Clustering-based Approach to Identify Solutions for the Inference of Regulatory Networks

Christian Spieth, Felix Streichert, Nora Speer, and Andreas Zell
Centre for Bioinformatics Tübingen (ZBIT), University of Tübingen,
Sand 1, D-72076 Tübingen, Germany
spieth@informatik.uni-tuebingen.de

Abstract- In this paper we address the problem of finding valid solutions for the problem of inferring gene regulatory networks. Different approaches to directly infer the dependencies of gene regulatory networks by identifying parameters of mathematical models can be found in literature. The problem of reconstructing regulatory systems from experimental data is often multimodal and thus appropriate optimization strategies become necessary. Thus, we propose to use a clustering based niching evolutionary algorithm to maintain diversity in the optimization population to prevent premature convergence and to raise the probability of finding the global optimum by identifying multiple alternative networks. With this set of alternatives, the identification of the true solution has then to be addressed in a second post-processing step.

1 INTRODUCTION

With the description of complete genome sequences, DNA microarray technology has become a powerful tool for genome-wide expression profiling and analysis [31]. It allows the simultaneous examination of thousands of genes in a single experiment and thus the cellular dynamics under various environments. The analysis of such behavior is one of the key elements in the functional analysis of the genome. A large amount of knowledge on various biological systems, e.g. gene regulation, metabolic regulations, and signal transduction are being continually accumulated over the years, though there remains a large portion that is not well understood.

The construction of a network model, which can describe the behavior of the biochemical system, is an important but very difficult task addressed in recent bioinformatics. The purpose of such a gene regulatory network (GRN) is to represent the rules of regulation defining the gene expression. It is regarded as an abstract mapping of the more complicated biochemical network, which includes other components such as proteins, metabolites, etc. The knowledge of the genetic network may then be used as the guidance for further biological experiments to explore higher level of interaction. Fig. 1 shows an example of such a regulatory system.

In the literature, several mathematical models can be found that address the problem of analyzing gene regulation. A good overview of related work can be found in [4, 3]. The models used to simulate GRNs are divided into two major classes, stochastic and deterministic models. In stochastic models, e.g. Bayesian networks, the dependencies between the components of a system are modeled by

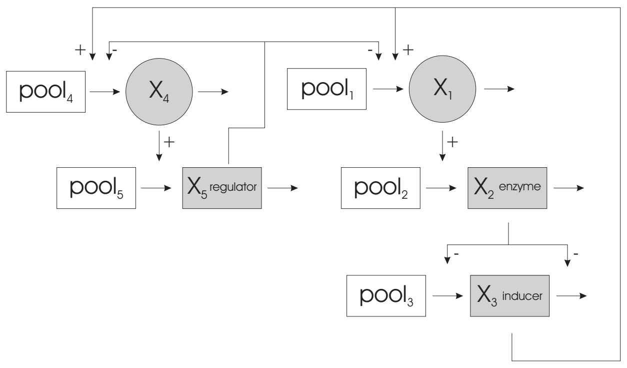

Figure 1: Genetic network introduced in [9]. This gene network consists of two genes (gene 1 and gene 4). X_1 and X_4 are mRNA concentrations produced by gene 1 and 4, respectively. X_2 is an enzyme translated from X_1 and X_3 is an inducer protein catalyzed by X_2. X_5 is a regulator protein translated from X_4. X_3 and X_5 are assumed to supress/activate the mRNA transcription of genes 1 and 4. The five components of our analysis are indicated in gray.

probabilistic transition values. There are many publications on this kind of model, [11, 6, 12]. The second class models these dependencies in a deterministic manner. Several deterministic models have been examined over the last years such as linear weight matrices introduced in [30, 16, 17]. Other deterministic models include S-systems, which consist of a set of differential equations describing the changes in expression over time. S-Systems have been examined in [13, 15, 28] but also by others. Arbitrary differential equations can be used to model regulatory structures as showed with Genetic Programming (GP) in [21, 1].

The analysis of gene expression using evolutionary algorithms (EAs) is a frequently used approach and common to the methods mentioned above. EAs have proven to be successful in identifying parameters of mathematical models representing GRNs. Their general principles and application will be described in the following sections.

The mathematical modeling of regulatory systems raises two problems:

1. Solutions that appear optimal under the objective function, but which do not correspond to the true model. These exist, because the system is underdetermined.

2. Suboptimal solutions, to which the optimization methods may converge.

The first mentioned problem results from the ambiguity in the experimental data. Due to the high number of network components in contrast to the small number of experimental samples, the overall problem is highly under-determined. Hence, there exist several network structures that satisfy the given time course dynamics of the experiment. This issue has already been addressed by several publications, one example can be found in [25], where the problem is resolved by exploiting additional data in the inference process. A similar approach has been proposed in [18].

But these strategies remove the cause of the previously mentioned problem of ambiguity of under-determined systems rather than offering a solution to it. Therefore, the present paper addresses the second problem and focusses mainly on constructing multiple alternative genetic networks automatically from gene expression data. These alternatives can then be evaluated either by an expert or by further processing with the algorithms mentioned above. Further on, biological knowledge can be used to determine the correct network by incorporating additional constraints into the optimization process [26].

Standard methods using optimization techniques like EAs suffer from the fact that they quickly focus on a single valid solution that satisfies the given experimental data or are not able to find any valid solution. Thus, the true system is found only in some of the inference processes. Therefore, finding multiple solutions, which all similarly comply with the experimental constraints, is the first step in finding the true solution. The second step is then to identify the correct solution in the list of possible network structures gained by step one. In this work, we suggest to use niching EAs to address the first step. Using this type of algorithm, we are able to show that niching algorithms reliably meet the requirements of step one by generating a limited set of alternative solutions, which contains the true system with a high probability.

The remainder of this paper is structured as follows. Section 2 describes the proposed algorithm and the mathematical model used in the optimization process. The results are listed in section 3 and the conclusions and an outlook are given in section 4.

2 METHOD

On an abstract level, the behavior of a cell is represented by a directed graph with N nodes representing N genes. Each gene g_i produces a certain amount of RNA x_i when expressed and changes the concentration of the RNA 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 the vector of expression levels from one state to the next.

2.1 S-Systems

As already mentioned, there are several approaches to mathematically model a regulatory system. S-Systems are one possibility. They employ a general formalism, which allows for capturing the non-linearity and general dynamics of the gene regulation. S-Systems are a type of power-law formalism suggested by [23], and can be described by a set of nonlinear differential equations:

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

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 Eqn. 1 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.

Since the modeling of genetic networks involves abstraction and mapping of interactions, the generality of this formula is essential. However, they suffer from the high number of model parameters ($2N^2 + 2N$). And although high performance computing has made significant progress over the last decade, this is still a major issue for larger biological systems.

2.2 Linear Weight Matrices

To overcome this problem, network models can be used that do not rely on that high numbers of parameters. One model of this class are linear weight matrices [30]. 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 $r_i(t)$ (see Eqn. 2).

$$r_i(t) = \sum_j w_{ij} x_j(t) \qquad (2)$$

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 are now 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 (Eqn. 3).

$$x_i(t+1) = \frac{m_i}{1 + e^{-(\alpha_i r_i(t) + \beta_i)}} \qquad (3)$$

where $r_i(t)$ is the mentioned regulatory state of gene g_i, and α_i and β_i are gene specific constants that define the shape of the squashing function for gene g_i. 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 $x_i(t+1)$ as shown in equation (3).

2.3 Evolutionary Algorithms

Evolutionary algorithms are stochastic optimization techniques that mimic the natural evolution process of repeated mutation and selection as proposed by Charles Darwin. They have proven to be a powerful tool for solving complex optimization problems and in particular combinatorial problems. Three main types of evolutionary algorithms have been proposed in the last decades: Genetic Algorithms (GA), mainly developed by J.H. Holland [10], Evolution Strategies (ES), developed by I. Rechenberg [20] and H.-P. Schwefel [24], and Genetic Programming (GP) by J.R. Koza [14]. Each of these uses different solution representations and different operators working on them.

The following sections list the details and the settings for each optimization algorithm that was used in the comparison experiments. Overall, four different algorithms were used for comparison. The experiment settings of the algorithms, which are described in the following, were repeated 20 times to gain sound statistics.

2.3.1 Multi-Start Hill-Climber (MS-HC)

As a first algorithm, we examined the performance of a standard hill climber method on the inference problem. MS-HCs represent a rather primitive technique but they can become useful in simple and low dimensional multi-modal solution spaces. The MS-HC performs several local hill climbing searches in parallel, where each one is randomly initialized. In the present work, we used a 10-start hill climber (**10S-HC**)with a real-value variable encoding and a fixed mutation step size with a total number of $1,000,000$ fitness evaluations per run.

2.3.2 Genetic Algorithm (GA)

The second algorithm was a real-value encoding GA that used a population of 200 individuals, tournament selection strategy with a tournament group size of 8 and 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 $5,000$ generations resulting in a total number of $1,000,000$ fitness evaluations per run.

Additionally, we used a multi-start GA to directly compare the results to those of the proposed method. This 10-start GA (**10S-GA**) was run with 50 individuals over $2,000$ generations to have the same total number of fitness evaluations per run.

2.3.3 Evolution Strategies (ES)

The third algorithm was a standard (μ,λ)-ES with $\mu = 10$ parents and $\lambda = 100$ offspring together with a Covariance Matrix Adaptation (CMA) mutation operator [8] 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 $10,000$ generations resulting in the same number of total fitness evaluations per run as the GA to allow the results to be compared.

As in case of the GA, a multi-start ES was also tested. We used a 10-start (5,20)-ES (**10S-ES**) with the same settings as described above but with a total number of $5,000$ generations.

The parameters of the GA and the ES have been used in previous experiments and especially the CMA mutation operator has been found extremely useful for the inference problem. Unfortunately, most niching techniques are not suited for this type of optimization problem. For example sequential niching by Beasely *et al.* [2] results in pseudo optima, which can only be identified by an additional expensive post-processing step. Fitness sharing [7], on the other hand, alters the search space dynamically and changes the fitness values by adding something similar to noise. But this renders the CMA ineffective. An alternative niching technique that leaves the search space unchanged is the clustering based niching EA, which is described in the following.

2.3.4 Clustering Based Niching EA (CBN)

This method is designed to identify multiple global and local optima in a multi-modal solution space. The basic idea of CBN-EA is to transfer the biological concept of non-interbreeding species living in separated ecological niches into evolutionary algorithms to preserve diversity in the optimization population. Thus, the CBN-EA searches for clusters of EA individuals in the solution space by means of clustering algorithms. These clusters can then be separated into isolated sub-populations. Individuals of different sub-populations are not allowed to interbreed, while within a sub-population ordinary evolutionary processes occur. Additional mechanisms to split sub-populations, if necessary, enable dynamic specialization, and the merging of sub-populations that become too similar enforces only one sub-population per niche.

The methods of choice to track multiple global/local optima in a multi-modal search space are either so called multi-start approaches or niching techniques for Evolutionary Algorithms. Unfortunately, the performance of multi-start approaches like the parallel LC algorithm (multiple local searches with clustering) [29] or the sequential niching approach [2] strongly depend on the search method applied and may even introduce deceptive local optima as in case of the sequential niching approach. Evolutionary niching techniques on the other hand often tend to alter the search space and to disrupt the evolutionary optimization process. For example the well established fitness sharing technique [7] may prevent convergence below the level of the sharing distance σ_{share} in the search space.

The Clustering Based Niching EA (CBN-EA) joins evolutionary approaches with the multi-start approaches by using clustering methods to identify sub-populations converging to global/local optima in an evolutionary run and then separating each into independent evolutionary optimization processes. This approach has two main advantages over most alternative evolutionary techniques. First, the CBN-EA approach does not interfere with the general optimization procedure and does not alter the search space, thus sophisticated search operators, like for example an ES with CMA mutation, can be applied without any additional precautions. Secondly, with additional techniques of population state management a re-initialization the CBN-EA returns a compact list of global/local optima based on a halting window criterion on each sub-population and it is also able to track arbitrary numbers of global/local optima.

```
S_0 = createInitialPop();
R = emptyList();
while isNotTerminated() do
    // species evolution phase;
    foreach $S_i$ do simulateEAGeneration(S_i);
    // species differentiation phase;
    if numOfClusters(S_0) $\geq$ 1 then split(S_0);
    foreach $S_{i\neq0}$ do
        if numOfClusters(S_i) $\geq$ 1 then split(S_i);
        S_0.add(S_i.getLoners());
    end
    // species convergence phase;
    TLP = createEmptyPop();
    foreach $S_{i\neq0}$ do TLP.addCentroidOf(S_i);
    foreach $R_i$ do TLP.add(R_i);
    if numOfClusters(TLP) $\geq$ 1 then mergeSpecies();
    // population state management;
    foreach $S_{i\neq0}$ do
        if $S_i$.isConverged() then
            R.add(S_i.getBest());
            S_i.reinitialize();
            S_0.add(S_i);
            S_i.remove();
        end
    end
end
```

Algorithm 1: Pseudo-code of the CBN-EA procedure with species evolution, differentiation, convergence and population state management phases.

The general CBN-EA procedure is given in Alg. 1. After the initialization of the undifferentiated sub-population S_0 and the initialization of an empty result list, the generational cycle of CBN-EA is entered, which has four distinct phases. First, the *species evolution phase*, where the sub-populations pass through one or multiple standard EA generation cycles. Secondly, the *species differentiation phase*. In this phase a clustering method is used to decide whether or not specialized sub-populations are to be created and whether some individuals (referred to by the authors as *straying loners*) should belong to the undifferentiated sub-population S_0. Then, the *species convergence phase* is entered, where it is decided whether or not multiple sub-populations converge on the same niche and are thus to be merged. And finally, the *population state management phase*, which allows to balance the sub-population size if necessary, tracks the state of convergence for each sub-population and allows re-initialization of converged sub-populations while memorizing the obtained result in the list of identified global/local optima R. The final result of the CBN-EA is then the list of identified global/local optima R without any further post-processing. A more detailed description of the algorithm and its implementation can be found in publications from the authors [27].

The CBN-EA proposed in this publication is based on an ES (**CBN-ES**). To reduce the computational complexity of the optimization process, we used a halting window of $h = 15$ generations without improvement to determine a sub-population to be converged. The solution of a converged sub-population is then memorized and the individuals are randomly re-initialized. *DBScan* was used for clustering of the individuals of the optimization population, which allows for arbitrary numbers of clusters. This clustering algorithm identifies clusters by connecting individuals if the distance $\|x_i - x_j\|$ between them is lower than a given threshold value σ_{dist}. All interconnected groups of individuals, whose group size exceeds a minimum value $MinPts$ are identified as cluster. The 'density-based' clustering algorithm offers several advantages: first, it allows clusters of varying size and shape. Secondly, it can identify clusters of a priori unknown number. Thirdly, the algorithm allows for loners, which do not belong to any species and finally, it requires only two parameters that are easy to interpret. Details of the clustering algorithm can be found in [22, 5].

In the present implementation, we used a maximum distance to differentiate between clusters $\sigma_{\text{dist}} = 0.1$ and the minimum number of individuals required to form a cluster $MinPts = 3$. To be comparable to the other algorithms, the CBN-ES optimized 200 individuals over $5,000$ generations using also a total number of $1,000,000$ fitness evaluations per run.

2.4 Fitness

For evaluating the fitness of the individuals, i.e. the similarity of the time dynamics between the experimental and the simulated data resulting from the parameters coded in the individuals, we used the following equation for calculation of the fitness value, referred to as the relative squared error or relative standard error (RSE).

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

where N is the total number of genes in the system, T is the 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. The overall optimization problem is then to minimize the fitness values of objective function f_{RSE}. This fitness function has already been used by several publications on this problem.

3 RESULTS

Our approach was tested on artificial gene regulatory networks with a total number of genes of $5 \leq N \leq 20$ using the two mathematical models described in section 2, namely linear weight matrices and S-Systems. To test the proposed method, we created artificial microarray data sets with randomly created models. These data sets were then used to reverse engineer the underlying regulatory system by the compared algorithms. Because GRNs are sparse systems in nature, we created regulatory networks randomly with a maximum cardinality of $k \leq 3$, i.e. each of the N genes depends on three or less other genes within the network.

To evaluate the different algorithms, we counted the number of different solutions found in the inference process and the number of hits, i.e. the number of times the algorithm was able to find the true system. A solution was counted as a hit if the euclidian distance between the parameters of the evolved model and the true system was smaller than a threshold $d_{hit} = 1.0$. Typically, the standard GA and ES return only one solution. In case of the multi-start experiments, we used the same *DBScan* clustering method as a postprocessing step to identify unique solutions. The CBN-ES on the other hand only proposes converged subpopulations as potential solutions.

3.1 5-dimensional example

As a first example, we used a randomly created linear weight matrix and an S-System with a total number of system components of $N = 5$. An example for such a regulatory network is given with the template for the topology of the 5-dimensional network shown in Fig. 1. Each of these artificial regulatory systems was then simulated to gain data sets and inferred by the four different algorithms.

Table 1 gives the statistics for each model and optimization algorithm. Listed in the table are the overall number of different network solutions found, the number of runs in which the true system has been found, the ratio of hits, the total number of runs, and the fitness value averaged over the best individuals.

As can be seen, the standard evolutionary algorithms are able to find the correct solution in about 20% to 25% of the runs. In the remaining runs, both the GA and the ES get stuck in local optima indicated by the low number of hits. The multi-start hill climber performs equally to the standard evolutionary algorithms, whereas the MS-GA and the MS-ES perform relatively well with hit ratios of 35% to 40%. The CBN-ES on the other hand is able to find the true system in 80% and 75% of the runs, respectively. This is also represented by very good fitness values in respect to the fitness function given in equation 4. The averaged fitness for the MS-GA and the MS-ES are comparably high because some of the resulting models yielded very high fitness values.

Further on, it can be seen that in case of the S-Systems, generally more possible solutions are found by the algorithms. This can be explained by the flexibility of the S-System, which allows this model to represent the given dynamics with different network solutions and thus with different genetic interactions. This behavior illustrates the already mentioned problem of ambiguity.

3.2 10-dimensional example

For the second experiment setting, we used randomly created networks with a total number of system components of $N = 10$ for both model types, weight matrix as well as S-System. As in the 5-dimensional example before, each algorithm was repeated 20 times to gain a sound multi-run statistic.

The results of the inference process is given in Tab. 2 for each of the model and optimization algorithm, respectively. As in the results of the 5-dimensional example, the CBN-ES outperforms standard GA and ES, the multi-start EA, and the multi-start hill climber. The proposed method identifies the correct solution in around 55% to 60% of the runs. The multi-start EAs are again superior to the standard methods, as can be seen from both the hit ratio and the averaged fitness values.

3.3 20-dimensional example

For the last example for evaluating the inference algorithms, we used an artificial system with $N = 20$ components.

The statistics for the 20-dimensional regulatory network are given in Tab. 3. As expected, the CBN-ES again performs better than all of the other algorithms, finding more solutions with overall better fitness as indicated by the average fitness value. Due to the higher dimension, the CBN-ES finds the true system in 35.0% and 20.0% of the optimization runs, respectively. The standard algorithms find the true system in only 5.0% to 15.0% of the experimental runs.

4 DISCUSSION

The following sections summarize the clustering based approach and its results and give an outlook on future work.

4.1 Conclusions

The problem of inferring GRNs is a very difficult process due to the limited data available and the large number of unknown variables in the system. One of the problems found in the literature is that conventional methods repeatedly run into local optima, thus not necessarily being able to find the

Table 1: Inference results for the 5-dimensional regulatory net using linear weight matrices and S-Systems. The table gives the overall number of different network solutions found in the optimization process. The next two columns give the number of runs in which the true system has been found and the ratio of hits and number of runs. Additionally, the averaged fitness value of the best individuals of all runs is given.

Model	Algorithm	Solutions	Hits	Hit ratio	Average fitness
Weight Matrix (40 parameters)	GA	1	5	25.0%	3.976
	ES	1	4	20.0%	3.451
	10S-HC	3	3	15.0%	4.187
	10S-GA	6	8	40.0%	2.111
	10S-ES	4	7	35.0%	2.454
	CBN-ES	**8**	**16**	**80.0**%	0.124
S-System (60 parameters)	GA	1	4	20.0%	3.351
	ES	1	4	20.0%	3.342
	10S-HC	7	2	10.0%	4.571
	10S-GA	11	8	40.0%	2.158
	10S-ES	10	6	30.0%	2.591
	CBN-ES	**15**	**15**	**75.0**%	0.152

Table 2: Inference results for the 10-dimensional regulatory net using linear weight matrices and S-Systems. The table gives the overall number of different network solutions found, the number of runs in which the true system has been found, the ratio of hits, the number of runs, and the averaged fitness value.

Model	Algorithm	Solutions	Hits	Hit ratio	Average fitness
Weight Matrix (130 parameters)	GA	1	4	20.0%	5.883
	ES	1	5	25.0%	5.231
	10S-HC	4	2	10.0%	6.510
	10S-GA	6	6	30.0%	4.851
	10S-ES	5	7	35.0%	4.678
	CBN-ES	**11**	**12**	**60.0**%	0.899
S-System (220 parameters)	GA	1	3	15.0%	7.212
	ES	1	3	15.0%	7.266
	10S-HC	7	3	15.0%	8.045
	10S-GA	13	5	25.0%	5.004
	10S-ES	12	5	25.0%	5.055
	CBN-ES	**19**	**11**	**55.0**%	0.822

Table 3: Inference results for the 20-dimensional regulatory net using linear weight matrices and S-Systems. The table gives the overall number of different network solutions found, the number of runs in which the true system has been found, the ratio of hits, the number of runs, and the averaged fitness value.

Model	Algorithm	Solutions	Hits	Hit ratio	Average fitness
Weight Matrix (460 parameters)	GA	1	3	15.0%	10.483
	ES	1	2	10.0%	11.311
	10S-HC	3	1	5.0%	15.160
	10S-GA	9	6	30.0%	8.463
	10S-ES	10	7	35.0%	8.666
	CBN-ES	**12**	**7**	**35.0**%	4.083
S-System (840 parameters)	GA	1	2	10.0%	11.212
	ES	1	1	5.0%	12.266
	10S-HC	7	1	5.0%	14.143
	10S-GA	13	2	10.0%	10.600
	10S-ES	12	2	10.0%	10.805
	CBN-ES	**26**	**4**	**20.0**%	5.689

optimal solution. Therefore, we introduced an algorithm that increases the probability of finding the correct regulatory network by inferring experimental microarray data in this paper. We showed that standard evolutionary algorithms suffer from the problem of finding solutions within the solution space that comply with the data but do not resemble the original system. The proposed cluster-based niching algorithm efficiently preserves the diversity of network candidates in the optimization process and results in multiple alternative solutions, thus addressing the first problem mentioned in the introduction. Moreover, the algorithm was often able to find the correct network in the multi-modal search space.

We showed that the CBN-EA was able to find better solutions with respect to the fitness than the standard methods independent of the mathematical model used for the simulation of the regulatory network. Further on, the GA based inference algorithm performed slightly better than the one with an ES implementation. This is most likely because GAs have the advantage of a larger initial population size and thus likely having a better coverage of the solution space. However, this issue has to be addressed in future work to be verified. Both standard EAs outperform the naive hill climber due to the multi-modal nature of the solution space. One potential reason for the superior performance of the CBN-EA relative to other algorithms is that if an individual has converged it is then re-initialized and thus more solutions can be found. This halting-window strategy can be implemented for GA and ES as well and will be examined in a future publication. Another possibility to overcome the problem of multi-start EAs to explore the same optima several times is an extension called clearing procedure, which was introduced by Petrowski in [19].

The proposed method yielded more than one valid network solution and although this seems like a disadvantage on the first glimpse, only this enables us to actually find the correct solution in an under-determined and ambiguous environment. To overcome the resulting problem of picking the true solution from the list of valid solutions, one has to use the CBN-EA together with other techniques to clearly identify the overall optimal network model. This postprocessing step can either be done by incorporating additional microarray data sets to decrease the ambiguity in the data or by using a priori known biological or biochemical constraints to eliminate some of the solutions found during optimization. Furthermore, biologists are able to select from the list of the resulting network solutions to exclude some solutions that are not biologically plausible.

4.2 Outlook

In future work, we will exploit the ability of CBN-EA algorithms to result in better solutions by combining them with other enhancements of the inferring process. For example, iterative methods [25] can be used to iteratively identify the correct regulatory network model by incorporating additional microarray data sets. Further on, enhancements to the CBN-EA will be implemented, which introduce an upper limit for the cluster size. With this, the CBN-EA should be able to perform even better, because single basins of attraction cannot hog the majority of the individuals.

Furthermore, we will continue to test our method with real microarray data in close collaboration with biological researchers at our facility. In future work we plan to use real microarray data sets and 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 in public databases.

Additionally, other models for gene regulatory networks will be examined for simulation of the non-linear interaction system as listed in Section 1 to overcome the problems of deterministic models used to infer stochastic processes like intracellular signalling.

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.

Bibliography

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

[2] D. Beasley, D. Bull, and R. Martin. A sequential niche technique for multimodal function optimization. *Evolutionary Computation*, 1(2):101–125, 1993.

[3] D. Corne and C. Pridgeon. Investigating issues in the reconstructability of genetic regulatory networks. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 582–589, 2004.

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

[5] M. Ester, H.-P. Kriegel, J. Sander, and X. Xiaowei. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, pages 226–231, 1996.

[6] N. Friedman, M. Linial, I. Nachman, and D. Pe'er. Using Bayesian networks to analyze expression data. In *Proceedings of the Conference on Research in Computational Molecular Biology*, pages 127–135, 2000.

[7] D. E. Goldberg and J. Richardson. Genetic algorithms with sharing for multimodal function optimization. In *Proceedings of the International Conference on Genetic Algorithms*, pages 41–49, 1987.

[8] N. Hansen and A. 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.

[9] W. Hlavacek and M. Savageau. Rules for coupled expression of regulator and effector genes in inducible circuits. *Journal of Molecular Biology*, 255:121–139, 1996.

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

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

[15] 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.

[16] T. Moriyama, A. Shinohara., M. Takeda, O. Maruyama, T. Goto, S. Miyano, and S. Kuhara. A system to find genetic networks using weighted network model. In *Genome Informatics*, volume 10, pages 186–195, 1999.

[17] K. Noda, A. Shinohara, M. Takeda, S. Matsumoto, S. Miyano, and S. Ku. Finding genetic network from experiments by weighted network model. *Genome Informatics*, 9:141–150, 1998.

[18] I. Ono, R. Yoshiaki Seike, N. Ono, and M. 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] A. Petrowski. A clearing procedure as a niching method for genetic algorithms. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 798–803, 1996.

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

[21] E. Sakamoto and H. Iba. Inferring a system of differential equations for a gene regulatory network by using genetic programming. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 720–726, 2001.

[22] J. Sander, M. Ester, H.-P. Kriegel, and X. Xiaowei. Density-based clustering in spatial databases, the algorithm GDBSCAN and its applications. *Data Mining and Knowledge Discovery*, 2(2):169–194, 1998.

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

[24] H.-P. Schwefel. *Numerical optimization of computer models*. John Wiley & Sons, 1981.

[25] C. Spieth, F. Streichert, N. Speer, and A. 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.

[26] C. Spieth, F. Streichert, N. Speer, and A. Zell. Multiobjective model optimization for inferring gene regulatory networks. In E. Z. 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.

[27] F. Streichert, G. Stein, H. Ulmer, and A. Zell. A clustering based niching ea for multimodal search spaces. In *Proceedings of the International Conference Evolution Artificielle*, volume 2936 of *Lecture Notes in Computer Science*, pages 293–304, 2003.

[28] D. Tominaga, N. Kog, and M. 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.

[29] A. Törn. Cluster analysis using seed points and density-determined hyperspheres as an aid to global optimization. *IEEE Transactions on Systems, Man and Cybernetics*, 7:610–616, 1977.

[30] 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.

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