# Feedback Memetic Algorithms for Modeling Gene Regulatory Networks

C. Spieth, F. Streichert, J. Supper, N. Speer, and A. Zell
Centre for Bioinformatics
University of Tübingen
Sand 1, 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. We focus on the evaluation of the performance of memetic algorithms on the inference problem. These algorithms are used to evolve an underlying quantitative mathematical model. The dynamics of the regulatory system are modeled with two commonly used approaches, namely linear weight matrices and S-systems. Due to the complexity of the inference problem, some researchers suggested evolutionary algorithms for this purpose. We introduce memetic enhancements to this optimization process to infer the parameters of sparsely connected nonlinear systems from the observed data. Due to the limited number of available data, the inferring problem is underdetermined and ambiguous. Further on, the problem often is multimodal and therefore appropriate optimization strategies become necessary. We propose a memetic method, which separates the overall inference problem into two subproblems to find the correct network: first, the search for a valid topology, and secondly, the optimization of the parameters of the mathematical model. The performance and the properties of the proposed methods are evaluated and compared to standard algorithms found in the literature.**

## I. Introduction

Systems biology has become one of the major research areas in biology over the past few years. Due to tremendous progress in experimental methods like DNA microarrays, several thousand expression levels of genes in an organism can be measured in parallel under specific environmental conditions [30]. This enables researchers to examine intracellular processes on a systemic level. The inference of gene regulatory networks from experimental data is one of the main unsolved problems in the post-genomic area. A gene regulatory network (GRN) is an abstract model representing dependencies between genes using a directed graph. In this graph, each node is a gene or component of the regulatory system and each edge represents a regulatory impact from one component to the other (e.g. activation or suppression of the transcription and translation of the dependent gene).

Several publications addressing the problem of inferring gene regulatory networks can be found in the literature. De Jong gives a good overview about related work in [5]. A major part of the work done in this field is using deterministic mathematical models to simulate regulatory networks. One kind of those deterministic models are linear models like the weighted matrix model [28], [29]. These models have only a small number of system parameters compared to S-systems but are often not flexible enough to model biological systems in detail, since they model the dependencies linearly. S-systems, on the other hand, model dynamic systems in a nonlinear manner. They consists of a set of differential equations describing the changes in expression over time. However, they show a significant higher number of system parameters. S-systems have been recently examined in [14], [15], [17]. The most flexible type of mathematical model are ordinary differential equations, which have been investigated in [4] and [3]. Most applications of deterministic models use evolutionary algorithms (EA) to determine the correct parameters of the mathematical model. EAs have proven to be successful in finding parameters of mathematical models representing GRNs. Another major focus of recent work are stochastic methods like Bayesian networks, where the dependencies are modeled by probabilistic transition values. Examples for this kind of models can be found in recent publications [8], [11], [12].

So far, only small networks have been successfully inferred by computational methods. Larger networks could be reconstructed, only if the participating genes show very similar time dynamical behavior as the target system. However, the correctness of the connections in a large graph cannot be verified. The main obstacle is the ambiguity in the data and the resulting high number of possible network structures. This is caused by the limited number of microarrays compared to the number of variables in the network model, thus making the estimation of the underlying system a very difficult task. We introduce a method, which separates the inference problem into two subproblems. The first task is to find a valid topology or structure of the network with a genetic algorithm (GA). In the second task, the parameters of a mathematical model are optimized by local search steps on the given topology with an evolution strategy (ES). In contrast to a previous paper [24], we additionally examine enhanced versions of the memetic algorithms, extended by a feedback mechanism. Here, the local search returns a feedback to the global optimizer to reduce the complexity of the search space. This was done by deleting dependencies in the model that had already been assigned a small interaction strength. Thus, the number of model parameters are significantly reduced. We examine two different implementations of the Feedback MA (FMA). The first uses a fixed threshold value determining the upper limit

0-7803-9387-2/05/$20.00 ©2005 IEEE

of values to be filtered. In the second implementation, this threshold value is itself evolved and optimized by the MA. Both types of FMA, the previous published MA and several standard optimization algorithms are compared on the inference problem using the two most commonly used deterministic type of models, namely linear weight matrices and S-systems. Both recently have found increased attention in the literature and were thus selected to examine the performance of the mentioned algorithms. For evaluation, we will apply the proposed methods in comparison to standard algorithms on artificial expression data of regulatory networks with 5-10 genes.

The remainder of this paper is structured as follows. Section II and III describe the mathematical models and the proposed algorithm used in the optimization process. Applications and results are listed in section V and the conclusions and an outlook are given in section VI.

## II. Mathematical Modeling

The genetic dependencies of a cell can be abstracted by a directed graph with $N$ nodes representing $N$ genes. Each gene $g_i$ produces a certain amount of mRNA $x_i$ when expressed and changes the concentration of the mRNA level over time: $\vec{x}(t+1) = h(\vec{x}(t))\,, \vec{x}(t) = (x_1, \cdots, x_n)$. Here, function $h$ represents the changes of expression levels from one state to the next. To model this function, several approaches can be found in the literature. We decided to use the two most popular deterministic models, namely linear weight matrices and S-systems. These models are described in details in the following sections.

### A. Linear Weight Matrices

Linear weight matrices (WM) have been originally introduced in [28]. In this approach, the regulative interactions between the genes are represented by a weight matrix, $\mathcal{W}$, where each row of $\mathcal{W}$ represents all the regulatory inputs for a specific gene. The regulatory effect of gene $g_j$ on gene $g_i$ at time $t$ is simply the expression level of $g_j$ multiplied by its regulatory influence on $g_i$, $w_{ij}$. The total regulatory input to $g_i$ is derived by summing across all the genes in the system and in the following referred to as $r_i(t)$:

$$r_i(t) = \sum_j w_{ij} x_j(t) \tag{1}$$

Here, a positive value for $w_{ij}$ indicates that gene $g_j$ is stimulating the expression of gene $g_i$. Similarly, a negative value indicates repression, while a value of zero indicates that gene $g_j$ does not influence the transcription of gene $g_i$. By modeling regulatory interactions with a weight matrix, we can use mathematical matrix approaches found in the field of neural networks for subsequent analyses of the resultant models.

With the regulatory state of each gene, we now are able to model the response of each gene to the given input. The impact of $r_i(t)$ on gene $g_i$ is calculated using a so called "squashing" function. The resulting expression level is only a relative value between 0 and 1, with 0 representing complete repression and 1 representing maximal expression. Thus, these relative levels have to be converted into the real expression space. In addition, the genes can have different levels of maximal expression. Hence, we multiply the calculated relative gene expression level $x_i$ by the maximal expression level for each gene $m_i$, to get the final expression level for $g_i$ $x_i(t+1)$:

$$x_i(t+1) = \frac{m_i}{1 + e^{-(\alpha_i r_i(t) + \beta_i)}} \tag{2}$$

where $r_i(t)$ is the mentioned regulatory state of gene $g_i$, and $\alpha_i$ and $\beta_i$ are gene specific constants that define the shape of the squashing function for gene $g_i$.

### B. S-systems

Another, more flexible type of model, are S-systems (SS). They employ a general formalism, which allow for capturing the nonlinearity and general dynamics of the gene regulation. S-systems are a type of power-law formalism, which have been suggested by [20] and can be described by a set of nonlinear differential equations:

$$\frac{dx_i(t)}{dt} = \alpha_i \prod_{j=1}^{N} x_j(t)^{\mathcal{G}_{i,j}} - \beta_i \prod_{j=1}^{N} x_j(t)^{\mathcal{H}_{i,j}} \tag{3}$$

where $\mathcal{G}_{i,j}$ and $\mathcal{H}_{i,j}$ are kinetic exponents, $\alpha_i$ and $\beta_i$ are positive rate constants and $N$ is the number of genes in the system. The equations in (3) can be seen as divided into two components: an excitatory and an inhibitory 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 $g_i$.

## III. Memetic Algorithm

Evolutionary Algorithms have proven to be a powerful tool for solving complex optimization problems. Four main types of Evolutionary Algorithms have evolved during the last 30 years: genetic algorithms, mainly developed by J.H. Holland [10], evolution strategies, developed by I. Rechenberg [19] and H.-P. Schwefel [21], Genetic Programming (GP) by J.R. Koza [16] and Evolutionary Programming by David Fogel *et al.* [7], [6]. Each of these uses different data representations and apply different main operators to them. They are, however, inspired by the same principles of natural evolution.

In the present work we use a combination of a global search for optimizing the topology together with a local search to find the best parameters for a given topology. This basic definition of a memetic algorithm (MA) differs in the implementation details as listed in the following.

### A. Global Search

In all implementations, a genetic algorithm evolves 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 thus of non-zero parameters in the mathematical model. The evaluation of the fitness of each individual within the global GA population uses directly the fitness values of the local search described in the following section.

### B. Local Search

To evaluate each structure suggested by the global optimizer an evolution strategy is used, which is well suited for optimizing problems based on real values. The ES optimizes the parameters of the mathematical model representing the regulatory network. For determining the fitness of each individual, the similarity of the time dynamics between the experimental and the simulated data (resulting from the parameters coded in the individuals) are evaluated. We use the following equation to calculate the fitness value, referred to as the relative squared error or relative standard error (RSE):

$$f_{RSE} = \sum_{i=1}^{N} \sum_{k=1}^{T} \left\{ \left( \frac{\hat{x}_i(t_k) - x_i(t_k)}{x_i(t_k)} \right)^2 \right\} \tag{4}$$

where $N$ is the total number of genes in the system, $T$ is the total number of sampling points taken from the experimental time series and $\hat{x}$ and $x$ distinguish between estimated data of the simulated model and data sampled in the experiment. Here, the models run from the initial conditions using the current model output as input for the next integration. The overall optimization problem is then to minimize the fitness values of objective function $f_{RSE}$ and thus maximize the similarity between the simulated model and the experiment. This fitness function is straight-forward and has already been used by several publications on this problem [23], [27].

### C. Feedback

We examined the performance of three different types of memetic algorithms. The first algorithm is a standard memetic method without any feedback from the local search as described in [24]. The other two implementations return the values for each system parameter from the local search to the global topology optimizer, which in turn is filtering all gene dependencies with a strength below a certain feedback threshold $t_{fb}$. One implementation is using a fixed threshold value $t_{fb}$ whereas the second feedback algorithm uses a parameter value $t_{fb}$ that itself is optimized during the inference process.

## IV. Experiment Settings

To compare the results with established inference methods we also used a standard GA, a standard ES, and an enhancement to a GA developed by Tominaga *et al.* [27] to infer the parameters of the network as described in the following. The following sections list the details and the settings for each optimization algorithm that was used in the comparison experiments. All settings for the evolutionary algorithms were determined in preliminary experiments. Overall, six different algorithms were used for comparison. To gain sound statistics, the experiment settings of the algorithms were repeated 20 times.

### A. Evolution Strategy - *ES*

The first algorithm was a standard ($\mu$,$\lambda$)-ES with $\mu = 10$ parents and $\lambda = 100$ offspring together with a Covariance Matrix Adaptation (CMA) mutation operator [9] without recombination. In case of the **ES**, the probabilities of crossover and mutation were chosen as $p_c = 0.0$ and $p_m = 1.0$. Overall, the ES evolved the individuals for $50,000$ generations.

*1) Genetic Algorithm - **GA**:* The second algorithm was a real-value encoding GA that used a population of $500$ individuals, a tournament selection strategy with tournament group size of $8$, a uniform-crossover-operator with a crossover probability of $p_c = 1.0$, and a mutation probability of $p_m = 0.1$. The decision variables were binary encoded and one-point mutation was applied to the genotype. Each GA optimization process evolved the individuals in the population for $10,000$ generations resulting in a total number of $5,000,000$ fitness evaluations per run.

### B. Skeletalizing Genetic Algorithm - *SkGA*

The third algorithm is an extension to a standard real-coded GA using tournament selection with a tournament group size of $t_{group} = 8$, uniform-crossover recombination with $p_c = 1.0$ and a mutation probability $p_m = 0.1$. This method was 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 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 $0.0$.

### C. Memetic Algorithm - *MA*

The main focus of this publication is the performance of the proposed memetic algorithms as presented in sect. III. The global GA evolved a population of $100$ possible structures with a tournament selection, a tournament group size of $t_{group} = 8$, uniform-crossover with $p_c = 1.0$ and a mutation probability $p_m = 0.1$. The local optimization was started using a ($\mu$,$\lambda$)-ES with $\mu = 2$ parents and $\lambda = 5$ offsprings together with a Covariance Matrix Adaptation (CMA) mutation operator without recombination as in the case of the standard ES. The number of total fitness evaluations were as for all algorithms $5,000,000$.

### D. Feedback Memetic Algorithm - *FMA*

For the feedback algorithms we used the same settings as for the MA. Further on, feedback algorithm FMA-I used a fixed threshold value $t_{fb} = 0.05$ as in case of the skeletalizing GA. As described in the previous section, FMA-II evolved this value during optimization with initial values for $t_{fb} \in [0.01, 0.2]$.

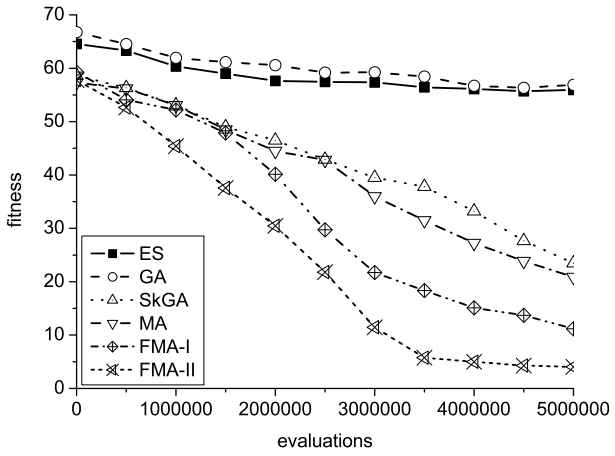


Fig. 1. Results of the inference of small-sized systems modeled with a weight matrix. Given are the fitness courses for each examined optimization algorithm: ES, GA, Skeletalizing GA, Memetic Algorithm, and the proposed Feedback MAs.

## V. Results

To evaluate the proposed methods we created several artificial gene regulatory networks for both model types, which were simulated to gain microarray expression data sets. These data sets were then reverse-engineered by our algorithms. The data sets represented two classes of N-dimensional artificial regulatory systems. The first test class were small-sized networks with a number of system components of $N \in [2, 5]$, i.e. there existed relationship between at most 5 genes. These relationships were randomly assigned and the resulting model simulated to gain data sets. The second class of example data were artificially created medium-sized GRNs, which consists of $N \in [6, 10]$ components. The regulatory models of both classes were sampled equidistantly with $T = 40$ time points.

### A. Small-sized Networks

Due to the fact that GRNs in nature are sparse systems [26], we randomly created 4 small-sized (($N \in [2, 5]$)) regulatory networks for each model type with a cardinality of $0 \leq k \leq 3$, i.e. each of the $N$ genes depends on three or less other genes within the network. With the 20 repetitions of the algorithms on each of these networks, we examined 80 experimental runs.

The compared 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 graphs (figure 1 for the weight matrices and figure 2 for the S-systems).

As can be seen, the standard optimizers ES and GA are not able to find a solution to the optimizing problem. Obviously, they both got stuck in local optima without being able to escape. This is reflected by the high averaged fitness values of $56.88$ and $57.99$, respectively. The skeletalizing GA was able to converge to relative good fitness values (23.93), which is comparable with the slightly better results obtained by the plain MA (22.08). The graphs of both suggest that they can achieve even better results with an increased number of fitness evaluations. In contrast to the other methods the proposed Feedback MAs converged quickly and reliably to very good fitness values (11.26 and $4.02$). The self-tuning FMA-II clearly outperforms the other algorithms. The results suggest that the ability to change the threshold value enables the method to adapt to each of the problem instances.

TABLE I

Averaged best fitness values for the small-sized inference problem after a total number of $5.000.000$ fitness evaluations.

| Model | WM | SS | Averaged |
|---|---|---|---|
| ES | 55.93 | 57.82 | **56.88** |
| GA | 56.8 | 59.1 | **57.99** |
| SkGA | 23.4 | 24.38 | **23.93** |
| MA | 20.8 | 23.29 | **22.08** |
| FMA-I | 11.16 | 11.36 | **11.26** |
| FMA-II | 4.05 | 3.98 | **4.02** |

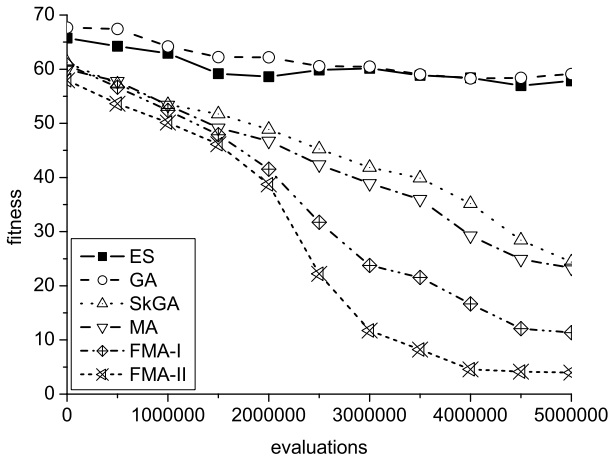


Fig. 2. Results of the inference of small-sized systems modeled with an S-system. Given are the fitness courses for each examined optimization algorithm: ES, GA, Skeletalizing GA, Memetic Algorithm, and the proposed Feedback MAs.

### B. Medium-sized Networks

As a second test class we created $4$ medium-sized ($N \in [6, 10]$) regulatory networks randomly with a cardinality of $0 \leq k \leq 3$. The optimization processes were performed as in the example before.

The compared algorithms were again used to infer the underlying regulatory dependencies with the parameter settings given in the previous sections. The resulting averaged fitness courses are plotted in the following graphs (figure 3 for the weight matrices and figure 4 for the S-systems).

As illustrated by the fitness plots (figures 3 and 4) the standard ES and GA were again not able to find a solution for the optimization problem. Again, both seemed unable to escape a local optimum and therefore no further improvement

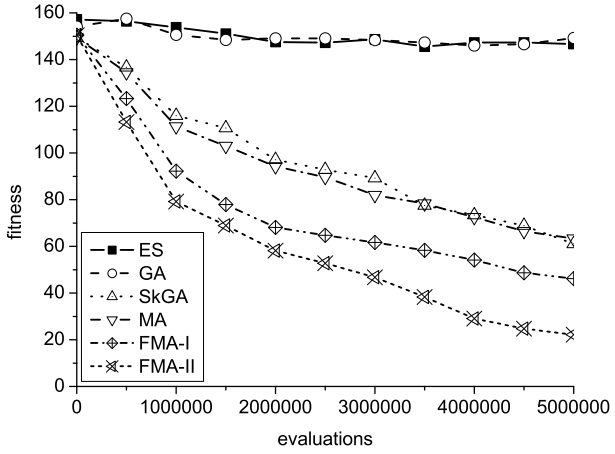


Fig. 3. Results of the inference of medium-sized systems modeled with a weight matrix. Given are the fitness courses for each examined optimization algorithm: ES, GA, Skeletalizing GA, Memetic Algorithm, and the proposed Feedback MAs.

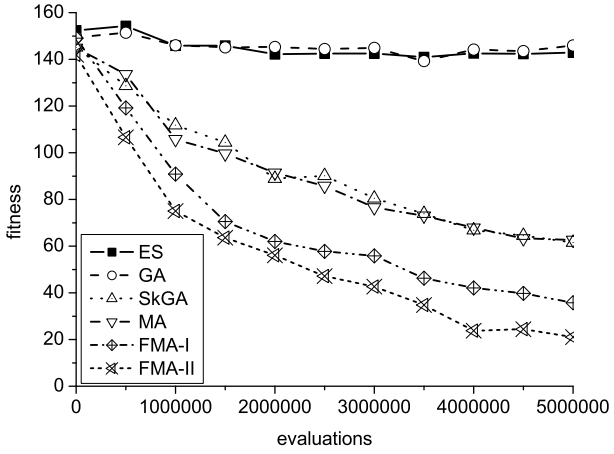


Fig. 4. Results of the inference of medium-sized systems modeled with an S-system. Given are the fitness courses for each examined optimization algorithm: ES, GA, Skeletalizing GA, Memetic Algorithm, and the proposed Feedback MAs.

TABLE II

AVERAGED BEST FITNESS VALUES FOR THE MEDIUM-SIZED INFERENCE PROBLEM AFTER A TOTAL NUMBER OF 5.000.000 FITNESS EVALUATIONS.

| Model | WM | SS | Averaged |
|---|---|---|---|
| ES | 146.67 | 142.90 | **144.79** |
| GA | 149.23 | 145.93 | **47.58** |
| SkGA | 61.00 | 61.54 | **61.27** |
| MA | 63.25 | 62.66 | **62.95** |
| FMA-I | 46.16 | 35.73 | **40.94** |
| FMA-II | 22.16 | 21.01 | **21.58** |

is achieved. The skeletalizing SkGA and the plain MA yielded both similar good results (61.27 and 62.95). But in the higher dimensional solution space, the SkGA performed slightly better than the MA. Again, both seem to have the potential to reach better fitness values with a larger number of fitness evaluations. Our proposed Feedback MAs outperformed the other methods by finding very good solutions with respect to the fitness values and with a high speed of convergence. Especially the self-adapting FMA-II resulted in the best fitness values.

### C. Ambiguity

Unfortunately, the resulting networks face a problem not captured by the fitness values. Looking closer at the results it becomes clear that good fitness values do not necessarily correspond to the correct network topology. Table III gives the absolute numbers of runs and the ratio of runs in which the correct network was found with respect to the topology and parameter values.

Neither the standard evolutionary algorithms (ES and GA) nor the MA found the correct solutions, with respect to the topology and parameter values, in a large number of the repeated optimizing runs. The skeletalizing GA found the correct target system in a slightly larger number of experiment runs. The Feedback MAs identified the largest number of correct solutions but the total number of correct networks found is still by far not sufficient. In the remaining optimization runs, systems were found, which fitted the experimental data but showed different relationships between the component genes.

As can be seen from table III, the ambiguity is a serious issue, which has to be considered in future publications on the inference problem. However, the issue has been addressed recently by only a few publications. Some approaches try to increase the diversity of the solution population to increase the probability of finding the correct network topology [17], [18]. To further identify the correct network in a set of possible solutions, approaches like an iterative search, where additional experiments enables the algorithm to find the correct solution, multi-objective methods [25] or methods that incorporate biological information [1], [2], [13] have recently been published.

## VI. CONCLUSION

In this paper we compared different optimization methods to infer gene regulatory networks from time-series microarray data. We showed that memetic algorithms are superior to standard optimization approaches found in the literature. The comparison to our previous paper showed significant differences in the performance of the algorithms. The reason therefore is the high impact of the network properties. The performance of the methods differ on each of the problems and it is thus necessary to average over several network structures.

In case of the weight matrices, the standard ES and GA were not able to find good solutions to the inference problem. Both resulted in similar fitness values for the weight matrices and the S-systems. Further on, they were able to find models that fit the given data in only a small fraction of the test instances. This is likely due to the fact that they were trapped

TABLE III

AMBIGUOUS RESULTS OF THE INFERENCE. GIVEN ARE THE TOTAL NUMBER OF RUNS AND THE RATIO OF RUNS IN WHICH THE CORRECT NETWORK WAS FOUND WITH RESPECT TO THE TRUE TOPOLOGY AND THE CORRESPONDING PARAMETER VALUES.

| Size | small | | medium | | **total** |
|---|---|---|---|---|---|
| Model | WM | SS | WM | SS | |
| ES | 2/80 (2.5%) | 1/80 (1.25%) | 1/80 (1.25%) | 0/80 (0%) | **4/320 (1.25%)** |
| GA | 1/80 (1.25%) | 0/80 (0%) | 1/80 (2.5%) | 0/80 (0%) | **2/320 (0.63%)** |
| SkGA | 2/80 (2.5%) | 1/80 (1.25%) | 3/80 (3.75%) | 1/80 (1.25%) | **7/320 (2.19%)** |
| MA | 2/80 (2.5%) | 1/80 (1.25%) | 2/80 (2.5%) | 1/80 (1.25%) | **6/320 (1.88%)** |
| FMA-I | 4/80 (5%) | 2/80 (2.5%) | 4/80 (5%) | 2/80 (2.5%) | **12/320 (3.75%)** |
| FMA-II | 4/80 (5%) | 3/80 (3.75%) | 5/80 (6.25%) | 3/80 (3.75%) | **19/320 (5.93%)** |

in local optima in most of the test cases. The skeletalizing algorithm proposed by Tominaga *et al.* was 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. This algorithm performed better than the standard optimization techniques and found the correct solution in more test problems. Furthermore, it resulted in sparse matrices, which is biologically more plausible than fully connected networks. Our plain memetic algorithm (MA) yielded better fitness values as the SkGA, but the resulting network models were often fully connected and thus showed huge differences to the true solution. This can also be seen in the small number of correct identified models. The proposed feedback algorithm FMA-I performed better than the skeletalizing algorithm with respect to the fitness values and showed slightly better identification rates. 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. And finally, the self-tuning feedback algorithm FMA-II showed the best fitness values as well as the best identification rates. The ability to fine-tune the parameter value of $t_{fb}$ enabled the algorithm to adapt to the network models and thus yielded the best solutions.

The MAs proved to work even in medium-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 $\leq 10$ 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 thus makes it possible to infer larger systems. Future experiments on high performance computers will address large-scale systems with 50 genes or more.

Furthermore, the solutions found by the MAs and the SkGA are sparse due to the preceding structure optimization. Because in nature GRNs are sparse systems the solutions of those algorithms represent better resemblance to biological systems than the standard optimizer, which always resulted 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 underdetermined. Therefore, multiple solutions exist, which fit the given data, but show only little resemblance to the original target system. In future work we plan to incorporate additional methods to identify the correct network as suggested in section V-C.

Further on, we plan to include a-priori information into the inference process, like partially known pathways or information about co-regulated genes, which can be found in literature or public databases. Additionally, other models for gene regulatory networks will be examined for simulation of the nonlinear interaction system to overcome the problems with those gene regulatory networks that hardly can be modeled by S-systems. To further reduce the dimensionality of the data set, we plan to use cluster methods. Recently, we developed a new method that incorporates biological ontologies [22] and thus yields biologically more plausible clustering results.

## ACKNOWLEDGEMENT

This work was supported by the National Genome Research Network (NGFN) of the Federal Ministry of Education and Research in Germany under contract number 0313323.

## REFERENCES

[1] Hong-Chu Chen, Hsiao-Ching Lee, Tsai-Yun Lin, Wen-Hsiung Li, and Bor-Sen Chen. Quantitative characterization of the transcriptional regulatory network in the yeast cell cycle. *Bioinformatics*, 20(12):1914–1927, 2004.

[2] Kuang-Chi Chen, Tse-Yi Wang, Huei-Hun Tseng, Chi-Ying F. Huang, and Cheng-Yan Kao. A stochastic differential equation model for quantifying transcriptional regulatory network in Saccharomyces cerevisiae. *Bioinformatics*, 21(12):2883–2890, 2005.

[3] Ting Chen, Hongyu L. He, and George M. Church. Modeling gene expression with differential equations. In *Proceedings of the Pacific Symposium on Biocomputing*, 1999.

[4] Michiel J.L. de Hoon, Seiya Imoto, Kazuo Kobayashi, Naotake Ogasawara, and Satoru Miyano. Inferring gene regulatory networks from time-ordered gene expression data of Bacillus Subtilis using differential equations. In *Proceedings of the Pacific Symposium on Biocomputing*, volume 8, pages 17–28, 2003.

[5] Hidde de Jong. Modeling and simulation of genetic regulatory systems: A literature review. *Journal of Computational Biology*, 9(1):67–103, January 2002.

[6] David B. Fogel. *Evolving Artificial Intelligence*. Phd thesis, University of California, 1992.

[7] L.J. Fogel, A.J. Owens, and M.J. Walsh. *Artificial Intelligence through Simulated Evolution*. John Wiley & Sons, 1966.

[8] Nir Friedman, Michal Linial, Iftach Nachman, and Dana Pe'er. Using Bayesian networks to analyze expression data. *Journal of Computational Biology*, 7:601?620, 2000.

[9] Nikolaus Hansen and Andreas Ostermeier. Adapting arbitrary normal mutation distributions in evolution strategies: the covariance matrix adaptation. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 312–317, 1996.

[10] John H. Holland. *Adaption in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Systems*. University Press of Michigan, 1975.

[11] Dirk Husmeier. Sensitivity and specificity of inferring genetic regulatory interactions from microarray experiments with dynamic Bayesian networks. *Bioinformatics*, 19(17):2271–2282, 2003.

[12] S. Imoto, T. Goto, and S. Miyano. Estimation of genetic networks and functional structures between genes by using Bayesian networks and nonparametric regression. In *Proceedings of the Pacific Symposium on Biocomputing*, pages 175–186, 2002.

[13] Phil Hyoun Kao and Doheon Lee. Modularized learning of genetic interaction networks from biological annotations and mRNA expression data. *Bioinformatics*, 21(11):406, 2005.

[14] Shinichi Kikuchi, Daisuke Tominaga, Masanori Arita, Katsutoshi Takahashi, and Masaru Tomita. Dynamic modeling of genetic networks using genetic algorithm and S-system. *Bioinformatics*, 19(5):643–650, 2003.

[15] Shuhei Kimura, Kaori Ide, Aiko Kashihara, Makoto Kano, Mariko Hatakeyama, Ryoji Masui, Noriko Nakagawa, Shigeyuki Yokoyama, Seiki Kuramitsu, and Akihiko Konagaya. Inference of S-system models of genetic networks using a cooperative coevolutionary algorithm. *Bioinformatics*, 21(7):1154–1163, 2005.

[16] John R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, 1992.

[17] Ryohei Morishita, Hiroaki Imade, Isao Ono, Norihiko Ono, and Masahiro Okamoto. Finding multiple solutions based on an evolutionary algorithm for inference of genetic networks by S-system. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 603–612, 2003.

[18] Isao Ono, Ryohei Yoshiaki Seike, Norihiko Ono, and Masahiko Matsui. An evolutionary algorithm taking account of mutual interactions among substances for inference of genetic networks. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 2060–2067, 2004.

[19] Ingo Rechenberg. *Evolutionsstrategie - Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Frommann-Holzboog, 1973.

[20] Michael A. Savageau. 20 years of S-systems. In E.O. Voit, editor, *Canonical Nonlinear Modeling. S-systems Approach to Understand Complexity*, pages 1–44, 1991.

[21] Hans-Paul Schwefel. *Numerical optimization of computer models*. John Wiley & Sons, 1981.

[22] Nora Speer, Christian Spieth, and Andreas Zell. A memetic clustering algorithm for the functional partition of genes based on the gene ontology. In *Proceedings of the IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology*, pages 252–259, 2004.

[23] Christian Spieth, Felix Streichert, Nora Speer, and Andreas Zell. Iteratively inferring gene regulatory networks with virtual knockout experiments. In *Proceedings of the European Workshop on Evolutionary Bioinformatics*, volume 3005 of *Lecture Notes in Computer Science*, pages 102–111, 2004.

[24] Christian Spieth, Felix Streichert, Nora Speer, and Andreas Zell. A memetic inference method for gene regulatory networks based on S-systems. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 152–157, 2004.

[25] Christian Spieth, Felix Streichert, Nora Speer, and Andreas Zell. Multi-objective model optimization for inferring gene regulatory networks. In Eckart Zitzler Carlos A. Coello Coello, Arturo Hernndez Aguirre, editor, *Proceedings of the Conference on Evolutionary Multi-Criterion Optimization*, volume 3410 of *Lecture Notes in Computer Science*, pages 607–620, 2005.

[26] Denis Thieffry, A. Huerta, E. Perez-Rueda, and J. Collado-Vides. From specific gene regulation to genomic networks: a global analysis of transcriptional regulation in escherichia coli. *BioEssays*, 20:433–440, 1998.

[27] Daisuke Tominaga, Nobuto Kog, and Masahiro Okamoto. Efficient numeral optimization technique based on genetic algorithm for inverse problem. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 251–258, 2000.

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

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

[30] Michael Q. Zhang. Large-scale gene expression data analysis: A new challenge to computational biologists. *Genome Research*, 9(8):681–688, 1999.