Using Datamining Techniques to Help Metaheuristics: A Short Survey

Laetitia Jourdan[1], Clarisse Dhaenens[2], and El-Ghazali Talbi[1,2]

[1] INRIA Futurs, Bât M3, Cité Scientifique 59655 Villeneuve d'Ascq France
[2] LIFL, CNRS, Université de Lille I, 59655 Villeneuve d'Ascq France
{jourdan, dhaenens, talbi}@lifl.fr

Abstract. Hybridizing metaheuristic approaches becomes a common way to improve the efficiency of optimization methods. Many hybridizations deal with the combination of several optimization methods. In this paper we are interested in another type of hybridization, where datamining approaches are combined within an optimization process. Hence, we propose to study the interest of combining metaheuristics and datamining through a short survey that enumerates the different opportunities of such combinations based on literature examples.

1 Introduction

Hybrid metaheuristics are more and more studied and a first taxonomy has been proposed in [44]. Many works propose to combine two or more metaheuristics, but other works present also hybridizations between exact optimization methods and metaheuristics. Another promising approach to hybridization is to use datamining techniques to improve metaheuristics. Datamining (DM), also known as Knowledge-Discovery in Databases (KDD), is the process of automatically exploring large volumes of data e.g., instances described according to several attributes, to discover patterns. In order to achieve this, datamining uses computational techniques from statistics, machine learning, pattern recognition or combinatorial optimization.

Datamining tasks can be organized into a taxonomy, based on the desired outcome of the algorithm. Usually a distinction is made between supervised and unsupervised learning. Classical tasks of supervised learning are:

- Classification: examining the attributes of a given instance to assign it to a predefined category or class.
- Classification rule learners: discovering a set of rules in the database which forms an accurate classifier.

The most common tasks of unsupervised learning are:

- Clustering: partitioning a data set into subsets (clusters), so that data in each subset share some common aspects. Partitioning is often indicated by a proximity evaluated using a distance measure.

F. Almeida et al. (Eds.): HM 2006, LNCS 4030, pp. 57–69, 2006.

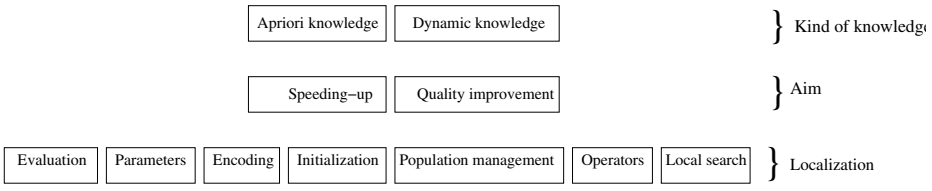

Fig. 1. The proposed taxonomy

– Association rule learners: discovering elements that occur in common within a given data set.

Using metaheuristics for knowledge extraction has become common while the other way which consists in using knowledge discovery to improve metaheuristic is less studied. This research way can be refereed as knowledge incorporation in metaheuristics and may be performed by using informed operators, approximation of fitness, etc.

To illustrate the different ways to integrate knowledge into metaheuristics, a small taxonomy that summarizes compositions found in several articles, is proposed in Figure 1.

– Two kinds of knowledge can be distinguished: a previously acquired knowledge which is called *Apriori Knowledge* and a dynamically acquired knowledge which is extracted or discovered during the search.
– Another useful information to classify algorithms is to distinguish the aim of the cooperation. Either the cooperation is used to reduce the computational time *i.e.*, speed up techniques, by simplification of the fitness *i.e.*, fitness approximation, or by significantly reducing the search space e.g., leading the metaheuristic in promising area; or the cooperation is used to improve the quality of the search by introducing knowledge in operators or in other parts of the metaheuristic. In fact, the insertion of datamining techniques often leads to both speeding up the metaheuristic and improving the quality.
– The last point used to distinguish the hybridizations is to determine which part of the metaheuristic is concerned by the knowledge incorporation. Hybridization can occur in each part of the metaheuristic: parameters, encoding, evaluation, initialization, operators, etc.

This paper aims to provide a quick comprehensive picture of the interest of combining datamining techniques and metaheuristics. We do not consider here the vast topic of incorporating knowledge but the use of knowledge algorithms also called datamining algorithms. In order to present this literature review, we have chosen to classify references with respect to the localization of the knowledge integration.

The remainder of this paper is set out as follows. Section 2 highlights the potential of datamining to speed-up metaheuristics by using datamining during

the evaluation. Section 3 discusses how datamining can help to set the parameters of the metaheuristic. Section 4 presents the use of datamining techniques for the initialization of metaheuristics. Section 5 is devoted to population management. Section 6 details the benefit of datamining in crossover operators. Section 7 shows the local search datamining applications in evolutionary computation. Section 8 exhibits that some metaheuristics are based on datamining incorporation. Finally, conclusions and perspectives are drawn in the last section.

2 Using Datamining During the Evaluation

In some real cases, the fitness function can be very expensive to compute. Thus decreasing the number of complete evaluations would be beneficial. To achieve this, some approximations of the fitness functions could be used, and datamining techniques may be interesting to obtain good approximations. A very complete and comprehensive survey on fitness approximation has been proposed in [15]. It shows that fitness approximation can be used either for expensive fitness functions or multi-modal fitness functions and may be realized by several approaches exposed below.

2.1 Replace the Evaluation Function by a Datamining Algorithm

Datamining techniques can be used to build approximate models of the fitness function. In this context, previously calculated fitnesses are learned by a datamining algorithm to approximate the fitness of new individuals. Many works use neural networks (both multi-layer perceptrons and radial-basis-function networks) to realize an approximation of the function to optimize. For example in [6], the authors use an artificial neural network (ANN) with a multiple objective genetic algorithm. They evaluate a large part of the population with an ANN and a small part is still simultaneously evaluated with the original function. Rasheed et al. propose to cluster data and to construct separate approximation models for the different clusters [30,32,29]. The approximation model can be used each time or alternatively with the real objective function.

2.2 Using Datamining Techniques to Avoid Evaluations

When the fitness function can be approximated, some authors use the approximation only within operators, such as initialization, mutation, crossover and selection. This approach avoids the calculation of time consuming fitness for individuals that may be of very bad quality and that will not be kept in the population. For example, in [31], the authors use approximations to improve operators. They generate several possible new individuals and then choose the best according to a reduced model. To compute the model, they maintain a large sample of points encountered during the course of the optimization and divide it into dynamic clusters. To compute the approximate model of an individual, they use the weighted k-nearest-neighbor approach which is a classification technique.

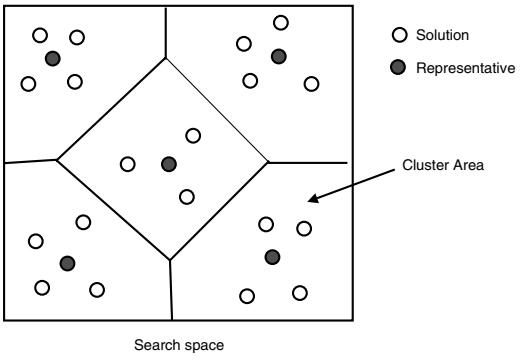

Fig. 2. Using representatives for the evaluation process through clustering

2.3 Using a Datamining Algorithm to Estimate a Representative

Approximating the fitness could be not satisfying because of the quality of the approximation for example. Another way to speed up the metaheuristic is to reduce the number of calls to the fitness function. This may be realized using fitness imitation. In this kind of approach, a set of individuals is considered as similar to another one and their fitness is fixed as equal to the reference individual which is called the representative. To determine the different sets, clustering techniques are often used.

Hence, in [16,18,47], the author proposes to maintain a large population size by using clustering algorithms. For example, in [47] Yoo et al. propose to use a fuzzy clustering approach to divide the population and to elect a representative of each cluster. Only the representatives are evaluated which allows the reduction of the evaluation costs. The fitness value of an individual of a cluster is estimated in respect with its associated representative (Figure 2).

3 How Datamining May Help to Set Parameters

A very difficult part in designing metaheuristics deals with the setting of the parameters of such methods. How can we fix in advance parameters such as the probability of application of a given operator, the size of the population or the number of iterations, for example? Two approaches may be used in this context:

- A first approach which is empirical consists in both executing several times the method with different parameter values and trying to select the best values. If the number of executions or the number of parameters are high, determining the best set of parameters may require statistical analyses. This may be seen as a datamining help.
- To set the probability of application of an operator, another approach may be used. It consists in analyzing the performance of the operators during the algorithm execution. In particular, this approach may be used when several

operators are available for the same operation (crossover or mutation, for example). In [12], the author proposes to compute the rate of appliance of a mutation operator by calculating the progress of the last applications of this operator. Hence, it becomes possible to determine the probabilities of appliance of a given operator in an adaptively way where the more efficient an operator is, the more it will be used. Another approach could be to analyse in details the new individuals generated by operators (in term of quality, diversity) using clustering algorithms for example. This would give valuable information that can help to set the new application probabilities.

These two approaches give examples on the way the datamining techniques may help to set parameters.

4 Using Datamining for Initialization

Generally, metaheuristics generate their initial solution(s) randomly. In continuous optimization, this generation may also be done using a grid initialization. It could be also interesting to cleverly generate the initial population in order, for example, to reduce the search space by leading the metaheuristic to promising area.

For example, in [28], Ramsey et al. propose to initialize a genetic algorithm with case-based reasoning in a tracker/target simulation with a periodically changing environment. Case-based initialization allows the system to automatically bias the search of the genetic algorithm toward relevant areas of the search space.

5 Datamining and Population Management

Datamining techniques are often used to manage the population. Several works deal with introducing new individuals in the population. Some common methods try to inject new individuals into the population. This could be realized to diversify the population like in the random immigrant strategy. In order to lead the search to promising search spaces it could be also interesting to regularly introduce individuals that are built based on information of the past encountered solutions.

In [20,21], Louis presents CIGAR (Case Injected Genetic AlgoRithm). The aim of CIGAR is to provide periodically to the genetic algorithm solutions that suit to similar problems. CIGAR has been successfully applied to several problems such as jobshop, circuit modelling, etc.

In [4,39], the authors propose to hybridize a genetic algorithm and the Apriori algorithm (Apriori is a classical algorithm to generate association rules [1]) to discover interesting subroutines for the oil collecting vehicle routing problem. They insert the found subroutines into the new individuals.

In the work of Ribeiro et al. [37,38,40], the authors present a GRASP hybridized with several frequent item set mining algorithms: the Direct Count and

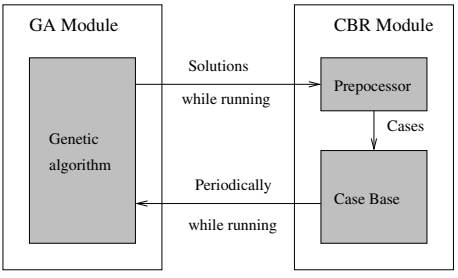

Fig. 3. Case Injected Genetic AlgoRithm (CIGAR)

the Intersect algorithms in [38] and the FPMax* in [40], which are Apriori-like approaches. These algorithms are used to extract patterns that are promising only on elite solutions. The hybridization is realized after a fixed number of seconds or iterations and new starting solutions for the GRASP are computed thanks to the found patterns. The authors apply their approach to the Set Packing Problem and the Maximum Diversity Problem. The method allows for the speed up of the convergence of the algorithm and for the improvement of the robustness of the GRASP.

Another example of the use of datamining techniques in the population management is the use of clustering algorithms in Multi-objective population metaheuristics where the result to produce is a set of solutions of best compromise (Pareto solutions). An archive is often used to store these solutions and the clustering is used to avoid a bias towards a certain region of the search space. Such a bias would lead to an unbalanced distribution of the Pareto solutions. Authors often use the average linkage method as this clustering algorithm performs well for Pareto optimization [48].

6 Using Datamining Within Operators

Incorporating knowledge in operators could be useful if, for example, it allows to cleverly exploit the search space. In the following section, some examples using machine learning approaches in crossover to explore the search space are presented.

Handa et al. propose a co-evolutionary genetic algorithm, which uses an hybridization between a GA and C4.5 (a classification algorithm) in order to discover the schemata to use in the crossover [10,9]. In their early work [8], Handa et al. have proposed a co-evolutionary algorithm in order to discover the good schemata to use that have not been discovered by the GA. The method works well but was just presented for bit representation.

LEM [22,23] integrates a symbolic learning component to evolutionary computation; it seeks out rules explaining the differences between the better and worse performers in the population, and generates new individuals based on the templates specified in these rules. An example of behaviour of LEM in the search

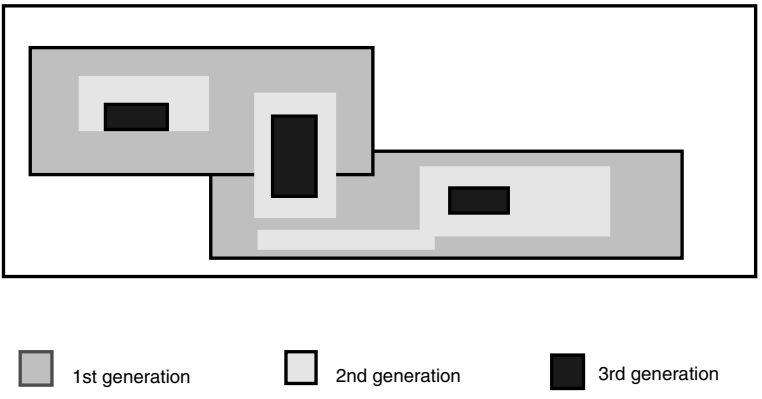

Fig. 4. LEM: Example of search region reductions defined by description of the 1st, 2nd and 3rd generations

space is shown on Figure 4 where the search regions associated to each generation are illustrated. This figure shows how the search space is reduced. The LEM methodology has proved able to improve the efficiency of the evolutionary process. LEM uses AQ learning algorithm (a general decision rules learning algorithm) in order to produce rules. Let us remark that, existing implementations, such as AQ11 [24], AQ15 [25] handle noise with pre and post-processing techniques. The basic AQ algorithm however, heavily depends on specific training examples during the search (the algorithm actually employs a beam search). This approach has been used with the C4.5 algorithm for mono-objective jobshop problems in the work of Huyet [14,13].

In the work of Jourdan et al. [17,45], the authors propose to extend LEM to LEMMO for the multi-objective case in order to seek for rules that explain why some individuals dominate others in a multi-objective point of view and why some individuals are dominated by other. They generate new individuals thanks to the rules by creating solutions that match to positive rules and do not match to negative rules. This approach has shown good results on a water system application both in speeding up the multi-objective algorithm and in improving the quality of the solutions.

We can remark that usually the authors use classification methods (C4.5, AQ, etc) to identify the genes that induce the good quality of the individuals. Some authors propose to also determine the genes that characterize bad quality solutions and to use them to repair the constructed solutions [17,45].

7 Datamining in Local Search

Metaheuristics are often hybridized with local search methods to improve the intensification part. Some datamining algorithms are themselves local searches and could be used as part of the metaheuristics.

In [6], the authors propose to use inverse Artificial Neural Networks (ANN) as local search to exploit specific region for candidate solutions. They note that ANNs can be adequate as they construct a smooth mapping. The ANN is trained in a reverse way as the input layer presents the criteria and the output the parameters to be optimised.

Moreover, datamining problems can be often modelized as optimization problems and in this case, the hybridization of the metaheuristic with a machine learning algorithm could be realized such that the machine learning algorithm treats a subproblem. For example, when clustering or grouping problems are solved using a metaheuristic, the metaheuristic searches for the optimal subset of genes that act as initial cluster centers. At the lower level, a local learning method performs local clustering from these initial centers. The objective is to combine the strength of EAs and clustering methods to produce a global efficient clustering algorithm. Kmeans is often used as a local search [5,11,46] for initialization of the solutions or to realize a local search during the search. Another approach, presented in [7], uses fuzzy c-means and hard c-means as an objective function. This article shows the importance of the initialization of the solution(s). In [3], the authors also use Expectation Maximization (EM) as a local search to analyze gene trajectory.

8 Datamining Based Metaheuristics

Some metaheuristics are designed to directly care of dynamic knowledge. We decide to create a specific part for them as they are now considered as new metaheuristic and not as improvement of previous ones. A lot of these algorithms are classified as Non-Darwinian evolutionary computation as they replace Darwinian operators by other operators. As identified in the taxonomy (Figure 1), the cooperation can appear in different localizations but we observe that in the proposed metaheuristics, the integration is often localised in the operator part. For example, the *Population-based Incremental Learning* (PBIL) creates a real-valued probability vector characterizing high fitness solutions [2] (Figure 5). This vector is then used to build new solutions. Generally, PBIL does not use mutation and crossover. PBIL can be considered as both encoding and initialisation localization hybridizations.

Specifically, Muhlenbein and Paass have estimated the probability density functions of binary variables in their chromosomes by the product of the individual probability density functions in the UMDA (Univariate Marginal Distribution Algorithm) [26]. Hence, UMDA is a special class of the PBIL algorithm.

Pelikan and Goldberg developed an algorithm "BOA" (Bayesian Optimization Algorithm) that extends the above ideas by using Bayesian Networks to model the chromosomes of superior fitness [27] (Figure 6). BOA can be classified as localization operator algorithm with dynamic knowledge.

A similar approach has also been proposed by Larranaga and Lozano, who have given the term "EDA" (Estimation of Distribution Algorithms) to the statistical estimation approach to EC [19].

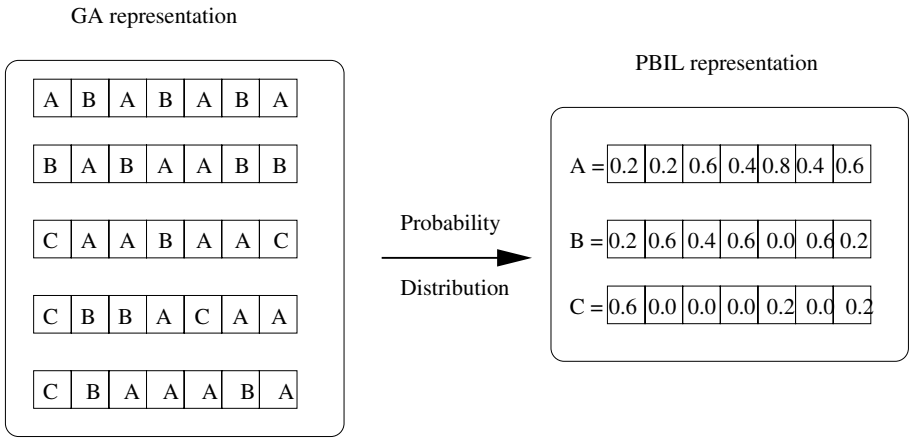

Fig. 5. The PBIL probability vector

```
begin
    t=0;
    Initialise randomly Population POP(0);
    Evaluate(POP(0));
    repeat
        Select a set of promising strings S(t) from POP(t);
        Construct the network B using a given metric and constraints;
        Generate a set of new strings O(t) according to the joint distribution
         encoded by B;
        Create a new population POP(t+1) by replacing some strings from P(t)
         with O(t);
        Evaluate(POP(t));
        t=t+1;
    until (termination condition)
end
```

Fig. 6. Overview of the Bayesian Optimization Algorithm

Similarly, cultural algorithms use high performing individuals to develop beliefs constraining the way in which individuals are modified by genetic operators [35,36] (Figure 7). In cultural algorithm, beliefs are formed based on each entity's individual experiences. The reasoning behind this, as outlined by [35], is that cultural evolution allows populations to learn and adapt at a rate faster than pure biological evolution alone. Importantly, the learning which takes place individually by each entity is passed on to the remainder of the group, allowing learning to take place at a much faster rate. Cultural algorithm can be classified as operator localization algorithm in the taxonomy.

Ravise and Sebag [43,34,41,33,42] worked on civilized genetic algorithms that differ from Darwinian's ones as they keep information of the population in order to avoid doing the same errors. The knowledge is dynamically updated during

```
begin
    t=0;
    Initialise Population POP(0);
    Initialise Belief Network BLF(0);
    Initialise Communication Channel CHL(0);
    Evaluate(POP(0));
    t=1;
    repeat
        Communicate(POP(0), BLF(t));
        Adjust(BLF(t));
        Communicate(BLF(t), POP(t));
        Modulate Fitness (BLF(t), POP(t));
        t=t+1;
        Select POP(t) from POP(t-1);
        Evolve(POP(t));
        Evaluate(POP(t));
    until (termination condition)
end
```

Fig. 7. Overview of the cultural evolution algorithm (Reynolds 1994)

generations. The datamining hybridization accelerates and improves the convergence of the algorithm. But it has been tested only on bit representation. The authors have observed that the GA must be run first with Darwinian operator in order to have diversity in its population. After a fixed number of generations, the civilized operator is used. They keep history of the past results in order to not reproduce the same error (and produce bad individuals). Civilized genetic algorithms can be classified as operator based dynamic knowledge.

9 Discussion and Conclusion

We have seen that there are multiple reasons to integrate datamining methods within metaheuristics. It could be to approximate the fitness function, to improve the convergence of the metaheuristics or to create an operator that is adapted to the problem.

In a research point of view, the actual major interest is to use datamining to extract useful information from the history of the metaheuristic in order to move the search in interesting space areas. Moreover, the hybridization between metaheuristics and datamining techniques have not been studied a lot in multi-objective optimization.

The major drawback of hybridization is the setting of parameters. When applying the datamining method, how many solutions have to be stored in dynamic knowledge, etc ? Many articles realize experimentally the parameter settings and many authors remark that clearly the performances are correlated with the parameters. A very promising search investigation is to automatically determine during the search all these parameters for designing adaptive efficient metaheuristics.

References

1. Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules. In Jorge B. Bocca, Matthias Jarke, and Carlo Zaniolo, editors, *Proceeding 20th International Conference Very Large Data Bases, VLDB*, pages 487–499. Morgan Kaufmann, 12–15 1994.
2. S. Baluja. Population based incremental learning. Technical Report CMU-CS-94-163, Carnegie Mellon University, 1994. http://www.cs.cmu.edu/afs/cs/user/baluja/www/techreps.html.
3. Z.S.H. Chan and N. Kasabov. Gene trajectory clustering with a hybrid genetic algorithm and expectation maximization method. In *IEEE International Joint Conference on Neural Networks*, pages 1669–1674, 2004.
4. F.L. Dalboni, L.S. Ochi, and L.M.A. Drummond. On improving evolutionary algorithms by using data mining for the oil collector vehicle routing problem. International Network Optimization Conference, 2003.
5. Emanuel Falkenauer. A new representation and operators for genetic algorithms applied to grouping problems. *Evolutionary Computation*, 2(2):123–144, 1994.
6. A. Gaspar-Cunha and A.S. Vieira. A hybrid multi-objective evolutionary algorithm using an inverse neural network. In *Hybrid Metaheuristic*, pages 25–30, 2004.
7. L. O. Hall, I. B. Özyurt, and J. C. Bezdek. Clustering with a genetically optimized approach. *IEEE Trans. on Evolutionary Computation*, 3(2):103–112, 1999.
8. H. Handa, N. Baba, O. Katai, and T. Sawaragi. Coevolutionary genetic algorithm with effective exploration and exploitation of useful schemata. In *Proceedings of the International Conference on Neural Information Systems*, volume 1, pages 424–427, 1997.
9. H. Handa, T. Horiuchi, O. Katai, and M. Baba. A novel hybrid framework of coevolutionary GA and machine learning. *International Journal of Computational Intelligence and Applications*, 2002.
10. H. Handa, T. Horiuchi, O. Katai, T. Kaneko, T. Konishi, and M. Baba. Fusion of coevolutionary ga and machine learning techniques through effective schema extraction. In Lee Spector, Erik D. Goodman, Annie Wu, W.B. Langdon, Hans-Michael Voigt, Mitsuo Gen, Sandip Sen, Marco Dorigo, Shahram Pezeshk, Max H. Garzon, and Edmund Burke, editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, page 764, San Francisco, California, USA, 7-11 July 2001. Morgan Kaufmann.
11. J. Handl and J. Knowles. Improvements to the scalability of multiobjective clustering. In IEEE, editor, *IEEE Congress on Evolutionary Computation*, pages 438–445, 2005.
12. T.P. Hong, H. Wang, and W. Chen. Simultaneously applying multiple mutation operators in genetic algorithms. *Journal of heuristics*, 6:439–455, 2000.
13. A.-L. Huyet. *Extraction de connaissances pertinentes sur le comportement des systemes de production : une approche conjointe par optimisation evolutionniste via simulation et apprentissage.* PhD thesis, Université Blaise Pascal Clermont II, October 2004.
14. A.-L Huyet and J.-L. Paris. Configuration and analysis of a multiproduct kanban system using evolutionary optimisation coupled to machine learning. In *Proceedings of CESA 2003, the IMACS Multiconference Computational Engineering in Systems Applications*, July 2003. ISBN 2-9512309-5-8, CDROM.
15. Y. Jin. A comprehensive survey of fitness approximation in evolutionary computation. *Soft Computing Journal*, 9(1):3–12, 2005.

16. Y. Jin and B. Sendhoff. Reducing fitness evaluations using clustering techniques and neural networks ensembles. In *Genetic and Evolutionary Computation Conference*, volume 3102 of *LNCS*, pages 688–699. Springer, 2004.
17. L. Jourdan, D. Corne, D.A. Savic, and G A. Walters. Preliminary investigation of the learnable evolution model for faster/better multiobjective water systems design. In LNCS 3410, editor, *Third International Conference on Evolutionary Multi-Criterion Optimization (EMO'05)*, pages 841–855, 2005.
18. H.-S. Kim and S.-B. Cho. An efficient genetic algorithms with less fitness evaluation by clustering. In *Proceedings of IEEE Congress on Evolutionary Computation*, pages 887–894. IEEE, 2001.
19. P. Larranaga and J.A. Lozano. *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*. Kluwer Academic Publishers, 2002.
20. S. J. Louis. Genetic learning from experience. In IEEE, editor, *Congress on Evolutionary Computation (CEC'03)*, pages 2118 – 2125, Australia, Dec 2003. IEEE.
21. S. J. Louis. Learning for evolutionary design. In *Proceedings of the 2003 Nasa/DoD Conference on Evolutionary Hardware*, pages 17–23, July 2003.
22. R.S. Michalski. Learnable evolution model: Evolutionary processes guided by machine learning. *Machine Learning*, 38(1–2):9–40, 2000.
23. R.S. Michalski, G. Cervon, and K.A. Kaufman. Speeding up evolution through learning: Lem. In *Intelligent Information Systems 2000*, pages 243–256, 2000.
24. R.S. Michalski and J.B. Larson. Selection of most representative training examples and incremental generation of vl1 hypothesis: The underlying methodology and the descriptions of programs esel and aq11. Technical Report Report No. 867, Urbana, Illinois: Department of Computer Science, University of Illinois, 1978.
25. R.S. Michalski, I. Mozetic, J. Hong, and N. N. Lavrac. The multipurpose incremental learning system aq15 and its testing application to three medical domains. In *Proc. of the Fifth National Conference on Artificial Intelligence*, pages 1041–1045. PA: Morgan Kaufmann, 1986.
26. H. Muhlenbein and G. Paass. From recombination of genes to the estimation of distributions: I. binary parameters. *Lecture Notes in Computer Science*, 1141:178–187, 1996.
27. M. Pelikan, D. E. Goldberg, and E. Cantú-Paz. BOA: The Bayesian optimization algorithm. In Wolfgang Banzhaf, Jason Daida, Agoston E. Eiben, Max H. Garzon, Vasant Honavar, Mark Jakiela, and Robert E. Smith, editors, *Proceedings of the Genetic and Evolutionary Computation Conference GECCO-99*, volume I, pages 525–532, Orlando, FL, 13-17 1999. Morgan Kaufmann Publishers, San Fransisco, CA.
28. C. Ramsey and J. Grefenstette. Case-based initialization of genetic algorithms. In *Fifth International Conference on Genetic Algorithms*, pages 84–91, 1993.
29. K. Rasheed. An incremental-approximate-clustering approach for developing dynamic reduced models for design optimization. In *Proceedings of the 2000 Congress on Evolutionary Computation (CEC'00)*, pages 986–993, California, USA, 6-9 2000. IEEE Press.
30. K. Rasheed and H. Hirsh. Using case based learning to improve genetic algorithm based design optimization. In Thomas Bäck, editor, *Proceedings of the Seventh International Conference on Genetic Algorithms (ICGA97)*, San Francisco, CA, 1997. Morgan Kaufmann.
31. K. Rasheed and H. Hirsh. Informed operators: Speeding up genetic-algorithm-based design optimization using reduced models. In L. Darrell Whitley, David E. Goldberg, Erick Cantú-Paz, Lee Spector, Ian C. Parmee, and Hans-Georg Beyer, editors, *GECCO*, pages 628–635. Morgan Kaufmann, 2000.

32. K. Rasheed, S. Vattam, and X. Ni. Comparison of methods for developing dynamic reduced models for design optimization. In *Proceedings of the Congress on Evolutionary Computation (CEC'2002)*, pages 390–395, 2002.
33. C. Ravise and M. Sebag. An advanced evolution should not repeat its past errors. In *International Conference on Machine Learning*, pages 400–408, 1996.
34. C. Ravise, M. Sebag, and M. Schoenauer. A genetic algorithm led by induction. url: citeseer.ist.psu.edu/126354.html.
35. R. G. Reynolds, Z. Michalewicz, and Michael J. Cavaretta. Using cultural algorithms for constraint handling in genocop. In *Evolutionary Programming*, pages 289–305, 1995.
36. R. G. Reynolds and B. Peng. Cultural algorithms: computational modeling of how cultures learn to solve problems: an engineering example. *Cybernetics and Systems*, 36(8):753–771, 2005.
37. M. Ribeiro, A. Plastino, and S. Martins. Hybridization of grasp metaheuristic with data mining techniques. *Special Issue on Hybrid Metaheuristic of the Journal of Mathematical Modelling and Algorithms*, 5(1):23–41, April 2006.
38. M. Ribeiro, V. Trindade, A. Lastino, and S. Martins. Hybridization of GRASP metaheuristic with data mining techniques. In *Workshop on Hybrid Metaheuristics 16th European Conference on Artificial Intelligence (ECAI)*, pages 69–78, 2004.
39. H.G. Santos, L.S. Ochi, E.H. Marinho, and L.M.A. Drummond. Combining an evolutionary algorithm with data mining to solve a vehicle routing problem. *NEUROCOMPUTING*, 2006. (to appear).
40. L. Santos, M. Ribeiro, A. Plastino, and S. Martins. A hybrid GRASP with data mining for the maximum diversity problem. In LNCS 3636, editor, *Hybrid Metaheuristic*, pages 116–128, 2005.
41. M. Sebag, C. Ravise, and M. Schoenauer. Controlling evolution by means of machine learning. In *Evolutionary Programming*, pages 57–66, 1996.
42. M. Sebag and M. Schoenauer. Controlling crossover through inductive learning. In Yuval Davidor, Hans-Paul Schwefel, and Reinhard Männer, editors, *Parallel Problem Solving from Nature – PPSN III*, pages 209–218, Berlin, 1994. Springer.
43. M. Sebag, M. Schoenauer, and C. Ravise. Toward civilized evolution: Developing inhibitions. In Thomas Bäck, editor, *Proceeding of the Seventh Int. Conf. on Genetic Algorithms*, pages 291–298, San Francisco, CA, 1997. Morgan Kaufmann.
44. E-G. Talbi. A taxonomy of hybrid metaheuristics. *Journal of heuristics*, 8(2):541–564, Sept. 2002.
45. L. Vermeulen-Jourdan, D. Corne, D.A. Savic, and G.A. Walters. Hybridising rule induction and multi-objective evolutionary search for optimising water distribution systems. In *Proceeding of Fourth International Conference on Hybrid Intelligent Systems (HIS'04)*, pages 434–439, 2004.
46. L. Vermeulen-Jourdan, C. Dhaenens, and E-G. Talbi. Clustering nominal and numerical data: A new distance concept for a hybrid genetic algorithm. In Jens Gottlieb and Günther R. Raidl, editors, *Evolutionary Computation in Combinatorial Optimization – EvoCOP 2004*, volume 3004 of *LNCS*, pages 220–229, Coimbra, Portugal, 5-7 April 2004. Springer Verlag.
47. S-H. Yoo and S-B. Cho. Partially evaluated genetic algorithm based on fuzzy c-means algorithm. In LNCS 3242, editor, *Parallel Problem Solving From Nature (PPSN)*, pages 440–449, 2004.
48. E. Zitzler and L. Thiele. Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271, 1999.