Massively Parallel Simulated Annealing and its Relation to Evolutionary Algorithms

Günter Rudolph
Rudolph@LS11.Informatik.Uni-Dortmund.DE

Universität Dortmund
Fachbereich Informatik
Lehrstuhl für Systemanalyse
D-44221 Dortmund / Germany

Abstract

Simulated annealing and and single trial versions of evolution strategies possess a close relationship when they are designed for optimization over continuous variables. Analytical investigations of their differences and similarities lead to a cross-fertilization of both approaches, resulting in new theoretical results, new parallel population based algorithms, and a better understanding of the interrelationships.

Keywords: global optimization, parallel simulated annealing, parallel evolutionary algorithms, neighborhood algorithms

1 Introduction

Simulated annealing (SA) (Kirkpatrick, Gelatt, and Vecchi 1983) is a widely used method for combinatorial optimization problems. If this method is designed for optimization over continuous variables, i.e., $\min\{f(x) \,|\, x \in M \subseteq \mathbb{R}^n\}$, there is a close relationship between simulated annealing and evolution strategies (ES) (Rechenberg 1973; Schwefel 1977; Schwefel 1981) and also evolutionary programming (EP) (Fogel 1992; also see Bäck and Schwefel 1993). The investigation of these interrelationships will lead to a cross-fertilization of both approaches, resulting in new theoretical results, new parallel population based algorithms, and a better understanding of the importance of some algorithmic features of SA and ES.

In Section 2 SA is related to single trial versions of ES by a general Markov process model. From this model it is easy to see that potential differences between SA and ES only can arise by the choice of the sampling and acceptance distributions. Therefore, some SA algorithms given in the literature are investigated with respect to these distributions with the result, that the sampling distributions used in SA algorithms may be divided into fixed and non-fixed distributions. Results originally derived for single trial ES indicate that fixed sampling distributions are causing early stagnation in convergence speed, so that the massively parallel SA algorithm to be developed in Section 4 should use a non-fixed sampling distribution as it is used in ES.

The choice of the acceptance distribution affects the convergence properties. Therefore, some theoretical results are compiled and it is shown that it is not possible to achieve geometric convergence rates with SA algorithms that use acceptance distributions guaranteeing global convergence.

It is instructive to investigate multiple trial versions in the spirit of $(1,p)$–ES, where p denotes the number of trials that can be performed in parallel. This analysis given in Section 3 reveals that is now possible to achieve geometric convergence rates for SA with $p \geq 2$, when using appropriate non–fixed sampling distributions. This is possible because the competition between at least two new samples to be selected for the acceptance test mostly overrules the original acceptance rule. Although the gain in convergence speed when increasing the number of trials is shown to grow logarithmicly only, multiple trial versions have the potential to improve their properties: It is possible to supply $(1,p)$-type algorithms with the ability to self–adapt their sampling distributions as it was introduced by Schwefel (1977) in ES. This mechanism is transferred to $(1,p)$-SA and related algorithms.

In Section 4 this feature appears again in the massively parallel neighborhood algorithm in disguised form. The neighborhood algorithm (NA) is the first population based algorithm considered here. The close relationship between SA and ES makes it possible to develop an algorithmic frame so that the NA may be instantiated with ES, SA or other acceptance rules and sampling distributions.

Computational results presented in Section 5 indicate that this family of massively parallel optimization algorithms are powerful tools in global optimization.

2 Markovian optimization algorithms

Single trial variants of SA and ES algorithms can be studied in the general framework of Markovian processes. The general algorithmic frame can be formulated as:

```
choose $X_0 \in M \subseteq \mathbb{R}^n$ and set $t = 0$
repeat
    $Y_{t+1} = X_t + Z_t$
    $X_{t+1} = Y_{t+1} \cdot a(X_t, Y_{t+1}; .) + X_t \cdot (1 - a(X_t, Y_{t+1}; .))$
    increment $t$
until termination criterion satisfied
```

Here, $a(x,y;.)$ denotes the acceptance function which may depend on additional parameters. The distribution of random vector Z_t is chosen to be symmetric, i.e. $z \overset{d}{=} B\,z$ for every orthogonal matrix B. In this case z may be expressed in its stochastic representation $z \overset{d}{=} r\,u$, where r is a nonnegative random variable and u a random vector uniformly distributed on a hypersphere surface of dimension n (Fang, Kotz, and Ng 1990). This reveals that the trial point generation mechanism of the above algorithm is equivalent to that of a random direction method with some chosen distribution for the step size r (Rappl 1984; Rappl 1989).

Depending on the choice of the acceptance function $a(x, y; .)$ and of the generating distribution of z one obtains a family of Markovian optimization algorithms which can be identified by a sequence of transition probabilities $(P_t)_{t \in \mathbf{N}}$ (see Appendix):

$$P_t(x, A) = \int\limits_A Q_t(x, d\omega)\, q_t(x, \omega)\, d\omega + 1_A(x) \int\limits_M Q_t(x, d\omega)\, (1 - q_t(x, \omega))\, d\omega \tag{1}$$

with $A \subseteq M$, $x \in M$ and where $Q_t(.)$ denotes the generating distribution, $1_A(.)$ the indicator function of set A and $q_t(.)$ the acceptance probability function which is related to the acceptance function $a_t(.)$ via

$$a_t(x, y; \xi) = 1_{[0, q_t(x,y;.)]}(\xi) \quad , \tag{2}$$

where ξ is a random variable uniformly distributed on $[0, 1]$. Typical examples are:

$$\begin{aligned} q(x, y; .) &= 1_{\mathbf{R}_0^+}(f(x) - f(y) + T) \quad \text{or} && (3) \\ q(x, y; .) &= 1_{\mathbf{R}^+}(f(x) - f(y)) \quad \text{or} && (4) \\ q(x, y; .) &= 1_{\mathbf{R}_0^+}(f(x) - f(y)) + 1_{\mathbf{R}^-}(f(x) - f(y)) \cdot \exp\left(\frac{f(x) - f(y)}{T}\right) \quad , && (5) \end{aligned}$$

where (3) is used by *threshold accepting* methods proposed by Dueck and Scheuer (1988) for combinatorial problems and tested by Bertocchi and Di Odoardo (1992) for continuous variables, whereas (4) is applied by *evolution strategies* and (5) by *simulated annealing*.

Usually, the sampling distribution is chosen to be a uniform distribution on *bounded* regions, e.g., fixed (Wille and Vennik 1985; Khachaturyan 1986; Wille 1987) or adapted hypercubes (Vanderbilt and Louie 1984; Haines 1987; Corana, Marchesi, Martini, and Ridella 1987), and fixed (Bohachevsky, Johnson, and Stein 1986) or adapted hypersphere surfaces (Bertocchi and Sergi 1992). As it is not possible with those distributions to reach each state in M when trapped in a local minimum a mechanism must be provided that allows the possibility of transitioning to regions with worse objective function values. This is realized by using (5) in (2). However, in order to establish any convergence at all, the probability of accepting a worse point has to be decreased towards zero over time. Investigations into the global convergence properties of SA has mainly concentrated on the case of a finite or countable state space (see e.g. the review of Romeo and Sangiovelli-Vincentelli 1991). For continuous state spaces there are results in form of stochastic differential equations (Aluffi-Pentini, Parisi, and Zirilli 1985; Gelfand and Mitter 1991a; Gelfand and Mitter 1991b), whereas a global convergence proof of the original SA optimizing over general state spaces is given by Haario and Saksman (1991). Their result indicates that in case of SA the rate of decrease of parameter T in (5) has to be logarithmic as in the finite case: $T_t = T_0 / \log(t + 2)$.

Table 1: Typical cooling schedules used in practical applications

cooling type	schedule	references
logarithmic	$T_t = T_0/\log(t+2)$	Haario and Saksman (1991)
geometric	$T_t = c^t\, T_0$ with $c \in (0,1)$	Vanderbilt and Louie (1984) Wille and Vennik (1985) Wille (1987) Corana et al. (1987) Bertocchi and Sergi (1992)
subtractive	$T_t = \max\{0, T_0 - t\,\Delta T\}$	Haines (1987)
linear	$T_t = T_0/(t+1)$	Szu and Hartley (1987a) Szu and Hartley (1987b)
function value	$T_t = \alpha\, f(X_t) + \beta$	Bohachevsky et al. (1986)

Empirical results with this so–called *cooling schedule* T_t indicate that the time until convergence is of exponential order. Thus, other schedules are used in practical applications (see table 1) which provide faster but possibly nonglobal convergence. This problem can be circumvented by an appropriate choice of the generating distribution. Indeed, if M is bounded one might use the uniform distribution over M and global convergence for continuous functions follows from standard arguments (Devroye 1978) with $T_t \equiv 0$ for all $t \geq 0$. Szu and Hartley (1987a) claim that global convergence can be established by employing a multidimensional Cauchy distribution with density $g(x) = K_n\, T_t\, (T_t^2 + ||x||^2)^{-(n+1)/2}$, which concentrates trials around 0 according to the schedule $T_t = T_0/(t+1)$. The advantage achieved over the use of sampling distributions with bounded support is due to the fact that for each trial there exists a (small) probability to reach any state. Actually, under some conditions no cooling is necessary at all such that (5) becomes equivalent to (4) and global convergence can be guaranteed:

Theorem 1 (Solis and Wets 1981; Pintér 1984)
Let $f^* := \min\{f(x) \,|\, x \in M\} > -\infty$ and for the Lebesgue measure of the level sets $L_{f^*+\epsilon} := \{x \in M \,|\, f(x) \leq f^* + \epsilon\}$ holds $\mu(L_{f^*+\epsilon}) > 0$ for all $\epsilon > 0$. If

$$\sum_{t=0}^{\infty} Q(x_t, L_{f^*+\epsilon}) = \infty \;\; \forall \epsilon > 0 \tag{6}$$

then $f(X_t) \to f^*$ with probability one. □

For instance, let $Q_t(x_t, A) = \int_A g_t(y - x_t)\, dy$ with $A \in \mathrm{I\!R}^n$ be the generating distribution, where $g(.)$ denotes the density of a n–dimensional normal random vector with zero mean and covariance matrix $C_t = \sigma_t^2\, I$. If $\min\{\sigma_t \,|\, t > 0\} \geq \sigma > 0$ and $f(x) \to \infty$ for $||x|| \to \infty$, the lower level sets are bounded and there exists a minimum positive probability to hit the level set $L_{f^*+\epsilon}$ regardless of x_t. Thus, $\liminf_{t\to\infty} Q(x_t, L_{f^*+\epsilon}) > 0$ and the sum in (6) diverges. A related result is given in Bélisle (1992).
Although global convergence of the above type should be the minimum requirement of a probabilistic optimization algorithm it is more interesting to inquire into the finite

time behavior, i.e., the rate of convergence. For finite state spaces it is known that the convergence rate of the *probability* to reach the optimal state is of order $1 - O(t^{-a})$ with $a > 0$ depending on the problem (Chiang and Chow 1988). This is slow *asymptotic* convergence compared to the rate of convergence of pure random search which is of order $1 - O(\beta^t)$ with $\beta \in (0, 1)$. The latter expression converges much faster as $t \to \infty$, but empirical results indicate that SA is a much better optimization algorithm than pure random search. Therefore, this measure is of limited utility.

Another convergence measure is the expected error defined by $\delta_t := \mathrm{E}[f(X_t) - f^*]$. The following examples will reveal that the convergence rate of the sequence (δ_t) has different orders depending on whether a *fixed*, i.e., $Q_t(.) = Q_s(.)$ for all $s, t \geq 0$, or a *non-fixed* generating distribution $Q_t(.)$ is chosen.

EXAMPLE 1

Let $f(x) = \|x\|^2$, $M = S_n(r) \overset{def}{=} \{x \in \mathbb{R}^n : \|x\| \leq r\}$ with $0 < r < \infty$. Moreover, let Y_t have uniform distribution on M and use acceptance criterion (4). Then the c.d.f. of the objective function value per sample is

$$F(v) = \begin{cases} 0 & , v \leq 0 \\ v^{n/2}/r^n & , v \in (0, r^2) \\ 1 & , v \geq r^2 \ . \end{cases} \tag{7}$$

With $m_t := \min\{f(X_1), f(X_2), ..., f(X_t)\}$ and using results from extreme value statistics (Resnick 1987) one obtains $P\{m_t \leq v\} = 1 - (1 - F(v))^t$. With appropriate norming constants a_t, b_t there is weak convergence to a Weibull distribution:

$$F_{m_t}(v/a_t + b_t) = 1 - (1 - F(v/a_t + b_t))^t \overset{w}{\to} H_{3,\alpha}(v) = (1 - \exp(-v^\alpha)) \tag{8}$$

with $\alpha > 0$ and $v > 0$. A necessary and sufficient condition to apply (8) is that $v_L := \inf\{v \in \mathbb{R} \,|\, F(v) > 0\} > -\infty$ and that

$$\lim_{h \to 0+} \frac{F(v_L + v\,h)}{F(v_L + h)} = v^\alpha \tag{9}$$

for $v > 0$ and $\alpha > 0$. Then, a possible choice of a_t, b_t for the weak convergence is given by $a_t = 1/(\gamma_t - v_L)$, $b_t = v_L$ and $\gamma_t = F^{-1}(1/t)$.

Here, $v_L = 0$ and the limit (9) becomes $v^{n/2}$ such that $a_t = t^{2/n}/r^2$ and $b_t = 0$. It follows that δ_t converges to

$$\delta_t = \mathrm{E}[f(X_t) - f^*] = \mathrm{E}[m_t] \to a_t^{-1}\,\mathrm{E}[W] = r^2\,\Gamma(1 + \frac{2}{n})\,t^{-2/n} = O(t^{-2/n}) \ , \tag{10}$$

where W has Weibull distribution $H_{3,\alpha}$ with $\alpha = n/2$.

The above algorithm is nothing more than a pure random search algorithm. To recognize that there is not much gain from using a sampling distribution with fixed support, suppose that $M = \mathbb{R}^n$ and $Y_{t+1} = X_t + Z_t$ with $Z_t \sim U(S_n(r)), r \in (0, \infty)$ fixed. After t_0 steps this algorithm reaches a point x_{t_0} with $\|x_{t_0}\| < r/2$. The algorithm subsequently resembles pure random search on the region $S_n(r)$ and the convergence rate of δ_t declines to $O(t^{-2/n})$ for $t \geq t_0$. □

More generally:

Theorem 2 (Rappl 1984)
Let f be (m, M)–strongly convex, i.e., f is continuously differentiable and with some constants $m > 0, M \geq 1$ there holds

$$m \|x-y\|^2 \leq (\nabla f(x) - \nabla f(y))'(x-y) \leq m \cdot M \|x-y\|^2$$

for all $x, y \in M$. If the generating distribution $Q(.)$ is fixed, the expected error δ_t decreases with $O(t^{-2/n})$ for any starting point $x_0 \in M$. □

The next example will demonstrate that the convergence rate of δ_t can be accelerated substantially when using an appropriate non–fixed generating distribution.

Example 2
Again, consider the $(2,1)$–strongly convex problem with $f(x) = \|x\|^2$ and $M = \mathbb{R}^n$. The sampling vector Z_t is chosen to be multinormally distributed with zero mean and covariance matrix $C_t = \sigma^2 I$. Consequently, for the distribution of the objective function values we have $f(x_t + Z_t) \sim \sigma^2 \chi_n^2(\kappa)$, where $\chi_n^2(\kappa)$ denotes a noncentral χ^2–distribution with n degrees of freedom and noncentrality parameter $\kappa = \|x_t\|^2/\sigma^2$. Using the fact that as $n \to \infty$,

$$\frac{\chi_n^2(\kappa) - (n+\kappa)}{\sqrt{2(n+2\kappa)}} \to N \sim N(0,1) \ ,$$

(Johnson and Kotz 1970, p. 135) the limit distribution of the relative variation of objective function values defined as $V \overset{def}{=} (f(x_t) - f(X_{t+1}))/f(x_t)$ becomes

$$V \ = \ 1 - \frac{\sigma^2}{\| x_t \|^2} \chi_n^2(\kappa) \ \to \ -\frac{s^2}{n} - \frac{s^2}{n}\sqrt{\frac{2}{n} + \frac{4}{s^2}} N \ \approx \ -\frac{s^2}{n} - \frac{2s}{n} N \ \ (n >> 1)$$

with $\sigma_t = s \| x_t \|/n$. As the algorithm only accepts improvements, we are interested in the expectation of the random variable $V^+ = \max\{0, V\}$, which is given by

$$\mathrm{E}[V^+] \ = \ \frac{1}{n}\left\{ s\sqrt{\frac{2}{\pi}} \exp\left(-\frac{s^2}{8}\right) - s^2\left[1 - \Phi\left(\frac{s}{2}\right)\right]\right\}$$

where $\Phi(.)$ denotes the c.d.f. of a standard normal random variable. The expectation becomes maximal for $s^* = 1.224$ (see fig. 1) such that $\mathrm{E}[V^+] = 0.405/n$ and

$$\sigma^* = \frac{1.224}{n}\|x\| \ = \ \frac{0.612}{n}\|\nabla f(x)\| \ .$$

This value is also given in Rechenberg (1973). The dependence on the problem dimension n is of importance: Geometric convergence can still be guaranteed if this factor is omitted but it will be very slow compared to the optimal setting (see fig. 1). □

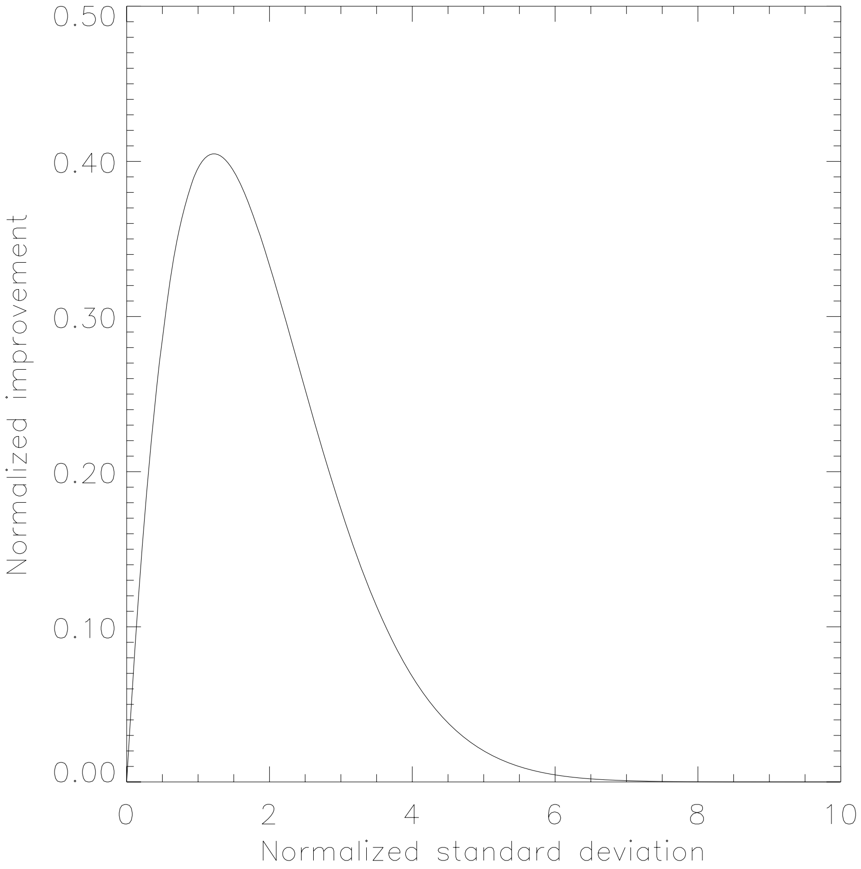

Figure 1: Normalized improvement versus normalized standard deviation

More generally:

THEOREM 3 (Rappl 1984; Rappl 1989)
Let f be (m, M)–strongly convex. If the generating distribution is non–fixed such that $Z_t \overset{d}{=} R_t \cdot U$, where U is a random vector uniformly distributed on a hypersphere surface of dimension n and $R_t = \|\nabla f(x_t)\| \cdot R$ with some positive random variable R with nonvoid support on $(0, a)$, $a < \infty$, then the expected error δ_t decreases with $O(\beta^t)$, $\beta \in (0, 1)$. □

It is clearly not possible to get convergence rates of order $O(\beta^t)$ for δ_t when using a logarithmic cooling schedule. Let $\Delta f_t = f(X_t) - f(x_{t-1})$. This random variable can be decomposed into $\Delta f_t = \Delta f_t^+ - \Delta f_t^-$ with $\Delta f_t^+ = \max\{\Delta f_t, 0\}$ and $\Delta f_t^- = \max\{-\Delta f_t, 0\}$. The expected variation of the objective function value $E(\Delta f_t)$ is the difference between expected improvement and expected deterioration: $E(\Delta f_t) = E(\Delta f_t^+) - E(\Delta f_t^-)$. From Example 2 we get $E[\Delta f_t^+] = 0.405\,(f(x_t) - f^*)/n$ and $E[\Delta f_t] = -1.224^2\,(f(x_t) - f^*)/n$ so that $E[\Delta f_t^-] = 1.903\,(f(x_t) - f^*)/n$. If the process would converge geometricly then the probability of accepting a worse point tends to one when using a logarithmic cooling $T_t = T_0 / \log(t+2)$:

$$\exp(-\Delta f_t^- / T_t) = \exp\left(-\left(\frac{1.903}{n}\right)^t \log(t+2)\,(f(x_0) - f^*)/T_0\right) \to 1 \text{ as } t \to \infty \ .$$

This might explain why several authors have used a geometric cooling schedule in their versions of simulated annealing, even though the guarantee of global convergence no longer holds. The conflicting goals of global convergence and fast convergence speed could be satisfied if it were possible to adapt and concentrate the support of the generating distribution to the lower level sets at each step t. In this case geometric convergence of δ_t can be shown even for Lipschitz–continuous functions with several local minima (Zabinsky and Smith 1992). This idea will be reconsidered when discussing massively parallel Markovian algorithms.

3 Parallel simulated annealing

The Markovian algorithms considered so far are not well–suited for parallelization. For finite state space variants some proposals are surveyed in Greening (1990). A straightforward method to take advantage of parallel hardware is to perform, say p, trials in parallel on p processors and to select the best. This is the idea of the so–called $(1,p)$–evolution strategies (Schwefel 1981) and it can be used for SA as well (Bertocchi and Sergi 1992). However, a simple example reveals that the speedup is less than $O(p)$ even for strongly convex functions:

Example 3
For the same problem as in Example 2 consider the following algorithm: instead of performing only one trial, perform $p > 1$ trials and accept the best trial among these according to some acceptance probability $q_t(.)$. Then the expected improvement given for parallel ES in Bäck, Rudolph, and Schwefel (1993) can be generalized to

$$\mathrm{E}[V_p] = \frac{1}{\eta} \int_{-\infty}^{\infty} u \cdot g\left(\frac{u-\theta}{\eta}, p\right)\, q_t(u;.)\, du$$

with $\theta = -2\sigma^2 n / \|\nabla f(x_t)\|^2$, $\eta = 4\,\sigma / \|\nabla f(x_t)\|$, and where

$$q_t(u;.) = \begin{cases} 1_{\mathbf{R}^+}(u) & \text{, for } (1+p)\text{–ES} \\ 1_{\mathbf{R}^+}(u) + 1_{\mathbf{R}_0^-}(u) \cdot \exp(u/T_t) & \text{, for SA}_p \\ 1_{\mathbf{R}^+}(u) + 1_{\mathbf{R}_0^-}(u) & \text{, for } (1,p)\text{–ES} \end{cases} \tag{11}$$

and

$$g(x,p) := \frac{d\Phi^p(x)}{dx} = \frac{p}{\sqrt{2\,\pi}} \exp(-x^2/2)\,\Phi(x)^{p-1} \ .$$

Obviously,

$$\mathrm{E}[V_p^+] \geq \mathrm{E}[V_p^{SA}] \geq \mathrm{E}[V_p] \tag{12}$$

where acceptance probabilities in (11) are used from top to bottom. $\mathrm{E}[V_p^+]$, $\mathrm{E}[V_p^{SA}]$ and $\mathrm{E}[V_p]$ denote the expected improvement for a $(1+p)$–ES, SA with p trials and a $(1,p)$–ES, respectively. Inequality (12) indicates, that the expected improvement for SA with p trials is somewhere between the expected improvement of a $(1+p)$–ES and a $(1,p)$–ES. For increasing p the gap between the inequalities (12) becomes successively closer leading to equality for $p \to \infty$. Therefore, the difference between evolution strategies and simulated annealing decreases as more trials are performed in parallel. Moreover, as pointed out in Bäck et al. (1993), the optimal expected improvement (i.e., using optimal $\sigma^* = \sqrt{2\log p}\,\|\nabla f(x)\|/2n)$ increases asymptotically as $\mathrm{E}[V_p] \approx 2\log p/n$, so that the speedup is only of order $O(\log\, p)$ if p processors are used. □

Although the expected speedup may appear quite low, these variants can be supplied with the property to self–adapt their generating distributions. This idea was first realized by Schwefel (1977) in the following manner:
Using biological terminology, let $(x_t, \sigma_t) \in \mathbb{R}^n \times \mathbb{R}^+$ represent an *individual* at step t, where the components of vector x_t are phenotypic traits and σ_t is regarded as a *gene*. In example 2 a new trial point was generated via $X_{t+1} = x_t + Z_t$, where Z_t was a random vector normally distributed with zero mean and covariance matrix $C_t = \sigma_t^2 \cdot I$, $\sigma_t = c \cdot \|\nabla f(x_t)\|$ for some constant $c > 0$. This operation may be regarded as a *pleiotropic mutation* of the original vector x_t . Here, the generating distribution was controlled deterministically by exploitation of the gradient information. Whenever the gradient information is not available another method can be employed: Before mutating vector x_t the distribution control parameter σ_t is mutated by multiplication with a lognormal random variable $S_t = \exp(N_t)$, where N_t is a normal random variable with zero mean and standard deviation $\tau = c \cdot n^{-1/2}$ with some constant $c > 0$.
With this mechanism the algorithm self–adapts its generating distribution: Assume that $\sigma_t > 0$ and that the optimal new value for σ_{t+1} is σ^*. Now σ_t is multiplied with a lognormally distributed random variable with support $(0, \infty)$. Therefore, the probability to sample a new σ_{t+1} in an ϵ–neighborhood of σ^* is $P(|\sigma_{t+1} - \sigma^*| < \epsilon) = p_\epsilon$, say. If we perform k independent trials, the probability to sample a new σ "close" to σ^* is $1 - (1 - p_\epsilon)^k \to 1$ as $k \to \infty$ for all $\epsilon > 0$.

Example 4
Let $f(x) = \|x\|^2$ with $n = 30$ and consider a $(1,p)$–ES with $p = 10$. The following experiment demonstrates the self–adapting capabilities of evolution strategies: Using the starting point $x_0 = (100, 100, \ldots, 100)'$, the optimal control parameter value would be $\sigma_0 = 28.0937$. In the experiment, this value was set to $\sigma_0 = 0.1$, which offers only small expected progress. As shown in Figure 2 a near–optimal value was obtained after about 260 generations. From then on the self–adapted control parameter σ_t was close to the optimal value, so that the convergence speed was accelerated substantially. Note that two quantities are optimized simultaneously: The objective function as well as the generating distribution. The feedback for adapting the generating distribution stems from the quality of the realized random vectors, i.e., the feedback is not direct. This is sometimes called *second level learning* (Hoffmeister and Bäck 1992). □

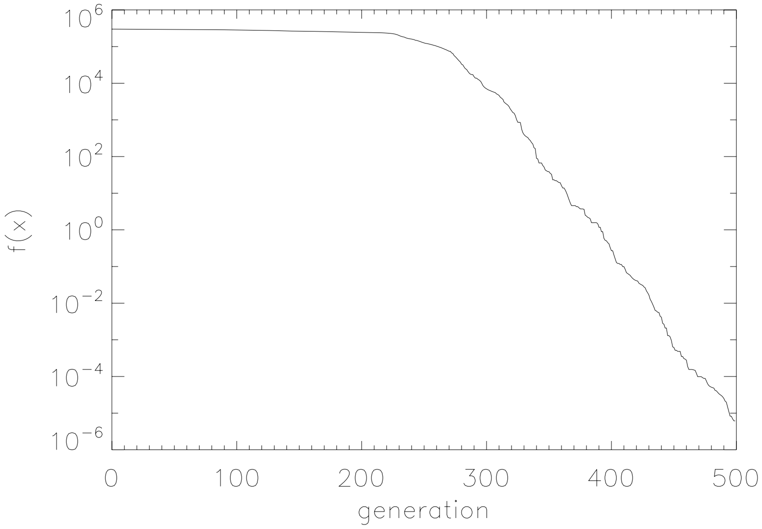

Figure 2: Best objective function value $f(x)$ over time

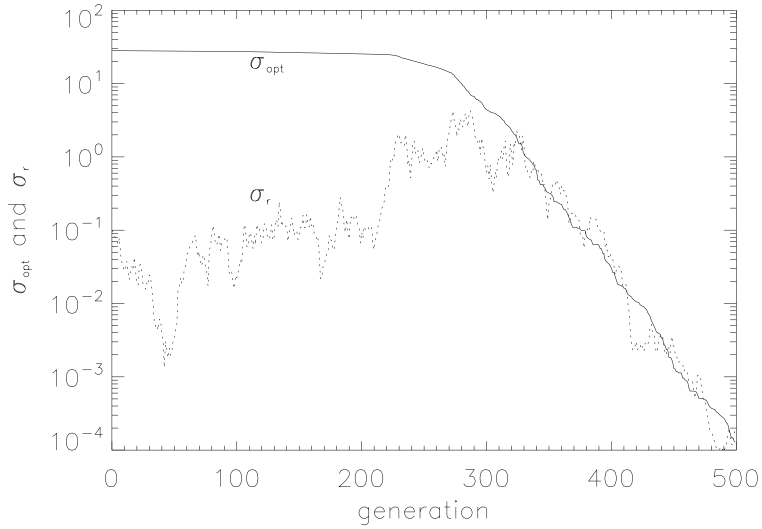

Figure 3: Self–adaptation of generating distribution: σ_{opt} denotes the optimal distribution control parameter whereas σ_r denotes the control parameter which was realized by the algorithm

4 Massively parallel simulated annealing

Another straightforward parallelization scheme is to run the sequential algorithm on p processors independently (Bertocchi and Sergi 1992) as a parallel version of the well–known multistart technique. This is well suited for SIMD parallel computers which perform the same instruction on p processors in parallel but on different data streams. A more sophisticated parallel algorithm should use the information gathered by the other processors. But in general it is an open question as to which information should be exchanged between the processors in order to accelerate the search on the one hand and to increase the global convergence reliability on the other. Again, natural evolution can be regarded as a guide for some design decisions.

4.1 Parallel neighborhood models

The essential idea of neighborhood models is to supply the population with a spatial structure which may be defined as any connected graph. On each node there is an individual which communicates with its nearest neighbors in the graph. These nearest neighbors are regarded to be the neighborhood of the individual. This model was used by Mühlenbein, Gorges-Schleuter, and Krämer (1988) and appeared later under terms like *plant pollination model* (Goldberg 1989), *parallel individual model* (Hoffmeister 1991) or *diffusion model*. The first implementations of the model onto a parallel machine has been conducted by Gorges-Schleuter (1989), Sprave (1990), Palmer and Smith (1991) and Spiessens and Manderick (1991), whereas Manderick and Spiessens (1989) tested a simulation on a uniprocessor machine. All these implementations have been developed for genetic algorithms in order to solve combinatorial or pseudoboolean optimization problems. The first usage of this model in the context of optimization over continuous variables appeared in Rudolph (1992).
In most cases the spatial structure of the population was adapted to the processor communication network of the underlying parallel machine. As demonstrated in Sprave (1990) it is useful to design algorithms of the above type for SIMD–type parallel computers, because these algorithm can be mapped onto MIMD–type computers easily. Therefore, the most usual spatial population structures are meshes or toroids.

4.2 Design of the neighborhood algorithm

Here, the neighborhood algorithm (NA) was designed to run on a torus topology, where each node of the torus has a label (i,j) with $i = 0, 1, \ldots, K_1$ and $j = 0, 1, \ldots, K_2$. In particular:

```
On each node $(i,j)$:
    initialize $x_0 \in M \subseteq \mathbb{R}^n$ and $\sigma_0 > 0$
    set $t = 0$
    repeat
        if recombination = true then
            let $(x_{t,0}, \sigma_{t,0})$ be a recombination of $(x_t, \sigma_t)$ and a neighbor
        else
            $(x_{t,0}, \sigma_{t,0}) = (x_t, \sigma_t)$
        endif
        $\sigma_{t,0} := \sigma_{t,0} \cdot S_t$
        $x_{t,0} := x_{t,0} + \sigma_{t,0} \cdot Z_t$
        get neighbors $(x_{t,1}, \sigma_{t,1}), (x_{t,2}, \sigma_{t,2}), \ldots, (x_{t,m}, \sigma_{t,m})$
        select $x_{t,b}$ with $f(x_{t,b}) = \min\{f(x_{t,0}), f(x_{t,1}), \ldots, f(x_{t,m})\}$
        if $a(x_t, x_{t,b}; .) = 1$ then
            $(x_{t+1}, \sigma_{t+1}) = (x_{t,b}, \sigma_{t,b})$
        else
            $(x_{t+1}, \sigma_{t+1}) = (x_t, \sigma_t)$
        endif
        increment $t$
    until termination criterion fulfilled
```

In the above algorithm (x_t, σ_t) denotes the sequence of accepted points. The pair $(x_{t,0}, \sigma_{t,0})$ contains the new generated (not yet accepted) individual on the node. The random vector Z_t is normally distributed with zero mean and the unit matrix as its covariance matrix, whereas S_t is a lognormally distributed random variable with parameter $\tau = n^{-1/2}$. The size m of the neighborhood is assumed to be constant for all nodes which seems most natural due to the regularity of the processor communication network. The experiments have been performed with several neighborhood sizes and structures, which will be discussed in section 5. The acceptance function $a(.)$ already described in (2) can be chosen. It remains to clarify how to choose a neighbor when recombination is desired. In this implementation two variants have been tried: Choose a neighbor at random or choose the best one of the neighborhood. Other variants can be imagined (Gorges-Schleuter 1992). Similarly, many recombination operators are possible. Here, the following variant (*hypercube recombination*) has been used: Let $x, y \in \mathbb{R}^n$ be the parent and $v \in \mathbb{R}^n$ be the result of recombination. Then $v = x + (y - x) \cdot \xi$, where ξ is a random vector uniformly distributed in $[0,1]^n$. Consequently, the offspring v is always located within the hypercube defined by the parent x and y.
The self-adaptation mechanism described in Section 3 does also work for the neighborhood algorithm: The initialization routine produces individuals with diverse generating distributions so that the search has global character in the first phase. After some generations there emerge several clusters of "similar" neighboring individuals. The neighborhood of individuals not located at the border of a cluster consists of nearly identical individuals and each of them generates an offspring with nearly the same generating distribution, so that we may view the generation scheme of individuals in the inner part of the cluster as a $(1, m+1)$-type scheme.

Convergence to the global optimum can be guaranteed under the conditions of Theorem 1. But the main advantage of NA is its property that local solutions, which are the best solutions known at step t, spread over the population quite slowly preventing the extinction of individuals, which may lead to the global solution. This behavior can be demonstrated by considering the evolution of the mixture density induced by the generating densities of all individuals within the population.

4.3 Evolution of the mixture distribution

Let $f_i(v;\theta_i)$ denote probability density functions with parameter vectors θ_i. Then

$$f(v) = \sum_{i=1}^{p} c_i \cdot f_i(v;\theta_i) \tag{13}$$

with mixing weights $c_i > 0$, $\sum_{i=1}^{p} = 1$ and finite p is called a *finite mixture density function* (Titterington, Smith, and Makov 1985).

Regarding $f_1, \ldots f_p$ to be the generating densities of the individuals (x_i, σ_i) and setting $c_i = 1/p$ $(i = 1, \ldots, p)$, then the mixture density (13) may be viewed as the generating density of the entire population. Here, parameter vector θ_i gathers the mean x_i (the object variable vector) and the covariance matrix $C_i = \sigma_i^2 I$ of the generating normal density $N(x_i, C_i)$ of each individual (x_i, σ_i). Using (13) it is possible to visualize the evolution and the motion of the mixture density of the population for problems of dimension $n = 2$.

Example 5

Let the feasible region be $M = [-2, 2]^2$ for the following objective function:

$$f(x) = \begin{cases} (x_1 - 1)^2 + (x_2 - 1)^2 & , \text{if } x_1 \geq 0, x_2 \geq 0 \\ (x_1 - 1)^2 + (x_2 + 1)^2 + \epsilon & , \text{if } x_1 \geq 0, x_2 \leq 0 \\ (x_1 + 1)^2 + (x_2 - 1)^2 + \epsilon & , \text{if } x_1 < 0, x_2 > 0 \\ (x_1 + 1)^2 + (x_2 + 1)^2 + \epsilon & , \text{if } x_1 < 0, x_2 < 0 \end{cases} \tag{14}$$

with $\epsilon = 0.001$. The global minimum is $f^* = 0$ at point $(1,1)'$ and there are 3 local minimal points $(-1,-1)'$, $(-1,1)$ and $(1,-1)$ with $f = \epsilon > 0$. The NA used recombination and acceptance criterion (4) on a torus of size 10 × 10. Figures 4 and 5 illustrate the evolution of the mixture distribution. □

If there is only one global optimum point x^* and the NA converges to the global minimum, it is necessary that the means x_i tend to x^*, i.e., the mixing density concentrates its probability mass around the global optimum point. Theorem 1 guarantees that the NA converges to the global optimum if $\sigma = const$ over time. Therefore, it is not necessary that the mixing distribution converges weakly to the Dirac delta distribution with expectation x^* and zero covariance matrix. But Theorem 2 indicates that the convergence rate will be of the same order as the rate for pure random search. Zabinsky and Smith (1992) proved that a geometric rate, as in Theorem 3, could be achieved even for multimodal Lipschitzian functions, if it was be possible to adapt the support of the generating distributions to the lower level sets. Of course, this is possible only for special problems and with additional knowledge about the objective function, because the support will split into disconnected sets, which are unknown in general. As every distribution can be described as a mixture of an *infinite* number of normal densities, it might be possible to approximate such a distribution if the population size were sufficiently large and if the individuals' generating distributions were adapted appropriately. In this case the covariance matrix should tend to a zero matrix, i.e., $\sigma \to 0$. This happens in Example 5. It is, however, an open question as to which problem classes the probabilistic mechanism for adapting the generating distribution provides global convergence as well as fast convergence. Therefore, we only can refer to empirical results indicating that this mechanism works well in most cases.

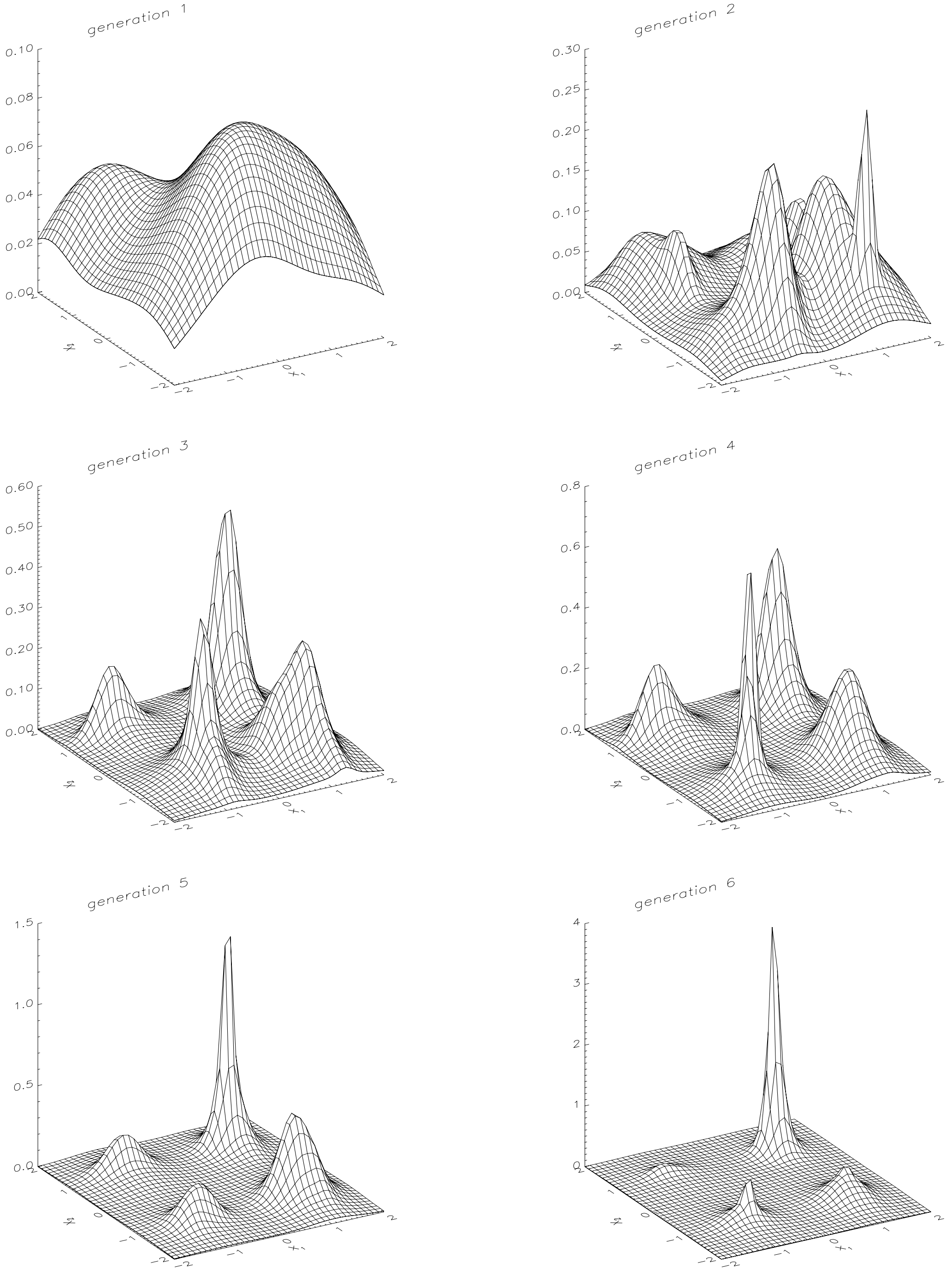

Figure 4: Evolution of the density of the joint generating distribution: At the first generation there is nearly a uniform distribution over the feasible region. Note that the scaling of the z–values are different for all plots. At the second generation the probability mass is moved towards the four local minimal points. The global minimal point $(1, 1)'$ attracts only little probability mass compared to the other three local minimal points. This changes at the third generation. Now the global minimal point attracts more and more probability mass at the expense of the local solutions, whose regions however are still explored.

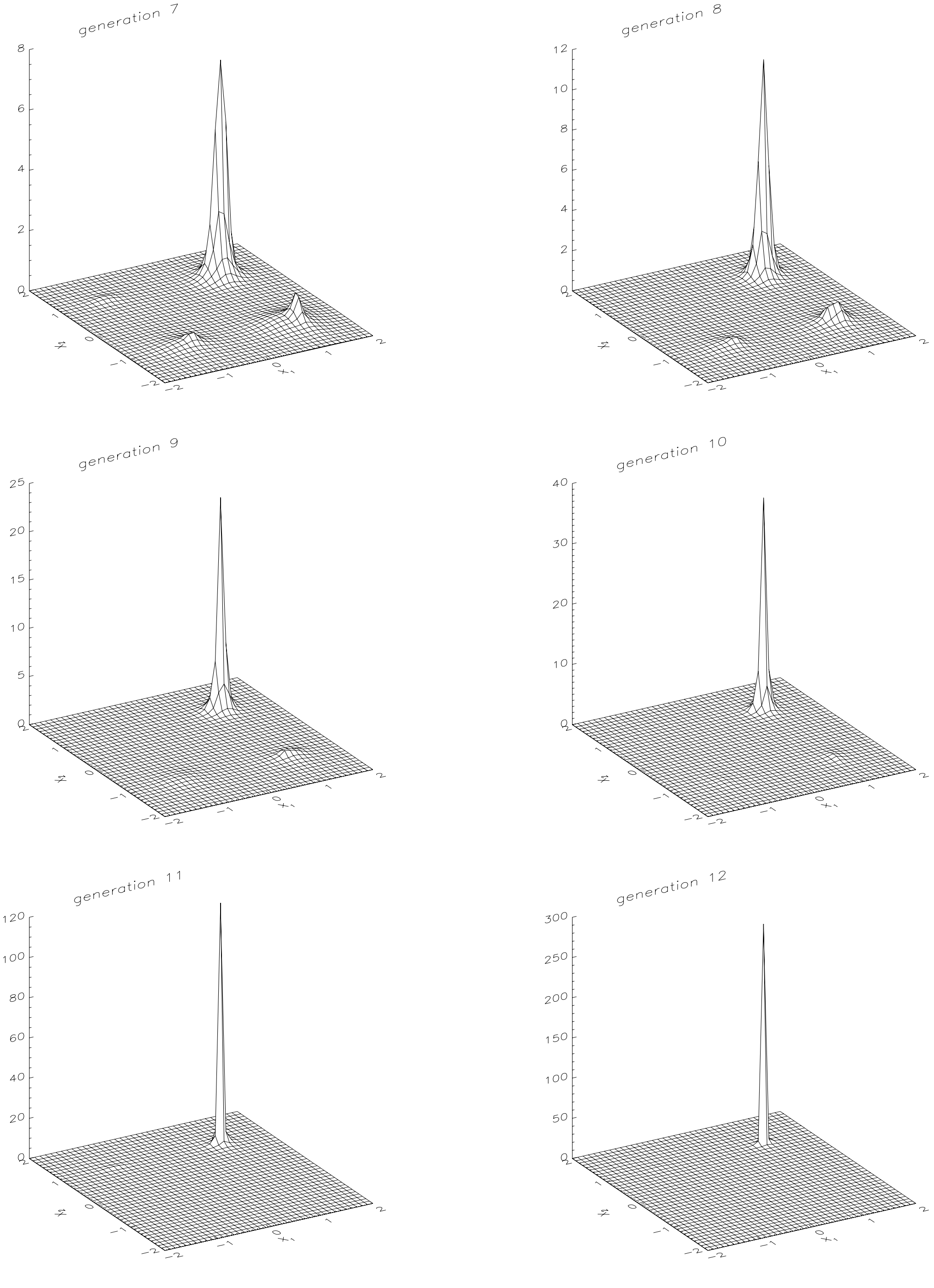

Figure 5: Evolution of the density of the joint generating distribution: The distribution becomes more and more peaky around the global optimum point. At the 12–th generation the distribution has lost its multimodality and converges to the Dirac distribution at the global minimal point $(1, 1)'$.

5 Computational results

Although convergence to the global minimum can be guaranteed under some conditions, nothing is known about the time complexity. Therefore, the NA was tested for several global optimization problems to achieve a preliminary assessment of its behavior.

5.1 Test problems

Test problem f_1 is a strictly convex problem and should not pose a problem for the NA. It was included into the testbed to check its convergence rate if the population is attracted by a local minimum point, whereas test problem f_2 was included to check the algorithm's ability to conquer plateaus. Test problems f_3 and f_4 taken from Schaffer et al. (1989) possess local minimal points which are arranged in sets with a spherical structure. Test problems f_5–f_7 are known as the Shekel problems, which have 5, 7 and 10 local minima, respectively. For test problems f_8 and f_9 the number of local minima increases exponentially with the problem dimension. In both problems the local minimum points are arranged regularly in the feasible region. Test problems f_5–f_9 are taken from Törn and Zilinskas (1989).

Test problem	n	M	f^*
$f_1(x) = \sum_{i=1}^{n} x_i^2 = \|x\|^2$	30	$[-100, 100]^n$	0
$f_2(x) = \sum_{i=1}^{n} \lfloor x_i + 0.5 \rfloor^2$	30	$[-100, 100]^n$	0
$f_3(x) = 0.5 + (\sin^2 \|x\| - 0.5)/(1 + 0.001\|x\|^2)^2$	2	$[-100, 100]^n$	0
$f_4(x) = \|x\|^{1/2} [\sin^2(50\|x\|^{1/5}) + 1]$	2	$[-100, 100]^n$	0
$f_5(x) = -\sum_{i=1}^{5} [(x - A_i)(x - A_i)' + c_i]^{-1}$	4	$[0, 10]^n$	-10.1532
$f_6(x) = -\sum_{i=1}^{7} [(x - A_i)(x - A_i)' + c_i]^{-1}$	4	$[0, 10]^n$	-10.4029
$f_7(x) = -\sum_{i=1}^{10} [(x - A_i)(x - A_i)' + c_i]^{-1}$	4	$[0, 10]^n$	-10.5364
$f_8(x) = \|x\|^2/4000 - \prod_{i=1}^{n} \cos(x_i/\sqrt{i}) + 1$	10	$[-600, 600]^n$	0
$f_9(x) = \|x\|^2 + 10 \sum_{i=1}^{n} [1 - \cos(2\pi x_i)]$	30	$[-5, 5]^n$	0

The coefficients of the Shekel functions are summarized below:

i	A_i				c_i
1	4	4	4	4	0.1
2	1	1	1	1	0.2
3	8	8	8	8	0.2
4	6	6	6	6	0.4
5	3	7	3	7	0.4
6	2	9	2	9	0.6
7	5	5	3	3	0.3
8	8	1	8	1	0.7
9	6	2	6	2	0.5
10	7	3.6	7	3.6	0.5

5.2 Parametrization of the neighborhood algorithm

The NA was tested on a 64 × 256–torus, so that 16384 individuals were evolved in parallel. To check whether the potential success of the algorithm is caused by the large number of individuals (search trajectories), the algorithm was applied to each problem without any communication. In other words, the simple sequential algorithm was run 16384 times in parallel with different random generator seeds (*multistart technique*). All other runs used either a *von Neumann* or a *Moore* neighborhood for communication. Figure 6 illustrates the difference.

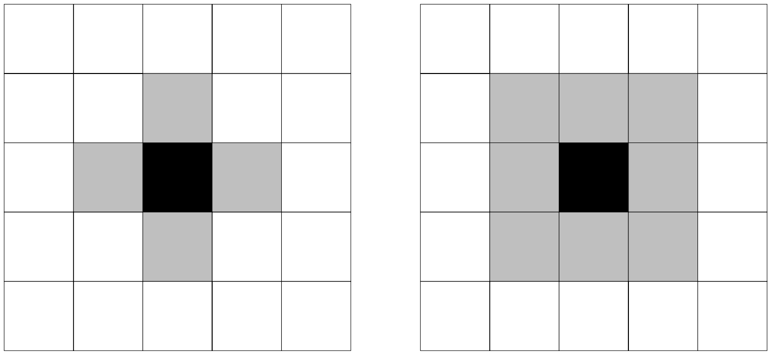

Figure 6: An individual (black) has four neighbors (grey) in a von Neumann neighborhood (left side) and eight neighbors in a Moore neighborhood (right side).

Two mating strategies were tested: Either choose the best individual from the neighborhood for recombination or choose it at random from the neighborhood.
A geometricly temperature schedule $T_t = \beta^t$ was chosen with $\beta = 0.99$, $\beta = 0.97$ and $\beta = 0$. Note that the variant $\beta = 0$ is an evolution strategy variant: An offspring is selected if and only if it is better than its parent on the specific grid element. The maximal number of generations was limited to 300 for problems f_1 and f_2, 200 for f_3 and f_4, 100 for f_5–f_7 and 400 for f_8 and f_9.

5.3 Test summary

The NA was applied ten times for each parameter setting. Each problem was regarded as solved when the algorithms hit the level set $L_\epsilon = \{x \in M : f(x) - f^* < \epsilon\}$ with $\epsilon = 10^{-5}$. These first hitting times were averaged over the 10 runs. The following tables summarize the results obtained. The mating strategy is labeled as *no, random* and *best* in the column *Reco.* The neighborhood structure (*NBH*) is abbreviated with *no*, *vN* (von Neumann) and *Mo* (Moore). Column *Hits* denotes the number of events where the level set was hit at all, which was the basis for the mean first hitting times (*Mean*) and the empirical standard deviation (*Dev*).
None of the 10 × 16384 "sequential" runs approximated the solution of problem f_1 up to the desired accuracy within 300 steps. Although there was actual convergence, the rate was very low because the self–adaptation of the sampling distribution only works

if competition is present: As the neighborhood increases, fewer iterations are required. A further source of acceleration is offered by recombination: It only takes half the time to hit the desired level set. The choice of the acceptance function does not seem to have a significant effect. This trend is continued for problem f_2. Competition is necessary to self–adapt the sampling distribution and recombination accelerates the search substantially.

The situation changes for the the multimodal test problems $f_3 - f_8$: Now competition is the major source of improvement. Note that the results for f_3 and f_4 without recombination/mating are better than those for random mating and only slightly worse than those for best mating. But these problems are of low dimension, so it may be that mutation alone is sufficient to search the feasible region effectively. Seemingly, random mating is not a good advice: The results for best mating are better than those for random mating regardless of the test problem under consideration.

Table 2: Test results for problems f_1–f_4.

f_1		$\beta = 0.99$			$\beta = 0.97$			$\beta = 0.00$		
Reco	NBH	Hits	Mean	Dev	Hits	Mean	Dev	Hits	Mean	Dev
no	no	0	—	—	0	—	—	0	—	—
	vN	9	296.89	1.85	4	300.75	0.43	10	289.90	3.08
	Mo	10	219.90	2.30	10	217.90	3.33	10	218.80	2.04
random	vN	10	141.20	1.89	10	140.70	0.90	10	153.70	0.78
	Mo	10	119.60	0.92	10	118.50	0.92	10	128.10	0.70
best	vN	10	126.80	0.75	10	126.40	0.92	10	134.50	1.20
	Mo	10	105.10	0.94	10	105.40	0.49	10	110.80	0.75

f_2		$\beta = 0.99$			$\beta = 0.97$			$\beta = 0.00$		
Reco	NBH	Hits	Mean	Dev	Hits	Mean	Dev	Hits	Mean	Dev
no	no	0	—	—	0	—	—	0	—	—
	vN	10	168.80	6.63	10	167.10	7.06	10	157.80	6.72
	Mo	10	119.80	3.25	10	118.90	6.04	10	121.10	4.61
random	vN	10	67.40	0.80	10	67.40	0.80	10	80.10	1.97
	Mo	10	57.00	1.18	10	56.60	0.92	10	66.10	1.22
best	vN	10	60.30	0.78	10	60.50	1.12	10	69.60	1.11
	Mo	10	51.00	0.63	10	50.70	0.78	10	56.70	0.78

f_3		$\beta = 0.99$			$\beta = 0.97$			$\beta = 0.00$		
Reco	NBH	Hits	Mean	Dev	Hits	Mean	Dev	Hits	Mean	Dev
no	no	3	166.67	54.82	1	104.00	0.00	8	202.00	55.95
	vN	10	39.60	19.42	9	28.00	9.09	10	37.30	14.69
	Mo	10	25.30	9.21	10	24.60	6.41	10	26.00	6.02
random	vN	10	41.90	3.99	10	39.80	5.38	10	48.70	8.52
	Mo	10	31.50	3.72	10	28.10	4.32	10	32.10	3.73
best	vN	10	30.80	4.58	10	29.80	3.16	10	32.40	4.08
	Mo	10	26.60	3.95	10	24.90	4.91	10	26.00	2.61

f_4		$\beta = 0.99$			$\beta = 0.97$			$\beta = 0.00$		
Reco	NBH	Hits	Mean	Dev	Hits	Mean	Dev	Hits	Mean	Dev
no	no	0	—	—	0	—	—	0	—	—
	vN	10	58.00	1.84	10	58.80	1.60	10	58.90	2.39
	Mo	10	48.60	0.92	10	48.20	1.08	10	47.20	1.33
random	vN	10	75.10	1.70	10	74.20	2.64	10	79.10	2.62
	Mo	10	55.60	1.91	10	56.60	1.69	10	56.70	2.19
best	vN	10	57.30	1.42	10	56.30	1.55	10	57.60	2.29
	Mo	10	48.60	1.02	10	48.70	1.73	10	50.70	1.27

Table 3: Test results for problems f_5–f_8

f_5		$\beta = 0.99$			$\beta = 0.97$			$\beta = 0.00$		
Reco	NBH	Hits	Mean	Dev	Hits	Mean	Dev	Hits	Mean	Dev
no	no	0	—	—	0	—	—	0	—	—
	vN	10	29.00	1.26	10	29.20	0.98	10	29.60	1.11
	Mo	10	23.30	0.64	10	23.10	0.94	10	23.00	1.18
random	vN	10	29.60	1.11	10	29.50	1.63	10	31.30	1.00
	Mo	10	24.00	1.00	10	23.50	1.20	10	24.40	1.36
best	vN	10	24.10	0.94	10	23.90	0.70	10	24.60	1.28
	Mo	10	20.60	1.02	10	21.30	0.90	10	21.00	0.89

f_6		$\beta = 0.99$			$\beta = 0.97$			$\beta = 0.00$		
Reco	NBH	Hits	Mean	Dev	Hits	Mean	Dev	Hits	Mean	Dev
no	no	0	—	—	0	—	—	2	91.50	2.50
	vN	10	28.00	2.05	10	27.90	1.37	10	27.50	1.36
	Mo	10	22.90	0.83	10	23.30	1.00	10	22.10	1.70
random	vN	10	28.40	2.01	10	27.80	1.40	10	29.20	1.60
	Mo	10	23.00	0.77	10	22.00	1.79	10	23.10	1.64
best	vN	10	23.30	0.90	10	22.60	1.28	10	23.70	1.00
	Mo	10	20.40	0.92	10	19.80	0.87	10	20.60	0.49

f_7		$\beta = 0.99$			$\beta = 0.97$			$\beta = 0.00$		
Reco	NBH	Hits	Mean	Dev	Hits	Mean	Dev	Hits	Mean	Dev
no	no	0	—	—	0	—	—	0	—	—
	vN	10	28.70	1.79	10	29.20	1.40	10	28.40	2.01
	Mo	10	23.00	0.77	10	23.10	0.70	10	23.60	0.80
random	vN	10	29.30	0.90	10	28.60	1.50	10	30.20	1.89
	Mo	10	21.60	6.93	10	23.00	1.48	10	24.50	1.12
best	vN	10	23.90	0.94	10	24.20	1.54	10	24.70	0.90
	Mo	10	21.00	1.00	10	20.30	1.35	10	20.90	1.14

f_8		$\beta = 0.99$			$\beta = 0.97$			$\beta = 0.00$		
Reco	NBH	Hits	Mean	Dev	Hits	Mean	Dev	Hits	Mean	Dev
no	no	0	—	—	0	—	—	0	—	—
	vN	10	112.90	4.50	9	112.33	3.43	9	110.44	2.67
	Mo	8	84.50	2.40	10	86.20	1.60	10	86.60	4.22
random	vN	10	82.40	1.50	10	80.80	0.87	10	87.10	1.30
	Mo	10	67.50	1.12	10	67.20	1.66	10	70.50	1.28
best	vN	10	69.20	1.72	10	69.10	1.37	10	72.60	1.02
	Mo	10	58.40	2.91	10	58.80	1.08	10	60.40	1.36

Table 4: Test result for problem f_9 with discrete recombination

f_9		$\beta = 0.00$		
Reco	NBH	Hits	Mean	Dev
random	vN	10	340.3	11.93
	Mo	10	245.3	7.52
best	vN	10	264.3	16.95
	Mo	10	213.7	10.73

Similar conclusions may be drawn for test problems $f_5 - f_7$: Random mating is as good as no mating and best mating produces only slightly better results. Moreover, a larger neighborhood seems to accelerate the search, whereas the choice of a specific acceptance criterion does not offer a significant advantage.

Test problem f_8 is 10–dimensional and possesses many local minima. Now recombination helps in searching the feasible region more effectively and produces better results than mutation alone. Again, best mating is better than random mating and mating in a Moore neighborhood is better than mating in a von Neumann neighborhood.

Test problem f_9 was not solved by any variant. Although recombination helped to obtain better local solutions, the best discovered solutions always had a fitness value larger than 10. These results seem to indicate that recombination is not very useful for multimodal optimization. But this question turns out to be more complicated: When using *discrete recombination*, i.e., $v_i = x_i + (y_i - x_i) \cdot \xi_i$ for all $i = 1, 2, \ldots, n$ with $P\{\xi = 0\} = P\{\xi = 1\} = 1/2$, the results for f_9 are much better (see table 4). The success of this recombination scheme for problem f_9 has a simple explanation (Rudolph 1990): During the first iterations most individuals approximate a local optimal point. As all local optimum points are arranged regularly on a lattice, discrete recombination always assembles a local optimum. In other words, for this problem the search space is restricted to the space of local minimum points. In general, this problem is NP–hard but in this case it is much easier because as the distance to the global optimum point decreases the objective function values of the local optima are lower.

Recently, Mühlenbein and Schlierkamp–Voosen (1993) also used problems f_8 and f_9 to test their *breeder genetic algorithm* (BGA). The success of the BGA for these problems relies on its mutation operator, because more than one third of all mutations occur on a line that is parallel to a coordinate axis. Indeed, the probability to generate a point in a subspace of dimension k is of order $1/(e \cdot k!)$, where $e = \exp(1)$. As soon as the objective function possesses nonlinearities of higher order with several variables, the BGA would face the same problems as any other coordinate strategy that searches preferably in low dimensional subspaces.

As the local minima of problems f_8 and f_9 are arranged on a regular lattice, mutations on a line parallel to a coordinate axis very often result in an improvement provided that a local minimum was approximated previously. Therefore, the BGA is a specialist for solving such problems and a fair comparison with more general methods hardly seems possible. But one should keep in mind that every recombination/crossover scheme specializes an algorithm in solving a certain class of problems. The usage of many different interacting specialized methods can therefore be a fruitful alternative.

6 Conclusions

Although simulated annealing and evolution strategies are inspired from rather different disciplines, their Markov chain formulation reveals their close relationship. The first essential difference concerns the acceptance criterion. But as soon as parallel versions of both approaches are implemented, this difference becomes less essential because competition plays the major role in accepting a new trial point (individual). In addition, competition was identified as the necessary condition for self-adaptability.
The second essential difference between SA and ES relies on the usage of multiple search trajectories in ES. It was demonstrated that the ideas of parallel ES can be used to design a massively parallel SA algorithm in a straightforward manner. Even the recombination mechanism can be incorporated.
The recombination mechanism used in this computational study (*hypercube recombination*) does not seem to be very useful for the test suite. It accelerates the search for more or less unimodal problems and offers slight advantages when tackling high dimensional multimodal problems. Other recombination mechanisms (*discrete recombination*) can operate on this test suite with greater success. This observation supports the author's belief that a useful general purpose recombination operator does not exist: Positive effects of specific recombination operators depend on the problem under consideration. Therefore, it might be useful to employ different recombination operators within the massively parallel NA.

Acknowledgment

The author is supported by the Research Initiative *Parallel Computing* of the Ministry of Science and Research of North Rhine–Westphalia, Germany, under project no. 107 004 91. The computational results have been produced on the Connection Machine 2 at the High Performance Computing Center of the GMD, St. Augustin, Germany. Special thanks to David B. Fogel, the anonymous referees and the editor-in-chief, whose comments and suggestions led to a significant improvement of the paper.

Appendix: Derivation of Equation (1)

Let $x \notin A \subseteq M$. The probability to transition from $x \notin A$ to an element in A is

$$P(x, A) = \int_A Q(x, d\omega)\, q(x, \omega)\, d\omega \quad , \tag{15}$$

where $Q(x, d\omega)$ is the probability to generate a point within the cylinder set $d\omega \subset A$ and $q(x, \omega)$ is the probability to accept the point ω. If $x \in A$, then the transition probability is

$$\begin{aligned} P(x, A) &= 1 - \int_{M \setminus A} Q(x, d\omega)\, q(x, \omega)\, d\omega \\ &= 1 - \int_M Q(x, d\omega)\, q(x, \omega)\, d\omega + \int_A Q(x, d\omega)\, q(x, \omega)\, d\omega \end{aligned}$$

$$
\begin{aligned}
&= \int_M Q(x,d\omega) - \int_M Q(x,d\omega)\,q(x,\omega)\,d\omega + \int_A Q(x,d\omega)\,q(x,\omega)\,d\omega \\
&= \int_M Q(x,d\omega)\,(1-q(x,\omega))\,d\omega + \int_A Q(x,d\omega)\,q(x,\omega)\,d\omega \quad . \qquad (16)
\end{aligned}
$$

Combining (15) and (16) into one formula gives

$$P(x,A) = \int_A Q(x,d\omega)\,q(x,\omega)\,d\omega + 1_A(x) \cdot \int_M Q(x,d\omega)\,(1-q(x,\omega))\,d\omega \quad .$$

References

Aluffi-Pentini, F., V. Parisi, and F. Zirilli (1985). Global optimization and stochastic differential equations. *Journal of Optimization Theory and Applications 47*(1), 1–16.

Bäck, T., G. Rudolph, and H.-P. Schwefel (1993). Evolutionary programming and evolution strategies: Similarities and differences. In D. Fogel and W. Atmar (Eds.), *Proceedings of the 2nd Annual Conference on Evolutionary Programming.* La Jolla (CA): Evolutionary Programming Society, pp. 11–22.

Bäck, T. and H.-P. Schwefel (1993). An overview of evolutionary algorithms for parameter optimization. *Evolutionary Computation 1*(1), 1–23.

Bélisle, C. (1992). Convergence theorems for a class of simulated annealing algorithms on $\mathbb{R}^d$. *J. Appl. Prob. 29*, 885–895.

Bertocchi, M. and C. Di Odoardo (1992). A stochastic algorithm for global optimization based on threshold accepting technique. In Phua et al. (Eds.), *Optimization Techniques and Applications*, Volume 1. World Scientific.

Bertocchi, M. and P. Sergi (1992). Parallel global optimization over continuous domain by simulated annealing. In P. Messina and A. Murli (Eds.), *Proceedings of Parallel Computing: Problems, Methods and Applications.* Amsterdam: Elsevier, pp. 87–97.

Bohachevsky, I., M. Johnson, and M. Stein (1986). Generalized simulated annealing for function optimization. *Technometrics 28*(3), 209–217.

Chiang, T.-S. and Y. Chow (1988). On the convergence rate of annealing processes. *SIAM Journal on Control and Optimization 26*(6), 1455–1470.

Corana, A., M. Marchesi, C. Martini, and S. Ridella (1987). Minimizing multimodal functions of continuous variables with the “Simulated Annealing” algorithm. *ACM Transactions on Mathematical Software 13*(3), 262–280.

Devroye, L. (1978). Progressive global random search of continuous functions. *Mathematical Programming 15*, 330–342.

Dueck, G. and T. Scheuer (1988). Threshold accepting: A general purpose optimization algorithm appearing superior to simulated annealing. Technical Report TR 88.10.011, IBM Scientific Center, Heidelberg, Germany.

Fang, K.-T., S. Kotz, and K.-W. Ng (1990). *Symmetric Multivariate and Related Distributions.* London and New York: Chapman and Hall.

Fogel, D. (1992). *Evolving Artificial Intelligence.* Ph. D. thesis, University of California, San Diego.

Gelfand, S. and S. Mitter (1991a). Simulated annealing type algorithms for multivariate optimization. *Algorithmica 6*, 419–436.

Gelfand, S. and S. Mitter (1991b). Weak convergence of markov chain sampling methods and annealing algorithms to diffusions. *Journal of Optimization Theory and Applications 68*(3), 483–498.

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

Gorges-Schleuter, M. (1989). ASPARAGOS: an asynchronous parallel genetic optimization strategy. See Schaffer (1989), pp. 422–427.

Gorges-Schleuter, M. (1992). Comparison of local mating strategies in massively parallel genetic algorithms. In R. Männer and B. Manderick (Eds.), *Parallel Problem Solving from Nature, 2*, Amsterdam: North Holland, pp. 553–562.

Greening, D. (1990). Parallel simulated annealing techniques. *Physica D 42*, 293–306.

Haario, H. and E. Saksman (1991). Simulated annealing process in general state space. *Adv. Appl. Prob. 23*, 866–893.

Haines, L. (1987). The application of the annealing algorithm to the construction of exact optimal design for linear regression models. *Technometrics 29*(4), 439–448.

Hoffmeister, F. (1991). Scalable parallelism by evolutionary algorithms. In M. Grauer and D. B. Pressmar (Eds.), *Applied Parallel and Distributed Optimization*. Berlin: Springer, pp. 175–198.

Hoffmeister, F. and T. Bäck (1992). Genetic self–learning. In *Proceedings of the First European Conference on Artificial Life, December 11-13, 1991*, Paris, France, pp. 227–235. The MIT Press.

Johnson, N. and S. Kotz (1970). *Distributions in Statistics: Continuous Distributions - 2*. Boston: Houghton Mifflin.

Khachaturyan, A. (1986). Statistical mechanics approach in minimizing a multivariable function. *Journal of Mathematical Physics 27*(7), 1834–1838.

Kirkpatrick, S., C. Gelatt, and M. Vecchi (1983). Optimization by simulated annealing. *Science 220*, 671–680.

Manderick, B. and P. Spiessens (1989). Fine–grained parallel genetic algorithms. See Schaffer (1989), pp. 428–433.

Mühlenbein, H., M. Gorges-Schleuter, and O. Krämer (1988). Evolution algorithms in combinatorial optimization. *Parallel Computing 7*, 65–88.

Mühlenbein, H. and D. Schlierkamp-Voosen (1993). Predictive models for the breeder genetic algorithm I: Continuous parameter optimization. *Evolutionary Computation 1*(1), 25–49.

Palmer, M. and S. Smith (1991). Improved evolutionary optimization of difficult landscapes: Control of premature convergence through scheduled sharing. *Complex Systems 5*, 443–458.

Pintér, J. (1984). Convergence properties of stochastic optimization procedures. *Math. Operat. Stat. , Ser. Optimization 15*, 405–427.

Rappl, G. (1984). *Konvergenzraten von Random Search Verfahren zur globalen Optimierung.* Dissertation, HSBw München, Germany.

Rappl, G. (1989). On linear convergence of a class of random search algorithms. *Zeitschrift f. angew. Math. Mech. (ZAMM) 69*(1), 37–45.

Rechenberg, I. (1973). *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution.* Stuttgart: Frommann–Holzboog.

Resnick, S. (1987). *Extreme values, regular variation, and point processes.* New York: Springer.

Romeo, F. and A. Sangiovelli-Vincentelli (1991). A theoretical framework for simulated annealing. *Algorithmica 6*, 302–345.

Rudolph, G. (1990). Globale Optimierung mit parallelen Evolutionsstrategien. Diploma thesis, University of Dortmund, Department of Computer Science.

Rudolph, G. (1992). Parallel approaches to stochastic global optimization. In Joosen, W. and E. Milgrom (Eds.), *Parallel Computing: From Theory to Sound Practice, Proceedings of the European Workshop on Parallel Computing (EWPC 92).* Amsterdam: IOS Press, pp. 256–267.

Schaffer, J. (Ed.) (1989). *Genetic Algorithms, Proceedings of the 3rd International Conference on Genetic Algorithms.* San Mateo: Morgan Kaufman.

Schaffer, J., R. Caruana, L. Eshelman, and R. Das (1989). A study of control parameters affecting online performance of genetic algorithms for function optimization. See Schaffer (1989), pp. 51–60.

Schwefel, H.-P. (1977). *Numerische Optimierung von Computer-Modellen mittels der Evolutionsstrategie.* Interdisciplinary systems research; 26. Basel: Birkhäuser.

Schwefel, H.-P. (1981). *Numerical Optimization of Computer Models.* Chichester: Wiley.

Solis, F. and R.-B. Wets (1981). Minimization by random search techniques. *Math. Operations Research 6*, 19–30.

Spiessens, P. and B. Manderick (1991). A massively parallel genetic algorithm: Implementation and first analysis. In R. Belew and L. Booker (Eds.), *Proceedings of the fourth Conference on Genetic Algorithms*, San Mateo: Morgan Kaufmann, pp. 279–286.

Sprave, J. (1990). Parallelisierung Genetischer Algorithmen zur Suche und Optimierung. Diploma thesis, University of Dortmund, Department of Computer Science.

Szu, H. and R. Hartley (1987a). Fast simulated annealing. *Physics Letters A 122*(3/4), 157–162.

Szu, H. and R. Hartley (1987b). Nonconvex optimization by fast simulated annealing. *Proceedings of the IEEE 75*(11), 1538–1540.

Titterington, D., A. Smith, and U. Makov (1985). *Statistical analysis of finite mixture distributions.* Chichester: Wiley.

Törn, A. and A. Žilinskas (1989). *Global Optimization.* Berlin and Heidelberg: Springer.

Vanderbilt, D. and S. Louie (1984). A Monte Carlo Simulated Annealing approach to optimization over continuous variables. *Journal of Computational Physics 56*, 259–271.

Wille, L. (1987). Minimum energy configurations of atomic clusters: New results obtained by simulated annealing. *Chemical Physics Letters 133*, 405–410.

Wille, L. and J. Vennik (1985). Electrostatic energy minimization by simulated annealing. *Journal of Physics A 18*, L1113–1117. (Corrigendum, 19:1983, 1986).

Zabinsky, Z. and R. Smith (1992). Pure adaptive search in global optimization. *Mathematical Programming 53*, 323–338.