FDA – A scalable evolutionary algorithm for the optimization of additively decomposed functions

Heinz Mühlenbein
Theoretical Foundation GMD Lab.
Real World Computing Partnership
GMD FZ Informationstechnik
53754 St. Augustin
muehlenbein@gmd.de

Thilo Mahnig
Theoretical Foundation GMD Lab.
Real World Computing Partnership
GMD FZ Informationstechnik
53754 St. Augustin
mahnig@gmd.de

Abstract

FDA - the Factorized Distribution Algorithm - is an evolutionary algorithm which combines mutation and recombination by using a distribution instead. The distribution is estimated from a set of selected points. In general a discrete distribution defined for n binary variables has 2^n parameters. Therefore it is too expensive to compute. For additively decomposed discrete functions (ADFs) there exist algorithms which factor the distribution into conditional and marginal distributions. This factorization is used by FDA. The scaling of FDA is investigated theoretically and numerically. The scaling depends on the ADF structure and the specific assignment of function values. Difficult functions on a chain or a tree structure are solved in about $O(n\sqrt{n})$ operations. More standard genetic algorithms are not able to optimize these functions. FDA is not restricted to exact factorizations. It also works for approximate factorizations as is shown for a circle and a grid structure. By using results from Bayes networks, FDA is extended to LFDA. LFDA computes an approximate factorization using only the data, not the ADF structure. The scaling of LFDA is compared to the scaling of FDA.

Keywords

Genetic algorithms, Boltzmann distribution, simulated annealing, Bayes network, learning of Bayes networks, convergence, factorization of distributions.

1 Introduction

Numerically the deficiencies of genetic algorithms using Mendelian string based recombination methods have been first demonstrated with a simple class of fitness functions, called deceptive functions of order k. They are defined as a sum of more elementary deceptive functions f_k of k variables (Goldberg et al. 1993).

$$f(\mathbf{x}) = \sum_{j=1}^{l} f_k(\boldsymbol{s}_j), \tag{1}$$

where $\boldsymbol{s}_j$ are non-overlapping substrings of $\mathbf{x}$ containing k elements.

In a deceptive function the global optimum $x = (1, \ldots, 1)$ is isolated, whereas the neighbors of the second best fitness value $x = (0, \ldots, 0)$ have large fitness values. Genetic algorithms (GAs) are "deceived" by the fitness distribution. Most GAs will converge to $x = (0, \ldots, 0)$.

Deceptive functions are separable. They are trivial to optimize by mathematical methods. In this paper we consider functions which are additively decomposed (ADF),

but they need not be separable. This class of functions is of great theoretical and practical importance. Optimization of an arbitrary function in this space is NP complete.

A number of new evolutionary algorithms have been proposed which optimize ADFs better than genetic algorithms. These algorithms try to detect and exploit the structure of an ADF. The methods used can be classified as follows:

- Adaptive recombination
- Explicit detection of relations (Kargupta & Goldberg, 1997)
- Dependency trees (Baluja & Davies, 1997)
- Bivariate marginal distributions (Pelikan & Mühlenbein, 1998)
- Estimation of distributions (Mühlenbein & Paaß(1996), De Bonet et al., (1997), Harik (1999), Pelikan et al. (1999))

Adaptive recombination uses a number of heuristics to modify two-parent recombination. Kargupta's (1997) Gene Expression Messy Genetic Algorithm (GEMGA) tries to detect dependency relations by manipulating individual substrings.

The last three methods are based on probability theory and statistics. They use the statistical information contained in the population of selected points to detect dependencies. In this paper an algorithm called the Factorized Distribution Algorithm (FDA) will be investigated. FDA uses a factorization of the distribution of selected points. For Boltzmann distributions FDA is based on a solid mathematical foundation. Many results can be derived by mathematical analysis. Therefore this paper is a mixture between theoretical analysis and numerical experiments. The experiments are mainly used to confirm the theoretical analysis.

The outline of the paper is as follows. In Sections 2 and 3 some basic theorems about factorization and FDA are cited. Then FDA is analyzed for large (infinite) populations. A comparison is made between Boltzmann selection and truncation selection. In Section 5 finite populations are investigated. The concept of critical population size is introduced. Numerical results for an ADF test suite on simple regular ADFs are presented in Section 6. The problem of computing a factorization of an ADF with unknown structure is discussed in Section 7 with LFDA.

2 Factorization Theorem

In this section we recall the main results proven in Mühlenbein et al. (1999a). We use the notation most common in statistics. Large symbols denote variables, small symbols assignments, bold symbols are vectors. x_s denotes the sub vector of $\boldsymbol{X}$ with indices from s.

Definition: An *additively decomposed function* (ADF) is defined by

$$f(\mathbf{x}) = \sum_{s_i \in S} f_i(x_{s_i}) \qquad S = \{s_1, \ldots, s_l\} \quad s_i \subseteq \boldsymbol{X}. \tag{2}$$

Next we define a search distribution. For theoretical analysis we will use a generalization of a Gibbs or Boltzmann distribution.

Definition: *The Gibbs or Boltzmann distribution of a function f is defined for* $u \geq 1$ *by*

$$p(\mathbf{x}) := \frac{\mathrm{Exp}_u f(\mathbf{x})}{F_u} \tag{3}$$

where for notational convenience

$$\mathrm{Exp}_u f(\mathbf{x}) := u^{f(\mathbf{x})} \quad F_u := \sum_{\mathbf{y}} \mathrm{Exp}_u f(\mathbf{y})$$

Remark: The Boltzmann distribution is usually defined as $e^{-\frac{g(\mathbf{x})}{T}}/Z$. The term $g(\mathbf{x})$ is called the energy. Setting $g(\mathbf{x}) = -f(\mathbf{x})$ and $u = e^{\frac{1}{T}}$ gives Equation 3. $Z = F_u$ is called the *partition function.*

The Boltzmann distribution has the following property: the larger the function value $f(\mathbf{x})$, the larger $p(\mathbf{x})$ (for $u > 1$). Such a search distribution is obviously suitable for an optimization problem. Unfortunately the computation of the Boltzmann distribution needs an exponential effort (in the size of the problem). There are at least two approaches to reduce the computation: to approximate the Boltzmann distribution or to look for ADFs where the distribution can be computed in polynomial time. The first approach is used by *Simulated Annealing* (Aarts et al., 1997). FDA is based on the second approach. The distribution is factored into a product of marginal and conditional probabilities. They are defined for $b_i, c_i \subseteq \boldsymbol{X}$

$$p(x_{c_i}) = \sum_{\substack{\mathbf{y} \\ y_{c_i} = x_{c_i}}} p(\mathbf{y}) \tag{4}$$

$$p(x_{b_i} | x_{c_i}) = \frac{p(x_{b_i}, x_{c_i})}{p(x_{c_i})} \tag{5}$$

The basic factorization theorem uses the following sequence of sets as input.

Definition: *Given a set of sets* $S = \{s_1, \ldots, s_l\}$*, we define for* $i = 1, 2, \ldots, l$ *sets* d_i,b_i *and* c_i

$$d_i := \bigcup_{j=1}^{i} s_j \tag{6}$$

$$b_i := s_i \setminus d_{i-1} \tag{7}$$

$$c_i := s_i \cap d_{i-1} \tag{8}$$

We set $d_0 = \emptyset$.
In the theory of decomposable graphs, d_i are called *histories*, b_i *residuals* and c_i *separators* (Lauritzen 1996).

Theorem 1 (Factorization Theorem). *Let* $p(\mathbf{x})$ *be a Boltzmann distribution on* X *with*

$$p(\mathbf{x}) = \frac{\mathrm{Exp}_u f(\mathbf{x})}{F_u} \quad \textit{with } u > 1 \textit{ arbitrarily.} \tag{9}$$

If

$$b_i \neq \emptyset \quad \forall i = 1, \ldots, l; \quad d_l = \boldsymbol{X}, \tag{10}$$

$$\forall i \geq 2 \; \exists j < i \;\; \textit{such that } c_i \subseteq s_j \tag{11}$$

then

$$p(\mathbf{x}) = \prod_{i=1}^{l} p(x_{b_i} | x_{c_i}) \tag{12}$$

The proof can be found in Mühlenbein et al. (1999a). Equation 11 is called the *running intersection property*. The class of ADFs allowing an exact factorization which can be computed in polynomial time in n is severely restricted. Many ADFs are defined on a two dimensional grid. Here the sets needed for an exact factorization grow like $O(\sqrt{n})$ where n is the number of variables of the grid. Therefore the computational complexity scales exponentially.

3 The Factorized Distribution Algorithm

We assume that an ADF and a factorization of the probability distribution is given. The factorization can also be used at the initialization. For faster convergence a proportion of $r * N$ individuals will be generated with a local approximation of the conditional marginal distributions. The method will be explained in Section 3.2

FDA$_\mathbf{r}$

- **STEP 0:** Set $t \Leftarrow 0$. Generate $(1-r) * N \gg 0$ points randomly and $r * N$ points according to Equation 16.
- **STEP 1:** Selection
- **STEP 2:** Compute the conditional probabilities $p^s(x_{b_i} | x_{c_i}, t)$ using the selected points.
- **STEP 3:** Generate a new population according to $p(\mathbf{x}, t+1) = \prod_{i=1}^{l} p^s(x_{b_i} | x_{c_i}, t)$
- **STEP 4:** If termination criteria is met, FINISH.
- **STEP 5:** Add the best point of the previous generation to the generated points (elitist).
- **STEP 6:** Set $t \Leftarrow t+1$. Go to STEP 2.

FDA can be used with an exact or an approximate factorization. It uses *finite samples* of points. Convergence of FDA to the optimum will depend on the size of the samples. FDA can be run with any popular selection method.

3.1 Analysis of the Factorization Algorithm

The computational complexity of FDA depends on the factorization and the population size N. The number of function evaluations to obtain a solution is given by

$$FE = GEN_e * N \tag{13}$$

GEN_e denotes the number of generations till convergence. Convergence means that $p(x, t+1) = p(x, t)$. The computational complexity of computing N new search points is given by

$$compl(Npoints) \approx l * N \tag{14}$$

The computational complexity of computing the probability is given by

$$compl(p) \approx (\sum_{i=1}^{l} 2^{|s_i|}) * M \tag{15}$$

where $|s_i|$ denotes the number of elements in set s_i, and M is the number of selected points. Therefore we obtain that the amount of computation of FDA mainly depends on l, the size of the defining sets s_i, and the size of the selected population. In order to exactly compute the probabilities an infinite population is needed. But a numerical efficient *FDA* should use a minimal population size N^* still giving good numerical results. The computation of N^* is a difficult problem for any search method using a population of points. This problem will be discussed in Section 5.
We have implemented a simple factorization algorithm which computes a factorization for any given ADF.

FDA – FAC

- **STEP 0:** Set i=1. Let $\tilde{s}_i$ be the sub-function which is maximally non-linear. Non-linearity is defined as the square distance from the linear regression.
- **STEP 1:** Compute $\tilde{d}_i := \bigcup_{j=1}^{i} \tilde{s}_j$.
- **STEP 2:** Select s_k which has maximal overlap with $\tilde{d}_i$ and $s_k \cap \tilde{d}_i \neq \emptyset$.
- **STEP 3:** If no set is found: STEP 5
- **STEP 4:** Set $\tilde{s}_{i+1} = s_k$, $i := i + 1$. If $i < l$ go to STEP 1.
- **STEP 5:** Compute the factorization using Equation 6 with sets $\tilde{s}_i$.

For simple structures like chains or trees, FDA-FAC computes an exact factorization, for complex structures an approximate factorization. To compute a factorization with minimal complexity for an arbitrary ADF is a very difficult task. We conjecture that this problem is in NP. This research needs deep results from graph theory. The problem of factorization of a probability distribution is also dealt with in the theory of *graphical models* (Frey, 1998).

3.2 Generation of the Initial Population

Normally the initial population is generated randomly. But if an ADF is given, initial points can be generated using this information. The idea is to generate subsets x_{s_i} with high local fitness values (i.e high f_i) more often than subsets with lower values.

The following method has been implemented. The true Boltzmann distribution $p(x)$ is approximated by a distribution $\tilde{p}(x)$ which uses the same factorization as $p(x)$. But the conditional probabilities are computed using the local fitness functions f_i only.

$$\tilde{p}(x_{b_i}|x_{c_i}) = \frac{\tilde{p}(x_{s_i})}{\tilde{p}(x_{c_i})} := \frac{\mathrm{Exp}_u\, f_i(x_{s_i})}{\sum_{y_{c_i}=x_{c_i}^{y_{s_i}}} \mathrm{Exp}_u\, f_i(y_{s_i})} \tag{16}$$

with $u \geq 1$. The larger u, the "steeper" the distribution. $u = 1$ yields a uniform distribution. u can be chosen so that

$$\forall x_{s_i}, y_{s_i}: \qquad \frac{1}{10} \leq \frac{\tilde{p}(x_{b_i}|x_{c_i})}{\tilde{p}(y_{b_i}|y_{c_i})} \leq 10 \qquad i = 1, 2, \ldots, l$$

by setting

$$\begin{aligned} \mathrm{span} &:= \max_i \{\max_{x,y} |f_i(x) - f_i(y)|\} \\ u &:= 10^{1/\mathrm{span}} \end{aligned}$$

Let us take the the function $OneMax(n) = \sum x_i$ as an example. Here we have the factorization

$$p(\mathbf{x}) = \prod_{i=1}^{n} p(x_i)$$

FDA computes span $= 1$ and thus $u = 10$. This leads to

$$\tilde{p}(x_i = 0) = \frac{1}{11} \qquad \tilde{p}(x_i = 1) = \frac{10}{11}$$

There will be ten times more 1s than 0s in the initial population.

Such an initial population might not give a Boltzmann distribution. Therefore we generate only half of the population by this method. The other half is generated randomly.

Next we will investigate FDA with Boltzmann selection and truncation selection in infinite populations.

3.3 Convergence of FDA

Mühlenbein et al. (1999a) proved convergence of FDA if points are selected according to a Boltzmann distribution with a given $v > 1$. In this case the distribution p^s of the selected points is given by

$$p^s(\mathbf{x}, t) = p(\mathbf{x}, t) \frac{\mathrm{Exp}_v\, f(\mathbf{x})}{\sum_{\mathbf{x}} p(\mathbf{x}, t)\, \mathrm{Exp}_v\, f(\mathbf{x})} \tag{17}$$

One can easily show that if $p(\mathbf{x}, t)$ is a Boltzmann distribution, then $p^s(\mathbf{x}, t)$ is also a Boltzmann distribution. Because FDA computes new search points according to

$$p(\mathbf{x}, t+1) = p^s(\mathbf{x}, t),$$

the following theorem easily follows (Mühlenbein et al. (1999a)).

Theorem 2. *If the initial points are distributed according to* $p(\mathbf{x},0) = \frac{\mathrm{Exp}_u f(x)}{F_u}$ *with* $u \geq 1$, *then for FDA the distribution at generation* t *is given by*

$$p(\mathbf{x},t) = \frac{\mathrm{Exp}_w f(\mathbf{x})}{\sum_{\mathbf{y}} \mathrm{Exp}_w f(\mathbf{y})} \tag{18}$$

with $w = u \cdot v^t$.

Remark: Boltzmann selection with fixed basis $v > 1$ defines an annealing schedule with $T(t) = 1/(t*\ln(v) + \ln(u))$, where t denotes the number of generations. Theorem 3 remains valid for any annealing schedule with $\lim_{t\to\infty} T(t) = 0$.

Theorem 3 (Convergence). *Let* $X_{opt} = \{x_{1opt}, x_{2opt}, ..\}$ *be the set of optima. Then under the assumptions of Theorem 2*

$$\lim_{t\to\infty} p(\mathbf{x},t) = \begin{cases} \frac{1}{|X_{opt}|} & x \in X_{opt} \\ 0 & else \end{cases} \tag{19}$$

Therefore FDA with Boltzmann selection has a solid theoretical foundation. FDA with Boltzmann selection can be seen as an "exact" simulated annealing algorithm. Simulated annealing is controlled by two parameters – the number of trials $N(T)$ for a fixed temperature T and the annealing schedule of the temperatures. These two parameters are also important for FDA. FDA generates all N points for a given temperature by using the Boltzmann distribution. Therefore N can be called the population size. The second parameter of FDA remains the annealing schedule. We will investigate in Section 5 the difficult relation between N and annealing schedule for an efficient numerical algorithm.

Numerically *truncation selection* is easier to implement. It works as follows. Given is a truncation threshold τ. The best $\tau * N$ individuals are selected. We estimate the conditional probabilities of the selected points $p^s(x_{b_i}|x_{c_i},t)$ from the empirical distribution. Then the factorization theorem is used to generate new search points according to

$$p(\mathbf{x},t+1) = \prod_{i=1}^{l} p^s(x_{b_i}|x_{c_i},t)$$

Now the following problem arises. After truncation selection the distribution is *not* a Boltzmann distribution. Therefore in general

$$p^s(\mathbf{x},t) \neq \prod_{i=1}^{l} p^s(x_{b_i}|x_{c_i},t)$$

Because of this inequality we might have $p(x_{opt},t+1) < p^s(x_{opt},t)$. This makes a convergence proof difficult. For proportionate selection convergence has been shown by Mühlenbein and Mahnig (1999b) for separable functions.

4 Theoretical Analysis for Infinite Populations

We will investigate two linear functions with very different fitness distributions.

$$OneMax_n(x) = \sum_{i=1}^{n} x_i \tag{20}$$

$$Int_n(x) = \sum_{i=1}^{n} 2^{i-1} x_i \tag{21}$$

$OneMax$ has $(n+1)$ different different fitness values which are multinomially distributed. Int has 2^n different fitness values. For ADFs the multinomial distribution is "typical", i.e it occurs fairly often. The distribution generated by Int is more special. Both functions are linear and therefore the following factorization is used

$$p(\mathbf{x}, t+1) = \prod_{i=1}^{n} p(x_i, t) \tag{22}$$

We first analyze $OneMax$. Using Equation 18 we obtain.

Theorem 4. *For Boltzmann selection with basis v the probability distribution for $OneMax$ is given by*

$$p(\mathbf{x}, t) = \frac{v^{tf(x)}}{(1+v^t)^n} \tag{23}$$

The number of generations needed to generate the optimum with probability $1-\epsilon$ is given by

$$GEN_\epsilon \approx \frac{\ln \frac{n}{\epsilon}}{\ln(v)} \tag{24}$$

For truncation selection an approximate analysis was already done in (Mühlenbein et al. (1993b), Mühlenbein, (1998)). For simplicity we assume that in the initial population all univariate marginal distributions are equal ($p_i(x_i = 1, t = 0) := p_0$). Then $p_i(x_i = 1, t) := p(t)$ for all t.

Theorem 5. *For truncation selection τ with selection intensity I_τ the marginal probability $p(t)$ obeys for $OneMax$*

$$p(t+1) = p(t) + \frac{I_\tau}{n}\sqrt{np(t)(1-p(t))}. \tag{25}$$

This equation has the approximate solution

$$p(t) = 0.5\left(1 + sin\left(\frac{I_\tau}{\sqrt{n}}t + arcsin(2p_0 - 1)\right)\right) \tag{26}$$

where

$$t \leq \left(\frac{\pi}{2} - arcsin(2p_0 - 1)\right)\frac{\sqrt{n}}{I_\tau}$$

The number of generations till convergence is given by

$$GEN_e = \left(\frac{\pi}{2} - arcsin(2p_0 - 1)\right)\frac{\sqrt{n}}{I_\tau}. \tag{27}$$

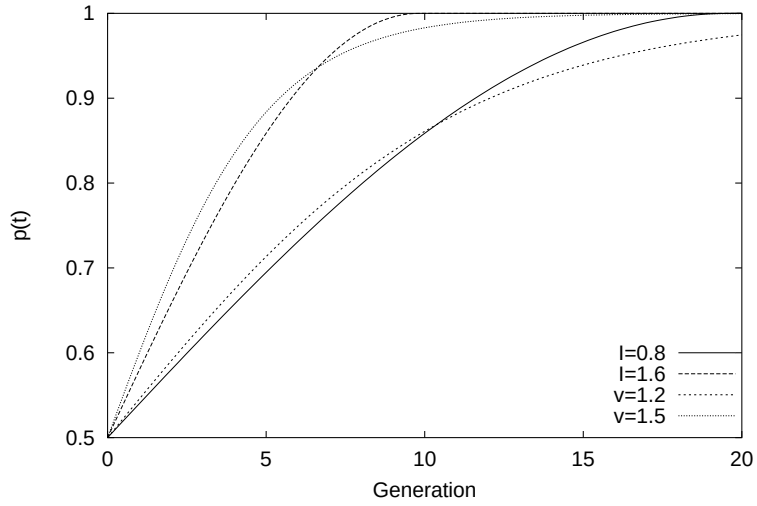

Figure 1: Probability $p(t)$ for $OneMax(100)$ with Truncation selection and Boltzmann selection

The relation between τ and I_τ depends on the fitness distribution (Mühlenbein, 1998). Assuming that the fitness distribution is normal, I_τ can be computed from the error integral. We approximately obtain

$$I_\tau \approx 0.6k \qquad \text{for} \qquad \tau = 2^{-k} \quad 1 \leq k \leq 3$$

Asymptotically truncation selection needs more number of generations to convergence than Boltzmann selection. GEN_e is of order $O(\ln(n))$ for Boltzmann selection and of order $O(\sqrt{n})$ for truncation selection. But if the basis v is small (e.g. $v = 1.2$), and $\epsilon = 0.01$ then even for $n = 1000$ truncation selection converges faster than Boltzmann selection.

The different behaviour of Boltzmann selection and truncation selection is shown in Figure 1. Equations 23 and 26 are plotted for reasonable values of v and I. For $v = 1.2$ Boltzmann selection selects slightly stronger than truncation selection with $I = 0.8$ at the beginning. Boltzmann selection gets weak when the population approaches the optimum. The same behaviour can be observed for $v = 1.5$. In fact, all selection methods using proportionate or exponential proportionate selection have this problem. If the fitness values in the population differ only slightly, selection gets weak. Truncation selection does not have this problem. It selects much stronger than Boltzmann selection when approaching the optimum. Therefore truncation selection with $I = 1.6$ converges faster than Boltzmann selection for $v = 1.5$.

Recall that Boltzmann selection with fixed v gives an annealing schedule of $1/T(t) = t \cdot \ln(v)$. The convergence of Boltzmann selection can be speeded up if the basis v is changed during the run. But annealing schedules have to be discussed with *finite* populations. For infinite populations strongest selection is obviously the best. For finite populations the computation of an optimal annealing schedule is very difficult. This will be discussed in the next section. Here we will only show that for $OneMax$ FDA with truncation selection also generates a Boltzmann distribution.

Lemma: *Let the distribution be generated by* $p(\mathbf{x}) = \prod_{i=1}^{n} p(x_i)$. *Let* $p(x_i = 1) := p_i$. *Then there exists* $T_1, \ldots, T_n$ *with*

$$\frac{1}{T_i} = \ln \frac{p_i}{1 - p_i} \tag{28}$$

so that for $OneMax(n)$

$$p(\mathbf{x}) = \frac{e^{\sum_{i=1}^{n} \frac{x_i}{T_i}}}{Z} \tag{29}$$

Z *is the partition function, defined by* $Z = \sum_{\mathbf{y}} e^{\sum_{i=1}^{n} x_i / T_i}$.

Proof: We have to show that $\prod p(x_i) = exp(\sum \frac{x_i}{T_i})/Z$. Because $exp\,(\ln(p_i/(1-p_i))) = p_i/(1-p_i)$ one obtains

$$Z = 1 + \frac{p_1}{1-p_1} + \cdots + \frac{p_n}{1-p_n} + \frac{p_1 p_2}{(1-p_1)(1-p_2)} + \cdots + \frac{\prod_i p_i}{\prod_i (1-p_i)}$$

This can be simplified to

$$Z = \frac{1}{\prod_{i=1}^{n}(1-p_i)}$$

Noting that $p(x_i = 0) = 1 - p_i$ the conjecture follows. □

Corollary *If* $p_1 = \ldots = p_n := p$ *then* $p(\mathbf{x})$ *is a Boltzmann distribution with*

$$p(\mathbf{x}) = \frac{e^{\frac{f(x)}{T}}}{Z} \tag{30}$$

where

$$\frac{1}{T} = \ln \frac{p}{1-p}. \tag{31}$$

For $p = 1/2$ we have $T = \infty$ and for $p = 1$ we get $T = 0$. We can use Equation 25 to compute $p(t)$. Because the assumptions of Theorem 5 and of the corollary are identical, FDA with truncation selection generates a Boltzmann distribution with annealing schedule

$$\frac{1}{T(t)} = \ln \frac{p(t)}{1-p(t)} = \ln \frac{np^2(t)}{np(t)(1-p(t))} = \ln \frac{\bar{f}^2(t)}{n \cdot Var(t)}. \tag{32}$$

The annealing schedule depends on the average fitness $\bar{f}(t)$ and the variance $Var(t)$ of the population. In Table 1 the schedule is shown for $n \in \{32, 64, 256\}$. $1/T(t)$ first grows linearly in t. This is the standard annealing schedule. But $1/T(t)$ increases nonlinear when approaching the optimum. For the first generation we have approximately $1/T(1) \approx 2I_\tau/\sqrt{n}$.

Let us now turn to the analysis of the function Int. We first consider truncation selection with $\tau = 0.5$ and a large population size. After one generation of selection the

t	$n = 32$	$n = 64$	$n = 256$
1	0.2848	0.2006	0.1000
2	0.5785	0.4044	0.2005
3	0.8884	0.6135	0.3016
7	2.5878	1.5658	0.7179
8	3.3510	1.8574	0.8265
9	4.7853	2.1880	0.9376
10	∞	2.5756	1.0517
11		3.0530	1.1693
12		3.6911	1.2907
13		4.7096	1.4168
14		∞	1.5482
28			5.6610
29			7.5458
30			∞

Table 1: Value of $1/T(t)$ for $OneMax$ and $\tau = 0.5$

n-th bit will be fixed. The other bits will not be affected by selection. After the next generation bit $(n-1)$ will be fixed etc. Convergence to the optimum is achieved after n generations.

For truncation selection with $\tau = 0.25$ two bits will be fixed in every generation. Convergence will be reached after $n/2$ generations. Therefore we obtain for Int

Theorem 6. *For truncation selection with $\tau = 2^{-k}; k \geq 1$ we have for Int*

$$GEN_e = \frac{n}{k} \tag{33}$$

Setting the selection intensity $I_\tau = k$ for $\tau = 2^{-k}$ we obtain the same result as for $OneMax$: GEN_e scales inversely proportionate to I_τ. But GEN_e scales proportionate to the problem size n. This is the worst case, as the following theorem shows:

Theorem 7. *Let the optimum be unique. Let the population size be very large $(N > 2^n)$. Assume that for truncation selection with $\tau = 2^{-k}$ we have $p(\mathbf{x}_{opt}) \geq p^s(\mathbf{x}_{opt})$. Then*

$$GEN_e \leq \frac{n}{k}. \tag{34}$$

Proof: In an infinite population the optimum is contained with probability $1/2^n$. After one step of selection the probability will be increased at least to $2^k/2^n$. In about n/k steps the probability of the optimum has increased to 1. □

Next we analyze Boltzmann selection.

Theorem 8. *Let $f(\mathbf{x}) = Int(n)$. Then for a Boltzmann distribution with $v > 1$ we have*

$$p(1, \ldots, 1, 1) \approx \tfrac{v-1}{v} \tag{35}$$

$$p(1, \ldots, 1, 0) \approx \tfrac{v-1}{v^2} \tag{36}$$

Proof: By definition

$$p(\mathbf{x}) = \frac{v^{f(\mathbf{x})}}{\sum_{\mathbf{x}} v^{f(\mathbf{x})}}$$

Observing that $\sum_{\mathbf{x}} v^{f(\mathbf{x})} = 1 + v + v^2 + \cdots + v^{2^n-1}$ we obtain

$$\sum v^{f(\mathbf{x})} = \frac{v^{2^n} - 1}{v - 1}$$

Now the theorem easily follows. □

The theorem shows that for Int the Boltzmann distribution is concentrated around the optimum, even for small values of v. For instance with $v = 1.2$ the global optimum is contained with probability $p = 0.167$ in the population. Even a small $v = 1.01$ gives a probability of about $p = 0.01$ for the optimum. Therefore the selected population has a small diversity. In finite populations this will cause a problem. Some genes will get fixed to wrong alleles. This will be investigated next.

5 Analysis of FDA for Finite Populations

In finite populations convergence of FDA can only be probabilistic.

Definition: Let ϵ be given. Let $P_{conv}(N)$ denote the probability that FDA with a population size of N converges to the optima. Then the critical population size is defined as

$$N^*(\epsilon) = \min_N P_{conv}(N) \geq 1 - \epsilon \tag{37}$$

If FDA with a finite population does not convergence to an optimum, then a gene is fixed to a wrong value. The probability of fixation is reduced if the population size is increased. We obviously have for FDA

$$P_{conv}(N_1) \leq P_{conv}(N_2) \quad N_1 \leq N_2$$

We show the cumulative fixation probability in Table 2 for $Int(16)$. The fixation probability is larger for stronger selection. For a given truncation selection the maximum fixation probability is at generation 1 for very small N. For larger values of N the fixation probability increases until a maximum is reached and then decreases again. This behaviour has been observed for many fitness distributions.

Boltzmann selection with $v = 1.01$ gives a temperature of about $T = 100$. Even for this temperature the selection is very strong for the fitness distribution given by $Int(16)$. For $N = 700$ the largest fixation probability is still at the first generation. Therefore the critical population size for Boltzmann selection for $v = 1.01$ is very high ($N^* > 700$). For truncation selection with $\tau = 0.25$ we have $N^*(0.1) \leq 80$.
Because Boltzmann selection in finite populations critically depends on a good annealing schedule, we normally run FDA with truncation selection. This selection method is a good compromise. It has an important property, which we formulate as an empirical law. It has been confirmed by many numerical experiments.

Empirical law: Let ϵ be reasonable small, e.g. $\epsilon = 0.1$. Then the number of generations to converge to the optimum remains constant for $N \geq N^*(\epsilon)$.

$$GEN_e(N^*(\epsilon)) = GEN_e(N) = GEN_e(N = \infty) \quad N \geq N^*(\epsilon) \tag{38}$$

t	$\tau = 0.25$ $N = 30$	$\tau = 0.5$ $N = 30$	$\tau = 0.25$ $N = 80$	$\tau = 0.5$ $N = 60$	$Boltz.$ $N = 500$	$Boltz.$ $N = 700$
1	0.0955	0.0035	0.0	0.0	0.2520	0.0885
2	0.4065	0.0255	0.0025	0.0095	0.2980	0.1110
3	0.5955	0.1040	0.0165	0.0205	0.3180	0.1275
4	0.6880	0.2220	0.0355	0.0325	0.3295	0.1375
5	0.7210	0.3270	0.0575	0.0490	0.3385	0.1455
6	0.7310	0.4030	0.0695	0.0630	0.3435	0.1510
7	0.7310	0.4470	0.0740	0.0715	0.3505	0.1555
8	0.7310	0.4705	0.0740	0.0780	0.3530	0.1565
9	0.7310	0.4840	0.0740	0.0806	0.3555	0.1575

Table 2: Cumulative fixation probability for $Int(16)$. Truncation selection vs. Boltzmann selection with $v = 1.01$.

Truncation selection has a free parameter, the truncation threshold τ. It seems obvious that the smaller the threshold τ, the larger N^* has to be. But numerical experiments have shown that there exists a threshold τ_{min} which leads to a minimal N^*_{min}. This means that N^* also increases for very low selection. The reason for this phenomenon is genetic drift. Slow selection leads to a large number of generations which increases the probability of gene fixation. This problem has been first investigated by Mühlenbein and Schlierkamp-Voosen (1993b) for $OneMax$ and genetic algorithms. A more detailed investigation can be found in Mühlenbein and Schlierkamp-Voosen (1994).

We denote the critical population size for given τ by $N^*(\epsilon, \tau)$. Because ϵ is fixed, we omit ϵ and write just $N^*(\tau)$. For Int we have approximately computed $N^*(\tau)$ by a a Markov chain analysis. The Markov model was simplified, therefore we formulate the result as a conjecture.

Conjecture: *Let $\tau_k = 2^{-k}$. For FDA with fitness function Int the critical population size $N^*(\tau)$ is approximately given by*

$$N^*(\tau_k) \approx N^*(\tau_1) * 2^{\frac{k-1}{2}} \qquad k \geq 1$$

If $N^*(\tau)$ has been determined, then an optimal truncation threshold τ_{opt} can be computed. This threshold gives the minimum number of function evaluations FE.

Definition: The optimum truncation threshold τ_{opt} is defined by

$$\tau_{opt} = \min_{\tau} FE(\tau) = \min_{\tau} GEN_e(\tau) * N^*(\tau) \tag{39}$$

In general τ_{opt} is different from τ_{min} which needs the minimal population size. The following result follows from $k > 1$ from the above conjecture.

Empirical Law: *For Int the optimal truncation threshold τ is contained in the interval* $[0.125, 0.4]$.

Proof: Part of the result follows from the approximate formulas. For $\tau = 2^{-k}$ we obtain using the critical population size

$$FE \approx \frac{n}{k} * N^*(\tau_1) * 2^{\frac{k-1}{2}} \propto \frac{1}{\sqrt{\tau}\log(1/\tau)}, \quad k \geq 1 \tag{40}$$

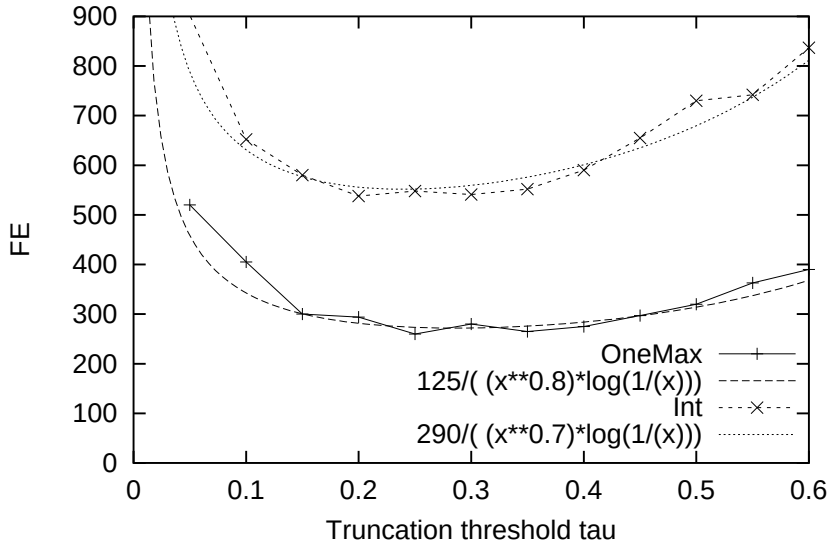

Figure 2: Minimum number of function evaluations for given τ, $Int(16)$, $OneMax(32)$.

The minimum is at $k = 0.5(1 + \sqrt{17})$. □

The empirical law has been investigated in detail by numerical experiments. The determination of the optimal population size by simulations is very difficult and error prone. We have done extensive simulations for two distributions generated by $OneMax$ and Int. In Figure 2 the results are shown. The critical population size is determined from the condition $\epsilon = 0.1$, i.e. out of 1000 runs 900 find the optimum. The best numerical fit was obtained by using $\tau^{0.7}$ instead of $\tau^{0.5}$ for Int. For $OneMax$, $\tau^{0.8}$ gave a good fit.

These intensive simulations have been made to eliminate the truncation threshold as a free parameter. We formulate our result as a rule.

Rule of Thumb: *A good truncation threshold for FDA is* $\tau \approx 0.3$.

It is interesting to note that the problem of an optimal truncation threshold has been also investigated for animal breeding. A discussion can be found in Robertson (1960). Using a too simple model (in fact assuming an infinite number of loci n) he obtained that $\tau = 0.5$ should be the optimal threshold. This result is not in agreement with our analysis, but also not with actual selection experiments. Robertson (1960) writes: In most selection programmes that are at all efficient, I_τ lies between 1 and 2. This corresponds to $\tau = 0.4$ and $\tau = 0.06$.

Note that we have assumed that the population size remains fixed to N. It is possible to reduce the function evaluations further by using different population sizes at each generation. But this is only a theoretical option, because there are no techniques known how to choose the population sizes.

We summarize our scaling results. *Let a family of functions to be optimized be defined for arbitrary* n. *For a given truncation threshold the function evaluations* FE

of FDA scale as the product of the number of generations to converge, GEN_e, and the critical population size, $N^*(\tau)$. GEN_e can be bounded. It is less than n/τ, normally it will be of order $O(\sqrt{n})$. The scaling of FDA mainly depends on $N^*(\tau)$. Unfortunately the estimation of $N^*(\tau)$ is difficult, even for linear functions.

6 Numerical Results

This section has two purposes. First, we want to show that FDA behaves like the theory predicts. Second, we will show that it can solve difficult optimization problems.
We will first investigate the conjecture concerning the number of generations until equilibrium. In addition to $F_1(\mathbf{x}) = OneMax(n)$ the following two functions will be investigated

$$F_2(\mathbf{x}) = \sum_{i=1}^{l} f_2(x_{s_i}) \quad s_i = \{x_{3i-2}, x_{3i-1}, x_{3i}\}$$

$$F_3(\mathbf{x}) = \sum_{i=1}^{l} f_3(x_{s_i}) \quad s_i = \{x_{3i-2}, x_{3i-1}, x_{3i}\}$$

In F_2 we set $f_2(x_{s_i})$ to the values of the *OneMax* function of order three. F_2 is thus identical to F_1. But FDA will use a factorization which consists of marginal distributions of size 3. Thus the number of free parameters is more than twice as large. For F_3 we set $f_3(1,1,1) = 10$ and all other values to zero.

Given our theory we expect the following results. GEN_e should be equal for $F1$ and $F2$. GEN_e should be smaller for $F3$ because here FDA has to test only two main alternatives — $(1,1,1)$ and all the rest. For FDA F_3 is just like a $OneMax$ function of size $n/3$, where the probability of generating the important substring $(1,1,1)$ is smaller. With random initialization the string with $(1,1,1)$ will be generated with probability $p_0 = 0.125$. For all cases the expected number of generations Gen_e can be computed from Equation 27.

n	F_1	F_2	F_3	$GEN_e(p_0 = 0.5))$	$GEN_e(n/3, p_0 = 0.125)$
30	7.0	7.0	6.2	7.2	6.4
60	10.0	10.0	9.0	10.1	9.0
90	12.2	12.3	11.0	12.4	11.0
120	14.2	14.4	12.9	14.4	12.7
120_{GA}	18.8	18.8	21.3		
150	16.0	16.3	14.1	16.0	14.3
180	17.2	17.8	15.9	17.5	15.6

Table 3: Generations until convergence, truncation threshold 0.3

Note how precisely Equation 27 predicts GEN_e obtained from actual simulation with FDA. GA is a genetic algorithm with truncation selection and uniform crossover. It needs slightly more generations for *OneMax* than UMDA. This was already observed in (Mühlenbein et al., 1993b). For the function F_3 the genetic algorithm needs almost twice as many generations as FDA, which has knowledge about the micro-structure of $F3$.
It is outside the scope of this paper to test FDA on an exhaustive set of typical functions.

We have decided to use in this paper separable ADFs, furthermore ADFs with a chain-like structure, a tree-like structure and a grid-like structure.

Given the structure, different sub-functions have been used to generate the test function. The first function is a deceptive function of order three. It is defined as follows. Let u denote the number of 1s in the string. Then

$$f_{dec3} = \begin{cases} 0.9 & \text{for} \quad u = 0 \\ 0.8 & \text{for} \quad u = 1 \\ 0.0 & \text{for} \quad u = 2 \\ 1.0 & \text{for} \quad u = 3 \end{cases}$$

Next we used a deceptive function of order 5

$$f_{dec5} = \begin{cases} 0.9 - 0.1i & \text{for} \quad u = i \quad i < 4 \\ 0 & \text{for} \quad u = 4 \\ 1 & \text{for} \quad u = 5 \end{cases}$$

F_{Dec5} is a separable function of subset size 5.

$$F_{Dec5} = \sum_{i=1}^{l} f_{dec5}(x_{5i-4}, x_{5i-3}, x_{5i-2}, x_{5i-1}, x_{5i}) \tag{41}$$

A difficult function to optimize is $IsoChain$. It is defined as follows:

$$F_{IsoChain} = \sum_{i=1}^{l-1} Iso_1(x_{2i-1}, x_{2i}, x_{2i+1}) + Iso_2(x_{2l-1}, x_{2l}, x_{2l+1}) \tag{42}$$

u	0	1	2	3
Iso_1	l	0	0	$l-1$
Iso_2	0	0	0	l

with $n = 2l + 1$. The global optimum is $(1, 1, \ldots, 1)$ with value $l * (l - 1) + 1$. This optimum is triggered by Iso_2. It is very isolated. Six strings with leading zeroes give the second best value of $l(l - 1)$. These points are far away in Hamming distance from the optimum. For this chain FDA-FAC computes the factorization

$$p(x) = p(x_1, x_2, x_3)p(x_4, x_5|x_3)p(x_6, x_7|x_5) \cdots p(x_{n-1}, x_n|x_{n-2})$$

Next we define ADFs on a tree. The root is x_1. The variable x_i is linked with x_{2i} and x_{2i+1}, the descendents. For every triple a sub-function of three variables is used, the deceptive function of order 3 in the case of $F_{Dec3Tree}$ and functions Iso in the case of $F_{IsoTree}$. Thus

$$F_{IsoTree} = Iso_2(x_1, x_2, x_3) + \sum_{i=2}^{l} Iso_1(x_i, x_{2i}, x_{2i+1}) \tag{43}$$

$$F_{Dec3Tree} = \sum_{i=1}^{l} f_{dec3}(x_i, x_{2i}, x_{2i+1}) \tag{44}$$

We also included the function F_{Cuban} defined in (Mühlenbein et al., 1999a). This function is defined on a chain. The definition of the function is difficult, so it will be omitted. Two sub-functions are used alternating, making this function difficult to optimize.

Furthermore we investigate two structures where FDA-FAC computes an approximate factorization only. A simple structure of this kind is the circle. F_{IsoCir} is obtained from $F_{IsoChain}$ by closing the chain to a circle.

$$F_{IsoCir} = \sum_{i=1}^{l-1} Iso_1(x_{2i-1}, x_{2i}, x_{2i+1}) + Iso_2(x_{2l}, x_{2l+1}, x_1) \tag{45}$$

For this function our factorization algorithm determines the following factorization

$$p(x) = p(x_1, x_2, x_3)p(x_4, x_5|x_3) \cdots p(x_{n-2}, x_{n-1}|x_{n-3})p(x_n|x_1, x_{n-1}) \tag{46}$$

This factorization is not exact. It does not fulfill the running intersection property. For a circle an exact factorization can be theoretically derived.

$$p(x) = p(x_1, x_2, x_3, x_n)p(x_4, x_5|x_3, x_n) \cdots p(x_{n-2}, x_{n-1}|x_{n-3}, x_n) \tag{47}$$

The exact and the approximate factorization are not very different. It turns out that the numerical results for F_{IsoCir} are almost identical to $F_{IsoChain}$. Therefore they are omitted.

The next test function is like $IsoChain$, but defined on a torus of size $n = m * m$. The peak function $IsoT_2$ is used at the upper left corner of the torus. Let u denote the number of 1's in a string.

$$\begin{aligned} F_{IsoTorus} =& IsoT_2(x_{1-m+n}, x_{1-1+m}, x_1, x_2, x_{1+m}) \\ &+ \sum_{i=2}^{n} IsoT_1(x_{up}, x_{left}, x_i, x_{right}, x_{down}) \end{aligned} \tag{48}$$

u	0	1	2	3	4	5
$IsoT_1$	m	0	0	0	0	$m-1$
$IsoT_2$	0	0	0	0	0	m^2

where x_{up} etc. is defined as the appropriate neighbor, wrapping around. This function is even more difficult to optimize than IsoPeak. The best and second best strings have values $m^3 - m + 1$ and $m^3 - m$. This means that their relative difference is much lower than $IsoPeak$ on a chain.

An exact factorization of an ADF on a grid or torus of size $n = m^2$ needs subsets of size m. For an exact factorization the computational complexity of FDA scales exponentially. Our FDA-FAC generated the following approximate factorization for $n = 100$:

$$\begin{aligned} p(\mathbf{x}) = p(x_1, x_2, x_{10}, x_{11}, x_{91})p(x_3, x_{12}, x_{92}|x_1, x_2)p(x_{20}, x_{21}|x_1, x_{11}, x_{12}) \\ \cdots p(x_{90}|x_{70}, x_{71}, x_{79}, x_{80}) \end{aligned}$$

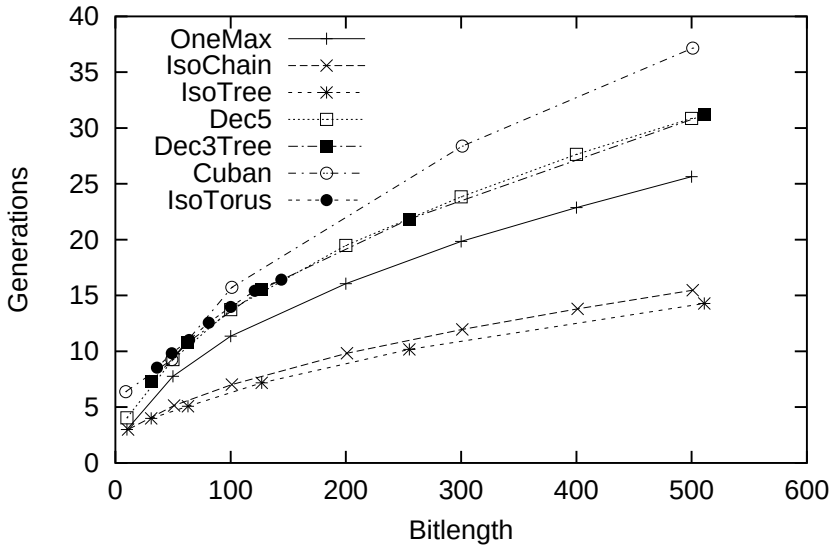

Figure 3: Number of generations until convergence

This factorization violates the running intersection property. The factorization does not use the conditional probabilities for one row and one column. In this case the exact and the approximate factorization are fairly different. FDA nevertheless can find the optimum, because the approximate factorization is still able to generate the optimum.

This test suite covers ADFs defined on simple regular graphs. We did simulations with a fixed parameter setting. Half of the population is initialized according to our local heuristic, the truncation threshold is set to $\tau = 0.3$. Only the population size N varies. We tried to use for all simulations the optimal $N^*(\tau)$ in order to obtain the lowest number of function evaluations.

The number of generations GEN_e needed to convergence is plotted in Figure 3. GEN_e is the smallest for $IsoChain$ and $IsoTree$. In the middle we find $OneMax$. The largest number of generations is needed for the function $Cuban$. In general GEN_e varies only slightly from run to run. For all test functions it scales approximately like $O(\sqrt{n})$. This confirms the conjecture derived from the theoretical results.
In Table 4 we show the number of function evaluations FE, defined as the product of GEN_e and critical population size.

n	OneMax	IsoChain	Dec5	Cuban	n	Dec3Tr.	IsoTr.	n	IsoTo.
10	45	120	410	179	31	1170	128	36	4000
50	280	460	3010	3290	63	2700	300	49	6200
100	580	1260	7410	15100	127	5290	790	64	8400
200	1250	3440	15980					81	15800
300	2080	6460	25510	56760	255	12670	1530	100	21800
400	2750	9800	35110					121	57000
500	3850	14700	46280	115180	511	28050	3570	144	67000

Table 4: Number of function evaluations

For $OneMax$, $IsoTree$, $Dec5$ and $Dec3Tree$ we have a scaling of about $O(n \ln n)$.

For $IsoChain$ the scaling is about $O(n\sqrt{n})$. There are not enough data points to estimate the scaling of $Cuban$. For $IsoTorus$ we conjecture despite some irregularities a scaling of $O(n^2)$.

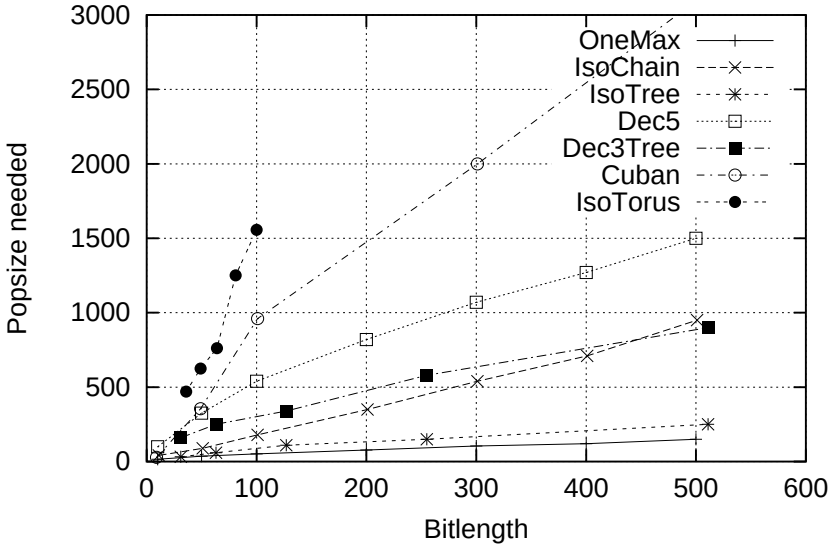

Figure 4: Critical population size for $\tau = 0.3$

Figure 4 shows the critical population size for all functions. It was already mentioned that the computation of the critical population size by simulations is a very difficult numerical task. In order to reduce the error bars, a huge number of runs have to be made. Our criterion for the optimal population size has been that about 90% of the runs converge to the optimum ($\epsilon = 0.1$). We have made 1000 runs for the cases needing a small population size, 100 runs for the medium sized problems and 20 runs only for the large problems. For all functions but $IsoTorus$ and $Cuban$ the critical population size scales less than linear in n.

7 LFDA - Computing a Bayes Factorization

The researchers of graphical models have already proposed several methods which determine a factorization from the data. This problem is called *learning*. The interested reader is referred to the book edited by Jordan (1999). We will investigate the most popular method developed for Bayes networks. In the context of optimization and FDA this method has been first used by Pelikan et al. (1999).

In order to apply this method, we have to recall that each factorization can be put into a normal form, where each variable occurs only once on the left side of a conditional marginal distribution.

Theorem 9 (Bayes Factorization). *Each probability can be factored into*

$$p(\mathbf{x}) = p(x_1) \prod_{i=2}^{n} p(x_i|pa_i) \tag{49}$$

Proof: By definition of conditional probabilities we have

$$p(\mathbf{x}) = p(x_1) \prod_{i=2}^{n} p(x_i|x_1, \cdots, x_{i-1}) \tag{50}$$

Let $pa_i \subset \{x_1, \cdots, x_{i-1}\}$. If x_i and $\{x_1, \cdots, x_{i-1}\} \setminus pa_i$ are conditional independent, we can simplify $p(x_i|x_1, \cdots, x_{i-1}) = p(x_i|pa_i)$. □

PA_i are called the parents of variable X_i. Thus each Bayes factorization defines a directed graph. In the context of graphical models the graph is called a Bayes network (Jordan (1999)).

We note that any FDA factorization fulfilling the running intersection property can be put into a normal form. We just take a simple example. Let $\mathbf{x}_{b_i} = \{x_i, x_k, x_l\}$ with $i < k < l$. Then

$$p(x_{b_i}|x_{c_i}) = p(x_i|x_{c_i})p(x_k|x_i, x_{c_i})p(x_l|x_i, x_j, x_{c_i})$$

Because of the running intersection property all variables of c_i have an index less than i. Therefore we obtain a valid normalized factorization.
We can now formulate the Bayes network learning problem. *Given a population of selected points $M(t)$, what is a good Bayes factorization fitting the data?* The most difficult part of the problem is to define a quality measure. The following discussion is a short summary of (Bouckaert, 1994). The interested reader is also referred to the two papers by Heckerman and Friedman et al. in (Jordan, 1999).

For Bayesian networks two quality measures are most frequently used - the BDe score and the *Minimal Description Length* (MDL) score. Let B denote a Bayes network, D the given data set and $M = |D|$ its size. Then MDL is given by

$$MDL(B, D) = -\,\mathrm{ld}(P(B)) + M \cdot H(B, D) + \tfrac{1}{2} PA \cdot \mathrm{ld}(M) \tag{51}$$

with $\mathrm{ld}(x) := \log_2(x)$. $P(B)$ denotes the prior probability of network B, $PA = \sum_i 2^{|pa_i|}$ gives the total number of probabilities to compute. $H(B, D)$ is the conditional entropy of the network structure B and data D. It is given by

$$H(B, D) = -\sum_{i=1}^{n} \sum_{pa_i} \sum_{x_i} \frac{m(x_i, pa_i)}{M} \,\mathrm{ld}\, \frac{m(x_i, pa_i)}{m(pa_i)} \tag{52}$$

where $m(x_i, pa_i)$ denotes the number of occurrences of x_i given configuration pa_i. $m(pa_i) = \sum_{x_i} m(x_i, pa_i)$. If $pa_i = \emptyset$, then $m(x_i, \emptyset)$ is set to the number of occurrences of x_i in D.

The term $\frac{1}{2} PA \cdot \mathrm{ld}(M)$ models the computational cost of estimating the probabilities. If no prior information is available, $P(B)$ is identical for all possible networks. In this case the MDL measure assigns high quality to networks that fit the data with as few arcs as possible. This principle is called *Occam's razor*. It has been intensively studied by Zhang and Mühlenbein (1997) for neural networks. In order to give more weight to sparse Bayes networks, we use a weight factor α. Therefore our score is

$$BIC(B, D, \alpha) = -M \cdot H(B, D) - \alpha PA \cdot \mathrm{ld}(M) \tag{53}$$

This measure has been also proposed by Schwarz (1978) as *Bayesian Information Criterion.* To compute a network B^* which maximizes BIC requires a search through the space of all Bayes networks. Such a search is more expensive than to search for the optima of the function. Therefore the following greedy algorithm has been used. k_{max} is the maximum number of incoming edges allowed.

$$\mathrm{BN}(\alpha, \mathrm{k_{max}})$$

- **STEP 0:** Start with an arc-less network.
- **STEP 1:** Add the arc (x_i, x_j) which gives the maximum increase of BIC(α) if $|PA_j| \leq k_{max}$ and adding the arc does not introduce a cycle.
- **STEP 2:** Stop if no arc is found.

Checking whether an arc would introduce a cycle can be easily done by maintaining for each node a list of parents an ancestors, i.e. parents of parents etc. $(x_i \rightarrow x_j)$ introduces a cycle if x_j is ancestor of x_i.

The BOA algorithm of Pelikan et al. (1999) uses the BDe score. This measure has the following drawback (Bouckaert, 1994). It is more sensitive to coincidental correlations implied by the data than the MDL measure. As a consequence, the BDe measure will prefer network structures with more arcs over simpler networks. The BIC measure with $\alpha = 1$ has also been proposed by Harik (1999). But Harik allows only factorizations with marginal distributions.

Given the BIC score we have several options to extend FDA to LFDA - the FDA which learns a factorization. Due to limitations of space we can only show results of an algorithm which computes a Bayes network at each generation using algorithm $BN(0.5, k_{max})$. FDA and LFDA should behave fairly similar, if LFDA computes factorizations which are in probability terms very similar to the FDA factorization. FDA uses the same factorization for all generations, whereas LFDA computes a new factorization at each step which depends on the given data M.

All numerical experiments show that LFDA and FDA behave very similar in number of generations to converge to the optima. The major difference occurs in the critical population size. One expects that LFDA needs a larger population size, because it has to estimate the network structure. We first compare the critical population sizes for three functions which are simple to optimize. The first function is $OneMax$, the other two functions are defined on a circle. We use two sub-functions

	00	01	10	11
f_{prisn}	1	5	0	3
f_{prisl}	1	5	0	4

Note that f_{prisl} is a linear function. Both functions are used to generate the functions F_{Prisl} and F_{Prisn} on a circle.

$$F_{Pris} = \sum_{i=1}^{n-1} f_{pris}(x_i, x_{i+1}) + f_{pris}(x_n, x_1) \tag{54}$$

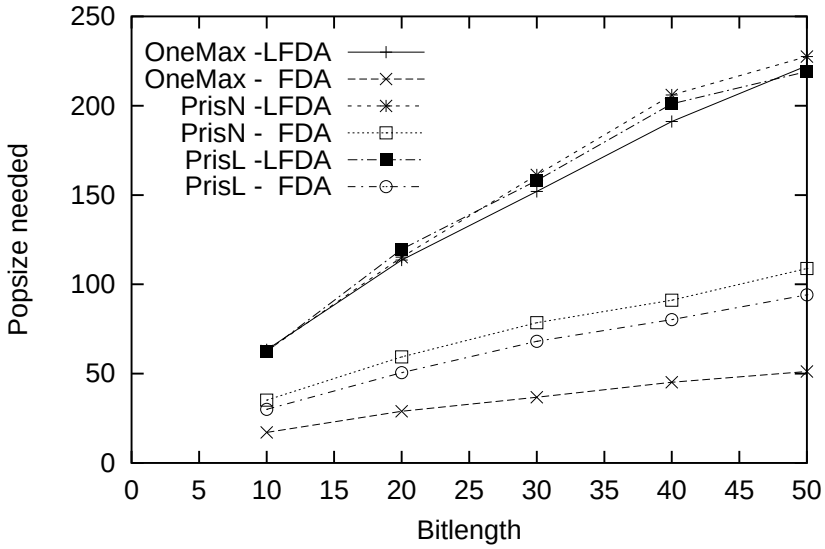

Figure 5: Critical population size for FDA and LFDA

For a circle FDA-FAC computes the approximate factorization

$$p(x) = p(x_1, x_2)p(x_2|x_1)\dots p(x_{n-1}|x_{n-2})p(x_n|x_{n-1}) \tag{55}$$

LFDA computed factorizations with slightly more edges. We expect the following results. The critical population size of FDA for $OneMax$ is smaller than the critical population size for F_{Pris}. There will be no difference between F_{Prisl} and F_{Prisn}, because FDA will use the same factorization. The critical population size of LFDA will be higher than of FDA. There will be almost no difference between the three functions, because LFDA will for all three compute a very similar Bayes network.

The actual results are shown in Figure 5. They confirm the expectations. For F_{Pris} we observe the relation $N^*_{LFDA} = 2N^*_{FDA}$.

LFDA is computationally much more expansive than FDA. First, the heuristic $BN(\alpha, k_{max})$ is only needed for LFDA. This computation is cubic in n. Second, LFDA needs a larger critical population size N^*.

Table 5 gives results for ADFs defined on a chain with 2,4, and 6 neighbors. The size of the problem is $n = 20$.

	$Chain2$	ms	$Chain4$	ms	$Chain6$	ms
FDA	130	60	650	300	1800	1200
LFDA	240	570	2460	9300	14000	60000

Table 5: Critical population sizes and computation time (ms) for FDA and LFDA

The computational time ms can be reduced by optimizing the LFDA program. But the critical population size N^* cannot be reduced for LFDA. Unfortunately the more parameter the network has, the larger the factor N^*_{LFDA}/N^*_{FDA} gets. The factor increases from 2 for $Chain2$ to almost 8 for $Chain6$. This empirical result confirms the conjec-

ture, that learning of structures will be computationally very expensive (Mühlenbein et al., 1999a).

8 Conclusion

The Factorized Distribution Algorithm FDA converges to the optima of the fitness function if Boltzmann selection is used. But Boltzmann selection has numerical drawbacks. For computational efficiency a good annealing schedule has to be determined for each problem. This is very difficult. A much simpler selection method is truncation selection as used by breeders. We showed for a representative fitness distribution that FDA with truncation selection behaves identical to FDA with a certain annealing schedule.

FDA is a true evolutionary algorithm. The population at generation t is used to generate the population at generation $t+1$. The population at generation $t-1$ is not used for generation $t+1$. There is no memory involved. The ADF decomposition is only used to compute a factorization of the distribution. The factorization is exact or approximate, depending on the ADF. FDA is depending on one parameter only. This is the population size N. The more difficult the optimization of the function is, the larger N has to be. We will try to develop methods where the population size can be adjusted during a run.

FDA uses the ADF structure to compute a factorization of the distribution. We have extended FDA to LFDA, which computes a Bayes factorization from the data without knowledge of the ADF structure. LFDA gave surprisingly good results, even for fairly complex structures. But for complex ADF structures LFDA is much more computationally expensive than FDA. There is lots of research to be done in improving LFDA.

References

Aarts, E.H. & Korst, H.M. & van Laarhoven, P.J. (1997). Simulated Annealing. In Aarts, E. & Lenstra, J.K. (Eds.),*Local Search in Combinatorial Optimization*.pp. 121-136. Chichester:Wiley

Baluja, S. & Davies, S. (1997). Using Optimal Dependency-Trees for Combinatorial Optimization: Learning the Structure of the Search Space. *Technical report CMU-CS-97-107* Carnegie-Mellon University Pittsburgh.

De Bonet, J.S.& Isbell, Ch. L. & Viola, P. (1997). MIMIC: Finding Optima by Estimating Probability Densities. In Mozer,M. & Jordan, M. & Petsche, Th. (Eds) *Advances in Neural Information Processing Systems 9* pp. 424–431x

Bouckaert, R.R. (1994). Properties of Bayesian network learning algorithms. In R. Lopez de Mantaras and D. Poole, (Eds.) *Proc. Tenth Conference on Uncertainty in Artificial Intelligence* pp. 102-109. San Francisco: Morgan Kaufmann.

Frey, B.J. (1998). *Graphical Models for Machine Learning and Digital Communication.* Cambridge: MIT Press.

Goldberg, D.E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning.* Reading: Addison-Wesley.

Goldberg, D.E & Deb, K. & Kargupta, H. & Harik, G. (1993). Rapid, Accurate Optimization of Difficult Problems Using Fast Messy Genetic Algorithms. In S. Forrest (Ed) *Proc. of the Fifth Int. Conf. on Genetic Algorithms* pp. 56-64, San Mateo, CA: Morgan Kaufman.

Harik, G. (1999). *Linkage Learning via probabilistic Modeling in the ECGA* IlliGal

Technical Report 99010, University of Illinois at Urbana-Champaign.
Jordan, M.I.(ed.) (1999). *Learning in Graphical Models* Cambridge:MIT Press.
Kargupta, H. & Goldberg, D.E. (1997). SEARCH, Blackbox Optimization, And Sample Complexity. In R.K. Belew & M. Vose (Eds.) *Foundations of Genetic Algorithms 4.* San Mateo, CA: Morgan Kaufman.
Kargupta, H. (1997). *Revisiting The GEMGA: Scalable Evolutionary Optimization Through Linkage Learning.* Personal Communication.
Lauritzen, S.L. (1996) *Graphical Models.* Oxford:Clarendon Press.
Mühlenbein, H. (1998). The Equation for Response to Selection and its Use for Prediction. *Evolutionary Computation*, 5:pp. 303-346.
Mühlenbein, H. & Schlierkamp-Voosen, D. (1993a). Predictive Models for the Breeder Genetic Algorithm I. Continuous Parameter Optimization *Evolutionary Computation*, 1:pp. 25–49.
Mühlenbein, H. & Schlierkamp-Voosen, D. (1993b). The science of breeding and its application to the breeder genetic algorithm. *Evolutionary Computation*, 1:pp. 335–360.
Mühlenbein, H. & Schlierkamp-Voosen, D. (1994). The Theory of Breeding and the Breeder Genetic Algorithm. In J. Stender & E. Hillebrand & J. Kingdon (eds.), pp. 27-64, Amsterdam: IOS Press.
Mühlenbein, H. & Mahnig, Th. & Ochoa, R. (1999a). Schemata, Distributions and Graphical Models in Evolutionary Optimization. *to appear in Journal of Heuristics*
Mühlenbein, H. & Mahnig, Th.(1999b). Convergence Theory and Applications of the Factorized Distribution Algorithm. *Journal of Computing and Information Technology* 7:pp 19–32.
Mühlenbein, H. & Paaß, G. (1996). From Recombination of Genes to the Estimation of Distributions I. Binary Parameters. In Voigt, H.-M et al. (eds.)*Lecture Notes in Computer Science 1141: Parallel Problem Solving from Nature - PPSN IV*, pp. 178-187, Berlin:Springer Press.
Pelikan, M. & Mühlenbein, H. (1998). The bivariate marginal distribution algorithm. In Roy,R. et al. (eds), *Advances in Soft Computing - Engineering Design and Manufacturing*, pp. 521–535, New York: Springer Press
Pelikan, M. & Goldberg, D.E. & Cantu-Paz, E. (1999). *BOA: The Bayesian optimization algorithm.* IlliGAL Technical Report 99003, University of Illinois at Urbana-Champaign.
Robertson, A. (1960). A theory of limits in artificial selection *Proc. Roy. Soc. London B* 153:pp. 234–249.
Schwarz, G. (1978). Estimating the dimension of a model. *Annals of Statistics*, 7:pp. 461–464.
Zhang, B.-T. & Ohm, P. & Mühlenbein, H. (1997). Evolutionary Induction of Sparse Neural Trees, *Evolutionary Computation*, 5:pp. 213–236.