Biol. Cybern. 57, 95–101 (1987)

Biological Cybernetics
© Springer-Verlag 1987

Optimization by Simulating Molecular Evolution

Qizhong Wang
Max-Planck-Institut für Biophysikalische Chemie, D-3400 Göttingen, Federal Republic of Germany

Abstract. Based on the analogy between mathematical optimization and molecular evolution and on Eigen's quasi-species model of molecular evolution, an evolutionary algorithm for combinatorial optimization has been developed. This algorithm consists of a versatile variation scheme and an innovative decision rule, the essence of which lies in a radical revision of the conventional philosophy of optimization: A number of configurations of variables with better values, instead of only a single best configuration, are selected as starting points for the next iteration. As a result the search proceeds in parallel along a number of routes and is unlikely to get trapped in local optima. An important innovation of the algorithm is introduction of a constraint to let the starting points always keep a certain distance from each other so that the search is able to cover a larger region of space effectively. The main advantage of the algorithm is that it has more chances to find the global optimum and as many local optima as possible in a single run. This has been demonstrated in preliminary computational experiments.

1 Introduction

Molecular evolution is in some respects an automatic optimum-searching process. It has been shown experimentally that a biomacromolecular system capable of self-replication, such as the single-stranded RNA of the bacteriophage Q_β, will, through mutation and selection, eventually acquire better properties (Spiegelman 1970; Mills et al. 1967). Under special conditions (e.g. de novo synthesis), this kind of evolutionary process can be much more efficient, and the molecular properties can be optimized at a much higher level (Sumper and Luce 1975; Biebricher et al. 1981). One of the explanations for this efficiency is based on Eigen's quasi-species model: During evolution, in addition to the fittest species, the unfavourable species also survive and play an important role – to make escape from "evolutionary traps" possible (Eigen 1971, 1985, 1986; Eigen and Schuster 1977, 1978a, b). Inspired by these experimental and theoretical studies in molecular biology and on the basis of the analogy between molecular evolution and mathematical optimization, we have developed an evolutionary algorithm for combinatorial optimization. This algorithm consists of a versatile variation scheme and an innovative decision rule. The variation scheme combines advantages of both random rearrangement and exhaustive enumeration, and thus is able to cope with a variety of objective functions. In the decision rule, the philosophy of optimization is radically revised: The less advantageous configurations are favoured and used as starting points for each iteration. Accordingly the search proceeds in parallel along a number of routes and is unlikely to get trapped in local optima. Furthermore, we introduce a constraint to let the starting points of iteration always keep a certain distance from each other so that a much larger region of space can be effectively searched. This provides more opportunities to find the global optimum and as many local optima as possible in a single run.

We have tested the performance of this evolutionary algorithm with five examples. These included a well-known difficult problem in communication engineering, namely to find binary sequences with minimal autocorrelations (Golay 1982). The preliminary results of computational experiments have convincingly demonstrated the power and efficiency of the algorithm. It compares favourably with other optimization methods, including the simulated annealing method (Kirkpatrick et al. 1983).

2 Combinatorial Optimization

The study of optimization is concerned with developing efficient methods to find the maximal or minimal values of a function with N independent variables. In combinatorial optimization these variables can only take finite, say k discrete values. At first sight there would seem to be no difficulty in finding the exact solution for problems of combinatorial optimization. Because both the number of variables (N) and the number of values (k) are finite, the total number T of all possible combinations of variable values is also finite. For the Selecting Type problem (e.g. the Knapsack Problem) $T=k^N$ and for the Sequencing Type problem (e.g. the Assignment Problem) $T=N!$. One can in principle obtain the exact solution by exhaustive enumeration, but in practice enumeration is really impossible for the problems with a large N, since T increases explosively with N. For example, for a moderate $N=100$ and $k=2$, $2^{100} \sim 10^{30}$, and $100! \sim 10^{157}$. Such a huge amount of enumeration can not be performed in reasonable time even on supercomputers.

Consequently, for large scale problems one has to resort to approximate methods. Many conventional approximate methods of optimization suffer a common drawback – getting trapped in local optima and failing to find the global one. For many so-called "*NP*-hard Problems" it is sometimes still difficult to obtain a good approximate solution. Because many combinatorial optimization problems are of great practical interest, many efforts have been devoted to the study of heuristic methods that aim at finding good approximate solutions, not necessarily the optimal ones, within reasonable computational times. One of efficient heuristic methods is the simulated annealing method (Kirkpatrick et al. 1983). It makes use of the analogy between finding the optimal values of a multivariate function and searching for ground states of a many-body system, taking advantage of the characteristics of an annealing procedure, i.e., by lowering the temperature slowly to bring the system to its true ground state (the global minimum) rather than freezing it into a metastable state (a local minimum).

This problem, avoiding getting stuck in a local optimum and leading towards the global optimum, however, exists not only in physical processes such as annealing. It can also be found in biological processes, including molecular evolution (Eigen 1985, 1986; Schuster and Sigmund 1985; Schuster 1986).

3 Molecular Evolution

Molecular evolution is concerned with exploring evolutionary laws at the molecular level, investigating the causal connection between molecular self-reproduction, selection and evolution. It is an important aspect of modern molecular biology. Experimental studies of the single-stranded RNA of bacteriophage Q_β showed that this macromolecule can reproduce in vitro (Spiegelman 1970; Mills et al. 1967; Sumper and Luce 1975; Biebricher et al. 1981). With the aid of a replication enzyme, Q_β replicase, a viral plus strand as template can reproduce a complementary sequence, the minus strand, which then acts as a template itself, again with the aid of replicase, to reproduce a plus strand. The replication is not 100% exact; mutations may occur. The process was found to obey the Darwinian principle of natural selection. After many generations some mutants eventually possessed better properties (e.g. resistance against inhibitors) than the wildtype (Kramer et al. 1974). The improvement, however, was quite limited, since the process became trapped in local optima. Much higher degrees of optimization of molecular properties (in particular, replication rate) was achieved by a de novo synthesis strategy (Sumper and Luce 1975), where the evolutionary process started from a large number of random sequences (Biebricher et al. 1981).

Eigen analyzed these experiments and formulated a mathematical theory of molecular evolution based on the "quasi-species" model (Eigen 1971; Eigen and Schuster 1977, 1978a, b). In his theory emphasis is shifted from a single surviving wildtype to a distribution of mutants that coexist and constitute so-called quasi-species. Selection and evolution can be derived from an extremum principle as inevitable consequences of self-replication.

If we compare this biological optimization process with a mathematical optimization process, we indeed find many similarities (Table 1). If we examine the strategies which both processes employ, we see that they have still more in common. The conventional strategy of mathematical optimization, Iterative Improvement, is just like evolution; its two elements, a variation scheme and a decision rule, are equivalent to a mutation mechanism and a selection principle. All of these suggested pursuing the analogy between molecular evolution and mathematical optimization, ta-

Table 1. Similarities between biological evolution and mathematical optimization

Mathematical optimization	Molecular evolution
Objective function $F(s_1, s_2, \ldots, s_N)$	Selective function $V(t_1, t_2, \ldots, t_N)$
Variables s_k $0, 1; \pm 1$	Nucleotides t_k A, C, G, U
Configuration $(s_1, s_2, \ldots, s_N)$	Sequence $(t_1, t_2, \ldots, t_N)$
Iterative improvement	Evolution
Variation scheme	Mutation mechanism
Decision rule	Selection principle

king full advantage of the characteristics of molecular evolution, and simulating its mutation mechanism and selection principle. We expect that this will enable us to obtain a better algorithm to do mathematical optimization.

4 Evolutionary Algorithm

A simulated evolutionary algorithm for optimization may be summarized as follows (Fig. 1):

1. Starting from an initial configuration of variables (template), and according to a specific variation scheme (mutation mechanism), generate a predetermined number (N_m) of configurations (mutants).

2. Evaluate the objective function for each configuration and sort the values in order.

3. Select a specified number (N_s) of configurations with the best values to form a "Survivor Set" according to the decision rule (natural selection).

4. Check the Survivor Set. IF it does not change after a specified number (N_g) of consecutive iterations THEN stop the process; ELSE use the Survivor Set as templates GOTO 1.

These main steps of the present algorithm are about the same as the other evolutionary algorithms

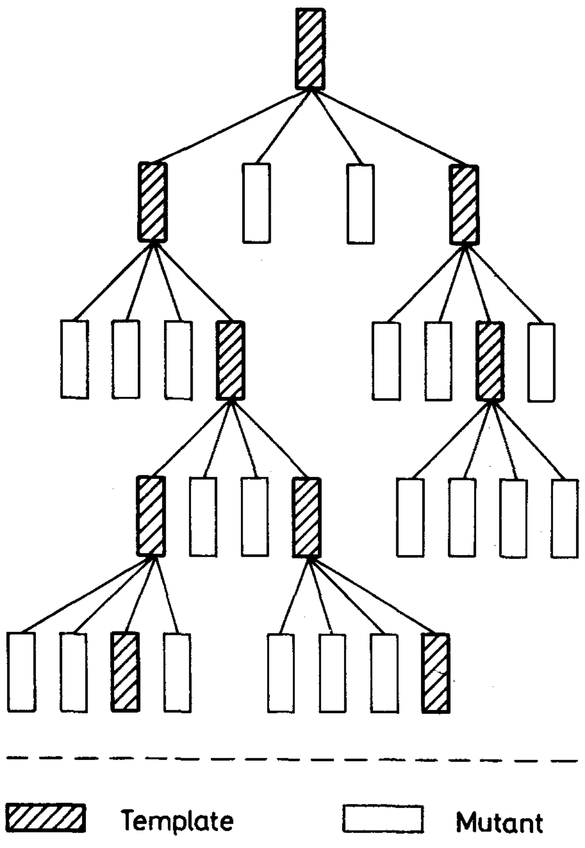

Fig. 1. Schematic illustration of the evolutionary algorithm of optimization

(Rechenberg 1973; Schwefel 1977), but its variation scheme and decision rule have some important differences.

4.1 Variation Scheme

In the case of binary variables, there are four different methods to make a rearrangement of a given configuration:

1. By simulating point mutation. Given a template $(x_1, x_2, \ldots, x_N)$ and a replication fidelity Q (the probability of correct replication), a copy $(x'_1, x'_2, \ldots, x'_N)$ of the template is randomly produced as follows: For each variable x'_k, generate a random number R which is uniformly distributed in the interval (0, 1) and compare it with Q. If $R < Q$, $x'_k = x_k$; otherwise $x'_k = \bar{x}_k$, $\bar{x}_k$ is the complement of x_k.

In this method, the critical point is choosing Q appropriately. Too large or too small values of Q would be no good. A large Q is favourable for maintaining a correct digit, but unfavourable for changing a wrong digit; a small Q is just on the contrary. Furthermore, at different stages of evolution one needs different values of Q. At the early stage, where the sequences in the Survivor Set are far from optimal, one should have a smaller Q to make larger mutations, while at the late stage, when sequences are close to optimal, one should have a larger Q to maintain the most correct digits. Thus one has to make some compromise. It is in fact impossible for a fixed Q to cope with both cases properly. A better way is to vary Q. Starting from an initial Q_0, after each generation let Q increase a small amount ΔQ, and after a specified number (K_g) of generations Q takes a maximal value Q_m and does not change any more. This method to produce new configurations of variables (new "moves") is more efficient than conventional variation schemes, such as that used in the Monte Carlo method, where only one variable changes at one time. The drawback of this option is that at the late stage, where only a few digits are wrong, it sometimes takes several generations to get them in order, especially in the case of long sequences, because of the random nature of the method. This can be overcome by using the next method.

2. By producing a complete mutant spectrum within a given mutation distance d. That means to produce in an exhaustive way all possible sequences which have a Hamming distance d to the template. This will ensure that all possible mutations within mutation distance d will be taken into account. Since the total number C of mutants with Hamming distance d to the template is

$$C = \binom{N}{d} \sim N^d,$$

which increases exponentially with d, this option can be used only in the case of small d, say 1 or 2. In order to produce larger distance mutations, one can use the option 3.

3. By producing a sample of mutant spectra within a larger mutation distance randomly. For instance, to produce a mutant with Hamming distance 10 to the template, select 10 variables of the template at random and flip their digits.

4. By simulating gene recombination. Select two corresponding segments at random out of two given sequences, and exchange them to form two new sequences.

The four options could be used separately or in different combinations to make efficient "moves" in a variety of value landscapes of the objective function.

4.2 Decision Rule

In the decision rule part of the algorithm, the Survivor Set is set up to keep not only the fittest sequence, but also some unfavourable sequences. In each generation, a number of sequences with better values, instead of only one best sequence, are selected as the starting points for the next iteration.

This is an essential revision of the philosophy of optimization. The conventional philosophy of optimization could be characterized as "Improvement Only". In each step a new variation of the configuration of variables would not be accepted unless it improved upon the preceding one. This proves to be a short-sighted strategy, corresponding to the Darwin's statement "Survival of the fittest". If only the fittest could survive, however, the development process will sooner or later get trapped in a local optimum, be it in biological evolution or mathematical optimization.

Consider the escape from a local trap (Fig. 2). It is clear that if one wants to move from the local maximum S_1 to the global maximum S_2 step by step, one has to go first downhill, through the valley S^* and then uphill again. Yet in the value landscape, to go downhill means to accept unfavourable sequences as starting points of the next iteration, i.e. to let them survive and act as templates. Consequently, in each generation one has to allow a number of survivors rather than only a single best one.

The Survivor Set must be large enough to keep unfavourable sequences, especially those sequences which may be the precursors of more optimal ones (Fig. 2a). On the other hand, a large Survivor Set results in an excessive amount of calculations. A remedy is to set a lower bound K_d of Hamming distance between sequences and let the sequences in the Survivor Set keep a certain distance ($> K_d$) from each other so that the size of Survivor Set may be reduced while relevant unfavourable sequences are not excluded from it (Fig. 2b).

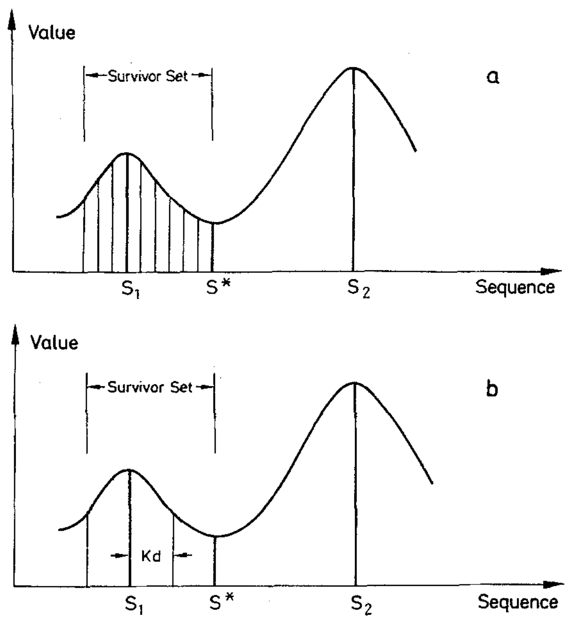

Fig. 2a and b. Schematic illustration of moving from the local maximum S_1 to the global maximum S_2. Before going uphill (improvement) to S_2 one has to first go downhill from S_1 (accepting unfavourable species) to the transition point S^*. **a** Without constraint on distance between sequences. The Survivor Set must consist of a large number of species. **b** With constraint (K_d) on distance between sequences. The Survivor Set can consist of a small number of species

This constraint on Survivor Set is the main innovation of this evolutionary algorithm which is intended to resolve the contradiction between thoroughness of search and reasonable amount of calculation. It enables one to reduce the amount of calculation greatly, sometimes by one order of magnitude, and thereby increase the efficiency of searching for the global optimum, as shown in the computational experiments on a number of examples.

5 Examples

To test the ability of the evolutionary algorithm to find the global maximum and as many local maxima as possible, four examples have been chosen, each has multiple local maxima that are known in advance or can be obtained by exhaustive enumeration. In order to compare the overall performance of the present algorithm with others, a hard optimization problem in communication engineering was used as the fifth example. Although the computations have not been performed by using the best combination of the relevant parameters, yet the results demonstrated the advantage and usefulness of the algorithm.

Example 1

The function to be maximized is

$$F(s_1, \dots, s_{20}) = 10000 - \left[\sum_{i=1}^{20} (s_i - a_i)^2\right]\left[\sum_{j=1}^{20} (s_j - b_j)^2\right] \times \left[\sum_{k=1}^{20} (s_k - c_k)^2\right]\left[\sum_{m=1}^{20} (s_m - d_m)^2\right].$$

Each variable can have values of 1 or 0. $\{a_i\}$, $\{b_j\}$, $\{c_k\}$, and $\{d_m\}$ are four given binary sequences:

$\{a_i\} = 10011\ \ 00101\ \ 11001\ \ 01000$

$\{b_j\} = 11101\ \ 10100\ \ 00011\ \ 10110$

$\{c_k\} = 00010\ \ 11010\ \ 01101\ \ 01111$

$\{d_m\} = 01000\ \ 01011\ \ 10110\ \ 11010.$

The function clearly has four equal global maxima 10000 at these four points. Starting from a random sequence, and after 7 iterations and 3400 function evaluations, the four maxima can be found in a single run.

Example 2

$$F(s_1, \dots, s_{20}) = 10^6 - \left[\sum_{i=1}^{20} (s_i - 1)^2\right]\left[\sum_{j=1}^{20} (s_j - 2)^2\right] \times \left[\sum_{k=1}^{20} (s_k - 3)^2\right]\left[\sum_{m=1}^{20} (s_m - 4)^2\right].$$

This function is like example 1, but a little more complicated. Each variable can take four values: 1, 2, 3, and 4. There are four equal global maxima at points $\{1, 1, \dots, 1\}$, $\{2, 2, \dots, 2\}$, $\{3, 3, \dots, 3\}$, and $\{4, 4, \dots, 4\}$. Starting from a random sequence, after 7 iterations and 11080 function evaluations one maximum was found; after 9 iterations and 13960 evaluations another maximum was also found; after 16 iterations and 24040 evaluations the third maximum was located; and after 17 iterations and 25480 evaluations the last maximum was obtained.

Example 3

The function to be maximized is the Hopfield Hamiltonian (Hopfield 1982):

$$F(s_1, \dots, s_{20}) = \sum_{i<j}^{20} T_{ij} S_i S_j, \qquad S_k = \pm 1,$$

where

$$T_{ij} = \sum_{k=1}^{3} t_{ki} t_{kj}, \qquad t_{km} = \pm 1.$$

It has the form of a spin Hamiltonian. The coefficients T_{ij} are constructed by following 3 randomly selected pattern sequences:

$\{t_{1m}\} = 01000\ \ 01011\ \ 10110\ \ 11011$

$\{t_{2m}\} = 00010\ \ 11011\ \ 10111\ \ 00011$

$\{t_{3m}\} = 10101\ \ 10100\ \ 01001\ \ 10101.$

The function has a number of maxima and minima. Starting from a random sequence, and after 4 iterations and 10980 function evaluations, the algorithm was able to find the best 20 sequences with the highest values (the best 10 sequences and their complementary sequences, which have the same values as the original ones), as compared with the exact solutions which are found by a separate exhaustive enumeration.

Example 4

The function to be maximized is the Spin Glass Hamiltonian (Sherrington and Kirkpatrick 1975)

$$F(s_1, \dots, s_{20}) = \sum_{i<j}^{20} T_{ij} S_i S_j, \qquad S_k = \pm 1,$$

where the T_{ij} are selected randomly according to a Gaussian distribution

$$P(T_{ij}) = \exp(-T_{ij}^2/2).$$

The function has many nearly degenerate random ground states. Starting from a random sequence, after 6 iterations and 22200 function evaluations the best 20 sequences were reproduced as compared with exhaustive enumeration.

Example 5

The function to be maximized is the so-called "merit factor", which is a measure of off-peak autocorrelations of a binary sequence and defined as

$$F(s_1, s_2, \dots, s_N) = N^2/2 \sum_{k=1}^{N-1} R_k^2,$$

where

$$R_k = \sum_{i=1}^{N-k} S_i S_{i+k}, \qquad S_i = \pm 1.$$

This is a hard optimization problem and has a long history (Golay 1982). Because the binary sequences with minimal autocorrelations are of importance in problems of communication engineering, much effort has been devoted to searching for such sequences. By exhaustive search, the maximal merit factors have been found for general sequences up to $N = 32$, and for skew-symmetric sequences up to $N = 59$. In the case of larger N, only restricted searches have been made. The best approximate solutions for skew-symmetric sequences

Table 2. Comparison of approximate maximal merit factors given by (1) Beenker et al. (1985), (2) this paper

N	F (1)	F (2)
101	6.058	6.911, 6.624, 6.556
103	5.900	7.766, 6.062, 6.007
105	6.071	7.614, 6.180

up to $N=199$ were obtained by means of five search methods, including a statistical cooling method, namely the simulated annealing method (Beenker et al. 1985). As no merit factors substantially larger than 6 have been found for long sequences ($N>100$) and hence the question of whether such sequences really exist has been left open, it is of interest to try to search for them by means of the evolutionary algorithm.

Our first trials were restricted to the skew-symmetric sequences of $N=101$, 103, and 105. In each case, after 9 to 10 iterations and 120000 to 150000 function evaluations we were able to obtain approximate optimal solutions better than the results given by Beenker et al. (1985) (Table 2). Our results show that long sequences with merit factors much larger than 6 do exist.

6 Concluding Remarks

Because of the rich similarities between biological evolution and mathematical optimization, it is very natural to simulate mutation mechanism and selection principle to do optimization. This idea has been put forward by a number of authors (Bremermann 1962; Rechenberg 1973; Holland 1975; Schwefel 1977). The work described here is in the same direction, but more concerned with exploiting the unique potential of an evolutionary algorithm in searching for the global optimum as well as local optima by simulating molecular evolution. The overall framework of our algorithm is about the same as the previous work, yet the way of implementing it has some important differences.

In the variation scheme of our evolutionary algorithm, random rearrangement of configurations is supplemented with limited enumeration. Large mutations are produced by random sampling while small mutations are produced by enumeration, so that the creation of "moves" is more efficient. In another option (simulating point mutation), the replication fidelity Q is made variable to adapt to different requirements of different stages of evolution. Accordingly, it is able to lead to improvement quickly.

In the decision rule of the algorithm, the target of selection is a number of sequences with better values rather than a single best sequence. Indeed, this is the common feature of all evolutionary algorithms, but seems not yet to have been fully utilized in practice for finding global optima. In designing our algorithm, however, we take this point seriously and make full use of it to search for global optima. Since Eigen's quasi-species model emphasized the importance of unfavourable sequences in leading the search process to escape from local optima, and on the other hand, the analysis of sequence space indicated that the characteristic structure of sequence space allows a large number of alternative routes to a target sequence and thus provides a suitable stage upon which unfavourable sequences can play their roles (Eigen 1985, 1986; Schuster and Sigmund 1985; Schuster 1986; Hamming 1980), we pay special attention to these unfavourable sequences. We introduce a lower bound of Hamming distance to let sequences in the Survivor Set maintain a certain distance from each other. It is analogous to imposing a selection pressure to limit similar, favourable species and leave more living space to unfavourable ones (Fig. 2b). As a result, a certain diversity of species, which would facilitate the evolution process, can be still maintained in a limited space. This constraint on the Survivor Set is the key feature of our evolutionary algorithm that ensures a higher probability of finding a global optimum quickly.

In so far as can be determined from the examples tested, our evolutionary algorithm is superior to the simulated annealing algorithm in that more and better approximate optimal solutions could be found in a single run. This is not surprising, because in the annealing algorithm the search is proceeding along a single route, and owing to the random nature of the process some parts of variable space may be searched repeatedly while some parts may never be reached at all. In our evolutionary algorithm, however, the search proceeds in parallel along many routes simultaneously; at each iteration the starting points always keep a certain distance from each other so that the search thoroughly and effectively covers a much larger region. Even if one route gets stuck the search along the other routes can still proceed.

This evolutionary algorithm is independent of the concrete form of the objective function and thus especially suitable for dealing with problems of nonlinear optimization, which usually can not be solved efficiently by conventional methods. The main advantage of the algorithm, with greater probability to find the global optimum and as many local optima as possible in a given domain and in a single run, is of great value to practical applications: more suboptimal solutions are available for choice, and a globally optimal scheme can be carried out through intermediate, locally optimal stages.

While the evolutionary algorithm is designed for combinatorial optimization, the extension to the case of continuous variables is straightforward. Moreover, owing to the inherent parallelism of search, the evolutionary algorithm is ideally suited for parallel processing computer systems, which will speed up the calculation enormously. The efficiency of the algorithm depends on a number of parameters: N_m, N_s, N_g, Q_0, Q_m, K_g, K_d and so on. Of course, different objective functions need different parameter combinations. To choose a best combination of these parameters for a given objective function is in itself an optimization problem that can only be solved by computer experiments. This would require a suitable method of experimental design, and experiences and tricks on the part of users. This kind of "intelligence" could eventually be programmed and combined with the program coded for the evolutionary algorithm itself to make a software package, which would provide a powerful tool to solve complicated and hard problems of optimization.

Full implementation of the evolutionary algorithm of optimization would open up new possibilities to biomolecular engineering and might find applications in biotechnology. In fact, an experimental realization of the algorithm, the construction of a so-called evolution machine is under way (Eigen 1985, 1986).

Acknowledgements. I wish to thank Prof. M. Eigen for encouragement and helpful discussions, and Prof. A. Dress for valuable comments. A Stipendium from Max-Planck-Gesellschaft is gratefully acknowledged.

References

Beenker GFM, Claasen TACM, Hermens PCW (1985) Binary sequences with maximally flat amplitude spectrum. Philips J Res 40:289–304

Biebricher CK, Eigen M, Luce R (1981a) Product analysis of RNA generated de novo by Q_β replicase. J Mol Biol 148:369–390

Biebricher CK, Eigen M, Luce R (1981b) Kinetic analysis of template-instructed and de novo RNA synthesis by Q_β replicase. J Mol Biol 148:391–410

Bremermann HJ (1962) Optimization through evolution and recombination. In: Yovits MC, Goldstein GD, Jacobi GT (eds) Self-organizing systems. Spartan, Washington D.C., pp 93–106

Eigen M (1971) Self-organization of matter and the evolution of biological molecules. Naturwissenschaften 58:465–523

Eigen M (1985) Macromolecular evolution: Dynamical ordering in sequence space. Ber Bunsenges Phys Chem 89:658–667

Eigen M (1986) The physics of molecular evolution. Chem Scr 26B:13–26

Eigen M, Schuster P (1977) The hypercycle – a principle of natural self-organization. Part A: Emergence of the hypercycle. Naturwissenschaften 64:541–565

Eigen M, Schuster P (1978a) The hypercycle – a principle of natural self-organization. Part B: The abstract hypercycle. Naturwissenschaften 65:7–41

Eigen M, Schuster P (1978b) The hypercycle – a principle of natural self-organization. Part C: The realistic hypercycle. Naturwissenschaften 65:341–369

Golay MJE (1982) The merit factor of long low autocorrelation binary sequences. IEEE, Trans Inform Theory IT-28:543–549

Hamming RW (1980) Coding and information theory. Prentice Hall, Englewood Cliffs, NJ

Holland JH (1975) Adaption in natural and artificial systems. University of Michigan, Ann Arbor

Hopfield JJ (1982) Neural networks and physical systems with emergent collective computational abilities. Proc Natl Acad Sci USA 79:2554–2558

Kirkpatrick S, Gelatt CD Jr, Vecchi MP (1983) Optimization by simulated annealing. Science 220:671–680

Kramer FR, Mills Dr, Cole PE, Nishihara T, Spiegelman S (1974) Evolution in vitro: sequence and phenotype of a mutant RNA resistant to ethidium bromide. J Mol Biol 89:719–736

Mills DR, Peterson RL, Spiegelman S (1967) An extracellular Darwinian experiment with a self-duplicating nucleic acid molecule. Proc Natl Acad Sci USA 58:217–224

Rechenberg I (1973) Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution, Frommann-Holzboog, Stuttgart

Schuster P (1986) The physical basis of molecular evolution. Chem Scr 26B:27–41

Schuster P, Sigmund K (1985) Dynamics of evolutionary optimization. Ber Bunsenges Phys Chem 89:668–682

Schwefel HP (1977) Numerische Optimierung von Computer-Modellen mittels der Evolutionsstrategie, Birkhäuser, Basel

Sherrington D, Kirkpatrick S (1975) Solvable model of a spin-glass. Phys Rev Lett 35:1792–1796

Spiegelman S (1970) The development and use of an extracellular RNA replicating system. In: The Harvey Lectures, Series 64, Academic Press, New York London, pp 1–67

Sumper M, Luce R (1975) Evidence for de novo production of self-replicating and environmentally adapted RNA structures by bacteriophage Q_β replicase. Proc Natl Acad Sci USA 72:162–166

Received: February 24, 1987

Qizhong Wang
Max-Planck-Institut für
Biophysikalische Chemie
D-3400 Göttingen
Federal Republic of Germany