A Memetic Inference Method for Gene Regulatory Networks Based on S-Systems

Christian Spieth, Felix Streichert, Nora Speer, and Andreas Zell
Centre for Bioinformatics Tübingen (ZBIT)
University of Tübingen
D-72076 Tübingen, Germany
Email: spieth@informatik.uni-tuebingen.de

***Abstract*— In this paper we address the problem of finding gene regulatory networks from experimental DNA microarray data. As underlying mathematical model we used S-Systems, a quantitative model, which recently has found increased attention in the literature. Due to the complexity of the inference problem some researchers suggested Evolutionary Algorithms for this purpose. We introduce enhancements to this optimization process to infer the parameters of sparsely connected non-linear systems given by the observed data more reliably and precisely. Due to the limited number of available data the inferring problem is underdetermined and ambiguous. Further on, the problem often is multi-modal and therefore appropriate optimization strategies become necessary. In this paper we propose a new method, which evolves the topology as well as the parameters of the mathematical model to find the correct network. This method is compared to standard algorithms found in the literature.**

I. INTRODUCTION

A number of technologies have been developed over the last few years enabling gene experiments in high-throughput set ups. These technologies include DNA microarrays, which are designed to measure gene expression levels in living cells. DNA chips allow the measurement of thousands of interactions between mRNA-derived molecules and genome-derived probes simultaneously. Since the number of probes on a chip is in the order of ten thousands per experiment, biologists encounter an enormous amount of raw data, which is almost impractical to handle by eyes only. With this, Systems Biology has become an important field in biology, which aims at deep insights into biological systems. The main focus of current research is mainly on identification of genes that show significant changes between different experiments, or genes that can be clustered by their course of expression over time. The next step is to understand the principles of biological systems grounded on the molecular level. To provide a deep understanding of life, we have to understand not only the components of the systems but also their dependencies, interactions, and structures. A system-level approach is necessary if we want to incorporate large amounts of data into a comprehensive model of the structure and functions of the complex mechanisms within an organism.

Gene expression is regulated at many molecular levels starting from the DNA level to mRNA to proteins. The proteins react and interact with others and with the DNA in many ways to build transcriptional molecules, which regulate the rate of transcription of a gene. Further regulation occurs at the level of RNA processing, transport and translation into proteins, which is not fully understood today. Advanced computational methods together with high-throughput experimental technologies like DNA arrays show promising results on the way to understand the regulatory mechanisms and regulatory network structures of gene expression.

II. COMPUTATIONAL BIOLOGY

On the computational side, several mathematical models have been applied to the problem of inferring gene regulatory networks.

The earliest models to simulate regulatory systems found in the literature are Boolean or Random Boolean Networks (RBN) [8]. In Boolean Networks gene expression levels can be in one of two states: either 1 (on) or 0 (off). The quantitative level of expression is not considered. Two examples for inferring Boolean Networks are given by Akutsu [1] and the REVEAL algorithm by Liang *et al.* [12]. These models have the advantage that they can be solved with only small computational effort. But they suffer from the disadvantage of being tied to discrete system states.

In contrast to discrete methods like RBNs, qualitative network models allow for multiple levels of gene regulation. An example for this kind of approach is given by Thieffry and Thomas in [20]. Akutsu *et al.* suggested a heuristic for inferring such models in [2]. But these models use only qualitative dependencies and therefore only a small part of the information hidden in the time series data.

Quantitative models based on linear models for gene regulatory networks like the weighted matrix model by Weaver *et al.* [22] or the singular value decomposition method by Yeung *et al.* [24] consider the continuous level of gene expression. But they are restricted to linear dependencies, which makes it difficult to model realistic networks.

Other approaches to infer regulatory systems from time series data using Artificial Neural Networks [9] or Bayesian Networks [7] have been recently published, but face some drawbacks as well. Bayesian networks, for example, do not allow for cyclic networks. Arbitrary differential equations can be used to model regulatory structures as well as Ando *et al.* showed with Genetic Programming (GP) in [3].

More general examples for mathematical non-linear models like S-Systems to infer regulatory mechanisms have been examined by Maki *et al.* [13], Kiguchi *et al.* [10] or Tominaga *et al.* [21]. The non-linear methods face a severe disadvantage of having many more model parameters to be inferred. Tominaga et al., for example, bypass this problem by inferring only a subset of genes of the original system (2 genes out of 5). So far, methods using S-Systems or arbitrary differential equations regard only a small number of components within the regulatory network to be reverse engineered, i.e. a total number of genes $N \leq 10$.

Nevertheless, we try to use the advantages of flexible mathematical models like S-Systems in our approach because they are able to represent complex structures of gene regulations. We introduce a Memetic Algorithm, which separates the inference problem into two subproblems. The first problem is to find a proper topology or structure of the network with a Genetic Algorithm (GA). In the second problem the parameters for the topology given by a GA individual are optimized with an Evolution Strategy (ES). The second problem can be seen as a local search phase of a Memetic Algorithm (MA).

The remainder of this paper is structured as follows. Sections III and IV describe the proposed algorithm and the mathematical model used in the optimization process. Applications and the results of the comparison are listed in section V, and the conclusions and an outlook are given in sections VI and VII.

III. INFERENCE METHOD

First, we describe the Memetic Algorithm, which optimizes the topology and the parameters followed by a description of the mathematical model used for representing the regulatory system.

A. Evolutionary Algorithms

Evolutionary Algorithms (EAs) have proven to be a powerful tool for solving complex optimization problems. Three main types of Evolutionary Algorithms have evolved during the last 30 years: Genetic Algorithms (GA), mainly developed by J.H. Holland [6], Evolution Strategies (ES), developed by I. Rechenberg [15] and H.-P. Schwefel [17] and Genetic Programming (GP) by J.R. Koza [11]. Each of these uses different representations of the data and different main operators working on them. They are, however, inspired by the same principles of natural evolution. Evolutionary Algorithms are a member of a family of stochastic search techniques that mimic the natural evolution of repeated mutation and selection as proposed by Charles Darwin.

A combination of an EA with a local search heuristic is referred to as Memetic Algorithm [14]. The proposed Memetic Algorithm uses a combination of two of the main types of EA, i.e. Genetic Algorithms and Evolution Strategies. These two are briefly outlined in the next sections.

1) Genetic Algorithm: Genetic Algorithms imitate the evolutionary processes with emphasis on the genetical mechanisms. The GA works on a population of artificial chromosomes, referred to as individuals. Each individual is represented by a string of L bits. Each segment of this string corresponds to a variable of the optimization problem in a binary encoded form.

The population is evolved in the optimization process mainly by crossover operations. This operation recombines the bit strings of individuals in the population with a certain probability p_c. Mutation is secondarily in most applications of a GA. It is responsible to ensure that some bits are changed, thus allowing the GA to explore the complete search space even if alleles are temporarily lost due to convergence.

2) Evolution Strategies: The second type of an Evolutionary Algorithm is the Evolution Strategy. ES differ mainly from GAs in respect to the representation of solutions and the selection operators.

The selection of the individuals forming a population is deterministic as in contrast to GAs where a stochastical method is used. In case of the (μ, λ)-ES selection strategy, the μ best individuals from a population of λ offsprings are selected to create the next population. An alternative implementation is the $(\mu + \lambda)$-strategy, which selects the μ best individuals from the population of the λ offsprings joined with the old population of μ parents.

The use of sophisticated mutation operators is emphasized in ES while recombination is only of lower importance.

B. Memetic Algorithm

In the current implementation we use a combination of a Genetic Algorithm for optimizing the topology together with an Evolution Strategy to locally find the best parameters for the given topology. The general principle is outlined in fig. 1.

The procedure ”eval(population)” used for evaluating the fitness for a given population of possible topologies is given schematically in fig. 2.

The Memetic Algorithm uses a Genetic Algorithm to evolve populations of structures of possible networks. These structures are encoded as bitsets, where each bit represents the existence or absence of an interaction between genes and therefore of non-zero parameters in the mathematical model. The evaluation of the fitness of each individual within the GA population applies a local search to find suitable parameters, which is described in the next section.

```
Algorithm MA
begin
   t:=0;
   P_GA(t):=initGAPop();
   eval(P_GA(t));
   while (termination criteria not met)
      P'_GA(t):=selectGAParentPop(P_GA(t));
      P''_GA(t):=createGAOffsprings(P'_GA(t));
      eval(P''_GA(t));
      P_GA(t+1):=selectNewGAPop(P''_GA(t));
      t:=t+1;
   endwhile
   end
```

Fig. 1. Pseudo-code describing the general principle of the Memetic Algorithm

```
eval(P_GA)
begin
   s:=0;
   for each individual in P_GA do
      P_ES(s):=initESPop();
      eval(P_ES(s));
      while (termination criteria not met)
         P'_ES(s):=selectESParentPop(P_ES(s));
         P''_ES(s):=createESOffsprings(P'_ES(s));
         eval(P''_ES(s));
         P_ES(s+1):=selectNewESPop(P''_ES(s));
         s:=s+1;
      endwhile
      setFitness(GAInd, bestESFitness);
   endfor
end
```

Fig. 2. Pseudo-code describing the general principle of the local search

Fig. 3 shows schematically the work flow of the optimizing process. A bitset (left) is suggested by the GA. It represents a specific network topology (middle) and this topology is then encoded into decision variables to be optimized by the ES.

For evaluation of each structure suggested by the global optimizer an Evolution Strategy is used, which is suited for the parameter optimizing problem, since it is based on real values. The ES optimizes the parameters of the mathematical model used for representation of the regulatory network.

C. Fitness

For assessing the quality of the locally obtained results we use the following equation for calculation of the fitness values for the ES optimization process:

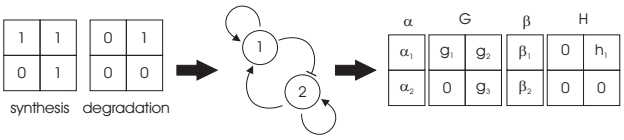

Fig. 3. Work flow of the MA optimizing process

$$f = \sum_{i=1}^{N} \sum_{k=1}^{T} \left(\frac{\hat{x}_i(t_k) - x_i(t_k)}{x_i(t_k)} \right)^2 \tag{1}$$

where N is the total number of genes in the regulatory system, T is the number of sampling points taken from the time series and $\hat{x}$ and x distinguish between estimated data and experimental data. The overall problem is to minimize the fitness value f. This function has been used in several publications [10], [13], [21].

D. S-Systems

On an abstract level, the behavior of a cell is represented by a gene regulatory network of N genes. Each gene g_i produces a certain amount of mRNA x_i when expressed and therefore changes the concentration of this mRNA over time: $x_i(t+1) = h_i(\vec{x}(t))$ with $\vec{x}(t) = (x_1, \cdots, x_n)$, where h_i describes the change of each RNA level depending on all or only on some RNA concentrations in the previous time step.

To model and to simulate regulatory networks we decided to use S-Systems since we think they are flexible enough to model important gene regulatory dependencies like feed back loops, etc. But there are alternatives as listed in section II, which will be the subject of research in future applications.

S-Systems are a type of power-law formalism, which has been suggested by Savageau [16] and can be described by a set of nonlinear differential equations:

$$\frac{dx_i(t)}{dt} = \alpha_i \prod_{j=1}^{N} x_j(t)^{\mathcal{G}_{i,j}} - \beta_i \prod_{j=1}^{N} x_j(t)^{\mathcal{H}_{i,j}} \tag{2}$$

where $\mathcal{G}_{i,j}$ and $\mathcal{H}_{i,j}$ are kinetic exponents, α_i and β_i are positive rate constants and N is the number of equations in the system. The equations in (eq. 2) can be seen as divided into two components: synthesizing and degrading component.

The kinetic exponents $\mathcal{G}_{i,j}$ and $\mathcal{H}_{i,j}$ determine the structure of the regulatory network. In the case $\mathcal{G}_{i,j} > 0$ gene g_j induces the synthesis of gene g_i. If $\mathcal{G}_{i,j} < 0$ gene g_j inhibits the synthesis of gene g_i. Analogously, a positive (negative) value of $\mathcal{H}_{i,j}$ indicates that gene g_j induces (suppresses) the degradation of the mRNA level of gene g_i.

The parameters of the S-System $\vec{\alpha}$, $\vec{\beta}$, $\mathcal{G}$, and $\mathcal{H}$ are optimized by the local ES method to fit the given time-series data according to the fitness function (eq. 1).

IV. ALGORITHM SETTINGS

To evaluate the proposed method we created two artificial gene regulatory networks, which were simulated to gain microarray expression data sets. These data sets were then to be re-engineered by our algorithm. To compare the results with established inference methods we also used a standard ES and an enhancement to a GA developed by Tominaga *et al.* [21] to infer the parameters of the network as described in the following. To obtain reliable results, each example setting was repeated $m = 20$ times.

Standard ES. The inference by a standard ES was performed using a (μ,λ)-ES with $\mu = 10$ parents and $\lambda = 50$ offsprings together with a Covariance Matrix Adaptation (CMA) mutation operator (see Hansen and Ostermeier [5]) without recombination.

Skeletalizing. The second algorithm is an extension to a standard real-coded GA using tournament selection with a tournament group size of $t_{group} = 8$, 3-Point-crossover recombination with $p_c = 1.0$ and a mutation probability $p_m = 0.1$, referred to as "skeletalizing" in several publications. This enhancement introduces a threshold value $t_{skel} = 0.05$, which represents a lower boundary for the parameters $\mathcal{G}_{i,j}$ and $\mathcal{H}_{i,j}$ in the mathematical model. If a decoded decision variable of the GA drops below this threshold during optimization the corresponding phenotype value is forced to 0.0.

Additionally to the original algorithm suggested by Tominaga *et al.* we implemented two versions of the skeletalizing method. SKEL$_A$ forced the phenotype values below the threshold to 0.0, thus relying only on the Baldwin effect whereas SKEL$_B$ encoded these changes back into the genotype of the evaluated individual (Lamarckism). Detailed descriptions of the Baldwin effect and Lamarckism can be found in Withley *et al.* [23].

Memetic Algorithm. The main focus of this publication is the performance of the proposed Memetic Algorithm as presented in sect. III. The GA evolved a population of possible structures with a tournament selection with a tournament group size of $t_{group} = 8$, 3-Point-crossover recombination with $p_c = 1.0$ and a mutation probability $p_m = 0.1$. The local optimization was started using a (μ,λ)-ES with $\mu = 10$ parents and $\lambda = 50$ offsprings together with a Covariance Matrix Adaptation (CMA) mutation operator without recombination as in the case of the standard ES.

V. RESULTS

A. Artificial Gene Regulatory Networks

The first data set represents the time dynamics of an artificial 5-dimensional regulatory system, i.e. the relationships between 5 genes, which were randomly assigned and simulated. The second example is an artificially created 10-dimensional GRN, which consists of 10 components.

1) 5-dimensional network: Due to the fact that GRNs in nature are sparse systems, we created regulatory networks randomly with a maximum cardinality of $k \leq 3$, i.e. each of the $N = 5$ genes depends on three or less other genes within the network. The dynamics of the first example can be seen in fig. 4.

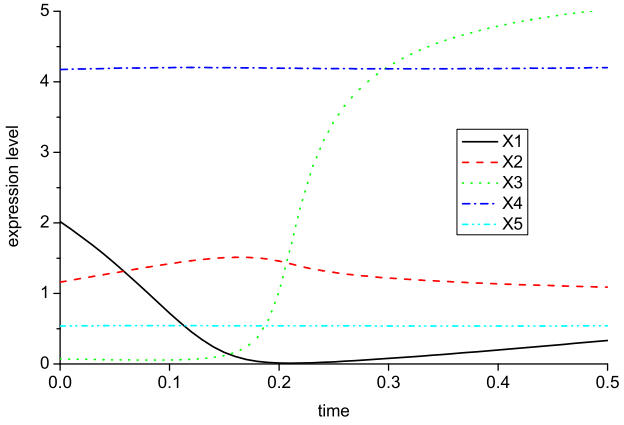

Fig. 4. Time dynamics of the 5-dim regulatory system

The four algorithms were used to infer the underlying regulatory dependencies with the parameter settings given in the previous section. The resulting averaged fitness courses are plotted in the following graph (fig. 5). To compare the results of the four methods a total number of fitness evaluations $N_{max} = 2,000,000$ was specified. In case of the Memetic Algorithm the optimizer performed $N_{max} = 2,000,000$ local ES evaluations with $N_{local} = 500$, i.e. 500 evaluations for each bitset. The fixed number of fitness evaluations resulted in 40 generations with 100 individuals each for the GA.

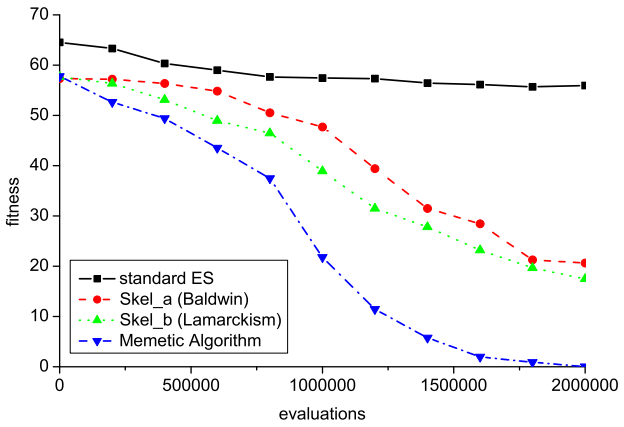

Fig. 5. Fitness course of the 5-dim regulatory system

Fig. 5 shows the fitness courses for the different methods. As can be seen, the standard ES is not able to find a solution to the optimizing problem. Obviously, it got stuck in a local optimum without being able to escape. Both skeletalizing GAs were able to converge to relative good fitness values, with a slight advantage of the implementation using Lamarckism. It seems that both did not converge with the specified number of evaluations. Further on, the GAs started with better fitness values than the ES due to the larger size of the initial population, which increases the probability of finding a good individual in the first steps of the optimization process. In contrast to the other methods the proposed MA converged quickly and reliably to very good fitness values, suggesting that it can achieve even better results with increased number of fitness evaluations. The MA also benefits from the advantage of the bigger population size of the GA that is used to evolve the bitsets representing different network topologies.

Unfortunately, the resulting networks face a weighty problem. Looking closer at the results it becomes clear that good fitness values do not necessarily correspond to the correct network topology. The skeletalizing GA did not find the correct solutions in any of the repeated optimizing runs. Our MA found only twice the correct target system with respect to the topology and parameter values. In the remaining 18 optimization runs, systems were found, which fitted the experimental data but showed different relationships between the component genes. We will address this problem in the discussion (sect. VI).

2) 10-dimensional network: As a second test case we created another regulatory network randomly again with a maximum cardinality of $k \leq 3$. The dynamics of the example are given in fig. 6.

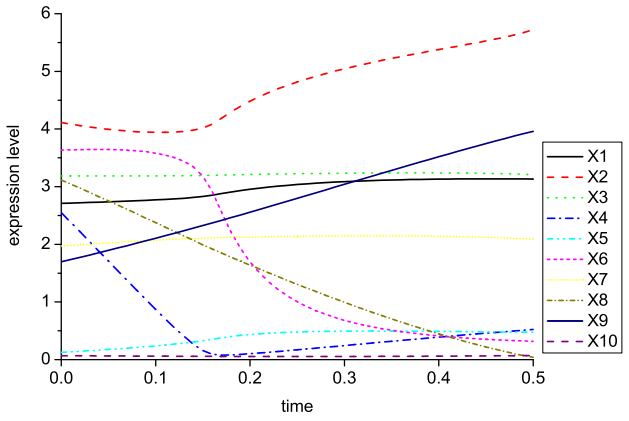

Fig. 6. Time dynamics of the 10-dim regulatory system

The optimization processes were performed as in the example before. Each algorithm was again terminated after a total number of fitness evaluations $N_{max} = 2,000,000$.

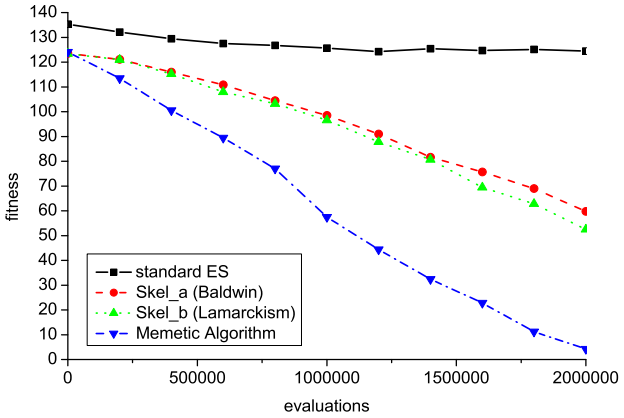

Fig. 7. Fitness course of the 10-dim regulatory system

As illustrated by the fitness plot the standard ES was again not able to find a solution for the optimization problem. Again, the ES is not able to escape a local optimum and therefore no further improvement is achieved. The advantage of the GA with Lamarckism was not as significant as in the example before. But again both seem to have the potential to reach good fitness values with a larger number of fitness evaluations. And all GAs started with better fitness values as in the example before. Our proposed MA outperformed again all other methods by finding very good solutions with respect to the fitness values with a very good speed of convergence.

As before in the 5-dimensional example the problem of finding the correct target system emerged again. The resulting time courses fitted the experiment data (yielding good fitness values) but showed different dependencies and interactions between the participating genes of the regulatory network.

VI. DISCUSSION

In this paper we introduced a new method to infer gene regulatory networks from time-series microarray data. Our method yielded far better fitness values compared to a standard ES, which was not able to find models that fit the given data in both test cases. This is due to the fact that the ES got stuck in local optima in each of the test cases. The algorithms proposed by Tominaga *et al.* were able to find comparably good sets of model parameters with respect to the fitness value but faced the major disadvantage of relying on large population sizes and therefore on a large number of total fitness evaluations. Our proposed algorithm performed best on the two test cases. This suggests that the algorithm is able to find a network structure that is either similar to the correct one or which represents a network topology with similar properties.

Additionally, our MA proved to work even in middle-sized examples. Most examples found in literature are artificial and

very small, i.e. with a total number of five genes or lower. The low dimensionality of these examples is by far not relevant to biological networks where even small systems have at least 50–100 components. We showed that our method is able to handle 10 genes, restricted currently only by computational performance. Because we use a bitset representation of the topology, the algorithm reduces the total number of parameters and makes it therefore possible to infer larger systems. Future experiments on high performance computers will address large-scale systems with 100 genes or more.

Further on, the solutions found by the MA are sparse due to the preceding structure optimization. Because in nature GRNs are sparse systems the solutions of the MA represent better resemblance to biological systems than the standard ES, which resulted always in complete and thus dense matrices. Therefore, the proposed MA is better suited to infer gene regulatory systems.

Due to the large number of model parameters and the small number of data sets available, the system of equations is highly under-determined. Therefore, multiple solutions exist, which fit the given data, but show only little resemblance with the original target system. This problem is known in literature but there are currently only few publications reflecting on this issue. Recently, the authors published a new method to incorporate data sets obtained by additional experiments [18]. In future enhancements of our algorithm we plan to incorporate such additional methods to identify the correct network.

VII. FUTURE WORK

In future work we also plan to include a-priori information into the inference process of real microarray data like partially known pathways or information about co-regulated genes, which can be found in literature. For better maintaining of the diversity we plan to use a cluster-based niching algorithm, which was developed in our group [19].

Additional models for gene regulatory networks will be examined for simulation of the non-linear interaction system as listed in sect. II to overcome the problems with a quadratic number of model parameters of the S-System.

Further on, we will continue to test our method with real microarray data in close collaboration with biological researchers at our university.

ACKNOWLEDGMENT

This work was partially supported by the National Genome Research Network (NGFN) [4] from the Federal Ministry of Education and Research in Germany.

REFERENCES

[1] T. Akutsu, S. Miyano, and S. Kuhura. Identification of genetic networks from a small number of gene expression patterns under the boolean network model. In *Proceedings of the Pacific Symposium on Biocomputing*, pages 17–28, 1999.

[2] T. Akutsu, S. Miyano, and S. Kuhura. Algorithms for identifying boolean networks and related biological networks based on matrix multiplication and fingerprint function. In *Proceedings of the fourth annual international conference on Computational molecular biology*, pages 8 – 14, Tokyo, Japan, 2000. ACM Press New York, NY, USA.

[3] S. Ando, E. Sakamotob, and H. Iba. Evolutionary modeling and inference of gene network. *Information Sciences*, 145(3-4):237–259, 2002.

[4] German Federal Ministry of Education and Research (BMBF). National Genome Research Network (NGFN). www.ngfn.de, 2001.

[5] N. Hansen and A. Ostermeier. Adapting arbitrary normal mutation distributions in evolution strategies: the covariance matrix adaptation. In *Proceedings of the 1996 IEEE Int. Conf. on Evolutionary Computation*, pages 312–317, Piscataway, NJ, 1996. IEEE Service Center.

[6] J. H. Holland. *Adaption in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Systems*. The University Press of Michigan Press, Ann Arbor, 1975.

[7] S. Imoto, T. Higuchi, T. Goto, K. Tashiro, S. Kuhara, and S. Miyano. Combining microarrays and biological knowledge for estimating gene networks via bayesian networks. In *Proceedings of the IEEE Computer Society Bioinformatics Conference (CSB 03)*, pages 104 –113. IEEE, 2003.

[8] S. A. Kauffman. *The Origins of Order*. Oxford University Press, New York, 1993.

[9] E. Keedwell, A. Narayanan, and D. Savic. Modelling gene regulatory data using artificial neural networks. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN 02)*, volume 1, pages 183–188, 2002.

[10] S. Kikuchi, D. Tominaga, M. Arita, K. Takahashi, and M. Tomita. Dynamic modeling of genetic netowrks using genetic algorithm and s-sytem. *Bioinformatics*, 19(5):643–650, 2003.

[11] J. R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, Cambridge, MA, USA, 1992.

[12] S. Liang, S. Fuhrman, and R. Somogyi. REVEAL, a general reverse engineering algorithm for inference of genetic network architectures. In *Proceedings of the Pacific Symposium on Biocomputing*, volume 3, pages 18–29, 1998.

[13] Y. Maki, D. Tominaga, M. Okamoto, S. Watanabe, and Y. Eguchi. Development of a system for the inference of large scale genetic networks. In *Proceedings of the Pacific Symposium on Biocomputing*, volume 6, pages 446–458, 2001.

[14] P. Moscato and M. G. Norman. A memetic approach for the traveling salesman problem. implementation of a computational ecology for combinatorial optimization on message-passing systems. In M. J. J. L. M. Valero, E. Onate and B. Suarez, editors, *Parallel Computing and Transputer Applications*, pages 187–194, 1992.

[15] I. Rechenberg. *Evolutionsstrategie - Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Frommann-Holzboog, Stuttgart, 1973.

[16] M. A. Savageau. 20 years of S-systems. In E. Voit, editor, *Canonical Nonlinear Modeling. S-systems Approach to Understand Complexity*, pages 1–44, New York, 1991. Van Nostrand Reinhold.

[17] H.-P. Schwefel. *Numerical optimization of computer models*. John Wiley and Sons Ltd, 1981.

[18] C. Spieth, F. Streichert, N. Speer, and A. Zell. Iteratively inferring gene regulatory networks with virtual knockout experiments. In R. et al., editor, *Proceedings of the 2nd European Workshop on Evolutionary Bioinformatics (EvoWorkshops 2004)*, LNCS. Springer, 2004.

[19] F. Streichert, G. Stein, H. Ulmer, and A. Zell. A clustering based niching ea for multimodal search spaces. In *In Proceedings of the 6th International Conference on Artificial Evolution*, Lecture Notes in Computer Science, pages 169–180, 2003.

[20] D. Thieffry and R. Thomas. Qualitative analysis of gene networks. In *Proceedings of the Pacific Symposium on Biocomputing*, pages 77–87, 1998.

[21] D. Tominaga, N. Kog, and M. Okamoto. Efficient numeral optimization technique based on genetic algorithm for inverse problem. In *Proceedings of German Conference on Bioinformatics*, pages 127–140, 1999.

[22] D. Weaver, C. Workman, and G. Stormo. Modeling regulatory networks with weight matrices. In *Proceedings of the Pacific Symposium on Biocomputing*, volume 4, pages 112–123, 1999.

[23] D. L. Whitley, V. S. Gordon, and K. E. Mathias. Lamarckian evolution, the baldwin effect and function optimization. In *Parallel Problem Solving from Nature – PPSN III*, pages 6–15. Springer, 1994.

[24] M. K. S. Yeung, J. Tegner, and J. J. Collins. Reverse engineering gene networks using singular value decomposition and robust regression. In *Proceedings of the National Academy of Science USA*, volume 99, pages 6163–6168, 2002.