \cdot 10^{-30}$, has similarity to a tape measure protein. Interestingly, several more hits of such tape measure proteins can be found within the first 50 BLAST hits. Areas in which the phage g27 gene generates hits with such tape measure proteins are indicated with tape 1 to 3 in Figure 6. The multiple occurrences of similar hits in a BLAST search indicates that this protein may be indeed a tape measure protein. Interestingly, the sequence of such proteins is under minimal constraints only. It determines the length of a phage tail very much like a ruler. A shorter tape measure protein means a shorter tail and vice versa. Therefore, other protein chains in overlapping reading frames may be easily encoded. Indeed, several additional ORFs can be found embedded in the tape measure protein gene. The largest embedded ORF in frame -1 comprises a protein of 947 amino acids. Amino acid positions 623 to 844 generate a BLAST hit to a viral A-type inclusion protein with an e-value of 0.081 (vtAfusion in Figure 6). Such proteins form inclusion bodies in the host cell during infection [37]. Surprisingly, two more ORFs with BLAST hits are encoded on the same locus of DNA, resulting in triple overlaps. One ORF with 225 amino acids in frame -2 generates a BLAST hit with a type V secretory pathway protein (ZP_04858685, e-value 0.035, T5S in Figure 6). Those proteins are autotransporters, which transport a protein domain across the membrane of a bacterium [38]. Finally, the last ORF generating a hit encodes 134 amino

acids in frame +3. The protein seems to be an integron integrase (e-value 0.031; int in Figure 6). Integrases belong to the large group of mobile genetic elements [38]. It is conceivable that at one point in time such a mobile element jumped into the tape measure protein gene and became incorporated in this DNA locus. Triple overlaps have only rarely been reported [14,39].

4 Conclusions and Future Prospects

We could demonstrate that in viral genomes several overlapping open reading frames can be found which generate a BLAST hit, which is usually considered as first evidence for a presumed biological function. To speed up BLAST searches for large datasets we implemented clustering strategies. By applying clustering methods previous to querying the database with one representative of each cluster a significant acceleration is possible (in our experiments up to 45%) while retaining a high quality of the results. Our initial results are promising and suggest that further research in this area might be fruitful. For reasons mentioned in the introduction it is comprehensible that embedded ORFs have been almost completely out of focus of experimental and bioinformatic research. Nevertheless, the lack of attention is about to change. Several databases with the aim to aid in the area of overlapping genes have been set up recently [19,40,27]. Molecular studies reveal overlapping genes in a diversity of organisms (e. g., see [27]). Therefore, given the availability of many completely sequenced genomes at the beginning of 2010, a number of which will increase steeply in the future [41,42], we expect the discovery of many yet unknown, but functional overlapping reading frames in natural DNA. Such genes must be tested in wet lab experiments whether they indeed have a biological function.

Acknowledgments. We want to thank Prof. Dr. Thomas Seidl for fruitful discussions.

References

1. Sakharkar, K., Sakharkar, M., Chow, V.: Gene fusion in *Helicobacter pylori*: making the ends meet. Antonie van Leeuwenhoek 89, 169–180 (2006)
2. Sakharkar, M.K., Perumal, B.S., Sakharkar, K.R., Kangueane, P.: An analysis on gene architecture in human and mouse genomes. In. Silico. Biol. 5 (2005)
3. Cock, P., Whitworth, D.: Evolution of gene overlaps: Relative reading frame bias in prokaryotic two-component system genes. J. Mol. Evol. 64, 457–462 (2007)
4. Fukuda, Y., Washio, T., Tomita, M.: Comparative study of overlapping genes in the genomes of *Mycoplasma genitalium* and *Mycoplasma pneumoniae*. Nucl. Acids Res. 27, 1847–1853 (1999)
5. Krakauer, D.C.: Stability and evolution of overlapping genes. Evolution 54, 731–739 (2000)
6. Lillo, F., Krakauer, D.: A statistical analysis of the three-fold evolution of genomic compression through frame overlaps in prokaryotes. Biol. Direct. 2, 22 (2007)

7. Luo, Y., Fu, C., Zhang, D.-Y., Lin, K.: Overlapping genes as rare genomic markers: the phylogeny of γ-Proteobacteria as a case study. Trends Genet. 22, 593–596 (2006)
8. Luo, Y., Fu, C., Zhang, D.-Y., Lin, K.: BPhyOG: An interactive server for genome-wide inference of bacterial phylogenies based on overlapping genes. BMC Bioinformatics 8, 266 (2007)
9. Sabath, N., Graur, D., Landan, G.: Same-strand overlapping genes in bacteria: compositional determinants of phase bias. Biol. Direct. 3, 36 (2008)
10. Yooseph, S., Sutton, G., Rusch, D.B. (and coworkers): The Sorcerer II Global Ocean Sampling Expedition: expanding the universe of protein families. PLoS Biol. 5, e16 (2007)
11. Zaaijer, H.L., van Hemert, F.J., Koppelman, M.H., Lukashov, V.V.: Independent evolution of overlapping polymerase and surface protein genes of hepatitis B virus. J. Gen. Virol. 88, 2137–2143 (2007)
12. Mizokami, M., Orito, E., Ohba, K., Ikeo, K., Lau, J.Y., Gojobori, T.: Constrained evolution with respect to gene overlap of hepatitis B virus. J. Mol. Evol. 44(suppl. 1), 83–90 (1997)
13. Nekrutenko, A., Wadhawan, S., Goetting-Minesky, P., Makova, K.D.: Oscillating evolution of a mammalian locus with overlapping reading frames: an XLαs/ALEX relay. PLoS Genet. 1, 18 (2005)
14. Sanger, F., Air, G.M., Barrell, B.G., Brown, N.L., Coulson, A.R., Fiddes, C.A., Hutchison, C.A., Slocombe, P.M., Smith, M.: Nucleotide sequence of bacteriophage ϕX174 DNA. Nature 265, 687–695 (1977)
15. Guyader, S., Ducray, D.G.: Sequence analysis of *Potato leafroll virus* isolates reveals genetic stability, major evolutionary events and differential selection pressure between overlapping reading frame products. J. Gen. Virol. 83, 1799–1807 (2002)
16. Lamb, R.A., Horvath, C.M.: Diversity of coding strategies in influenza viruses. Trends Genet. 7, 261–266 (1991)
17. McGirr, K.M., Buehuring, G.C.: Tax and rex: overlapping genes of the Deltaretrovirus group. Virus Genes 32, 229–239 (2006)
18. Firth, A.E., Atkins, J.F.: Analysis of the coding potential of the partially overlapping 3' ORF in segment 5 of the plant fijiviruses. Virol. J. 6, 32 (2009)
19. Pedroso, I., Rivera, G., Lazo, F., Chacon, M., Ossandon, F., Veloso, F.A., Holmes, D.S.: AlterORF: a database of alternate open reading frames. Nucleic Acids Res. 36, 517–518 (2008)
20. Kim, D.S., Cho, C.Y., Huh, J.W., Kim, H.S., Cho, H.G.: EVOG: a database for evolutionary analysis of overlapping genes. Nucleic Acids Res. 37, D698–D702 (2009)
21. Okamura, K., Feuk, L., Marques-Bonet, T., Navarro, A., Scherer, S.W.: Frequent appearance of novel protein-coding sequences by frameshift translation. Genomics 88, 690–697 (2006)
22. Majoros, W.H.: Methods for Computational Gene Prediction. Cambridge University Press, Cambridge (2007)
23. Di Gesù, V.: Data Analysis and Bioinformatics. In: Ghosh, A., De, R.K., Pal, S.K. (eds.) PReMI 2007. LNCS, vol. 4815, pp. 373–388. Springer, Heidelberg (2007)
24. Jain, A.K., Murty, M.N., Flynn, P.J.: Data clustering: a review. ACM Comput. Surv. 31, 264–323 (1999)
25. National Center for Biotechnology Information (NCBI). NCBI Homepage (2009), http://www.ncbi.nlm.nih.gov/
26. Rice, P., Longden, I., Bleasby, A.: EMBOSS: The European Molecular Biology Open Software Suite. Trends Genet. 16, 276–277 (2000)

27. Linial, M.: How incorrect annotations evolve – the case of short ORFs. Trends Biotechnol. 21, 298–300 (2003)
28. National Center for Biotechnology Information (NCBI). The BLAST Databases (2009), ftp://ftp.ncbi.nih.gov/blast/db/
29. Altschul, S.F., Gish, W., Miller, W., Myers, E.W., Lipman, D.J.: Basic Local Alignment Search Tool. J. Mol. Biol. 215(2), 403–410 (1990)
30. National Center for Biotechnology Information (NCBI). Documentation of the BLASTCLUST-algorithm, ftp://ftp.ncbi.nih.gov/blast/documents/blastclust.html
31. Sotiriades, E., Dollas, A.: A general reconfigurable architecture for the BLAST algorithm. J. VLSI Signal Process. 48, 189–208 (2007)
32. Sabath, N.: Molecular Evolution of Overlapping Genes. University of Houston (2009)
33. Belshaw, R., Pybus, O.G.G., Rambaut, A.: The evolution of genome compression and genomic novelty in RNA viruses. Genome Res. 17, 1496–1504 (2007)
34. Johnson, Z.I., Chisholm, S.W.: Properties of overlapping genes are conserved across microbial genomes. Genome Inform. 14, 2268–2272 (2004)
35. Tunca, S., Barreiro, C., Coque, J.J., Martin, J.F.: Two overlapping antiparallel genes encoding the iron regulator DmdR1 and the Adm proteins control siderophore and antibiotic biosynthesis in *Streptomyces coelicolor* A3(2). FEBS J. 276, 4814–4827 (2009)
36. Hatfull, G.F., Pedulla, M.L., Jacobs-Sera, D. (and coworkers): Exploring the mycobacteriophage metaproteome: phage genomics as an educational platform. PLoS Genet. 2, e92 (2006)
37. Okeke, M.I., Adekoya, O.A., Moens, U., Tryland, M., Traavik, T., Nilssen, O.: Comparative sequence analysis of A-type inclusion (ATI) and P4c proteins of orthopoxviruses that produce typical and atypical ATI phenotypes. Virus Genes 3, 200–209 (2009)
38. Dautin, N., Bernstein, H.D.: Protein secretion in gram-negative bacteria via the autotransporter pathway. Annu. Rev. Microbiol. 61, 89–112 (2007)
39. Zhao, X., McGirr, K.M., Buehring, G.C.: Potential evolutionary influences on overlapping reading frames in the bovine leukemia virus pXBL region. Genomics 89, 502–511 (2007)
40. Palleja, A., Reverter, T., Garcia-Vallve, S., Romeu, A.: PairWise Neighbours database: overlaps and spacers among prokaryote genomes. BMC Genomics 10, 281 (2009)
41. Zhulin, I.B.: It is computation time for bacteriology. J. Bacteriol. 191, 20–22 (2009)
42. Wul, D., Hugenholtz, P., Mavromatis, K. (coworkers): A phylogeny-driven genomic encyclopaedia of bacteria and archaea. Nature 462, 1056–1060 (2009)

Investigating Populational Evolutionary Algorithms to Add Vertical Meaning in Phylogenetic Trees

Francesco Cerutti[1,2,3], Luigi Bertolotti[1,2], Tony L. Goldberg[3], and Mario Giacobini[1,2]

[1] Department of Animal Production, Epidemiology and Ecology, Faculty of Veterinary Medicine, University of Torino, Italy
[2] Molecular Biotechnology Center, University of Torino, Italy
[3] Department of Pathobiological Sciences, School of Veterinary Medicine, University of Wisconsin-Madison, USA
{francesco.cerutti,luigi.bertolotti,mario.giacobini}@unito.it
tgoldberg@vetmed.wisc.edu

Abstract. In a typical "left-to-right" phylogenetic tree, the vertical order of taxa is meaningless, and the degree of similarity between taxa is only reflected by the branch path between them. We applied an Evolutionary Algorithm (EA) to improve the graphical representation of phylogenies, adding interpretability to the vertical order of taxa. We investigated the influence of different populations in the heuristic method to evaluate their influence on a $(\lambda + \mu)$-EA. In our example, the order of taxa linked to polytomic nodes is optimized using data from genetic distance matrices. However the vertical order of taxa on a phylogenetic tree can also be used to represent non-genetic features of interest.

1 Introduction

Phylogenetic trees are the most commonly used representations of relationships among living organisms. They consist of a combination of nodes connected by branches. Extant individuals are represented by terminal nodes (branch tips), linked together through an internal node, which represents a common ancestor. The trees, usually called additive, whose branches contain information about the degree of difference between nodes, are often used to show evolutionary features. Such trees are commonly based on genetic information and models of molecular evolution. In this case, information in the tree is contained only in the root-node direction, in the pattern of linkages between branches and nodes; or, in other words, in its topology. Indeed in a typical "left-to-right" phylogram, like the one shown in Fig. 1, the vertical order of taxa is meaningless, and the degree of similarity between taxa is only reflected by the branch path between them [1].

In a fully resolved tree, all internal nodes have a degree equal to three, but, because of simultaneous divergences of sequences or, more likely, because of insufficient resolution, nodes can be a polytomous, joining more than three branches.

C. Pizzuti, M.D. Ritchie, and M. Giacobini (Eds.): EvoBIO 2010, LNCS 6023, pp. 240–247, 2010.

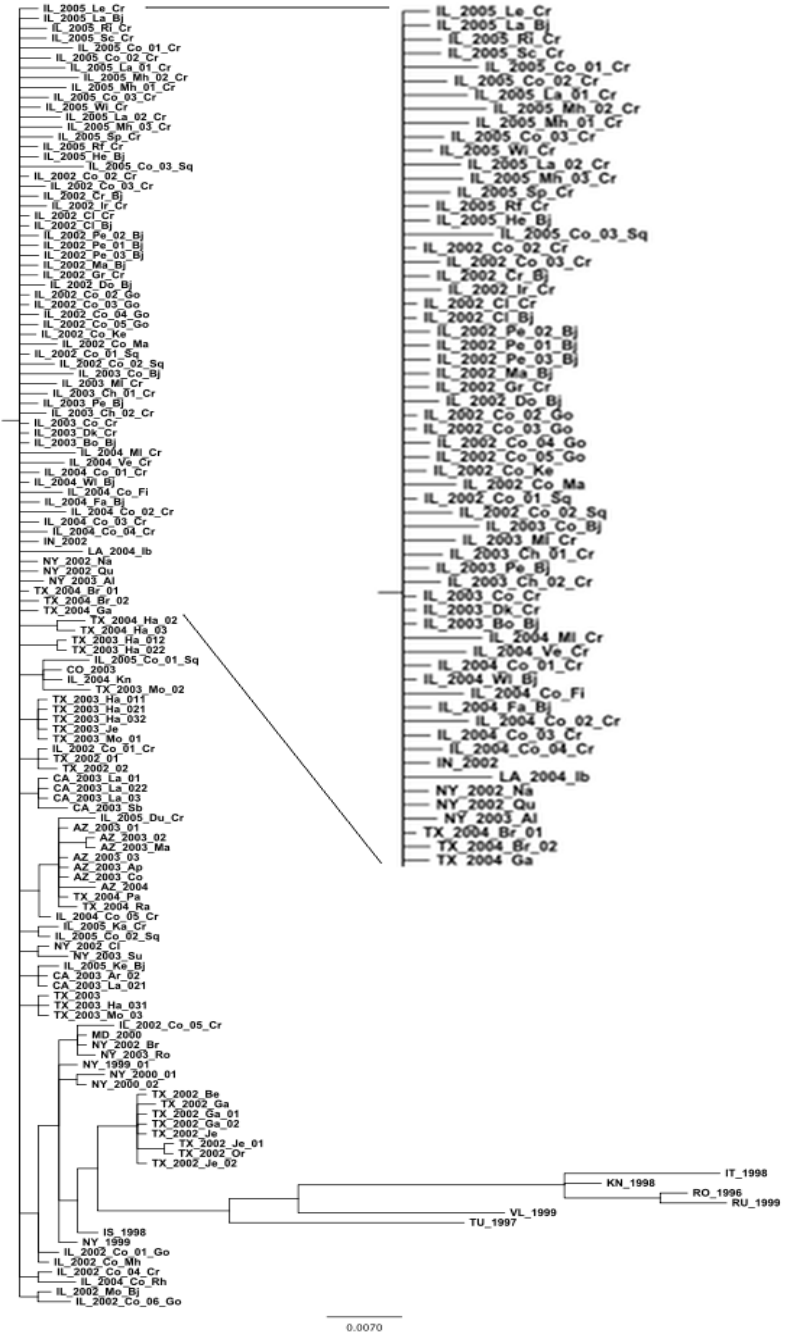


Fig. 1. West Nile virus phylogenetic topology obtained by the Bayesian approach using MrBayes software and proposed in [6]

In this latter case, trees are hard to interpret. Indeed, trees are often misinterpreted, with meaning mistakenly ascribed to the vertical proximity of taxa or clades. In cases where the vertical ordering of taxa on phylogenetic trees is flexible, the opportunity exists to ascribe biological meaning to this dimension [2]. In order to make unresolved trees more informative, we recently proposed a (1+1)-Evolutionary Algorithm (EA) to find the best graphical tree representation that includes vertical information [3]. The heuristic procedure is used to find trees that group samples with similar features in the vertical direction, and, starting from a given tree topology, it generates at each time step a new tentative solution by rotating internal nodes. Taxon order was searched in order to minimize relative distances according to their genetic distances, but this approach could be implemented using distance matrices created from other taxon features, such as temporal or geographical data, or any numerical data whose meaning is significant to build a distance matrix.

In this paper we investigate the influence of the use of populations in a $(\lambda+\mu)$-EA to solve this problem, both looking at accuracy and computational effort. In the next section, we introduce and discuss the proposed heuristic search approach, and the method is experimentally validated using a West Nile virus phylogenetic tree. Finally, in section 3 we present our conclusions and discuss possible future work.

2 Heuristic Search for West Nile Virus Phylogenetics

In a phylogenetic tree, given that each internal node can be freely rotated without changing tree's topology, it should be possible to find the best graphical representation that includes vertical information. This can be seen as an optimization problem, whose search space contains all the trees that can be obtained by node rotations from a topology previously obtained using other heuristic methods based on prior or posterior probability estimations. In this framework, the search problem would be to find the graphical representation that minimizes the distance between adjacent taxa, such distances being defined in matrices created from different sample features than were used to construct the tree, such as genetic, temporal or geographical data.

For phylogenetic trees containing large numbers of taxa, this optimization problem can not be solved by exhaustive searching, mainly because of the size of the search space. In fact, for a phylogenetic tree with 64 taxa, this dimension would range from $|S| = 2^{63}$ in the case of a completely resolved tree (thus containing 63 internal nodes all with degree 2), to $|S| = 64!$ for a completely unresolved tree. Heuristic search method is therefore needed to solve this problem. The simplest approach would be to use a hill-climbing algorithm starting from the topology obtained by the tree-building method. Most such methods output a tree in which taxon order in polytomic nodes is often alphabetical or is taken from the order of input of sequences. A simple hill-climber would, however, be computationally unfeasible in most cases, since for each tentative solution s the number of neighbors $|neigh_s|$ to be generated and evaluated would be too large.

As previously described [3], we initially applied a $(1 + 1)$-EA to explore the search space consisting of all phylogenetic trees with the same topology but with different permutations of the branches in internal polytomous nodes. The fitness of the original tree, considered as current tree at the first generation, was evaluated as sum of the distances between the considered tip to the r closest tips, based on a genetic distance matrix. Then, every generation, the current tree underwent a random swap between two branches connected to the same internal node. The fitness of the new tree was re-evaluated. Whenever the fitness of the new tree was better than the fitness of the current tree, the new tree replaces the current tree, and the search procedure continues. This process is iterated for 200000 generations, resulting in the creation and evaluation of 200000 new tentative solutions.

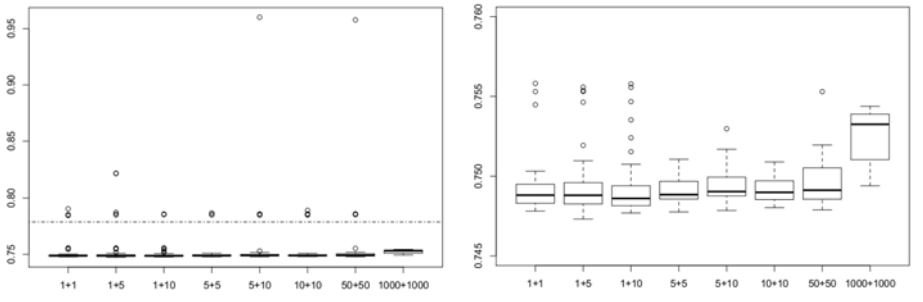
A key decision in the process described above is the choice of the fitness function. The most straightforward method would be to sum up, for all taxa in the tree, the distance between the taxon under consideration and the two taxa next to them, i.e. those taxa at radius $r = 1$. However, it is not obvious that such a choice would be optimal since, when reading a tree in the vertical dimension, one would also be inclined to consider as 'close' also taxa at distances greater than 1. To calculate the fitness of each tentative solution, we used the matrix of genetic distances among samples, corrected with the best fit molecular substitution model (GTR+Γ+I) [1,4]. In our preliminary work [3] we investigated the influence that different radii have on the search procedure. The radius $r = 8$ resulted in an acceptable balance between computational intensity and accuracy, and it was used for this study.

In order to investigate the influence of population size on the process of improving the interpretation of unresolved trees adding a vertical meaning, we explored different combinations of the λ and μ parameters in a classical $(\lambda+\mu)$-EA. First, we studied the algorithms when only the offspring population was augmented, evolving $(1 + 5)$ and $(1 + 10)$ – EAs, where $\lambda = 1$ is the starting tree to improve. Then, we evaluated the performances of the EAs when $\lambda = \mu \in \{5, 10, 50, 1000\}$. When $\lambda > 1$, in the initial generation $\lambda - 1$ randomly generated trees are added to the tree obtained by the Bayesian approach using MrBayes software. The next generation of λ parents is selected by performing μ tournaments between couples chosen by random sampling with reintroduction among the $\lambda + \mu$ individuals, selecting the best μ ones. In this way the best individual can potentially be selected more than once, without excluding the other tentative solutions. To have a fair comparison between algorithms with different population sizes, we set a maximum generation limit so that the number of new tentative solutions that each EA evaluates (i.e. its computational effort) is 200000. For $\mu = 5, 10, 50, 1000$ this results in 40000, 20000, 4000, and 200 generations, respectively. For each parameter combination, 50 independent runs each were performed.

The phylogenetic tree presented by Bertolotti et al. [6], represented in Fig. 1, has been used as a test case. West Nile virus (WNV; *Flaviviridae*; *Flavivirus*) is a single stranded, positive-sense RNA virus member of the Japanese encephalitis

serocomplex, transmitted by infected mosquitoes. Occasionally mosquitoes can transmit the virus to humans, horses and other mammals. Due to its recent introduction into North America, WNV phylogenies have tended to report highly unresolved trees [6,5,7]. In these cases, information on genetic, spatial or temporal clustering was not apparent, so that population substructure was investigated using different approaches. The original tree has a total of 132 taxa and 28 internal nodes. Up to 62 out of 76 branches from the root node are directly connected to terminal taxa. This portion of the tree, as shown magnified in Fig. 1, is therefore highly unresolved.

The tree obtained by the Bayesian approach within the MrBayes software has a fitness at $r = 8$ equal to 5.136. This value was used as reference in order to express the best final fitnesses of the EAs as relative fitness improvement (final-tree/original-tree fitnesses). The best tree was obtained in a $(1+5)$-EA run, but the fitness performances of the different parameters' combinations do not show a statistically significant difference among the groups, except for the $(1000+1000)$-EA that have p-values $<< 0.01$ in Wilcoxon Rank Sum tests (Shapiro-Wilk test $p << 0.01$ in every group) with all other groups. In the box plots shown in Fig. 2(a) all the values of the final fitness are depicted. The performance difference among $(1000 + 1000)$-EA and the other EAs is easily observable in the box plot of Fig. 2(b). In this case, the distribution of final fitness values is the only one in which all runs cross the threshold (see next paragraph).



(a) All final fitness values (b) Final fitness in the 0.745 - 0.76 range

Fig. 2. Boxplot of all the relative final fitness values (2(a)) and those in the range 0.745 - 0.76 (2(b)).The dotted line is the threshold used to evaluate the computational effort.

In order to compare the computational effort for the EAs to find "interesting" trees a threshold fitness value corresponding to a relative improvement of 0.78 was chosen (the same observation could be drawn if this threshold is set to any value in the $[0.757, 0.78]$ interval). The only algorithm able to always find solutions below this threshold is the $(1000 + 1000)$-EA, while all other λ and μ combinations succeed in between 90% and 94% of the runs. The computational effort of the $(1000 + 1000)$-EA is statistically lower than the other (p-values $<< 0.01$ in Wilcoxon Rank Sum tests), resulting in the best computational

Fig. 3. The best West Nile virus phylogenetic tree obtained by the $(1 + 5)$-EA

performances but the worse solution by one order of magnitude (Fig. 2(a) and 2(b)). Additionally the $(50 + 50)$-EA computational effort is statistically better than $(1 + 1)$, $(5 + 10)$ and $(10 + 10)$ - EAs.

From a biological point of view, the best obtained tree, shown in Fig. 3, has undergone some improvement. At first sight, the most notable change is the

long branch containing the European strains of WNV moved to the bottom of the tree from its previous middle position (Fig. 3a). Moreover, many samples collected in the same state are now close to each other (Fig. 3b). The algorithm has also moved closer those tips representing samples collected in the same year (Fig. 3b). Finally, as shown in [8], the strain of the first WNV epidemic in USA, in New York in 1999, is next to the strain associated with an epidemic in Israel in 1998, which arrived in the new world and likely sparked the North American WNV epidemic. In our final tree this relationship is highlighted, as shown in Fig. 3c.

3 Discussion and Future Works

We investigated the influence of population sizes in $(\lambda + \mu)$-EAs used to find the best graphical tree representation that includes vertical information in phylogenetic trees. To validate the proposed approach we applied it to the West Nile virus tree presented by Bertolotti and colleagues in 2007. This tree is highly unresolved, with a large number of samples belonged to highly polytomic internal nodes. To calculate the fitness value of each tentative solution we have employed the matrix of genetic distances among samples, corrected with the best fit molecular substitution model.

All different EAs result in interesting solutions, and this allow researchers to extend the readability of phylogenetic trees by enhancing the meaning of their vertical dimension. The use of populational EAs does not seem to improve the performance of the algorithms with respect to the initially proposed $(1 + 1)$-EAs [3]. However, our experiments showed a significant gain in computational effort of one order of magnitude by the largest population EA, leading to a worse fitness of the trees obtained by the $(1000 + 1000)$ one.

Generally, our results demonstrate that a heuristic search approach applied to tree graphical representation can improve the readability of phylogenetic trees, helping in their interpretation. This being a preliminary study to validate a novel analytical method, several issues need to be further investigated. This method will need to be validated and applied to other case studies, and using other types of data to generate distance matrices, such as temporal or spacial distances.

Acknowledgements. The authors thank the Supercomputing Group of the CINECA Systems & Tecnologies Department for supporting in computations. M. Giacobini acknowledges funding (60% grant) by the Ministero dell'Università e della Ricerca Scientifica e Tecnologica. L. Bertolotti gratefully acknowledges financial support by Ricerca Sanitaria Finalizzata 2008 - Regione Piemonte. This work is supported by the National Science Foundation/National Institutes of Health Ecology of Infectious Diseases Program under award number EF-0840403.

References

1. Page, R.D.M., Holmes, E.C.: Molecular evolution: a phylogenetic approach. Blackwell Science, Oxford (1998)
2. Maddison, W.: Reconstructing character evolution on polytomous cladograms reconstructing character evolution on polytomous cladograms. Cladistics 5(4), 365–377 (1989)
3. Cerutti, F., Bertolotti, L., Goldberg, T.L., Giacobini, M.: Adding Vertical Meaning to Phylogenetic Trees by Artificial Evolution. In: ECAL 2009 Proceeding. LNCS. Springer, Heidelberg (2009) (in press)
4. Nei, M.: Molecular evolutionary genetics. Columbia University Press, New York (1987)
5. Davis, C.T., Ebel, G.D., Lanciotti, R.S., Brault, A.C., Guzman, H., Siirin, M., Lambert, A., Parsons, R.E., Beasley, D.W.C., Novak, R.J., Elizondo-Quiroga, D., Green, E.N., Young, D.S., Stark, L.M., Drebot, M.A., Artsob, H., Tesh, R.B., Kramer, L.D., Barrett, A.D.T.: Phylogenetic analysis of north american west nile virus isolates, 2001-2004: evidence for the emergence of a dominant genotype. Virology 342(2), 252–265 (2005)
6. Bertolotti, L., Kitron, U., Goldberg, T.L.: Diversity and evolution of West Nile virus in Illinois and the United States, 2002-2005. Virology 360(1), 143–149 (2007)
7. Bertolotti, L., Kitron, U.D., Walker, E.D., Ruiz, M.O., Brawn, J.D., Loss, S.R., Hamer, G.L., Goldberg, T.L.: Fine-scale genetic variation and evolution of west nile virus in a transmission "hot spot" in suburban Chicago, USA. Virology 374(2), 381–389 (2008)
8. Lanciotti, R.S., Roehrig, J.T., Deubel, V., Smith, J., Parker, M., Steele, K., Crise, B., Volpe, K.E., Crabtree, M.B., Scherret, J.H., Hall, R.A., MacKenzie, J.S., Cropp, C.B., Panigrahy, B., Ostlund, E., Schmitt, B., Malkinson, M., Banet, C., Weissman, J., Komar, N., Savage, H.M., Stone, W., McNamara, T., Gubler, D.J.: Origin of the West Nile virus responsible for an outbreak of encephalitis in the northeastern United States. Science 286(5448), 2333–2337 (1999)

Author Index