communications in mathematical
match
and in computer chemistry

no. 34, October 1996

ISSN 0340-6253

MATCDY (34) 7-49 (1996)

SIMULATED ANNEALING

VLADIMÍR KVASNIČKA AND JIŘÍ POSPÍCHAL

Department of Mathematics, Slovak Technical University,

812 37 Bratislava, Slovak Republic

(email: {kvasnic,pospich}@cvt.stuba.sk)

Abstract

Simulated annealing belongs together with genetic algorithms and evolution strategies to major new optimization methods. It is used for optimization of highly multimodal functions, both for discrete as well as for continuous functions. Though origin of the method in physics has been nearly forgotten, it is useful to take into account also other terms connected with simulation of processes in statistical physics. These terms can be helpful for deeper understanding of the method. These terms can be also helpful in evaluation of newly proposed modifications of the method, like force and parallel simulated annealing in comparison with simple simulated annealing.

The paper introduces statistical physics terms into evaluation of minimization process of simulated annealing. New modifications of simulated annealing like forced simulated annealing and parallel simulated annealing are described in Pascal-like algorithms, their performance is illustrated by examples accompanied by figures describing processes in simulated annealing in graph plots of time vs. statistical physics terms.

1. INTRODUCTION

Minimization of multimodal functions and combinatorial problems belongs to basic problems in science, engineering or operations research. Recently, the majority of methods used for solution of this problem are based on stochastic approach, together with some kind of heuristic. Such methods are genetic algorithms [1-3], evolution strategies [4], and simulated annealing [5,6].

The connection of simulated annealing with its original purpose - to describe physical processes - has been nearly forgotten, but we believe, that many terms used in statistical physics can be still useful for deeper understanding of the method. The

simulated annealing algorithm [7,8,12,13] is based on the analogy between the simulated annealing of solids and the problem of solving large scale optimization problem. In physics annealing denotes a process in which a solid placed in a heat bath is heated up by increasing the temperature of the bath to a maximum value at which all particles (molecules or atoms) are randomly arranged so that solid body is melted. This process is followed by cooling by slowly lowering the temperature of the bath. All particles arrange themselves in stable positions, which results in the *low energy* state of a corresponding solid, assuming that the maximum temperature is sufficiently high and the cooling is carried out sufficiently slowly. Starting off at the maximum value of temperature, the cooling phase can be described as follows: At each temperature T, the solid is allowed to reach *thermal equilibrium*, described by the probability of being in a state i with energy E_i determined by the Boltzmann distribution

$$w_T(E_i) = \frac{1}{Q(T)} exp\left(-\frac{E_i}{kT}\right) \tag{1a}$$

where k is the Boltzmann constant and $Q(T)$ is a normalization factor called the *partition function*,

$$Q(T) = \sum_i exp\left(-\frac{E_i}{kT}\right) \tag{1b}$$

where the summation runs over all states i of the solid. As the temperature T decreases, the Boltzmann distribution favors states with lower energy, and finally, when temperature approaches zero, only the state with minimal energy has a nonzero (unit) probability of occurrence. Unfortunately, it is well known that if the cooling is too rapid (the solid is not allowed to reach thermal equilibrium for each temperature), defects can be "frozen" into the solid and metastable structures can be reached rather than the lattice structure with lowest energy.

To simulate the *evolution* towards the thermal equilibrium of a physical system (e.g. a many-particle crystalline solid) for a fixed value of the temperature T, Metropolis *et al.* [9] suggested the *Monte Carlo method*, which generates sequences of states of the system in the following way: Given the current state of system (determined by positions of particles) a small random perturbation is generated so that particles are "gently" displaced (the perturbation should be symmetric, that is, if A and B are two states, then the probability that a random perturbation of A leads to B must

equal the probability that a random perturbation of B leads to A). If the difference $\Delta E = E_{perturbed} - E_{current}$ between the perturbed state and the current state is negative ($E_{perturbed} < E_{current}$), then the process continues with the new perturbed state. In the opposite case, if $\Delta E \geq 0$, then the probability of acceptance of the perturbed state, *Pr(perturbed←current)*, is given by $exp(-\Delta E / kT)$ (see Fig. 1)

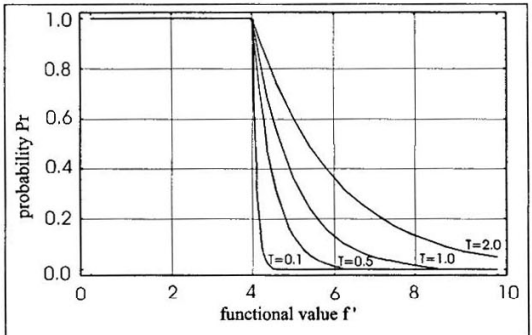

Figure 1. Plots of Metropolis criterion *Pr=min(1,exp(-(f'-f)/T))* for different temperatures T, where f is a fixed functional value (f=4) and f' is an independent variable taken from the closed interval [0,10]. For decreasing values of T and for $f' > f$ the probability $Pr \to 0$ as $T \to 0$.

$$Pr(perturbed \leftarrow current) = min(1, exp(-\Delta E / kT)) \qquad (2)$$

This acceptance rule of new states is called the *Metropolis criterion*. Following it, the system eventually evolves into thermal equilibrium; after a large number of perturbations, using the acceptance criterion (2), the probability distribution of states approaches the Boltzmann distribution (1). This form of the Monte Carlo method is known in statistical mechanics as the *Metropolis algorithm* [9]. In order to formalize the Metropolis algorithm we introduce the following notation (useful also for our forthcoming discussion on applications of the method of simulated annealing to optimizations of large scale problems): A state of system is determined by a *state variable* x (in general, a vector composed of many real entries) and an analogue of the energy $f(x)$ (treated as a function of x). A process of perturbation of the state x onto another state x' is represented by a stochastic function $x' = O_{pertur}(x)$. The stochastic

character of this function consists in its random changes of entries of x onto entries of x', see Algorithm 1.

```
procedure Metropolis_algorithm(x_ini, x_out, k_max, T);
begin k:=0; x:=x_ini;
      while k<k_max do
      begin k:=k+1;
            x':=O_pertur(x);
            Pr:=min(1,exp(-(f(x')-f(x))/T));
            if random<Pr then x:=x';
      end;
      x_out:=x;
end;
```

Algorithm 1. Implementation of the Metropolis algorithm. Input parameters are x_{ini}, k_{max}, T, and output parameter is x_{out}. The procedure is repeated k_{max} times, symbol O_{pertur} modifies a current state x onto another state x', its acceptance is solved by the Metropolis criterion performed for the temperature T. The procedure is initialised by the input state x_{ini}, after finishing the algorithm the current state is denoted by x_{out}.

In this algorithm random is a uniform random number from the semiopen interval [0,1). The algorithm contains a while-cycle repeated k_{max} times; this number should be sufficiently large to achieve the thermal equilibrium. What is very important, the Metropolis algorithm produces [5,6] a probability distribution approaching the Boltzmann distribution (1)

$$w_T(x) = \frac{1}{Q(T)} exp\left(-\frac{f(x)}{T}\right) \tag{3a}$$

$$Q(T) = \sum_x exp\left(-\frac{f(x)}{T}\right) \tag{3b}$$

where the summation runs over all states x. For simplicity, the Boltzmann constant k is omitted (cf. eqs. (1a-b)), formally, it is shifted to the temperature T.

The Metropolis algorithm can be used for the computer simulation of the method of simulated annealing. Simulated annealing can be now viewed as a sequence of Metropolis algorithms performed for a sequence of properly decreasing values of the temperature and moreover, an output state x_{out} from the current Metropolis algorithm serves as an input state x_{ini} for the next Metropolis algorithm. Initially, the temperature is set to a high value T_{max} and the Metropolis algorithm is applied until equilibrium is achieved (by k_{max} times, where k_{max} is the parameter of the Metropolis algorithm). The temperature is then lowered in steps (e.g. by $T:=\alpha T$, where

$0 << \alpha < 1$), with the system being allowed to approach equilibrium for each step by generating a sequence of states in the previously described way. The algorithm is terminated at some small value of the temperature T_{min}, for which virtually no states with higher functional value than the current one are accepted anymore. The final "frozen" state x_{out} is then taken as the solution of the problem at hand, see Algorithm 2. One of the major problems in simulated annealing is the specification of T_{max}, T_{min}, and the rate of decreasing of T. For the starting temperature and cooling scheme elaborate methods have been developed [5,6]. However, since the cooling scheme is not the main topic of this paper, for clarity we have chosen the most simple cooling scheme. Starting temperature was roughly set to a value, where about half of perturbations are accepted by the Metropolis algorithm.

```
procedure Simulated_annealing(x_opt, T_min, T_max, k_max, α);
begin x_ini:=randomly generated state vector;
      T:=T_max;
      while T>T_min do
      begin Metropolis_algorithm(x_ini, x_out, k_max, T);
            x_ini:=x_out;
            T:=α*T;
      end;
      x_opt:=x_out;
end;
```

Algorithm 2. Implementation of simulated annealing, input parameters are T_{min}, T_{max}, k_{max}, α, and output parameter is x_{out}. The algorithm is initialised by random construction of the initial state x_{ini} and by setting the temperature T to its initial value T_{max}. The cycle is repeated while the temperature $T > T_{min}$, the temperature T is decreased by parameter α. After finishing the cycle the current state x_{out} determines the solution of the algorithm denoted by x_{opt}.

The method of simulated annealing was formulated in a general way so that "physical ballast" was removed, though some physical concepts still play a role of fruitful heuristics useful for better intuitive understanding of the method. The main purpose of the simulated annealing is looking for global solution of optimization problems of the type

$$x_{opt} = arg \min_{x \in D} f(x) \tag{4}$$

where $f(x)$ is a real function determined over the domain D (usually discrete and finite), and x_{opt} is a value of variable corresponding to the global minimum of $f(x)$ over D. The variable x is considered as a state of hypothetical physical system and the function $f(x)$

expresses its "energy". Then, after the above considerations, the optimization problem (4) may be successfully approached by the method of simulated annealing. The parameter "temperature" plays now role of a control parameter of the method.

2. RELATIONS WITH STATISTICAL PHYSICS

As pointed out in the previous Section 1 there exists a clear analogy between the annealing of solids (or, in general, physical systems composed of many particles that are mutually interacting) and the solving of optimization problem (4). The physical annealing process can be successfully modeled by using computer simulation methods based on the Metropolis algorithm. This method in turn is based on the theory of statistical physics which can be viewed as the central discipline of condensed matter physics. We show that there exists very close relation between optimization problem (4) and statistical physics. This interpretation of the simulated annealing when applied to the finding of global solution of (4) is interesting not only because of a phenomenological interest in the analogy but as a possible framework to model the convergence and corresponding control of the simulated annealing.

The fundamental assumption of statistical physics [10] states that the physics of many particle systems is compatible with a statistical ensemble and that time averages of mechanical quantities of the system under microscopic equilibrium are equal to the corresponding ensemble averages. A number of useful macroscopic quantities can be then derived from the equilibrium distribution of states of the system. Applying the principle of equal probability [10], we can show that at the thermal equilibrium the probability that the system is in some microscopic state with a specified energy is determined by the Boltzmann distribution (1a-b). The relation between statistical physics and the optimization problem (4) can now be made more explicit. Given a hypothetical physical system in thermal equilibrium whose internal states x (variables from the domain D) are distributed according to the expressions (3a-b) (formally equivalent to (1a-b)) a set of microscopic quantities can be defined for the optimization problem (4) in a similar way as for standard physical systems.

Let us consider a function $f(x)$ defined over a discrete and finite domain D

$$f: D \rightarrow A \subset R_+ \tag{5}$$

where A is a domain (a discrete and finite subset of R_+ composed of nonnegative real numbers) of functional values of f. The probability distribution of a state x (i.e. variable x from D), resulting as an application of the Metropolis algorithm performed for a fixed temperature T, is determined by (3a-b). In our forthcoming considerations it will be useful to know also the corresponding density of functional values $y=f(x)$. In general, this entity is determined by

$$w_T(y) = \frac{|D(y)|}{Q(T)} exp\left(-\frac{y}{T}\right) \tag{6a}$$

$$Q(T) = \sum_{y \in A} |D(y)| exp\left(-\frac{y}{T}\right) \tag{6b}$$

where $D(y) \subset D$ is composed of all $x \in D$ that give the same functional value $y=f(x)$

$$D(y) = \{x \in D;\ y = f(x)\} \tag{6c}$$

and |D(y)| denotes its cardinality. It is easy to see that $Q(T)$ determined by (6b) is identical to its counterpart determined by (3b). The following "macroscopic" quantities are defined:

(1) The *mean value* of the function $f(x)$

$$\langle f \rangle_T = \sum_{y \in A} y\, w_T(y) \tag{7}$$

(2) The *mean value* of the function $f^2(x)$

$$\langle f^2 \rangle_T = \sum_{y \in A} y^2\ w_T(y) \tag{8}$$

(3) The *variance* of the function $f(x)$

$$\sigma^2(T) = \sum_{y \in A} w_T(y) \left(\langle f \rangle_T - y\right)^2 = \langle f^2 \rangle_T - \langle f \rangle_T^2 \tag{9}$$

(3) The *entropy*

$$S(T) = -\sum_{y \in A} w_T(y)\, ln\left(\frac{w_T(y)}{|D(y)|}\right) \tag{10}$$

(4) The *specific heat*

$$C(T) = \frac{\partial}{\partial T}\langle f \rangle_T = \frac{\sigma^2(T)}{T^2} = T\frac{\partial S(T)}{\partial T} \tag{11}$$

Both right-hand sides of (11) can be simply proved by differentiating either (7) or (10) with respect to T.

The density of functional values determined by (6a-b) satisfies the following asymptotic properties

$$\lim_{T\to\infty} w_T(y) = w_\infty(y) = \frac{1}{|D|}|D(y)| \tag{12a}$$

$$\lim_{T\to 0} w_T(y) = w_0(y) = \delta(y, y_{opt}) = \begin{cases} 1 & (if\ y = y_{opt}) \\ 0 & (if\ y \neq y_{opt}) \end{cases} \tag{12b}$$

where $y_{opt}=f(x_{opt})$ is the optimal value of the function $f(x)$ assigned to x_{opt} determined by (4) as the global minimum. The second limit property (12b) is very important for the method of simulated annealing; it states that as the temperature T approaches zero (through equilibrium states) the *probability of appearing of states corresponding to the global minimum is unity.*

The partial derivative of $\langle f \rangle_T$ with respect to T is determined by (11), we get

$$\frac{\partial \langle f \rangle_T}{\partial T} = \frac{\sigma^2(T)}{T^2} > 0 \tag{13}$$

This means that the mean value $\langle f \rangle_T$ is decreasing as the temperature T is also decreasing (through equilibrium states) . Then, according to (12a-b) ,we get

$$\lim_{T\to\infty} \langle f \rangle_T = \langle f \rangle_\infty = \frac{1}{|D|}\sum_{y\in A}|D(y)|\, y \tag{14a}$$

$$\lim_{T\to 0} \langle f \rangle_T = \langle f \rangle_0 = y_{opt} \tag{14b}$$

The mean value $\langle f^2 \rangle_T$ satisfies similar asymptotic properties

$$\lim_{T\to\infty} \langle f^2 \rangle_T = \langle f^2 \rangle_\infty = \frac{1}{|D|}\sum_{y\in A}|D(y)|\, y^2 \tag{15a}$$

$$\lim_{T\to 0} \langle f^2 \rangle_T = \langle f^2 \rangle_0 = y_{opt}^2 \tag{15b}$$

For the variance determined by (9) we get

$$\lim_{T\to\infty} \sigma^2(T) = \sigma^2(\infty) = \langle f^2 \rangle_\infty - \langle f \rangle_\infty^2 \tag{16a}$$

$$\lim_{T\to 0} \sigma^2(T) = \sigma^2(0) = \langle f^2 \rangle_0 - \langle f \rangle_0^2 = y_{opt}^2 - (y_{opt})^2 = 0 \tag{16b}$$

The entropy determined by (10) satisfies the following two well-known asymptotic properties

$$\lim_{T\to\infty} S(T) = S(\infty) = \ln|D| \tag{17a}$$

$$\begin{aligned}\lim_{T\to 0} S(T) = S(0) &= \ln\left|D(y_{opt})\right| \\ &= \ln 1 = 0 \quad \left(if\ \left|D(y_{opt})\right| = 1\right)\end{aligned} \tag{17b}$$

Their proof can be simply done introducing (6) into (10) and then turning the temperature T either to $T\to\infty$ or $T\to 0$. The second property (17b) is known in physics as the *third law of thermodynamics*. In the case of simulated annealing, the entropy can be interpreted as a *quantitative measure of the degree of optimality*.

According to (11) the partial derivative of the entropy with respect to the temperature T is determined by

$$\frac{\partial S(T)}{\partial T} = \frac{\sigma^2(T)}{T^3} > 0 \tag{18}$$

Then, with the decreasing temperature T (through equilibrium states), the entropy should be also decreasing, in particular as $T\to 0$ then $S(T)\to 0$ (cf. eq. (17b)). In the neighborhood of $T=0$ the entropy $S(T)$ may be approximated by a power series of T

$$S(T) = A_o + A_1 T + A_2 T^2 + \ldots \tag{19}$$

where $A_0=\ln(|D(y_{opt})|)$, and $A_1>0$. Then the heat capacity $C(T)$ in the neighborhood of $T=0$ (cf. eq. (11)) is approximated by

$$C(T) = T\frac{\partial S(T)}{\partial T} = T\left(A_1 + 2A_2 T + \ldots\right) \tag{20}$$

According to this formula and the asymptotic property (16a) of variance, the heat capacity C(T) satisfies

$$\lim_{T\to\infty} C(T) = \lim_{T\to 0} C(T) = 0 \tag{21}$$

This means that $C(T)$ should have for $0<T<\infty$ at least one maximum and as the temperature T is turned either to zero or infinity $C(T)$ is vanishing.

3. MODEL CALCULATIONS

Let us consider a function $f: D = \{0,1\}^N \rightarrow A \subset R_+$, with domain composed of all N-dimensional binary vectors, defined as follows

$$f(x) = f(x_1, x_2, \ldots, x_N) = \begin{cases} |x| + 1 & (\text{if } |x| \leq p) \\ N - |x| & (\text{if } |x| > p) \end{cases} \tag{22}$$

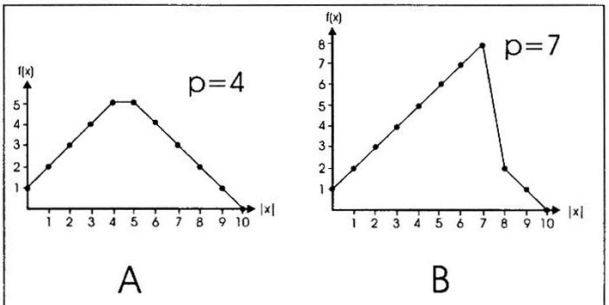

Figure 2. Plots of function (22) for N=10 and p=4, 7. For p=7 the function contains a lengthy foot beyond which it rapidly decreases to zero (global deceptive minimum).

where $|x| = \Sigma x_i$ is the L_1 norm of $x \in \{0,1\}^N$, $0 \leq |x| \leq N$. The parameter p is bounded by $0 \leq p \leq N$. For $p < N$ this function has two minima, the first minimum is $f(x)=1$, for $|x|=0$, and the second (global) minimum is $f(x)=0$, for $|x|=N$. For $p=N$, the function has only one (i.e. global) minimum $f(x)=1$, for $|x|=0$. The set A of functional values is composed of nonnegative integers bounded from above by f_{max}, $A=\{0,1,2,\ldots,f_{max}\}$, where $f_{max}=max(p+1,N-p-1)$. Schematic plots of $f(x)$ for $N=10$ and for $p=4$ and $p=7$ are displayed in Fig. 2. Parameter p determines the so-called deceptiveness [11] of the function $f(x)$. If p is slightly smaller than N, then the function contains lengthy foot starting at $|x|=0$ and ending at $|x|=p$, beyond which (for $p+1 \leq |x| \leq N$) the function rapidly decreases to zero (see Fig. 2, plot B). This feature of the function $f(x)$ is called the *deceptiveness* [11] and represents a hard task for evolution methods when applied to find global (deceptive) minima.

The operator $O_{perturb}$ from the Metropolis algorithm that transforms a binary vector $x \in \{0,1\}^N$ onto another vector $x' \in \{0,1\}^N$ is now realized by making use of the so-called *mutation operator* $O_{mut}^{(P_{mut})}$ (terminology used in the genetic algorithm [2])

$$\begin{aligned} x' = (x'_1, x'_2, \ldots, x'_N) &= O_{mut}^{(P_{mut})}(x) \\ &= O_{mut}^{(P_{mut})}(x_1, x_2, \ldots, x_N) \end{aligned} \tag{23}$$

where entries of x' are determined by

$$\forall i : x'_i = \begin{cases} 1 - x_i & (\text{if } random < P_{mut}) \\ x_i & (\text{otherwise}) \end{cases} \tag{24}$$

P_{mut} is a probability whether the entry x_i (for $\forall i$) will be changed to its complement (if $random < P_{mut}$) or not (otherwise), and *random* is a uniform random number from the semiopen interval [0,1). Its value should be small ($0 < P_{mut} << 1$) , in the opposite case the mutation operator produces new binary vectors that are very different from their original counterparts.

In order to calculate macroscopic entities defined in the previous Section 2, we have to know cardinalities of sets $D(y)$ for the function defined by (19). For instance, for $N=10$ and for $p=4$ and $p=7$, they are constructed after simple combinatorial considerations

$p=4 : |D(0)|=1, \quad |D(1)|=11, \quad |D(2)|=55, \quad |D(3)|=165, \; |D(4)|=330, \; |D(5)|=462$

$p=7 : |D(0)|=1, \quad |D(1)|=11, \quad |D(2)|=55, \quad |D(3)|=45, \quad |D(4)|=120, \; |D(5)|=210,$
$|D(6)|=252, \; |D(7)|=210, \; |D(8)|=120$

The plots of macroscopic quantities for the function $f(x)$ specified by $N=10$ and $p=1,4,7$, and 9 are displayed in Fig. 3, plots A to E. We see that all asymptotic properties discussed in the previous Section 2 are satisfied. In particular, the mean value $\langle f \rangle_T$ is vanishing as $T \to 0$. This means that for small temperatures the state with zero functional value is dominant with unit probability of appearance.

In the case of the simulated annealing an evaluation of macroscopic quantities can be done by approximating densities of functional values. In particular, they are approximated by appearances of functional values of states that have been accepted by the Metropolis algorithm for a fixed temperature T. Let $\chi_T(y)$ be a number of appearance of states x corresponding to y that have appeared in the course of

Metropolis algorithm for the temperature T. Then the densities $w_T(y)$ are approximated by

$$w_T(y) \approx \frac{\chi_T(y)}{\sum_{y' \in A} \chi_T(y')} \tag{25}$$

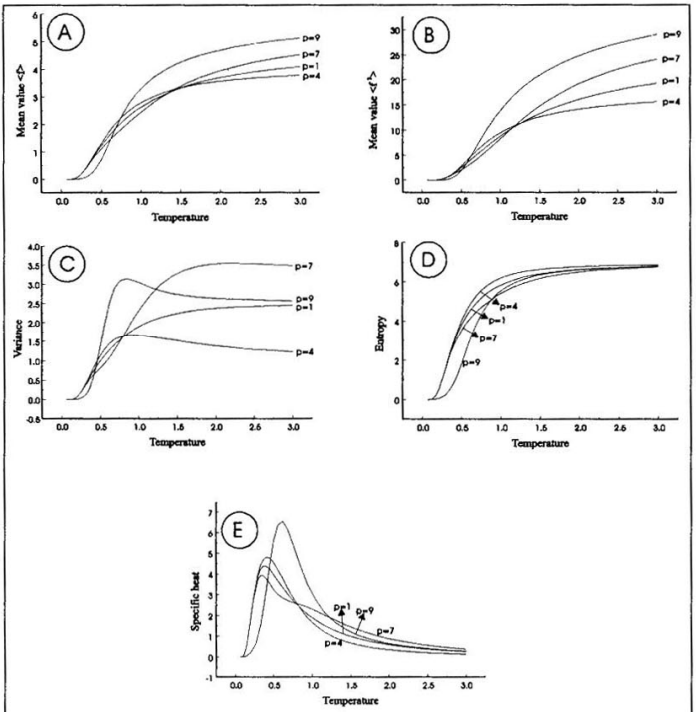

Figure 3. Plots of macroscopic quantities calculated by theoretical formulae (7-11) for parameters p=1,4,7, and 9 (plots A to E).

For k_{max} sufficiently large, this approximation should provide results that are closely related to the correct values of $w_T(y)$. The plots of all five macroscopic quantities determined by (7-11) with densities approximated by (25) are displayed in Fig. 4, plots A to E. Parameters of the simulated annealing used in calculations of these plots are

$$k_{max}=10^4,\ P_{mut}=0.1,\ T_{max}=3,\ T_{min}=3/50,\ and\ \alpha=0.95 \qquad (26)$$

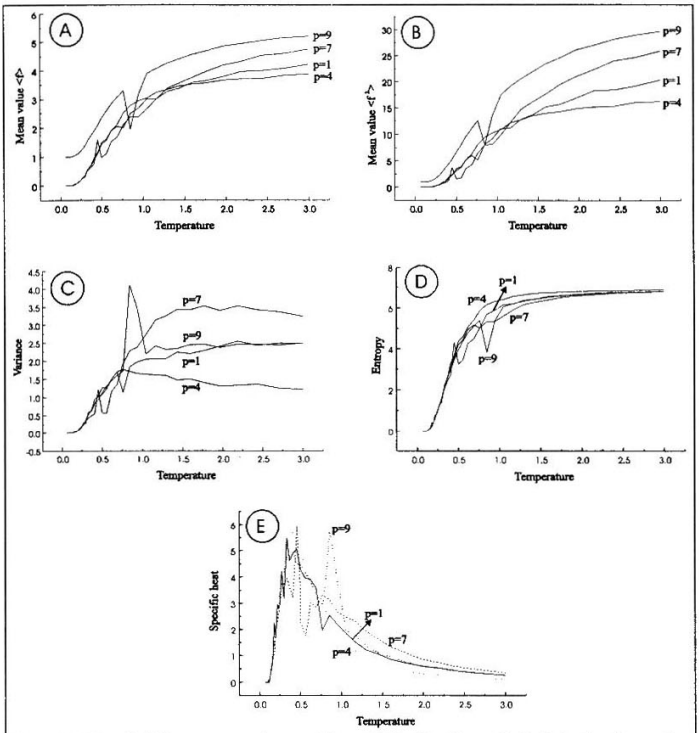

Figure 4. Plots of all five macroscopic quantities calculated by the method of simulated annealing specified by parameters (26). For the most deceptive case p=9, the final state x_{opt} corresponds to the unit functional value (local minimum), i.e. the method was unable to provide the correct global minimum.

We see that the simulated annealing for $p=1,4,$ and 7 has provided correct results for the mean value $\langle f\rangle_T$ (i.e. $\langle f\rangle_T \to 0$ as $T\to 0$), whereas for $p=9$ (the function $f(x)$ has the highest degree of deceptiveness) the method was unable to provide the correct result, in particular we got $\langle f\rangle_T \to 1$ as $T\to 0$. This means that for p=9 the simulated annealing was able to give only the suboptimal solution $x=(0,0,....,0)$, $|x|=0$, for which $f(x)=1$. A critical interval of temperatures for this case is 0.7-0.9, above this interval the method

provides solutions with densities of $y=0$ closely related to their theoretical values, but inside of this interval the incidence of states with $y=0$ was zero. Then, for lower temperature the Metropolis algorithm was unable to create states with $y=0$ (due to extended barrier of functional values, cf. Fig. 2, plot B). This fact is indicated, for example, by a large maximum on plots of the variance and the specific heat, which may be physically interpreted as "phase transition" to a configuration separated by great barrier from the configuration corresponding to the correct solution. Consequently, these macroscopic quantities might be of value as a valuable indicator of the range of temperature T where decrement in T should be smaller to overcome successfully the "phase transition".

Few remarks should be made about the simulated annealing when applied to the function $f(x)$ for the most deceptive case $p=9$. We have demonstrated that the simulated annealing with parameters (26) was unable to provide the correct solution. In general, this failure may be simply surmounted so that the probability P_{mut} is substantially increased, say $P_{mut}=0.5$. But then it is impossible to consider the simulated annealing as an evolutionary method; the scanning of search space is then more random than systematic. The simulated annealing for high values of P_{mut} is reduced, in fact, to a "blind" random searching. For fewer-dimensional cases this approach may be successful (e.g. searching space of $f(x)$ is composed of $2^{10}=1024$ binary vectors), but for higher-scale problems such an approach would be totally hopeless. In the forthcoming part of this Section we present two different approaches how to "gently" modify the standard version of simulated annealing so that it will be able to give correct solutions for highly deceptive function $f(x)$ (with $p=9$), and simultaneously the evolutionary character of simulated annealing will be saved.

3.1 Forced simulated annealing

In the original version of simulated annealing the resulting state from an application of the Metropolis algorithm (performed with temperature T) serves as an initial state of the next Metropolis algorithm (performed with decreased temperature αT). This basic assumption of the standard version of simulated version will be now relaxed so that each Metropolis algorithm is initialized by the best solution found so far in the previous history of the simulated annealing (i.e. for its runs with higher temperatures than the

current temperature); we shall call such simple modification the *forced simulated annealing*. A modification of procedures Metropolis_algorithm and Simulated_annealing outlined in Section 1 is displayed in Algorithm 3 and Algorithm 4.

```
procedure Forced_Metropolis_algorithm(x_opt, k_max, T);
begin k:=0; x:=x_opt;
      while k<k_max do
      begin k:=k+1;
            x':=O_pertur(x);
            Pr:=min(1,exp(-(f(x')-f(x))/T));
            if random<Pr then
            begin x:=x';
                  if f(x')<f(x_opt) then x_opt:=x';
            end;
      end;
end;
```

Algorithm 3. Implementation of the forced Metropolis algorithm with input parameters x_{opt}, k_{max}, T. The algorithm is initialized by x_{opt}, the best solution found in its course is recorded to x_{opt}. This means that x_{opt} serves simultaneously as the output parameter.

```
procedure Forced_Simulated_annealing(x_opt, T_min, T_max, k_max, α);
begin x_opt:=randomly generated state vector;
      T:=T_max;
      while T>T_min do
      begin Forced_Metropolis_algorithm(x_opt, k_max;T);
            T:= α*T;
              end;
end;
```

Algorithm 4. Implementation of the forced simulated annealing with input parameters T_{min}, T_{max}, k_{max}, α and output parameter x_{opt}. The state x_{opt}, initialized randomly, records the best solution achieved in the course of the whole history of the algorithm.

In general, the forced simulated annealing algorithm does not satisfy a very important condition that for sufficiently large k_{max} and for discrete temperature steps it will produce states distributed in accordance with the Boltzmann formula (3). Since in each temperature-decreasing step the Metropolis algorithm is now initialized by the best solution achieved so far in the previous history of the simulated annealing, it may happen (in particular, for lower temperatures) that the solution produced by the last performance of the Metropolis algorithm represents a local minimum different from the global minimum. However, k_{max} and therefore also the necessary computational time can be smaller than in simple simulated annealing.

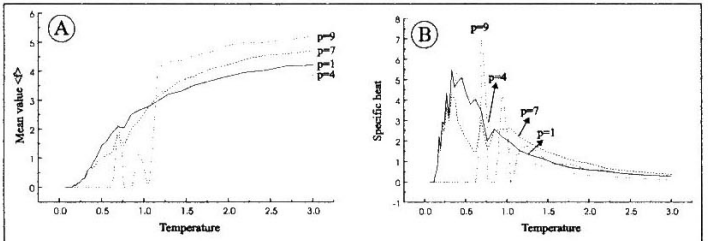

Figure 5. Plots of the mean value $\langle f \rangle$ and specific heat calculated by the method of forced simulated annealing. The resulting state x_{opt} corresponds for all values of the parameter p to the global minimum with zero functional value.

The numerical results obtained by the forced simulated annealing for the function (22) specified by $N=10$ and $p=1,4,7$, and 9 are displayed in Fig. 5 for mean value $\langle f \rangle$ and specific heat, plots A and B). We see that, in particular, for $p=9$ (the most deceptive form) changes of quantities for smaller values of temperature are very dramatic. For some medium range of temperature the Metropolis algorithm was unable to keep nonzero distribution of the state with zero functional value (i.e. a binary vector composed only of 1s). Since the best solution is for the presented example usually recorded in the very early stage of the process, the Metropolis algorithm produces for smaller temperatures ($T<0.6$) acceptable states that are identical with the best solution.

A weaker version of the forced simulated annealing is based on an assumption that the Metropolis algorithm is initialized by the best solution obtained not in the preceding history of the method but only in the previous step. Unfortunately, its application to our most deceptive case $p=9$ was not successful in overcoming the great barrier between the local minimum (with unit functional value) and the global minimum (with zero functional value).

3.2 Parallel simulated annealing

This version of the simulated annealing was successfully used for effective solution of complicated combinatorial graph-theoretical problems [12,13,14], where the standard version failed to give correct global solutions. Its basic idea consists in simultaneous

application of the simulated annealing to a pool P of states $x,x',x'',\ldots$ that are "synchronized" so that Metropolis algorithms running over them have the same temperature. In order to introduce an interaction between states the so-called *crossover* is applied. This operator was taken from the genetic algorithm [2], where it is usually applied to states x represented by binary vectors. However, the states x can be also other data structures like permutations. More complex data structures usually need a reparation process to be feasible after an application of crossover. Crossover applied to a randomly selected pair of states $x,x' \in P$ creates a new pair of states z, z'

$$(z,z') = O_{cross}(x,x') \tag{27}$$

Formally, let $x=(x_1,x_2,\ldots,x_N)$ and $x' = (x'_1,x'_2,\ldots,x'_N)$ be states from the right-hand side of (27), then the states z and z' are determined by

$$z = (x_1,\ldots,x_r,x'_{r+1},\ldots,x'_N) \tag{28a}$$

$$z' = (x'_1,\ldots,x'_r,x_{r+1},\ldots,x_N) \tag{28b}$$

where r, randomly selected from $1<r<N$, is called the *crossover point*.

An event of the interaction of two randomly selected states $x,x' \in P$ in the framework of the Metropolis algorithm is determined by a probability $0<P_{cross}<<1$. It determines a probability that instead of the standard mutation of a state $x \in P$ an interaction of two states $x,x' \in P$ will be performed by the crossover operator (27). Then, the problem whether the resulting new states z and z' are accepted or not is solved by the Metropolis criterion (2) applied separately for pairs (x,z) and (x',z'). For instance, if the Metropolis criterion accepts the new state z, then the pool P is updated so that the old state x is left out and the new state z is introduced into the pool P, formally $P:=(P\setminus\{x\})\cup\{z\}$. The resulting solution of the method is a state determined by

$$x_{opt} = arg \min_{x \in P} f(x) \tag{29}$$

where P is a pool resulting from the completed simulated annealing. The procedures Metropolis_algorithm and Simulated_annealing can be simply modified for the parallel version of simulated annealing, see Algorithm 5 and Algorithm 6.

```
procedure Parallel_Metropolis_algorithm(P,k_max,T);
begin k:=0;
      while k>k_max do
      begin k:=k+1;
            if random<P_cross then
            begin select randomly x∈P;
                  x':=O_pertur(x);
                  Pr:=min(1,exp(-(f(x')-f(x))/T));
                  if random<Pr then P:=(P\{x})∪{x'};
            end else
            begin select randomly x,x'∈P;
                  (z,z'):=O_cross(x,x');
                  Pr_1:=min(1,exp(-(f(z)-f(x))/T));
                  Pr_2:=min(1,exp(-(f(z')-f(x'))/T));
                  if random<Pr_1 then P:=(P\{x})∪{z};
                  if random<Pr_2 then P:=(P\{x'})∪{z'};
            end;
      end;
end;
```

Algorithm 5. Implementation of the parallel Metropolis algorithm, input parameters are P, k_{max}, T. In the while-cycle, repeated k_{max} times, a random choice (specified by the probability P_{cross}) of whether the algorithm will perform single perturbation of a randomly selected state x or a crossover between two randomly selected states x and x' is applied. In the latter case, the new states z and z' are accepted according to two Metropolis criteria performed independently for pairs (x,z) and (x',z').

```
procedure Parallel_Simulated_annealing(x_opt,T_max,T_min,k_max,α);
begin P:=randomly generated pool of states x,x',... ;
      T:=T_max;
      while T>T_min do
      begin Parallel_Metropolis_algorithm(P,k_max,T);
            T:=α*T;
      end;
      x_opt:=arg min {f(x),x∈P};
end;
```

Algorithm 6. Implementation of parallel simulated annealing, input parameters T_{max}, T_{min}, k_{max}, α, and output parameter is x_{opt}. The algorithm is initialized by random generation of states from the pool P. After its finishing the best state of the pool is recorded to x_{opt}.

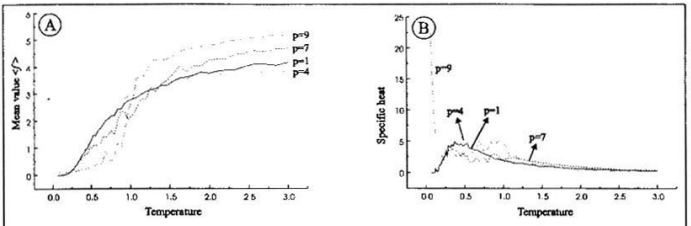

Figure 6. Plots of the mean value and specific heat calculated by the parallel simulated annealing. The pool P is composed of ten states. Since after the finishing of the algorithm the pool P still contains one state with unit functional value (local minimum), the plot **A** for the mean value $\langle f \rangle_T$ tends to small positive number as $T \to 0$. This is the main reason why the heat capacity (see plot **B**) rapidly increases as $T \to 0$.

Numerical results of the parallel simulated annealing for the mean value $\langle f \rangle$ and specific heat are displayed in Fig. 6. Plots for the parameter $p=1,4$, and 7 are similar to their theoretical counterparts (see Fig. 3). For $p=9$ (the most deceptive case of $f(x)$) the final pool (for $T \to 0$) contains almost entirely states with zero functional values but with a few (usually one) states corresponding to the local minimum with unit functional value (the state vector is composed of 0s). Therefore the mean value $\langle f \rangle_T$ as $T \to 0$ asymptotically tends to a small positive value (0.1-0.2). This means that the variance $\sigma^2(T)$ tends also to a small positive number and consequently the specific heat $C(T)$ is fast decreasing as $T \to 0$ (see plot B in Fig. 6). The final optimal state x_{opt} is constructed as a state of P with minimal functional value (see eq. (29)). In all our calculations we have obtained, for all values of the parameter p, the x_{opt} corresponding to the correct global minimum (with zero functional value). In summary, the parallel simulated annealing represents a very robust and stable method.

4. OPTIMIZATION OF CONTINUOS FUNCTIONS

The previous considerations about the method of simulated annealing (and its forced and parallel extensions) have been formulated for functions defined over the set $\{0,1\}^k$ composed of binary vectors of length k. It is easy to demonstrate that this theory is straightforwardly applicable also for optimization tasks of continuous functions defined over a hypercube $\langle a,b \rangle^N$,

$$f : D_f = \langle a,b \rangle^N \to A \subseteq R = (-\infty, \infty) \tag{30}$$

then

$$x_{opt} = arg \min_{x \in \langle a,b \rangle^N} f(x) \tag{31}$$

The vector of variables $x = (x_1, x_2, \ldots, x_N)$ is composed of N real entries bounded by $a \le x_i \le b$, for $i=1,2,\ldots,N$. Two different approaches will be used. The first one is based on the possibility to introduce simple binary representation of real numbers, where the length of used binary vectors determines the precision of representation. The second approach uses directly continuos variables without a necessity to apply an intermediate representation of variables. In this approach the operators of perturbation should be defined in another way than in the previous Section 3 (see eqs. (23-24) and (27-28)). It is quite difficult to say something about advantages of the first approach over the second approach. Loosely speaking, as seems from the current literature [3], that both are of the same importance and their effectiveness and robustness strongly depends on the type of optimized functions.

4.1 Binary representation [15]

The bit vectors $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_k) \in \{0,1\}^k$ may be interpreted as binary numbers assigned to nonnegative integers,

$$int(\alpha) = \sum_{i=0}^{k-1} 2^i \alpha_{k-i} \tag{32a}$$

Then, a real number $a \le x \le b$ assigned to α is determined by

$$\text{real}(\alpha) = a + \frac{b-a}{2^k - 1} int(\alpha) \tag{32b}$$

Its minimal (maximal) value a (b) is assigned to the bit vector α composed entirely of *0*s (*1*s). Consequently, the used binary representation of continuous real variables $a \le x_i \le b$ allows us to "translate" the continuos problem to a discrete one determined over the set of bit vectors.

In order to illustrate the algorithm let us consider simple case of bit strings of the length $k=3$ and assume that the produced real numbers are from the closed interval *[0,1]*, the results are summarized in Table 1.

Table 1

α	int(α)	real(α)
000	0	0
001	1	1/7
010	2	2/7
011	3	3/7
100	4	4/7
101	5	5/7
110	6	6/7
111	7	1

In the framework of this binary representation ($k=3$) the real variable $x \in [0,1]$ is partitioned by steps 1/7 so that the whole interval $[0,1]$ is composed of eight points (real numbers, see Table 1).

Accepting the binary representation of real variables, the continuous optimization problem (31) is transformed to a discrete optimization problem over a finite domain (it is in many cases of practical importance in spite enormous cardinality). This means that the method of simulated annealing (see Algorithms 1 to 6) can be used automatically so that the operator O_{pertur} is simply realized as the mutation operator O_{mut} (see eqs. (23-24)). Since the construction of the subsets $D(y)$ determined by (6c) is for continuous functions very difficult, we shall use another (equivalent) type of distribution of states x

$$w_T(x) = \frac{1}{Q(T)} exp\left(-f(x)/T\right) \tag{33a}$$

$$Q(T) = \sum_{x \in D} exp\left(-f(x)/T\right) \tag{33b}$$

where $x=real(\alpha)$, for $\forall \alpha \in \{0,1\}^k$. It means that the summation runs over all states assigned to binary vectors from $\{0,1\}^k$.

In order to illustrate the method of simulated annealing when applied to optimization of continuous functions with binary representation of variables we use the following simple function

$$f(x) = x\left(x^2 - 1\right) \tag{34}$$

for $x \in [-1,1]$. It has a local minimum at x=-1 and a global minimum at $x = 1/\sqrt{3}$, see Fig. 7. The used binary representation of the variable $x \in [-1,1]$ uses vectors of length k=10, i.e. the closed interval $[-1,1]$ is partitioned by 2^k=1024 points with increment determined by $\Delta x = (b-a)/(2^k - 1) = 2/1023$.

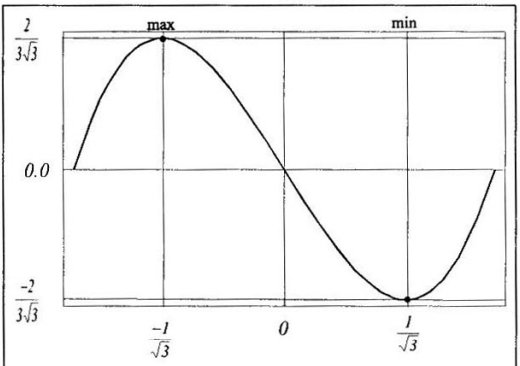

Figure 7. Plot of function $f(x)=x(x^2-1)$, for $x \in [-1,1]$

Numerical results obtained by the method of simulated annealing with parameters specified by (26) are displayed in Fig. 8, plots A to E. We see that for mean value $\langle f \rangle$, mean value $\langle f^2 \rangle$, variance $\sigma^2(t)$, and entropy $S(T)$ we get excellent agreement between the theoretically predicted and calculated plots. The calculated values of the specific heat $C(T)$ for decreasing sequence of temperatures are plagued by "fluctuations", which are caused by numerical errors produced by discrete states.

For more complicated continuous functions (e.g. highly multimodal or deceptive) the method of simulated annealing with binary representation of variables can be successfully used for the finding of their global minima. Moreover, approaches of forcing and/or parallelization are straightforwardly applicable. Usually, these approaches applied either separately or in their combination provide substantial increase of the robustness and effectiveness of the simulated annealing.

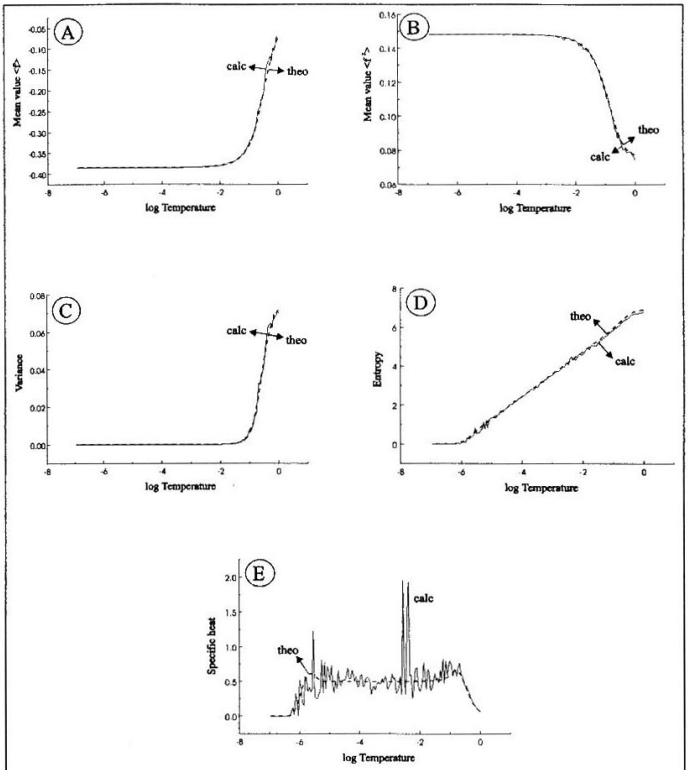

Figure 8. Plots of mean value $<f>$ (plot **A**), mean value $<f^2>$ (plot **B**), variance $\sigma^2(T)$ (plot **C**), entropy $S(T)$ (plot **D**), and specific heat $C(T)$ (plot **E**) for function (34) that are constructed theoretically or calculated by the method of simulated annealing. For the first four macroscopic quantities we obtain excellent agreement between those calculated from the theory and those ones obtained from distributions generated by the method of simulated annealing.

4.2 Real representation

The state vector $x \in \langle a,b \rangle^N$ of the function $f(x)$ determined by (30) can be immediately used in the method of simulated annealing when applied to finding the solution of optimization problem (31). This is simply achieved in that the perturbation operator O_{pertur} modifies a current state $x = (x_1, x_2, \ldots, x_N) \in \langle a,b \rangle^N$ to a new state

vectors $x' = (x'_1, x'_2, \ldots, x'_N) \in \langle a,b \rangle^N$ by addition random numbers with zero mean and "standard deviation" σ to its entries (cf. evolution strategy method [3,4])

$$x' = O_{pertur}(x) \tag{35a}$$

$$x'_i = x_i + r(0,\sigma) \tag{35b}$$

where $r(0,\sigma)$ is a random number specified below. The resulting new entries x'_i should again belong to the closed interval $\langle a,b \rangle$, therefore if they are outside of this interval we have to apply a repair process to ensure, that the resulting value belong to the interval $\langle a,b \rangle$.

It is often postulated that these random numbers satisfy Gauss distribution, then the quantity σ is correctly interpreted as the standard deviation. Since the random generator of numbers $r(0,\sigma)$ is in the simulated annealing used frequently and substantially influences the time consumption of the method, we turn our attention to another technique (much faster than generators of random numbers with Gauss distribution) based on the so-called logistic distribution (see Fig. 9)

$$f_\sigma(x) = \frac{1}{1 + e^{-x/T}} \tag{36a}$$

$$\omega_\sigma(x) = f'(x) = \frac{1}{\sigma} \frac{e^{-x/\sigma}}{\left(1 + e^{-x/\sigma}\right)^2} \tag{36b}$$

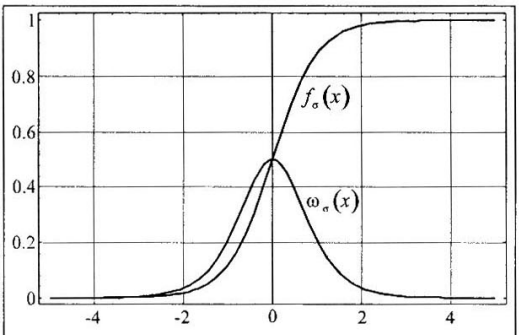

Figure 9. Plots of logistic distribution function $f_\sigma(x)$ and its density $\omega_\sigma(x)$ for $\sigma = 0.5$.

where the positive parameter σ (loosely speaking, a standard deviation of logistic distribution) determines the "width" of density $\omega_\sigma(x)$. The distribution $f_\sigma(x)$ maps the whole real axis on the open interval $(0,1)$, $f_\sigma : R \to (0,1)$. It may be used for simple construction of random generator of real numbers satisfying logistic distribution. Solving the equation

$$r = \frac{1}{1 + e^{-x/\sigma}} \tag{37}$$

for $r \in (0,1)$ we get

$$x = \sigma \ln \frac{r}{1 - r} \tag{38}$$

That is, if r is a random number with uniform distribution, then x determined by (38) is a random number with zero mean and fulfilling the logistic distribution. How to interpret and determine the parameter σ? Let us require that $100p\%$ random numbers (for $0<p<1$) are generated in the interval $(-\rho,\rho)$, then

$$\int_{-\rho}^{\rho} \omega_\sigma(x)\,dx = p \tag{39}$$

Introducing here (36) and after simple manipulations we get

$$\sigma = \rho \left(\ln \frac{1 + p}{1 - p} \right)^{-1} \tag{40}$$

For instance, if we put $p=0.99$ (i.e. 99% of generated random numbers belong to interval $(-\rho,\rho)$, then

$$\ln \frac{1 + p}{1 - p} = \ln 199 \approx 5.3 \tag{41}$$

This means that σ should be roughly five times shorter than ρ, provided that 99% random numbers belong to $(-\rho,\rho)$.

In the present version of simulated annealing, which performance is illustrated by finding of minima of function (34), we used the following parameters

$$k_{max}=10^4,\ T_{max}=1.0,\ T_{min}=10^{-7},\ \alpha=0.9,\ \sigma=0.02 \tag{42}$$

In order to calculate densities $\omega_T(x)$, the whole interval $[-1,1]$ is divided into 1000 subintervals, $[-1,1] = \bigcup_{(i=1)}^{(N)} I_i$, where $I_i=[x_{i-1},x_i]$, and $x_i=a+i(2/1000)$. If a state x accepted by the Metropolis criterion satisfies $x \in I_j$, then a frequency of appearance

$\chi_T(x_j)$ (initially empty) is increased by 1 (see eq. (25) and comment above). The density $\omega_T(x)$ is then approximated by normalized frequencies $\chi_T(x_j)$, i.e. they are multiplied by an inverse value of their sum. The obtained numerical results are displayed in Fig. 10.

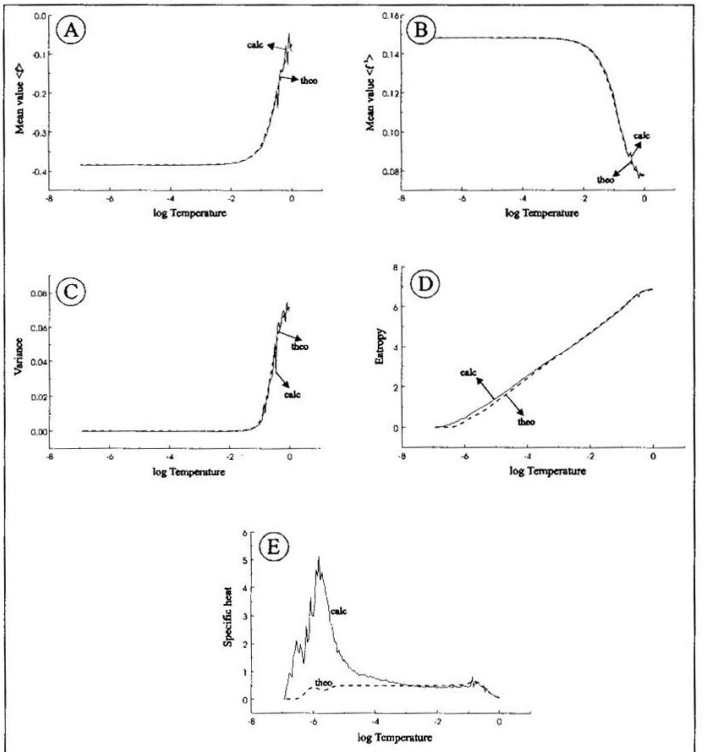

Figure 10. Plots of mean value $\langle f \rangle$ (plot **A**), mean value $\langle f^2 \rangle$ (plot **B**), variance $\sigma^2(T)$ (plot **C**), entropy $S(T)$ (plot **D**), and specific heat $C(T)$ (plot **E**) for function (34) constructed theoretically and calculated by the simulated annealing with real representation of variables. Similarly as for binary representation (see Fig. 8) an excellent agreement between theoretically predicted and calculated macroscopic quantities has been obtained for the first four quantities (plots **A** to **B**). For the specific heat (plot **E**) the calculated values for smaller temperature are considerable greater than those theoretically predicted.

5. COMBINATORIAL PROBLEMS

The first big success of the simulated annealing was recorded [7,8] in the field of *combinatorial problems* that are notorious due to their NP character [16] (i.e. their CPU time grows with respect to the problem dimension NonPolynomially, e.g. exponentially or factorially). We shall study two types of combinatorial problems that roughly cover their whole diversity.

Type 1. A function f is determined over the symmetric group S_N composed of all permutations of N objects,

$$f : S_N \to A \subset R \tag{43}$$

where permutations from S_N are denoted by

$$P = (p_1, p_2, \ldots, p_N) \tag{44}$$

Optimization problem (5) has now the following form

$$P_{opt} = arg \min_{P \in S_N} f(P) \tag{45}$$

The main difficulty of this problem lies in the fact that S_N contains $N!$ permutations. Therefore, for higher values of N its demand on CPU time roughly increases as $N!$.

Type 2. Let $R_{set}=\{1,2,\ldots,r\}$ be a set composed of integers $1,2,\ldots,r$, its direct product R_{set}^N contains N-tuples

$$\pi = (\pi_1, \pi_2, \ldots, \pi_N) \tag{46}$$

so that its entries belong to the closed interval of integers $[1,N]$. A function

$$f : R_{set}^N \to A \subset R \tag{47}$$

maps k-tuples (46) onto real numbers from A, an analogue of the optimization problem (45) is

$$\pi_{opt} = arg \min_{\pi \in R^N} f(\pi) \tag{48}$$

Since the cardinality of R_{set}^N is r^N, this task also belongs to the class of NP hard combinatorial problems, for larger values of r and N it may be even worse than the previous combinatorial problem, which is "only" a factorial NP-complete problem. For $r=2$ the present type of combinatorial problems is equivalent to optimization problems determined over binary vectors (see Section 3).

5.1 Traveling salesman problem (TSP)

A graph-theoretical formulation of TSP may be simply done as follows: Let G be a complete graph composed of N vertices (cities) and $N(N-1)/2$ edges (connections). Each edge $[i,j]$, connecting the vertices - cities i and j, is evaluated by a positive number $d(i,j)$ (called the distance between vertices i and j). A cyclical itinerary (a Hamiltonian cycle) for a traveling salesman, who must visit just once each of N cities and return to the starting position, is simply determined by a permutation on N objects, see (44). It corresponds to an itinerary of going through cities p_1, p_2, ..., p_N and returning to the initial city p_1. We assign to each itinerary P a distance determined by

$$f(P) = d(p_1, p_N) + \sum_{i=2}^{N} d(p_{i-1}, p_i) \tag{49}$$

The search for permutation P_{opt} which provides the minimal distance $f_{opt}=f(P_{opt})$, see eq. (45), is the main objective of TSP. We see that TSP belongs to the *type 1* of combinatorial problems, after classification presented at the beginning of section 5.

The NP-completeness of TSP causes that verification of results achieved by method, which does not search through all possibilities and gives only suboptimal results, is quite questionable. Therefore, in order to overcome these difficulties, a special positioning of cities is used. The vertices of graphs are situated at regular positions for which the optimum length can be deduced. One of such classes is a complete graph composed of $N=p^2$ vertices, which are situated at nodes of a square grid. For this special case, the optimum value f_{opt} is determined by (see Fig. 11)

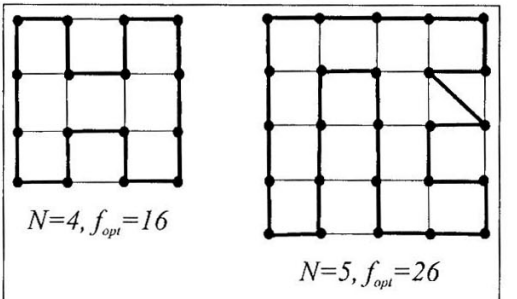

Figure 11. Two optimal paths on square grid of even and odd dimensions. For even N only horizontal and vertical lines appear, while for odd N there must be one cross line. All other paths which do not satisfy this condition have greater distance (49) than the presented ones.

$$f_{opt} = \begin{cases} p^2 & (\text{for even } p) \\ p^2 + 1 & (\text{for odd } p) \end{cases} \tag{50}$$

where a distance between two juxtaposed vertices on the same line or column is unit and other distances are calculated in the framework of Hamming (city block, L_1) metric (see Fig. 12).

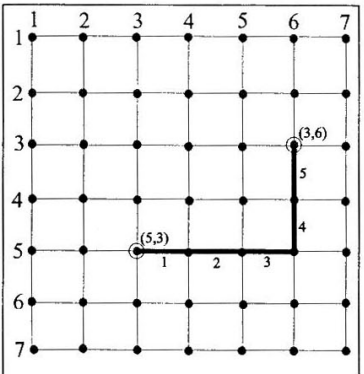

Figure 12. Square grid of vertices for $N=7^2=49$. The Hamming distance (L_1) between two vertices with row/column indices (5,3) and (3,6) is (see bold path) $d = |5-3|+|3-6|=5$. Two juxtaposed vertices on the same row or column of grid have the unit distance.

A state of TSP specified by the objective function (49) and by the minimization problem (45) is represented by a permutation $P \in S_N$. Its perturbation onto another permutation $P' \in S_N$

$$P' = O_{pertur}(P) \tag{51}$$

is realized by the following three different ways (see Fig. 13).

(1) The *transposing operator*, two randomly selected entries of permutation P indexed by $1 \le i < j \le N$ are mutually transposed

$$P = (p_1 \cdots p_i \cdots p_j \cdots p_N) \tag{52a}$$

$$P' = (p_1 \cdots p_j \cdots p_i \cdots p_N) \tag{52b}$$

(2) The *transporting operator*, a segment of permutation P (specified by indices i and j) is excised and then inserted into new position specified by k, indices are randomly selected and specified by $1 \leq i < j < k \leq N$,

$$P = \left(p_1, \ldots, p_{i-1}, p_i, \ldots, p_j, p_{j+1}, \ldots, p_k, p_{k+1}, \ldots, p_n\right) \tag{53a}$$

$$P' = \left(p_1, \ldots, p_{i-1}, p_{j+1}, \ldots, p_k, p_i, \ldots, p_j, p_{k+1}, \ldots, p_n\right) \tag{53b}$$

(3) The *reversing operator*, a segment of permutation P specified by randomly selected indices i and j, where $1 \leq i < j \leq N$, is reversed

$$P = \left(p_1, \ldots, p_{i-1}, p_i, p_{i+1}, \ldots, p_{j-1}, p_j, p_{j+1}, \ldots, p_N\right) \tag{54a}$$

$$P' = \left(p_1, \ldots, p_{i-1}, p_j, p_{j-1}, \ldots p_{i+1}, p_i, p_{j+1}, \ldots, p_N\right) \tag{54b}$$

Three types of the perturbation operator are introduced, and applied with the same probability. Theoretically, the first one (transposing operator) is fully sufficient, but the other two operators (i.e. transporting and reversing) may substantially influence [2] the convergence of simulated annealing to a path which is optimal or closely related to the optimal path.

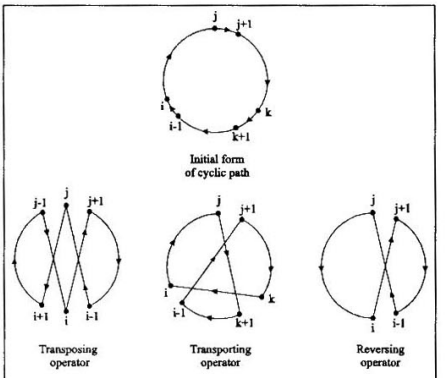

Figure 13. Schematic interpretation of transposing, transporting, and reversing operators defined by eqs. (52) to (54).

An application of the method of simulated annealing to TSP can be performed by Algorithms 1 and 2. The used square grid is 10×10 (i.e. TSP contains 100 vertices - cities), the shortest path has the length 10^2=100 (see eq. (50)). The used parameters are

$$k_{max}=10^4,\ T_{max}=7,\ T_{min}=0.01,\ \alpha=0.9 \tag{55}$$

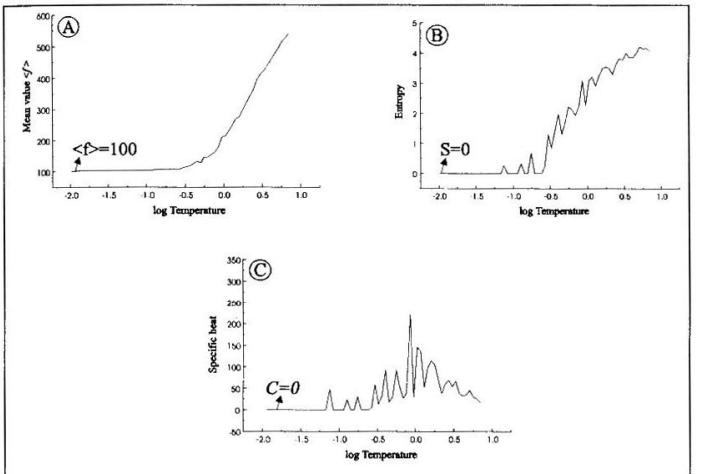

Figure 14. Plots of mean value $<f>$, entropy and specific heat calculated for TSP on grid 10×10 with parameters specified by (55). The entropy is approximated by the relation (25), where the cardinalities $|D(y)|=1$.

The densities $\omega_T(y)$ are approximated by (25), where values of y are discrete points - integers from the chosen interval [100,600]. Unfortunately, in the present case we are not able to calculate macroscopic quantities using the given theory. The main reason for this is, irrespective of the used densities (3) or (6), that not only we cannot "scan" all states of the system (their number is $N!$) but in addition we do not know the cardinalities $|D(y)|$ (see (6c)) (i.e. how many permutations correspond to the same length (functional value) $y=f(P)$). That is why entropy cannot be calculated correctly for the tested problem. Though we are able to approximately calculate the densities (see (25)), the definition of entropy contains cardinalities $|D(y)|$ that are, as was already mentioned, difficult to construct for combinatorial problems. We will try to

overcome this unpleasant situation in that we put $|D(y)|=1$, for each observed value of the length y. Then the entropy is approximated by

$$S(T) \approx \sum_{y \in A} \omega_T(y) ln\left[\omega_T(y)\right] \tag{56}$$

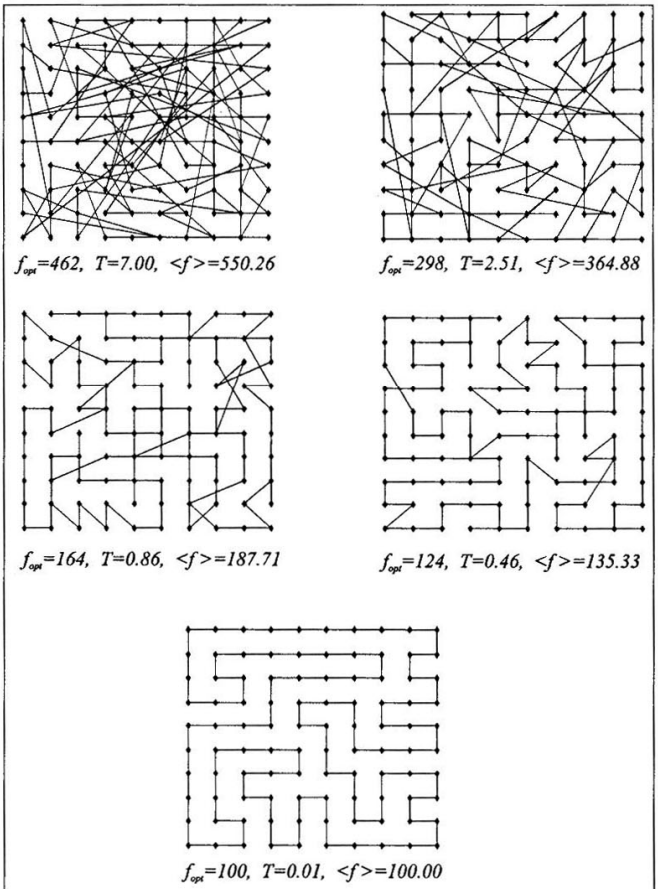

Figure 15. Plots of paths for grid 10×10 (i.e. 100 cities) for different temperatures, these paths were obtained by the simulated annealing method with parameters specified by (55).

where the densities $\omega_T(y)$ are constructed numerically by (25). The plots of mean value $<f>$, entropy and specific heat are displayed in Fig. 14. In Fig. 15 we present paths for four selected temperatures. These paths are constructed from the permutations P_{opt} that correspond to the currently best solution achieved so far by the simulated annealing. We see that an "order" is emerging from "chaos" as the temperature is decreasing. The final temperature T_{min} offers a path with length equal to its exact optimal value determined by (50).

5.2 Number partitioning problem (NPP) [17]

Let $Q=\{q_1,q_2,...,q_N\}$ be a set composed of N positive real numbers and let π be a mapping of Q onto a set $R_{set}=\{1,2,...,r\}$

$$\pi: Q \rightarrow R_{set} \tag{57}$$

The mapping π may be unambiguously expressed as an N-tuple $=(\pi_1,\pi_2,...,\pi_N) \in R_{set}^N$, its entries are interpreted so that π_i is an integer from R_{set} assigned to $q_i \in Q$. The set Q can be decomposed onto r disjoint subsets

$$Q = Q_1 \cup Q_2 \cup ... \cup Q_r \tag{58}$$

where

$$Q_i = \left\{ q \in Q;\ \pi(q) = i \right\} \tag{59}$$

The subset Q_i is composed of all numbers of Q that are mapped by π on the integer $i \in R$. The following type of objective function is defined

$$f(\pi) = \max_i \sum_{q \in Q_i} q - \min_i \sum_{q \in Q_i} q \tag{60}$$

It expresses a difference between maximal and minimal sums of numbers from subsets Q_1, Q_2, ..., Q_r. The main subject of NPP is to look for such a mapping π_{opt} that minimizes the objective function f

$$\pi_{opt} = arg \min_{\pi \in R_{set}^N} f(\pi) \tag{61}$$

After above classification, NPP belongs to combinatorial problems of the *type 2* (see beginning of section 5).

How to interpret the above described NPP? Let us imagine a store containing N articles with known prices q_1, q_2, ..., q_N. Our goal is to redistribute store articles onto r

heaps so that their total prices manifest minimal differences between them (see Fig. 16 and eq. (60)).

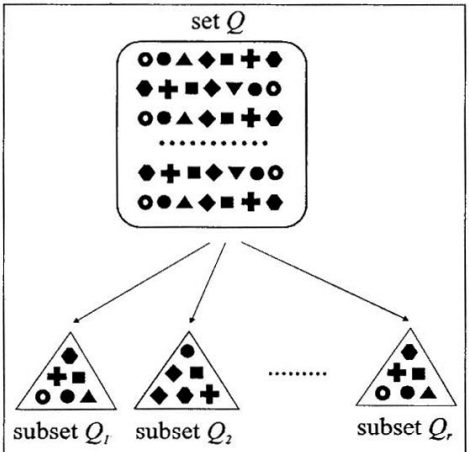

Figure 16. Schematic decomposition of the set Q, composed of N articles onto r disjoint subsets - heaps. The goal of NPP is to look for such a decomposition that differences between sums of "prices" of articles belonging to subsets are minimal.

The operator O_{pertur} from the Metropolis algorithm (see Algorithm 1), transforming a mapping π onto another mapping π', is now realized by two different ways (see Fig. 17).

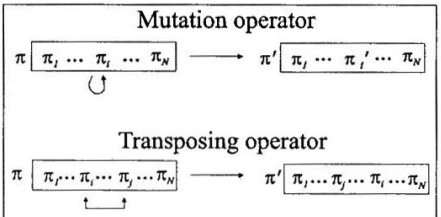

Figure 17. Diagrammatic illustration of mutation and transposing operators. The mutation operator changes a randomly selected entry of the mapping π to another feasible value. The transposing operator mutually exchanges randomly selected entries indexed by i and j.

(1) The *mutation operator*, let us have a mapping $\pi=(\pi_1,\pi_2,...,\pi_N)$ and let i be a randomly selected index, $1\le i\le N$. Then π_i' is also randomly chosen so that $\pi_i'\neq\pi_i$ and $1\le\pi_i'\le r$. The result of this mutation is a new mapping $\pi=(\pi_1,...,\pi_{i-1},\pi_i',\pi_{i+1},...,\pi_N)$.

(2) The *transposing operator*, this operator is determined in a similar way as in TSP. For a mapping $\pi=(\pi_1,\pi_2,...,\pi_N)$ we select randomly two indices i and j so that $\pi_i\neq\pi_j$ and then entries π_i and π_j are mutually transposed. The produced mapping is $\pi'=(\pi_1,...,\pi_{i-1},\pi_j,\pi_{i+1},...,\pi_{j-1},\pi_i,\pi_{j+1},...,\pi_N)$.

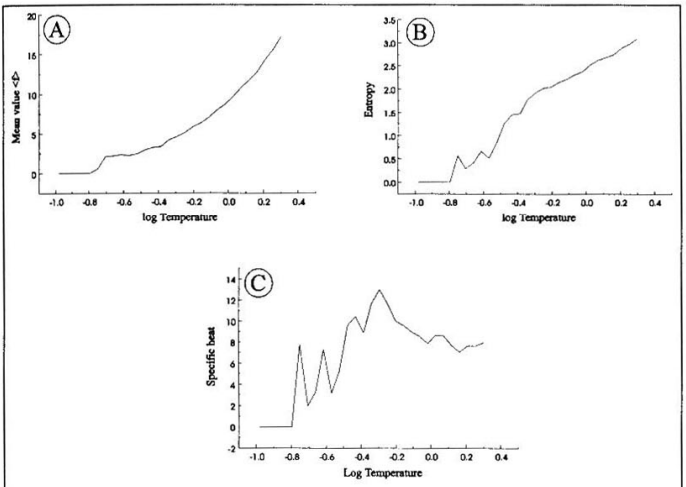

Figure 18. Plots of the mean value, entropy and specific heat calculated by the simulated annealing when applied to NPP. The mean value <f> (see plot **A**) converges to zero as temperature approaches zero. This means that simulated annealing provides a correct solution of NPP. The plot **B** of entropy was calculated approximately.

The method of simulated annealing (see Algorithms 1 and 2) is applied to solve the optimization problem (61) for $N=100$ and $r=10$. This means that the set R_{set}^N of all mappings π contains 10^{100} elements, and as will be demonstrated, the simulated annealing is successful in achieving the correct solution π_{opt} of this combinatorial problem of enormous dimension of searching space. In order to know the optimal solution π_{opt} in advance the set Q is constructed so that it contains 10-times integers 1

to 10. After simple considerations it is easy to see that a mapping π_{opt} separates Q onto 10 subsets Q_1, Q_2, ..., Q_{10}, composed of integers 1 to 10 just once. That is, the sum of all integers from subsets Q_i is the same, with distribution e.g. equal to the sequence 1...10 for each Q_i

$$\sum_{q \in Q_i} q = 1 + 2 + ... + 10 = 55 \tag{62}$$

for $i=1,2,...,10$, and therefore $f(\pi_{opt})=0$.

The parameters of simulated annealing used in our calculations are

$$T_{max}=7,\ T_{min},\ k_{max}=10^4,\ \alpha=0.9 \tag{63}$$

Plots of the mean value, entropy and specific heat are displayed in Fig. 18. The plot A shows that as T decreases, the mean value <f> converges to zero. Consequently, the simulated annealing provides correct solution π_{opt} with $f(\pi_{opt})=0$. The plot B corresponds to the entropy approximately calculated by (56), that is the cardinalities $|D(y)|$ were set equal to one.

6. Chemical Applications

There are many areas in chemistry, where optimization based on simulated annealing can be applied. Examples are geometry optimization of protein folding [18], QSAR [19], best wavelength design for spectroscopy concentration prediction, principal component analysis, chemical batch process scheduling and many other applications [20]. Any area with a difficult function to optimize is a potential field for successful application of simulated annealing.

However, most of applications concern real numbers in continuous space. The further presented application is not the most typical, as the search space is a discrete and finite space of structural formulas or molecular graphs. In principle any organic or inorganic class of compound with covalent bonds can be searched. Nevertheless, spatial isomerism is out of the scope of the presented example and aromatic bonds can be specified only by a sequence of alternating single and double bonds.

The optimized property should be easily computable from the structural formula, probably either with heuristics or with prediction based on regression, as typical optimization by simulated annealing requires tens of thousands evaluations of the optimized function. Quite interesting case is optimization of structural formula, so

that the result corresponds to a molecule with desired property/activity. This problem has been studied for decades from the opposite direction as a prediction of activity from a structure. Only when this task is solved, the opposite task can be approached, i.e. a compound of a certain biological or physicochemical activity is wanted, and it is up to a program to suggest a promising structure.

Any property or activity easily computable from structural formula would do, however the most simple example would be using topological indices (which are easily correlated with other interesting chemical properties like boiling or melting points or with some biological activity type). The task then would start from desired property by determining corresponding value of combination of some topological indices. The value would be obtained from an analytic expression derived from regression analysis constructed from a training set of a small database of structures and their activities. Simulated annealing would be then used to find out a structure with the most similar value of the combination of indices. Similar task has been dealt with during the last few years by exhaustive generation of structures, followed by structure - activity evaluation and screening [21,22].

Only recently new approaches based on optimization of a structural formula by genetic algorithms appear [23,24]. However, they are restricted to linear-type polymeric structures. The simulated annealing approach has no such restrictions [25,26]. For clear presentation of the principles of the method [25] the presented example is restricted only to a simple basic illustration.

For the sake of simplicity, only alkanes with 10 carbon atoms and one ring were considered. The chemical structures were after removing hydrogen atoms represented by graphs defined by adjacency matrices. The required properties of molecular graphs were represented by a convex combination of Wiener and Randic topological indices. Both these topological indices belong to the most frequently used topological indices in chemistry for studies of structure vs. property/activity correlation [27,28]. However, the restriction on carbohydrates and topological indices is not mandatory, in general any kind of molecules representable by structural formula could be considered as well as any property easily obtainable from structural formula.

For our application, the control parameters for the algorithm 2 were set at T_{max} = 5, T_{min} = 0.001, k_{max} = 10000, α = 0.9. The initial state was represented by a randomly generated lower triangle semicanonic [21] adjacency matrix A, i.e. for 10

atoms and 10 bonds it is a lower triangle of matrix 10x10 with entries a_{ij}=1 when atoms i and j are connected by a bond, and a_{ij} = 0 otherwise. The indexing of atoms is semicanonic, when for any two neighboring rows (the last entry of the longer row is deleted) their corresponding entries are either equal or for the first nonequal entries the upper row has 1, where the lower row has 0. This restriction reduces the search space, keeps us from dealing with many matrices describing the same molecule with different indexing of atoms. Each row of the lower-triangle part must also contain at least one "*1*" entry (this simple condition ensures that the resulting adjacency matrix corresponds to the connected graph). Moreover, restricting structures to molecular graphs, one has to check whether valences of vertices are not greater than the prescribed threshold value (e.g. four for saturated hydrocarbons).

A perturbation is performed by removing the lower part of the triangle matrix, starting from a randomly chosen row. The deleted part of the adjacency matrix is again randomly generated (see Fig. 19), satisfying semicanonicity and other restrictions. Of course, there exist many other ways how to define the perturbation, e.g. moving an arbitrary edge somewhere else, but there are always problems with preserving semicanonicity and other restrictions.

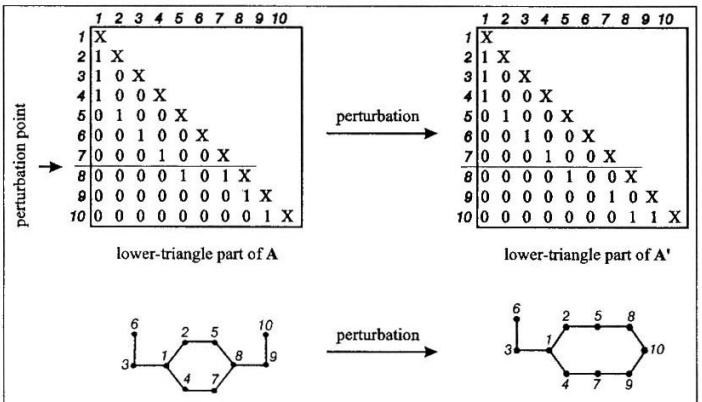

Figure 19. Illustrative example of the perturbation process of a semicanonic adjacency matrix A onto other semicanonic adjacency matrix A'. Starting from a randomly selected perturbation point, lower triangle rows are randomly generated so that the resulting matrix is semicanonic and corresponds to a graph with p vertices and q edges.

The energy of the system is replaced by a function defined as difference between the required (χ_{req}) and current property value (χ(G)) of graph G,

$$f(G) = \left| \chi(G) - \chi_{req} \right| \tag{64}$$

where the function χ(G) represents a hypothetical (physicochemical or biological) property determined as a convex combination of Randic and Wiener topological indices

$$\chi(G) = \omega \chi_R(G) + (1-\omega)\chi_W(G) \tag{65}$$

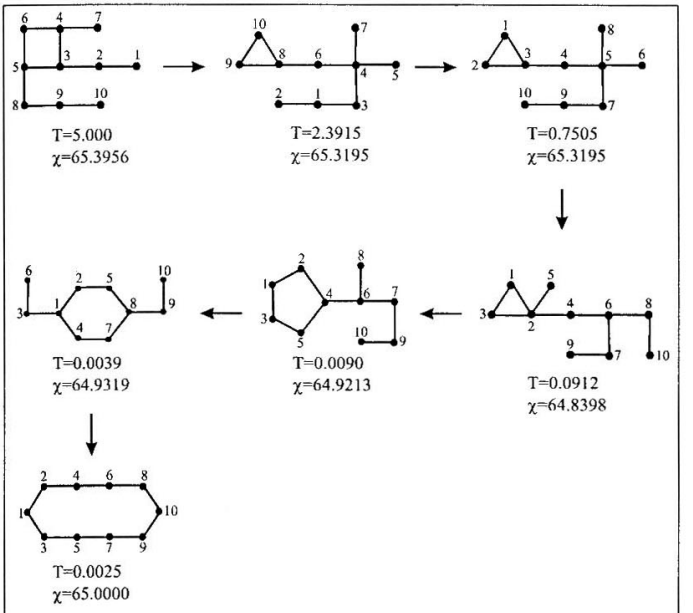

Figure 20. Graphs obtained at different values of formal parameter called temperature *T* in the course of simulated annealing reconstruction of a structural formula with a weighted sum of Wiener and Randic indices corresponding to a simple ring composed of 10 vertices, where all vertices are of the same valence equal to 2. As the temperature approaches zero, the graphs have larger and larger rings and their values χ(G) approach the desired value. Starting from the temperatures smaller than 0.0025 only the correct ring appears.

where $0 \le \omega \le 1$ (in our calculations we have used $\omega = 0.5$). The Randic topological index [27] is defined by $\chi_R(G) = \Sigma_{[v,v']\in E} \, 1/\sqrt{\mathrm{val}(v)\,\mathrm{val}(v')}$, where the summation runs

over all edges $[v,v']\in E$ of the graph G and symbols $val(v)$ and $val(v')$ denote valences of vertices v and v', respectively. The Wiener topological index [28] $\chi_W(G) = 1/2 \sum_{(v\neq v')}^{v,v'\in V} d(v,v')$, where the summation runs over all pairs of distinct vertices $v,v'\in V$, and $d(v,v')$ denotes the graph distance between vertices v and v'.

To test the approach we had to choose such properties, for which we already knew the correct answer. The testing was done with $\chi(G)$ corresponding to a simple ring of 10 vertices. The simulated annealing method gave this correct result. Moreover, we have recorded the best solutions obtained so far in the course of simulated annealing and it was observed that it was equal, after a few iterations of temperature decreases, to the correct results. In Figure 20 are displayed graphs, produced by the simulated annealing method for different temperatures. We see that as the temperature is decreasing the graphs are more and more similar to the required simple ring, for sufficiently low temperature they are identical to the simple ring.

Other test involved assigning a paticular ^{13}C NMR chemical shift to the corresponding alkane carbon atom from a set of all C_2-C_9 alkanes [25]. The program had always proposed suitable structures, the only inaccuracies were caused by regression.

7. DISCUSSION

The method of simulated annealing is formulated in the present paper as a generalization of "physical (simulated) annealing" applied to a hypothetical system determined by an objective function $f(x)$ defined over a domain D. Then many of concepts and notions of statistical physics can be immediately used as proper quantities to describe "global and/or macroscopic" properties of the simulated annealing. We did not discuss the very important parameters k_{max}, T_{max}, T_{min}, and α (see paragraph below eqs. (3a-b)). The main reason for this is, that although these parameters are frequently studied in the simulated annealing literature [5,6] and a lot of formulae and recommendations were suggested, the values of these parameters are strongly problem-dependent. That is, each author usually gives an independent recipe how to determine the basic parameters of his version and/or application of the simulated annealing method. The main purpose of this article was to give a general outline of

universal features of the simulated annealing as an optimization method of large scale problems of multimodal as well as of combinatorial character. In particular, its numerical efficiency is excellent for almost all applications if only suboptimal solutions closely related to the correct ones are required. The algorithmic implementation of the simulated annealing is very simple, it does not require any special programming tricks and techniques. On other hand, the simulated annealing method should not be understood as a universal algorithm in a sense that standard procedure may be applied to an arbitrary problem. Each new optimization problem requires some preliminary study in which one specifies (1) the numerical representation of state variables, (2) the definition of perturbation operator that transforms a state onto another state (which is in the neighborhood of the original state), and (3) the type of objective function to be minimized. If this first stage is accomplished successfully, then in the second stage we are ready to use the method of simulated annealing, e.g. in the form of Algorithms 1 and 2. Moreover, in order to test the used application of simulated annealing it is worthwhile to test its efficiency for simple model examples for which correct solutions may be simply deduced and numerical values of basic control parameters tuned.

In 1989 Goldberg et al. [29-31] tried to overcome some difficulties of genetic algorithm with the so-called *deceptive optimization problems*, i.e. problems that are multimodal with many local minima but only with one global (*deceptive*) minimum, separated by a barrier from other minima (see Fig. 2). They suggested the *messy genetic algorithm*, where the main departure from the standard genetic algorithm consists in the using special data structure called the *messy chromosome*. Recently, the concept of messy chromosomes was successfully used in the framework of simulated annealing by present authors [32]. The produced version of messy simulated annealing is very robust and effective, it solved correctly all model functions that were used by Goldberg as the test of effectiveness of the messy genetic algorithm.

In order to place simulated annealing within other stochastic optimization algorithms, it is necessary to compare it mainly with genetic algorithms. Although simulated annealing has theoretical advantage of the possibility to achieve absolute minimum, it can be achieved only by a very slow lowering of temperature, with enormous requirements on CPU time. Since in practical applications the temperature decreases faster than required the result may not be optimal. The quality of results in simulated annealing can be better controlled in comparison with genetic algorithms; in

particular, when using smaller k_{max} we can exchange optimality of result for speed of computation, a fate that can be hardly achieved in genetic algorithm. Genetic algorithms might give better results for more ragged landscape of function, but as it is shown in parallel simulated annealing, both methods can be merged to produce better results than each method separately. It cannot be stated in advance, which of the two methods works better for some problem, this question can be solved only by computations with tuned parameters. Genetic algorithms are generally better for problems of simulations of evolution in nature, while simulated annealing is better for large scale optimization, where it is enough to have as a result just one solution.

REFERENCES

1. J. H. Holland, *Adaptation in Natural and Artificial Systems.* The University of Michigan Press, Ann Arbor (1975).
2. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley, Reading, MA. (1989).
3. Z. Michalewicz, *Genetic Algorithm + Data Structure = Evolution Programs.* Springer Verlag, Berlin (1992).
4. H. P. Schwefel, *Numerical Optimization for Computer Models.* Wiley, Chichester (1981).
5. P. J. M. van Laarhoven and E. H. L. Aarts, *Simulated Annealing. Theory and Applications.* Reidel, Dordrecht (1987).
6. R. H. J. M. Otten and L. P. P. P. van Ginneken, *Annealing Algorithm.* Kluwer, Boston (1989).
7. S. Kirkpatrick, C. D. Gelatt Jr. and M. P. Vecchi, Optimization by Simulated Annealing. *Science* **220**, 671-680 (1983).
8. J. Cerny, Thermodynamical Approach to the Traveling Salesman Problem: An Efficient Simulation Algorithm. *J. Opt. Theory Appl.* **45**, 41-51 (1985).
9. N. Metropolis, A. W. Rosenbluth, N. N. Rosenbluth, A. H. Teller and E. Teller, Equation of State Calculations for Fast Computing Machines. *J. Chem. Phys.* **21**, 1087-1092 (1953).
10. M. Toda, R. Kubo, and N. Saito, *Statistical Physics.* Springer Verlag, Berlin (1983).
11. K. Deb, D. E. Goldberg, Sufficient Conditions for Deceptive and Easy Binary Functions. *(IlliGAL Report No. 91008),* Urbana: University of Illinois at Urbana Champaign, Illinois Genetic Algorithm Laboratory.
12. J. Pospichal and V. Kvasnicka, Fast Evaluation of Chemical Distance by Simulated-Annealing Algorithm. *J. Chem. Inf. Comp. Sci.* **33**, 879-884 (1993).
13. V. Kvasnicka, J. Pospichal and D. Hesek, Augmented Simulated Annealing Algorithm for the TSP. *Central European Journal for Operations Research and Economics* **2**, 307-317 (1993).
14. S. W. Mahfoud, D. E. Goldberg, A genetic algorithm for parallel simulated annealing. in *Parallel Problem Solving from Nature, 2*, ed. by R. Manner, B. Manderick, pp. 301-310, North-Holland, Amsterdam (1992).

15. K. A. De Jong, *An Analysis of the Behavior of a Class of Genetic Adaptive Systems.* PhD Thesis, University of Michigan (1975).
16. M. R. Garey and D. S. Johnson, *Computers and Intractability - A Guide to the Theory of NP-Completness.* Freeman and Co., San Francisco (1979).
17. D. S. Johnson, C. R. Aragon, L. A. McGeoch, and C. Schevon, Optimization by Simulated Annealing: an Experimental Evaluation, *List of Abstracts, Workshop on Statistical Physics in Engineering and Biology,* Yorktown Heights, April 1984, revised version, 1986.
18. M. E. Snow, Powerful Simulated Annealing Algorithm Locates Global Minimum of Protein-Folding Potentials From Multiple Starting Conformations. *J. Comput. Chem.* **13**, 579-584 (1992).
19. J. M. Sutter, S. L. Dixon, P. C. Jurs, Automated Descriptor Selection for Quantitative Structure-Activity Relationships Using Generalized Simulated Annealing. *J. Chem. Inf. Comput. Sci.* **35**, 77-84 (1995).
20. J. H. Kalivas, ed., *Adaption of Simulated Annealing to Chemical Optimization Problems.* Elsevier, Amsterdam (1995).
21. V. Kvasnicka, J. Pospichal, Canonical Indexing and Constructive Enumeration of Molecular Graphs. *J.Chem.Inf.Comp.Sci.* **30**, 99-105 (1990).
22. M. I. Skvortsova, I. I. Baskin, O.L. Slovokhotova, V. A. Palulin, N. S. Zefirov, Inverse Problem in QSAR/QSPR Studies for the Case of Topological Indices Characterizing Molecular Shape (Kier Indices). *J. Chem. Inf. Comput. Sci.* **33**, 630-634 (1993).
23. V. Venkatasubramanian, K. Chan, J. M. Caruthers, Evolutionary design of Molecules with Desired Properties Using the Genetic Algorithm. *J. Chem. Inf. Comp. Sci.* **35**, 188-195 (1995).
24. R. P. Sheridan, S. K. Kearsley, Using a Genetic Algorithm To Suggest Combinatorial Libraries. *J. Chem. Inf. Comput. Sci.* **35**, 310-320 (1995).
25. V. Kvasnicka, J. Pospichal, Simulated Annealing Construction of Molecular Graphs with Required Properties, *J. Chem. Inf. Comp. Sci.* **36**, 516-526 (1996).
26. J.-L. Fulton, Stochastic Generator of Chemical Structure. 2. Using Simulated Annealing to Search the Space of Constitutional Isomers. *J. Chem. Inf. Comput. Sci.* **36**, 731-740 (1996).
27. M. Randic, On Characterization of Molecular Branching. *J. Am. Chem. Soc.* **97**, 6609-6615 (1975).
28. H. Wiener, Structural Determination of Paraffin Boiling Points. *J. Am. Chem. Soc.* **69**, 17-20 (1947).
29. D. E. Goldberg, B. Korb, and K. Deb, Messy Genetic Algorithms: Motivation, Analysis and First Results. *Complex Systems* **3**, 493-530 (1989).
30. D. E. Goldberg, K. Deb, and B. Korb, Messy Genetic Algorithms: Studies in Mixed Size and Scale. *Complex Systems* **5**, 415-444 (1990).
31. K. Deb, *Binary and Floating-Point Function Optimization Using Messy Genetic Algorithm.* PhD Thesis, Department of General Engineering, University of Illinois at Urbana-Champaign (1991).
32. V. Kvasnicka and J. Pospichal, Messy Simulated Annealing, *Journal of Chemometrics* **9**, 309-322 (1995).