Generality of results obtained by the parameter calibration and relevance estimation method

Volker Nannen[1] and A.E. Eiben[2]

[1] Vrije Universiteit Amsterdam, volker@cs.vu.nl
[2] Vrije Universiteit Amsterdam, gusz@cs.vu.nl

Abstract. We present and evaluate a method for parameter selection and calibration for evolutionary systems. The general goal of the method is to help distinguishing between relevant and irrelevant algorithm parameters and to select good values for the relevant ones. We apply the method to a number of different problem instances (abstract fitness landscapes) and test how well the settings generated by our method generalize. The parameter settings obtained for problem A are then tested on problems B, C, etc.

1 Introduction

We have previously [1] introduced the Calibration and Relevance Estimation (CRE) method, a numerical method that estimates the relevance of the different parameters of an evolutionary algorithm, in the process calibrating these parameters in a robust way.

The objectives of the CRE method are:

1. To estimate the relevance of each parameter, i.e., how sensitive the performance of the evolutionary algorithm (EA) is to the exact calibration of each parameter.
2. To remove parameters that are irrelevant.
3. To calibrate the relevant parameters of an EA, i.e., to find good values for them.

The CRE method combines information theory with a particular Estimation of Density Algorithm (EDA) that calibrates the parameters of an evolutionary algorithm. An EDA manipulates distributions over parameter values and the CRE method uses the Shannon entropy of these distributions to measure the relevance of each parameter. By maximizing the Shannon entropy of the EDA the CRE method can estimate the minimum amount of information that is necessary to calibrate the parameters. The result is a quantitative measure of the minimum amount of calibration needed for an algorithm to reach a certain performance level, and how this information is distributed over the different parameters.

The rationale behind this search for simple calibrations is that the simpler the calibration, the broader the set of problems it can be applied to with success. That is, we expect that a simpler calibration can be more easily adapted to a new problem, and we expect that a simpler calibration is more robust against changes to the target application. This principle has long been known as Occam's Razor and has been put on sound mathematical footing by the theory of algorithmic complexity, formulated simultaneously by R. Solomonoff [2], A. Kolmogorov [3] and G. Chaitin [4].

The CRE method was motivated by the complexity of the original application, namely the simulation of agent behavior in a dynamic economic environment, and it was tailored to that application. In this paper we present a cleaner version of the method and test it on a set of abstract problems that we consider as a generalization of the application in our previous work. In particular, we test how robust the results of the CRE method are, i.e., how useful the calibrated parameters are when the target application is changed.

Related Work. Eiben e.a. established parameter control in EAs as an important research question [5]. A practical method to find good parameter settings for an EA was introduced by François and Lavergne in [6]. They use numerical simulations to estimate the functional relationship between

parameter values and the performance of the EA, both for a single test case and for multiple test cases (generalization)..

The groundwork of statistical experimental design was laid by R.A. Fisher in the 1920s and 1930s. Response surfaces methods that exploit sequential sampling were introduced in Box and Wilson [7]. A paradigm shift that emphasizes parameter robustness is due to G. Taguchi [8].

Information theory and Shannon entropy were introduced by C. Shannon [9]. K. Kolmogorov [3] developed algorithmic complexity as an uncomputable, yet precise, measure for parameter robustness. The relation between Shannon entropy and algorithmic complexity is discussed by Grünwald and Vitanyi [10].

2 The CRE Method

As discussed in the introduction, the main objective of the calibration and relevance estimation method (CRE) is to find a set of relevant parameters, to find good values for these parameters, and to establish how accurate these values have to be. One could say that we are after a good evolutionary system that achieves a high performance not only in one particular test case, but in as many other test cases as possible. Since we assume that this depends largely on the algorithmic complexity of the model, we define "good" as having a good tradeoff between performance and algorithmic complexity. Thus, the quality of any given list of parameters and a specific vector of values x for these parameters is determined by the performance $\mathcal{F}(x)$ obtained by running experiments using x, and the information or algorithmic complexity $\mathcal{K}(x)$ necessary to specify x. We start the formalization of this matter by defining the term *calibration* as a distribution over parameter vectors.

Calibration. Let $M = \langle \theta_1, \ldots, \theta_k \rangle$ be a list of k parameters with an initial finite[3] domain of possible values for each parameter θ_i and let $\mathcal{X}_M$ stand for the Cartesian product of these domains. Then a *calibration* is a distribution $\mathcal{C}(x)$ over all possible parameter settings $x \in \mathcal{X}_M$. We define $\mathcal{C}_0$ to be the uniform distribution over $\mathcal{X}_M$. $\mathcal{C}^\theta$ is be the marginal distribution of $\mathcal{C}(x)$ on parameter θ.

Calibration Performance. We can now define *calibration performance* $\mathcal{F}(\mathcal{C})$ as the expected performance when drawing the parameter settings from the calibration distribution $\mathcal{C}$

$$\mathcal{F}(\mathcal{C}) = \mathop{\mathrm{E}}_{x \in \mathcal{X}_M} \left[\mathcal{C}(x) \mathcal{F}(x) \right]. \tag{1}$$

Calibration Complexity. Algorithmic complexity as understood by A.N. Kolmogorov [3] is the minimum amount of information needed to compute a vector of values x. While algorithmic complexity itself is not computable, it is sufficient for our purpose to estimate it from the Shannon entropy[4] $H(\mathcal{C})$ of the distribution $\mathcal{C}(x)$. The probability that the true algorithmic complexity $\mathcal{K}(x)$ of any x chosen by the distribution $\mathcal{C}(x)$ is significantly lower than the Shannon entropy of $\mathcal{C}(x)$ is exponentially low [10] and can be neglected. We define *calibration complexity* as

$$\mathcal{K}(\mathcal{C}) = H_0 - H(\mathcal{C}). \tag{2}$$

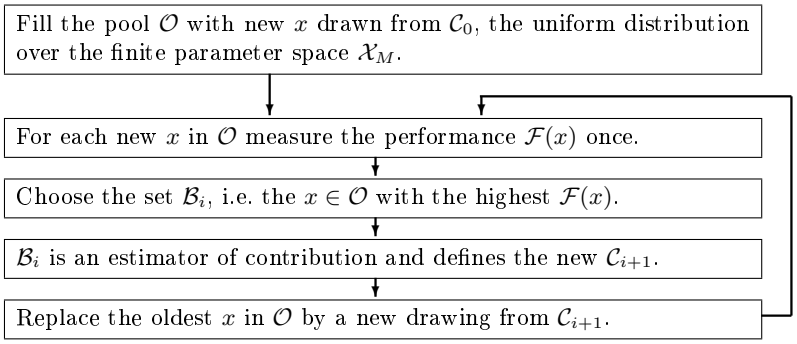
where H_0 is the average Shannon entropy of calibrations that the CRE produces when fed with white noise. This will be explained shortly. Calibration complexity estimates the amount of additional information needed to calibrate a set of parameters, given that we have already specified a finite parameter space $\mathcal{X}_M$. We use binary bits as the unit of Shannon entropy and calibration complexity.

Minimax Calibration. Not all possible calibrations are equally interesting to us. Of all calibrations that achieve a given performance we are only interested in those of the lowest complexity.

[3] If the initial domain is $\mathbb{R}$ we transform it by e.g. the sigmoid to make it finite.

[4] Differential Shannon entropy is defined as $H(\mathcal{C}) = -\int_x \mathcal{C}(x) \log \mathcal{C}(x)$.

Fig. 1. Diagram of the search method



And of all calibrations of a given complexity we are only interested in those that achieve the highest performance. While we assume that a calibration that satisfies both constraints generally does not exist, it is essential that the CRE method comes reasonably close to producing minimax calibrations because these are the only calibrations of any practical interest. We loosely define an approximate minimax calibration as a calibration $\mathcal{C}_i$ that has the following properties: no calibration $\mathcal{C}_j$ with performance $\mathcal{F}(\mathcal{C}_j) = \mathcal{F}(\mathcal{C}_i)$ is *significantly* less complex (i.e. no $\mathcal{K}(\mathcal{C}_j) \ll \mathcal{K}(\mathcal{C}_i)$) and no calibration $\mathcal{C}_k$ with complexity $\mathcal{K}(\mathcal{C}_k) = \mathcal{K}_(\mathcal{C}_i)$ can perform *significantly* better (i.e. no $\mathcal{F}(\mathcal{C}_k) \gg \mathcal{F}(\mathcal{C}_i)$).

Conditional Marginal Contribution. At the heart of the CRE method lies a reliable estimation of conditional marginal contribution of parameter values to performance. By this we mean the contribution of the values of a specific parameter to performance (hence *marginal*) if the distribution of values over the other parameters are fixed by a specific calibration (hence *conditional*).

The Search Process. We use an iterative Estimation of Density Algorithm (EDA) that produces a series of approximate minimax calibrations of increasing complexity and performance. Two features are peculiar to this EDA: the density over the domains is constructed from overlapping uniform distributions, and Shannon entropy is increased by letting more elementary uniform distributions overlap.

As shown in figure 1, we work with a pool $\mathcal{O}$ of p different parameter settings x. These are initially drawn from $\mathcal{C}_0$, the uniform distribution over the finite parameter space $\mathcal{X}$. At each iteration i we replace the oldest x from $\mathcal{O}$ by a new x drawn from the latest calibration $\mathcal{C}_i$, calculated below. Each x is applied once to the test case and we measure the performance $\mathcal{F}(x)$. We then rank all $x \in \mathcal{O}$ according to performance and select the best q settings $\mathcal{B}_i$.

The density of $\mathcal{B}_i$ is a good estimator of the conditional marginal contribution of the parameters to performance: the higher the density over a certain range of a parameter, the higher the marginal contribution of values from that range to performance, conditioned on the fact that the values for the other parameters were chosen from the current and previous calibrations. We use the Shannon entropy of this density to estimate parameter relevance. Figure 2 shows how the density of $\mathcal{C}(x)$ over two parameters changes during a search. The Shannon entropy of the density of $\mathcal{B}_i$ decreases fast for the parameter that specifies the probability to innovate. There is no significant reduction in Shannon entropy for the parameter that specifies the probability to rewire.

We use a two step process to generate the next distribution $\mathcal{C}_{i+1}$ from the density of $\mathcal{B}_i$. When drawing a new x from $\mathcal{C}_{i+1}$, we first draw a random member $y \in \mathcal{B}_i$ for each parameter $\theta \in M$. Next we determine the parameter range that is formed by the members $\in \mathcal{B}_i$ that are the h^{th} closest upper and lower neighbors of y along the parameter θ. We now determine the value of the parameter for the new x by drawing a random value from this range.

When defining the order h of the upper and lower neighbors of y along θ, it is absolutely essential that we use a value greater than 1. A value greater than 1 smoothes the density function so that a slightly higher probability is given to parameter ranges where the density of $\mathcal{B}_i$ is lower.

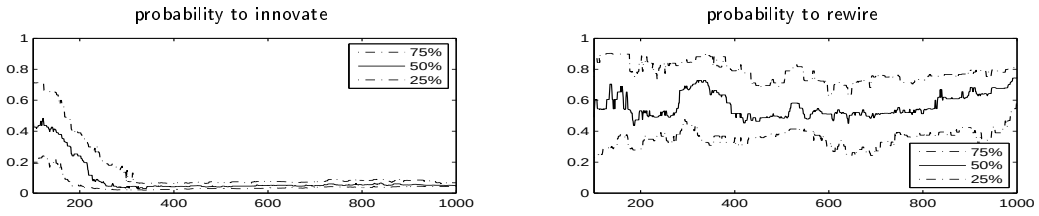


Fig. 2. Example of how the density of $\mathcal{B}$ changes during the search. The x-axis shows the number of evaluations that the calibration is based on, the y-axis shows the parameter ranges. The solid line in the middle shows the median of the respective distribution. The dashed lines show the 25^{th} and the 75^{th} percentile. The graphs are taken from the test case with $m = 10$ and $s = 0$.

This minimizes the Shannon entropy of $\mathcal{C}$ and ensures that it is reasonably close to being a minimax calibration.

To give an example, let us draw a new value for the chance to innovate. We first draw a random $y \in \mathcal{B}_i$, say with value 0.959. Assume the 5^{th} closest neighbors $\in \mathcal{B}_i$ with regard to the innovation rate have values 0.945 and 0.963. The final parameter value x for the innovation rate will now be drawn from the uniform distribution [0.935–0.963].

Calculating Complexity. When measuring the calibration complexity, we need to correct for the effects of noise. Due to the random nature of the algorithm, the Shannon entropy of the density of $\mathcal{B}_i$ is always lower than the Shannon entropy of the uniform distribution, even when the contribution to performance is uniform over all parameters. We can measure this trivial decrease in Shannon entropy by replacing the performance measurements from actual test cases by white noise and we found that for every parameter θ the resulting Shannon entropy H^θ is typically 1 bit lower than the Shannon entropy of the uniform distribution over θ. Since we are only interested to know how much the Shannon entropy of $\mathcal{C}_i$ differs from this trivial entropy level, we calculate the calibration complexity of each parameter θ as $\mathcal{K}(\mathcal{C}^\theta) = H^\theta - H(\mathcal{C}^\theta)$. And since the distribution of $\mathcal{C}_i(x)$ is constructed form the marginal distributions $\mathcal{C}_i^\theta(x)$, which therefore contain all necessary information, we have $\mathcal{K}(\mathcal{C}_i) = \sum_\theta \mathcal{K}(\mathcal{C}_i^\theta)$.

Interpretation of the Results. As the search progresses, $\mathcal{K}(\mathcal{C})$ and $\mathcal{F}(\mathcal{C})$ increase continuously, but not at equal rates. $\mathcal{K}(\mathcal{C})$ increases with a much more constant rate than $\mathcal{F}(\mathcal{C})$. The latter usually increases in one, maybe two sharp rises, after which no significant improvements are made. Even so, $\mathcal{K}(\mathcal{C})$ might still increase by some 25%. We can compare the calibration of different sets of parameters and plot the minimum complexity needed for any given performance. If calibration achieves equivalent performance with two different sets of parameters, the set that achieves this performance with less complexity will probably be of a more general value. We are also interested in the Shannon entropy per parameter. The lower this Shannon entropy, the more information is needed to calibrate the respective parameter. When applying the EA to a new problem, such a parameter is more likely to be miss-calibrated than a low Shannon entropy one and consequently deserves more attention from the practitioner.

The 25^{th} and 75^{th} percentile of the calibration distribution on each parameter are what we consider the practical calibration of the CRE method: the values from this range have the highest conditional marginal contribution. As long as the parameters of our EA stay within this range we are confident of a high performance.

3 The Evolutionary Algorithm

We asses the robustness of results obtained from the CRE method by calibrating an evolutionary algorithm from agent-based economics where economic strategies evolve through innovation and imitation in a social network. To avoid the complexity of an economic environment that the

agents have to adapt to, we use an enhanced version of the Multimodal Problem Generator[5] by W. Spears [11] with two dimensions of hardness typical for economic problems: the landscape ruggedness and the rate of change of this landscape. Ruggedness is measured by the number m of peaks, represented by random binary strings. Rate of change is represented by the probability s with which each bit of these binary strings flip at each cycle of the simulation. For each hardness we want to optimize the mean best fitness after 200 cycles of the simulation. We use 100 agents.

We use a discrete synchronous time model for the simulation where each cycle is divided into four steps:

1. *moving the peaks* — by bit-flip mutation of the strings that define peaks
2. *calculating the fitness* — multiplying the Hamming distance of a strategy to the nearest peak with the predefined height of that peak.
3. *updating all strategies* — by innovation and imitation
4. *updating the network* — agents can rewire their connections

Representation, Individuals and Population. It is important to note that in this EA agents and strategies are not the same: an agent carries or maintains a strategy, but it can change its strategy and we still consider it as the same agent. This dichotomy is necessary so that we can maintain a social network (among the *agents*), while evolving, i.e., changing, the strategies. Because every agent has exactly one strategy at a time, the number of active strategies is the same as the number of agents. The population size (in the search space of strategies) is thus constant and equals the number of agents.

Social Structure. The agents of our simulation are located in a complex bidirectional dynamic network. This network is generated by a stochastic growth process prior to each run of the simulation and can be changed by the agents during the simulation. Based on current theory on social networks we ensure that this network has random connectivity [12], a high cluster coefficient [13] and a scale free degree distribution [14], actually a gamma distribution with an exponent between 2 and 3. The average connectivity is a free parameter of the evolutionary algorithm and varies between 2 and 10.

Fitness and Selection. An agent is evaluated according to the strategy it carries. It is essential to our model that the fitness of a strategy is determined by the performance of its hosting agent *relative to that of its peers in the social network*. This implies that the structure of the social network has a direct effect on the definition of fitness, hence on the selection mechanism.

We introduce three probabilistic selection mechanisms: one to decide whether a given strategy will be changed by random innovation, one to decide whether it will be changed by imitation (recombining it with the strategy of a peer in the network), and one to decide whether an agent changes its position in the social peer network (rewiring). These choices are independent from the fitness of the agent and are parametrized by three scalars $\in (0, 1)$ that specify the respective probabilities.

Innovation in our simulation is implemented by random bit-flip mutation, introducing one additional parameter that controls the probability for each bit to flip. In the case of imitation the agent needs to select a peer to be imitated, i.e., its strategy to be partially adopted. A parameter is needed that controls the fraction of best performing peers to imitate from. Imitation is performed by replacing one or more binary values of the the strategy of the imitating agent by values from the strategy of the imitated agent, implemented by uniform crossover. The resulting strategy replaces the strategy of the imitating agent, the strategy of the imitated agent remains the same. This introduces one additional parameter to control the fraction of binary values that are replaced.

For rewiring we need to select the peer from which the agent disconnects, and a new agent to which the agent connects (the total number of connections is constant). This calls for two more

[5] consisting of a search space of $\{0, 1\}^n$ and a number of peaks, which are randomly chosen binary target strings. The fitness of a solution is its Hamming distance to the nearest target string, multiplied by the height associated with that target string.

parameters: the fraction of worst performing peers from which the agent randomly chooses the one to disconnect from and the fraction of best performing agents (of the total population) from which the agent can randomly choose a new peer.

Together with average connectivity, this makes 9 parameters that have to be calibrated. They are shown in the first column of Table 1.

4 Experiments

Our strategy space consists of binary strings of length $n = 100$. We use three different values $m = \{1, 10, 1000\}$ and three different values $s = \{0, 0.001, 0.01\}$. With $s = 0$ the problem is static. With $s = 0.001$ a target flips a bit about once every ten cycles of the simulation and with $s = 0.01$ a target flips about one bit every cycle. These combinations allow for 9 different test cases.

For the CRE method we use a pool size of $p = 100$, from which we select the $q = 50$ as possible parents for the next setting. To increase entropy we define the uniform distributions over $h = 5$ neighboring values when drawing a new setting x. We use a total of 1000 evaluations, 100 of which are used to evaluate the first filling of the pool, and 900 for the iterative search. A calibration based on 1000 evaluations leads to a significantly improved performance for all test cases.

After we have calibrated the parameters on all test cases we apply the final result $\mathcal{C}^i$ of each test case i to the other 9 test cases (including itself). We do so by uniformly drawing x 20 times from what we consider the practical calibration: the range formed by the 25^{th} and 75^{th} percentile of $\mathcal{C}^i$. We measure the resulting performance $F_{i\to j}$ and compare it to the native performance $F_{j\to j}$, the performance obtained by applying the $\mathcal{C}^j$ from test case j on itself.

To compare $F_{i\to j}$ and $F_{j\to j}$ we first substract from both $F_{i\to j}$ and $F_{j\to j}$ the default performance $F_{0\to j}$, i.e., the performance obtained by applying the default calibration $\mathcal{C}_0$ to test case j. Next we divide the resulting $F'_{i\to j}$ by $F'_{j\to j}$ and express this fraction in percent. If $\mathcal{C}^i$ generalizes well to test case j, the result will be close to 100%, occasionally even higher. If, on the other hand, the comparison produces a negative result, $\mathcal{C}^i$ should not be applied to test case j at all since even the default calibration $\mathcal{C}_0$ does better.

Results for Relevance Estimation. After we have calibrated the parameters on all test cases we compare the Shannon entropy that is measured for each parameter on each test case. Table 1 shows the results after 1000 evaluations. For each parameter the Shannon entropy measured over all 9 test cases is compared: the minimum Shannon entropy is shown, the maximum, the mean, and the standard deviation. There is much agreement between different test cases. The Shannon entropy is either uniformly low or height for almost all test cases. That means that if performance is highly sensitive to parameter θ in one test case, we can be almost certain that performance is highly sensitive to this parameter in most other test cases. And if performance is insensitive to a parameter in one test case, it will most likely be insensitive to that parameter in most other test cases.

This is particularly surprising in the case of the rewiring parameters. Our working assumption was that network structures improve or at least influence the performance of an EA on dynamic problems. However, not even in the case of the most dynamic test cases ($s = 0.1$) do we see real sensitivity. We conclude that these parameters are simply not relevant to the EA.

Results for Generalization Performance. Table 2 compares the generalization performance of the calibrations after 1000 evaluations.

All experiments were programmed and analyzed in Matlab. The code and additional information are online available at `http://www.cs.vu.nl/~volker/CRE`

5 Conclusion

We presented a method for numerical parameter calibration and relevance estimation that is based on robust parameter estimation. We conclude that results produced by the method on one partic-

Table 1. Comparing the Shannon entropy per parameter at the end of the search. Results are corrected for noise.

parameter of the EA	min. entropy	max. entropy	mean entropy	std. deviation
probability to innovate	-3.71	-0.43	-1.68	0.94
probability of bits to flip when innovating	-5.62	0.19	2.80	1.92
probability to imitate	-3.11	-1.05	-2.09	0.68
fraction of best perform. peers to imitate from	-2.20	-0.06	-1.12	0.69
fraction of bits to imitate	-2.91	-0.82	-2.01	0.74
average connectivity	-2.78	-0.09	-1.31	0.98
probability to rewire (relocate)	**-0.68**	**0.42**	**-0.11**	**0.33**
fraction of best perform. agents to connect to	**-0.72**	**0.27**	**-0.34**	**0.33**
fraction of worst perform. peers to disconnect from	**-0.90**	**0.37**	**-0.09**	**0.37**

Table 2. Comparing the generalization performance $F_{i \to j}$ with the native performance $F_{j \to j}$. m is the number of peaks. s is the rate of change of a peak. Values around 100 indicate excellent generalization between cases. Values lower than zero indicate incompatibility.

			i								
	m		1	10	1000	1	10	1000	1	10	1000
		s	0	0	0	0.001	0.001	0.001	0.01	0.01	0.01
j	1	0	**100**	100	100	100	100	100	100	100	100
	10	0	100	**100**	100	100	100	100	100	100	100
	1000	0	101	101	**100**	102	100	100	100	100	104
	1	0.001	100	100	100	**100**	100	100	100	100	100
	10	0.001	98	88	91	92	**100**	97	99	98	100
	1000	0.001	103	98	97	96	101	**100**	103	102	106
	1	0.01	87	69	69	70	96	79	**100**	98	107
	10	0.01	83	68	67	77	96	80	102	**100**	106
	1000	0.01	70	58	61	56	85	59	94	94	**100**
	mean		96	90	90	91	98	93	100	100	102

ular problem instance are robust against significant changes to the problem hardness. This holds for both the parameter relevance estimation as the parameter calibration.

References

1. Nannen, V., Eiben, A.E.: A Method for Parameter Calibration and Relevance Estimation in Evolutionary Algorithms. Genetic and Evolutionary Computation Conference (GECCO) (2006) to appear.
2. Solomonoff, R.: A formal theory of inductive inference, part 1 and part 2. Information and Control **7** (1964) 1–22, 224–254
3. Kolmogorov, A.N.: Three approaches to the quantitative definition of information. Problems of Information Transmission **1** (1965) 1–7
4. Chaitin, G.J.: On the length of programs for computing finite binary sequences. J. Assoc. Comput. Mach. **13** (1966) 547–569
5. Eiben, A.E., Hinterding, R., Michalewicz, Z.: Parameter control in evolutionary algorithms. IEEE Transactions on Evolutionary Computation **3**(2) (1999) 124–141
6. François, O., Lavergne, C.: Design of Evolutionary Algorithms—A Statistical Perspective. IEEE Transactions on Evolutionary Computation **5**(2) (2001) 129–148
7. Box, G.E.P., Wilson, K.G.: On the Experimental Attainment of Optimum Conditions. Journal of the Royal Statistical Society, Series B (Methodological) **13**(1) (1951) 1–45
8. Taguchi, G., Wu, Y.: Introdution to Off-Line Quality Control. Central Japan Quality Control Association, Nagoya, Japan (1980)

9. Shannon, C.E.: A mathematical theory of communication. Bell System Technical Journal **27** (1948) 379–423, 623–656
10. Grünwald, P., Vitányi, P.: Shannon Information and Kolmogorov Complexity. arXiv:cs.IT/0410002 (2004)
11. Spears, W.M.: The Role of Mutation and Recombination. Springer, Berlin, Heidelberg, New York (2000)
12. Erdős, P., Rényi, A.: On random graphs. Publicationes Mathematicae **6** (1959) 290–297
13. Watts, D.J., Strogatz, S.H.: Collective dynamics of 'small-world' networks. Nature **373** (1998) 440–442
14. Barabási, A.L., Albert, R.: Emergence of Scaling in Random Networks. Science **286** (1999) 509–511