# Simulated Annealing Construction of Molecular Graphs with Required Properties[†]

Vladimír Kvasnička* and Jiří Pospíchal

Department of Mathematics, Slovak Technical University, 812 37 Bratislava, Slovakia

Received July 26, 1995[⊗]

The method of simulated annealing for the construction of molecular graphs with required properties was studied. The method depends on the already available functional relationship that transforms molecular structural features into a numerical value of a property. The simulated annealing was initialized by a randomly generated molecular graph. A molecular graph was perturbed onto another molecular graph so that starting from a randomly selected point the rest of the numerical code of the current graph was replaced by a randomly generated code. The acceptance of the generated code to the next process of simulated annealing was solved by the Metropolis criterion. After the prescribed number of steps the temperature was multiplicatively decreased. Two types of molecular graphs were studied. The first type of molecular graphs was acyclic graphs (trees) that are simply represented by a numerical code composed of the same number of entries as the number of vertices in molecular graphs. Perturbation operations consist of simple changes of numerical codes. The second type of molecular graphs was cyclic connected graphs ($q > p - 1$, where $p$ and $q$ are numbers of vertices and edges, respectively) that are represented by the lower-triangle part of adjacency matrices. The efficiency of the proposed method is illustrated by model calculations, wherein molecular graphs with required properties are constructed.

## INTRODUCTION

Simulated annealing methods[1,2] belong to powerful and effective numerical tools to solve optimization tasks on highly multimodal objective functions with many tens or hundreds of variables. The methods are successfully used for large scale combinatorial problems that are plagued by notorious NP-hardness (e.g., traveling salesman problem in operations research[3,4] and the chemical distance evaluation problem in mathematical chemistry[5]). The simulated annealing methodology was successfully embraced by computational chemistry in looking for stable geometry of large molecular systems[6] and in chemometrics and the QSAR field as an efficient optimization procedure in regression analysis studies.[7] This technique may be considered as part of a broad class of *evolution methods*[8] (e.g., genetic algorithm, evolution strategy, etc.). It has a well established simple physical background in statistical physics on the basis of which it was proved[1,2] that global minima are potentially achievable with a probability asymptotically tending to unity if some control parameters are sufficiently great or small. The physical background of the simulated annealing is of great importance for its simple interpretation and understanding of processes occurring in the course of its performance.

The purpose of this paper is to outline a simulated annealing construction of molecular graphs with a prescribed property. The possibility of using computers for such a task has attracted attention only in the last few years.[9−12] One reason for the delay is that we have to be able to predict an activity from a given structure in the first place, and it must be done very fast. Thousands of structures per minute should be computed. That rules out quantum chemistry computations and leaves us either with heuristics or with predictions based on regression or on neural networks. The regression analysis as well as neural networks require a training set, i.e., some small database of structures and their required activities. An analytic expression (or neural network) is then constructed, where molecular descriptors serve as variables that are mapped onto the required property. This functional relationship is used for fast prediction of the required property of compounds outside the training set. Molecular descriptors must be easily attainable from the structure; good examples are topological indices[9,10] or the presence of active substructures.[11,12]

Early models[9,10] included a structure generator that was able to generate all possible molecular structures. These were evaluated afterwards by the value of a topological index, and only the structures with proper value were chosen. An exhaustive search, i.e., one which is not able to include efficient screening within the generation process, is limited only to smaller molecules or to structures combined only from a restricted number of fragments. (Fragments themselves can be large.)

In order to avoid a combinatorial catastrophe, we have to resort to an approach based on a random search, which may not give all solutions, but will provide us with a correct solution with a high probability. The generation of structures totally at random is a rather hopeless task.[13] The most popular modern heuristics for combinatorial optimization problems are genetic algorithms[14] and simulated annealing.[1,2] Unfortunately, genetic algorithms include as their most important feature the exchange of information between possible solutions, performed by an operation called crossover. There is no problem in exchange of pieces of information, however, if the information makes meaningful package as a whole code, mutual exchange of parts of two codes may destroy most of the information of both changed codes. The intentional exchange of information then works more as a "giant" perturbation for both concerned codes, preventing thus convergence to optimum. This especially concerns structures containing cycles outside building blocks, where creating a meaningful exchange of parts of structures

[†] Dedicated to Ivar Ugi on the occasion of his 65th birthday.

[⊗] Abstract published in *Advance ACS Abstracts,* February 15, 1996.

requires deeper analysis of the code, and such analysis is time consuming. Known applications of this method are therefore restricted only to linear polymers or molecules created by a linear link of some building blocks (groups).[11,12]

A much more flexible approach, one based on simulated annealing, allows for the creation of rings. In fact it allows us to deal with general molecular structures. For simplicity, a required property is, in most of our considerations, expressed as a convex combination of Randić[15] and Wiener[16] topological indices. It means that the simulated annealing is looking for those molecular graphs from a set of graphs specified by a class of restrictions, that have numerical values of a convex combination of Randić and Wiener topological indices closely related (or even equal) to the required value. The molecular structures (e.g., alkanes), after abstracting hydrogen atoms, are represented by graphs. Two types of molecular graphs are studied. The first type encompasses acyclic molecular graphs represented by trees. These molecular graphs are represented by a simple linear code invented by Read,[17] often used in mathematical chemistry for constructive enumeration of acyclic molecular graphs.[18−20] The graphs of the second type are general graphs that may contain cycles. They are numerically coded by the lower-triangle part of adjacency matrices. In order to avoid enormous redundancy of isomorphic graphs we use these matrices in the so-called semicanonic form. This concept of adjacency matrices was successfully used in a series of our papers[9,21−23] devoted to constructive enumeration of molecular graphs. The method of simulated annealing requires the so-called perturbation of one state onto another. In the present work this important operation is carried out so that, up to a randomly generated perturbation point, the current code is simply copied and then randomly completed to a new (perturbed) code. Of course, the second operation of completion is realized so that the produced code satisfies some restrictions (e.g., it corresponds to a graph with the required number of vertices and edges, etc.). The proposed simulated annealing approach to construction of molecular graphs is illustrated by model calculations done for both types of molecular graphs. It is demonstrated that this approach gives results (molecular graphs) that correctly solve the construction task of molecular graphs with required properties. In addition, the restriction on hydrocarbons 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 calculated from numerical descriptors that are deduced from structural formulas.

## SIMULATED ANNEALING

The simulated annealing algorithm[1−4] is based on the analogy between the simulated annealing of solids and large scale optimization problems. 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 of the solid are randomly arranged, followed by cooling through slowly lowering the temperature of the heat bath. All particles arrange themselves 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.

To simulate the *evolution* to thermal equilibrium of a physical system (e.g., a many-particle crystalline solid) for

**Chart 1.** Algorithm 1: Implementation of the Metropolis Algorithm[a]

```
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;
```

[a] Input parameters are $x_{\mathrm{ini}}$, $k_{\max}$, $T$, and output parameter is $x_{\mathrm{out}}$. The procedure is repeated $k_{\max}$ times, symbol $O_{\mathrm{pertur}}$ 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 initialized by the input state $x_{\mathrm{ini}}$, after finishing the algorithm the current state is denoted by $x_{\mathrm{out}}$.

a fixed value of the temperature $T$, Metropolis *et al.*[24] have suggested a *Monte Carlo method*, which generates sequences of states of the system in the following way: Given the current state of the system (determined by positions of particles) a small random perturbation is created so that particles are displaced. If the difference $\Delta E = E_{\mathrm{pertur}} - E_{\mathrm{current}}$ between the perturbed state and the current state is negative ($E_{\mathrm{pertur}} < E_{\mathrm{current}}$), then the process is continued 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 $\leftarrow$ current), is given by $\exp(-\Delta E/T)$

$$\Pr(\text{perturbed} \leftarrow \text{current}) = \min(1, \exp(-\Delta E/T)) \quad (1)$$

This acceptance rule of new states is called the *Metropolis criterion*.

In order to formalize the Metropolis algorithm we introduce the following notation: The state of a 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 operator $x' = O_{\mathrm{pertur}}(x)$. The stochastic character of this operator consists in its random changes of entries of $x$ onto entries of $x'$. A pseudo-Pascal implementation of the Metropolis algorithm is outlined in algorithm 1. What is very important, the Metropolis algorithm produces[1,2] a probability distribution approaching the Boltzmann distribution

$$w_T(x) = \frac{1}{Q(T)} \exp\left(-\frac{f(x)}{T}\right) \quad (2a)$$

$$Q(T) = \sum_x \exp\left(-\frac{f(x)}{T}\right) \quad (2b)$$

where the summation runs over all states $x$.

The Metropolis algorithm can be used for the computer simulation of the method of simulated annealing. Simulated annealing can now be viewed as a sequence of Metropolis algorithms. Metropolis algorithms in this sequence are performed properly decreasing the values of the temperature, and, moreover, an output state $x_{\mathrm{out}}$ from the Metropolis algorithm serves as an input state $x_{\mathrm{ini}}$ for the next Metropolis algorithm. Initially, the temperature is given by a high value $T_{\max}$ and the Metropolis algorithm is applied until equilibrium is achieved ($k_{\max}$ times, where $k_{\max}$ − number of sampling is a sufficiently great number and belongs to the main parameters of the Metropolis algorithm). The temperature is then lowered in steps (e.g., by $T{:=}\alpha T$, where $0 \ll \alpha <$

**Chart 2.** Algorithm 2: Implementation of Simulated Annealing[a]

```
procedure Simulated_annealing($x_{opt}$,$T_{min}$,$T_{max}$,$k_{max}$,$\alpha$);
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:=$\alpha$*T;
      end;
      $x_{opt}$:=$x_{out}$;
end;
```

[a] Input parameters are $T_{\min}$, $T_{\max}$, $k_{\max}$, $\alpha$, and output parameter is $x_{\text{out}}$. The algorithm is initialized by random construction of the initial state $x_{\text{ini}}$ and by setting the temperature $T$ to its initial value $T_{\max}$. The cycle is repeated while the temperature $T$ is greater than its smallest value $T_{\min}$; the temperature $T$ is decreased by the parameter $\alpha$. After finishing the cycle the current state $x_{\text{out}}$ determines the solution of the algorithm denoted by $x_{\text{opt}}$.

1), with the system being allowed to approach equilibrium in each step by generating a sequence of states in the previously described way. The algorithm is terminated for some small value of the temperature $T_{\min}$, for which virtually no further deterioration of a low-energy state is accepted. The final "frozen" state $x_{\text{out}}$ is then taken as the final solution of the problem, see algorithm 2.

The method of simulated annealing was formulated in a general way so that the "physical ballast" was removed, though some physical terminology still plays a role of fruitful heuristics useful for better intuitive understanding of the method. The main purpose of the simulated annealing is a search for global solutions of large scale optimization problems of the type

$$x_{\text{opt}} = \arg\min_{x\in D} f(x) \tag{3}$$

where $f(x)$ is a real function determined over the domain $D$ (usually discrete and finite), and $x_{\text{opt}}$ is a value of variable corresponding to the global minimum of $f(x)$ over $D$. The variable $x$ is considered as a state of a hypothetical physical system, and the function $f(x)$ expresses its "energy". Then, after the above considerations, the optimization problem (3) may be successfully approached by the method of simulated annealing. The parameter "temperature" now plays a role of a control parameter of the method.

Analogy between the annealing of solids and the solving of optimization problem (3) is interesting either because of a phenomenological interest in the analogy or because it furnishes a possible framework to model the convergence and corresponding control of the simulated annealing.

Starting off with the fundamental assumption of statistical physics a number of useful macroscopic quantities can be derived from the equilibrium distribution of states of the system in a way similar to that for standard physical systems.

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

$$f{:}D \rightarrow A \subset R \tag{4}$$

where $A$ is a domain (a discrete and finite subset of $R$ composed of 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 eqs 2a and 2b.

In the case of our model calculations an evaluation of macroscopic quantities (6−9) can be done by approximating densities of functional values $w_T(y)$. 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

$$w_T(y) \approx \frac{\chi_T(y)}{\sum_{y'\in A} \chi_T(y')} \tag{5}$$

The following "macroscopic" quantities are defined:

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

$$\langle f\rangle_T = \sum_{y\in A} y w_T(y) \tag{6}$$

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

$$\langle f^2\rangle_T = \sum_{y\in A} y^2 w_T(y) \tag{7}$$

(3) For a numerical evaluation of the *entropy* $S(T)$ we have to know also cardinalities $|D(y)|$ that determine numbers of states $x$ with the same functional value $y = f(x)$. For simplicity, these entities are approximated by $|D(y)| = 1$, for each functional value $y \in A$, then the entropy $S(T)$ is approximated by

$$S(T) = -\sum_{y\in A} w_T(y) \ln w_T(y) \tag{8}$$

where densities $w_T(y)$ are approximated by (5).

(4) The *specific heat*

$$C(T) = \frac{\langle f\rangle_T^2 - \langle f^2\rangle_T}{T^2} \tag{9}$$

The density of functional values determined by (5) satisfies the following asymptotic property

$$\lim_{T\to 0} w_T(y) = w_0(y) = \delta(y,y_{\text{opt}}) = \begin{cases} 1 \text{ (if } y = y_{\text{opt}}) \\ 0 \text{ (if } y \neq y_{\text{opt}}) \end{cases} \tag{10}$$

where $y_{\text{opt}} = f(x_{\text{opt}})$ is the optimal value of the function $f(x)$ assigned to $x_{\text{opt}}$ determined by (3) as the global minimum. This limit property (10) is very important for the method of simulated annealing: It states that as the temperature $T$ tends to zero (through equilibrium states) the *probability of appearance of states corresponding to the global minimum is unity*. In statistical physics[25] it has been demonstrated that the quantities $\langle f\rangle_T$, $\langle f^2\rangle_T$, and $S(T)$ are decreasing functions as the temperature $T$ is also decreasing. The heat capacity $C(T)$ satisfies the following conditions

$$\lim_{T\to\infty} C(T) = \lim_{T\to 0} C(T) = 0 \tag{11}$$

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

## SIMULATED ANNEALING AND MOLECULAR GRAPHS

The purpose of this section is to outline an application of the simulated annealing approach to molecular graph construction problem. Let $G = (V,E)$ be a molecular graph composed of vertices from the vertex set $V$ and edges from the edge set $E$. Cardinalities of these two sets $|V| = p$ and $|E| = q$ correspond to numbers of vertices and edges of $G$, respectively. The so-called *molecular graph construction problem* is determined as follows: Let $\chi(\cdot)$ be a function determined over a set $H$ composed of molecular graphs that are specified by a class of required conditions (e.g., by the number of vertices and edges). The function $\chi(\cdot)$ assigns to the graph $G \in H$ a real number that determines its "numerical" specification. In chemistry[26,27] similar functions are called *topological indices*. Let us define the following form of objective function

$$f(G) = |\chi(G) - \chi_{\rm req}| \tag{12}$$

where $\chi_{\rm req}$ is a required value of the function $\chi(\cdot)$. In the construction phase then we look for such a graph $G_{\rm opt}$ that minimizes the objective function $f(G)$ for the set $H$

$$G_{\rm opt} = \arg \min_{G\in H} f(G) \tag{13}$$

This optimization (13) searches for a graph $G \in H$ with a functional value closely related (or even equal) to the required value $\chi_{\rm req}$.

We shall focus our attention on the following two topological indices: The Randić topological index[15]

$$\chi_R(G) = \sum_{[v,v']\in E} \frac{1}{\sqrt{\mathrm{val}(v)\mathrm{val}(v')}} \tag{14}$$

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, and the Wiener topological index[16]

$$\chi_W(G) = {}^1/_2 \sum_{\substack{v,v'\in V\\(v\neq v')}} d(v,v') \tag{15}$$

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 $v'$. Both these topological indices belong to topological indices most frequently used in chemistry for studies of structure vs property/activity correlation.[26,27]

A property of graphs from the set $H$ may be expressed as a function $\chi$ of different topological indices assigned to the given graph

$$\chi(G) = \mathrm{property}(\chi_1(G),\chi_2(G),...) \tag{16}$$

For instance, a hypothetical property is determined as a convex combination of Randić and Wiener topological

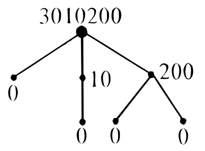

**Figure 1.** Illustrative example of construction of linear code of a tree. Going down-up in the tree, marginal vertices are evaluated by "0" (they have no successors), and then each nonmarginal vertex is evaluated by a code formed from its valence decreased by one and concatenated codes of its already coded successors on the lower level (arranged in an arbitrary way). The resulting code of the root serves also as a code of the whole tree.

indices

$$\chi(G) = \omega\chi_R(G) + (1 - \omega)\chi_W(G) \tag{17}$$

where $0 \leq \omega \leq 1$.

Two different types of molecular graphs are studied. The first type stands for acyclic molecular graphs that are graph-theoretically represented by trees. The second type represents connected graphs that may, in general, contain cycles. Although the first type is a subset of the second type, it is worthwhile to consider them separately. The numerical coding of the acyclic molecular graphs is much simpler than the coding of molecular graphs that may contain cycles.

## ACYCLIC MOLECULAR GRAPHS

The coding of trees can be carried out by many different methods. We turn our attention to the simplest of them invented by Read.[17] Let us consider a tree $T$ composed of $p = k$ vertices and $q = k - 1$ edges; its code is a $k$-tuple of nonnegative integers

$$\mathrm{code}(T) = \alpha = (\alpha_1\alpha_2...\alpha_k) \tag{18}$$

Read describes how to decode this code. For example, let us consider the code $\alpha = (3010200)$, see Figure 1. The first vertex (a root of the tree), with valence 3, has three branches. The first branch (going from the left to the right hand side), starting with the second vertex has no (0) further branches. The second branch, starting with the third vertex, has one (1) further branch on which lies the fourth vertex, which in turn has no (0) further branches. The third branch from vertex 1, starting with vertex 5, has two (2) further branches, each terminating in a marginal vertex (0,0). Unlike Read we do not require order of the codes of branches of the tree, so that one graph can have several codes. This enlarges the search space; on the other hand no recalculation is required, after the code is perturbed.

The sufficient conditions[17] for the sequence (18) to be graphable (i.e., there exists a tree with the same code) are simply determined by

$$\sum_{i=1}^{j}\alpha_i \geq j \quad (j = 1,2,...k - 1) \tag{19a}$$

$$\sum_{i=1}^{k}\alpha_i = k - 1 \tag{19b}$$

Let us define recurrently the following entities

$$d_1 = k - 1 \tag{20a}$$

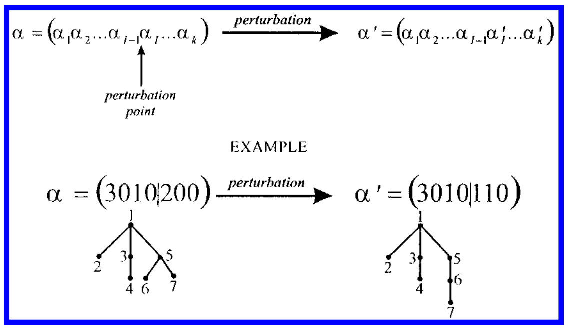
$\alpha = (\alpha_1\alpha_2...\alpha_{I-1}\alpha_I...\alpha_k)$
perturbation
$\alpha' = (\alpha_1\alpha_2...\alpha_{I-1}\alpha'_I...\alpha'_k)$
perturbation point
EXAMPLE
$\alpha = (3010|200)$
perturbation
$\alpha' = (3010|110)$
1 2 3 4 5 6 7
1 2 3 4 5 6 7

**Figure 2.** Schematic outline of the perturbation operation of the linear code. Starting from a randomly generated perturbation point the code is randomly modified so that conditions (19a,b) are satisfied. An example of this perturbation process shows diagrammatically the effect of perturbation on the tree from Figure 1.

$$d_j = d_{j-1} - \alpha_{j-1} \quad (j = 2,3,...,k) \tag{20b}$$

They bound from above the entries of the code α

$$1 \le \alpha_j \le d_j \quad (\text{if } d_j = k - j) \tag{21a}$$

$$0 \le \alpha_j \le d_j \quad (\text{if } d_j < k - j) \tag{21b}$$

for $j = 1, 2, ..., k$. These restrictions are very useful when tree codes are randomly generated; if their single entries are random nonnegative integers restricted by (21a,b), then the resulting codes automatically satisfy conditions (19a,b), see algorithm 3.

A perturbation of a code $\alpha = (\alpha_1\alpha_2...\alpha_k)$ onto another code of the same length $\alpha' = (\alpha'_1\alpha'_2...\alpha'_k)$ is formally realized by making use of the perturbation operator, $\alpha' = O_{\text{pertur}}(\alpha)$. This operator is applied so that the first part of original code α is copied up to a randomly generated index (perturbation point) $1 < I < k - 2$. The rest of the code is then randomly filled by nonnegative integers up to some randomly chosen length $k$ (within prescribed range) so that the resulting code satisfies conditions (19a,b), see Figure 2.

Crucial for the efficiency of the present approach is the calculation of topological indices for trees that are numerically represented by $k$-tuples (18). This problem is tightly related to the so-called parsing of the code, which means an unambiguous translation of the code to a form in which the vertices and edges are simply identified. We outline a simple *back-track approach* showing how to construct a spanning walk on the tree. This walk starts and terminates in the vertex indexed by 1 (root of tree). Moreover, in the course of its construction we are able to identify all vertices and edges that have not already been visited and to add their contribution to the Randić and/or Wiener topological indices. The first entry $\alpha_1 = 3$ means that the vertex (root) indexed by 1 has 3 branches. These branches will be successively visited in the forthcoming parsing so that, loosely speaking, the leftmost still nonvisited branch has the greatest priority and is to be used as the first one. The second entry $\alpha_2 = 0$ means that that the current branch is terminated by a vertex indexed by 2. We return to the first vertex, and the process of parsing is continued for the second branch, its form being determined by the third entry $\alpha_3 = 1$. It has an intermediate vertex indexed by 3, and the process of parsing is continued down by an edge terminated by a vertex indexed by 4 and specified by the fourth entry $\alpha_4 = 0$. In the next steps we have to return to the vertex indexed by 1, and the parsing process is continued for the third branch of the root. The single steps of the algorithm are illustrated by diagrams in Figure 3. An algorithmic back-track implementation of the parsing process is in algorithm 4.

**Chart 3.** Algorithm 3: Random Generation of Tree Code with $k$ Entries

```
Procedure Generation_code_tree(α,k);
begin $\alpha_1$:=1+random(k-1);
      $d_1$:=k-1;
      for j:=2 to k-1 do
      begin $d_j$:=$d_{j-1}$-$\alpha_{j-1}$;
            if $d_j$=k-j then
            $\alpha_j$:=1+random($d_j$) else
            $\alpha_j$:=random($d_j$+1);
      end;
      $\alpha_k$:=0;
end;
```

[a] I.e., the corresponding tree has $k$ vertices and $k - 1$ edges. The resulting codes automatically satisfy sufficient conditions (19a,b), it means that codes generated by this algorithm are graphable. Function random (d) generates a random integer ranged from 0 to $d$.

**Chart 4.** Algorithm 4: Back-Track Parsing Algorithm of Tree Linear Codes[a]

```
procedure Parsing_linear_code(α,k,V,E);
begin E:=∅;
      $index_0$:=1; $branch_0$:=$\alpha_1$; d:=0; i:=1;
      while d≥0 do
      if $branch_d$>0 then
      begin $branch_d$:=$branch_d$-1; i:=i+1; d:=d+1;
            $branch_d$:=$\alpha_i$; $index_d$:=i;
            E:=E∪{[$index_{d-1}$,$index_d$]};
      end else d:=d-1;
end;
```

[a] Single steps of the algorithm are illustrated by diagrams in Figure 3. Set variable E denotes the edge set of the tree constructed by the parsing of the code α. The depth of the parsing procedure is determined by the integer variable $d$. Indexed integer variables index $d$ determined the current vertex for the given depth $d$. Indexed integer variables branch$_d$ specify how many edges of the current vertex at the depth $d$ have not been used by the parsing process.

The Randić and Wiener topological indices for "linear trees" (e.g., trees with vertex valences bounded from the above by 2) can be explicitly evaluated. After simple considerations we get

$$\chi_R = \sqrt{2} + \frac{p-3}{2} \quad (\text{for } p \ge 3) \tag{22a}$$

$$\chi_W = \frac{p(p^2-1)}{6} \quad (\text{for } p \ge 1) \tag{22b}$$

We select such a required value of the property (17) which corresponds to linear trees. Then for a given $0 \le \omega \le 1$ we get

$$\chi_{\text{req}} = \omega\left(\sqrt{2} + \frac{p-3}{2}\right) + (1-\omega)\frac{p(p^2-1)}{6} \tag{23}$$

Then the objective function (12) is determined by

$$f_w(G) = |\omega\left(\sqrt{2} + \frac{p-3}{2} - \chi_R(G)\right) + (1-\omega)\left(\frac{p(p^2-1)}{6} - \chi_W(G)\right)| \tag{24}$$

In order to estimate roughly the complexity of the present inverse optimization problem, a few notes about the dimension of search space should be done. The considered search space would be composed of all $k$-tuples (18) that are restricted by conditions (19a,b). An average integer appearing in the first $k - 1$ positions is simply determined by

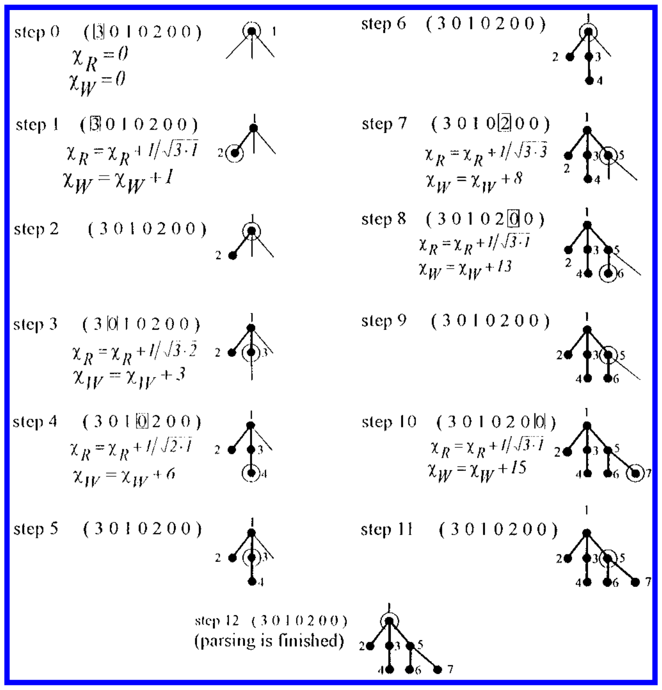

**Figure 3.** Parsing process of linear codes of the tree in Figure 1. The parsing is composed of 12 steps. The first step (initialization) reads the first entry (3) of the code, diagrammatically a root of the tree is created and three edges are delineated (still not terminated by vertices). In the second step we select most left delineated edge, and it is ended by the vertex indexed by 2. The second code entry (0) specifies that this vertex 2 has no successors; therefore in the next third step we return to the higher vertex indexed by 1. In the fourth step the second delineated edge is used, and it is ended by the vertex indexed by 3. The third code entry (1) specifies that this vertex has one successor, etc. Encircled vertices denote those vertices that are current in the given stage of the parsing process. Whole parsing process may be represented by a closed path starting and ending at the root (vertex indexed by 1) which visits all vertices of the parsed tree. The boxed code entry indicates that it is "read", i.e., its value is used for construction of delineated edges. The parsing process recognizes all edges of the tree, and during its course Randić and Wiener topological indices can be sequentially calculated.

$(k - 1)/(k - 1) = 1$, while the last integer is fixed to $\alpha_k = 0$. This means, loosely speaking, that in $k - 1$ positions at least two integers may appear, namely 0 and 1. Hence, a rough estimate of the number of all possible $k$-tuples is $2^{k-1}$. For instance, for $k = 25$ the search space is composed of about $2^{24} \approx 10^7$ *25*-tuples. It means that the simulated annealing is here looking for the correct solution among $10^7$ possible *25*-tuples.

## GENERAL MOLECULAR GRAPHS

The coding of general molecular graphs that may contain cycles is a much more complex problem than the similar problem for acyclic graphs. The complexity arises due to the fact that the isomorphism problem[28] for general graphs is a numerically very difficult problem likely of the NP type.[29] Search for the unique code of a structure could be plagued by existence of automorphisms of structures. Huge number of automorphisms can cause serious computational demands. Nevertheless, the condition for uniqueness of the code is not obligatory for stochastic approach. In the series of our recent papers[9,20−23] devoted to the constructive enumeration of molecular graphs we have used an approach based on the so-called semicanonic adjacency matrices. It proved to effectively reject those redundant adjacency submatrices that correspond to isomorphic subgraphs already constructed in the course of constructive enumeration process. In the present context the semicanonicity of adjacency matrices offers a simple and effective way of how to substantially reduce the number of different representations of isomorphic graphs. Those representatives appear in the process of simulated annealing when a graph is stochastically perturbed onto another graph.

Let the set $H$ from (13) be composed of connected graphs with $p$ vertices and $q$ edges (where $q \geq p - 1$). A graph $G \in H$ is represented by the adjacency matrix $\boldsymbol{A} = (a_{ij})$, a symmetric 0−1 matrix of the type $p \times p$. Its entries are determined by

$$a_{ij} = \begin{cases} 1 \text{ (if edge } [i,j] \in E(G)) \\ 0 \text{ (otherwise)} \end{cases} \tag{25}$$

For each index $1 < s < p$ we introduce two vectors

$$\boldsymbol{a}_s = (a_{s1}, a_{s2}, ..., a_{s,s-1}) \tag{26a}$$

$$\boldsymbol{b}_s = (a_{s+1,1}, a_{s+1,2}, ..., a_{s+1,s-1}) \tag{26b}$$

They are composed of the first $(s - 1)$ entries of the lower-triangle adjacency matrix taken from the $s$th and $(s + 1)$th rows, respectively. Let $\boldsymbol{a} = (a_i)$ and $\boldsymbol{b} = (b_i)$ be two $(s - 1)$-dimensional row vectors, they are equal, $\boldsymbol{a} = \boldsymbol{b}$, if $a_i = b_i$, for $i = 1, 2, ..., s-1$. These vectors may also be related by $(\boldsymbol{a} < \boldsymbol{b})$ $(\boldsymbol{a} > \boldsymbol{b})$, if there exists an integer $1 \leq i \leq s-1$ such that $a_j = b_j$, for $j = 1, 2, ..., i-1$, and $a_i < b_i$ $(a_i > b_i)$. The adjacency matrix $\boldsymbol{A}$ is called semicanonic, if for each $1 < s < p$ the row vectors $\boldsymbol{a}_s$ and $\boldsymbol{b}_s$, determined by (26a,b), satisfy

$$\boldsymbol{a}_s \geq \boldsymbol{b}_s \tag{27}$$

In our forthcoming considerations we shall manipulate semicanonic adjacency matrices. Their application allows us to avoid huge manipulations with redundant adjacency matrices and so will substantially reduce the search space of the optimization problem (15). We have to emphasize that our requirement to consider only the semicanonic adjacency matrices does not restrict forms of the appearing graphs. Each graph can be represented by a semicanonic adjacency matrix. A graph $G \in H$ may be indexed in such a way that the corresponding adjacency matrix is semicanonic; it means that many other forms of adjacency matrix assigned to the same graph are automatically omitted.

A semicanonic adjacency matrix that corresponds to a graph $G \in H$ can be subsequently randomly generated so that the condition (27) is fulfilled, see Figure 4. What we have to control in this extension process of adjacency matrices is restriction of the final number of "1" entries by the number $q$ of edges. Moreover, each row of the lower-triangle part must contain at least one "1" entry (this simple condition ensures that the resulting adjacency matrix corresponds to the connected graph). Restricting ourselves to molecular graphs, it is also necessary to check whether valences of vertices are not greater than a prescribed threshold value (e.g., four for saturated hydrocarbons).

Let $\boldsymbol{A}$ be a semicanonic adjacency matrix occurring in the current stage of the simulated annealing. Its perturbation onto another semicanonic adjacency matrix $\boldsymbol{A}'$ is formally performed by a perturbation operator, $\boldsymbol{A}' = O_{\text{pertur}}(\boldsymbol{A})$. A

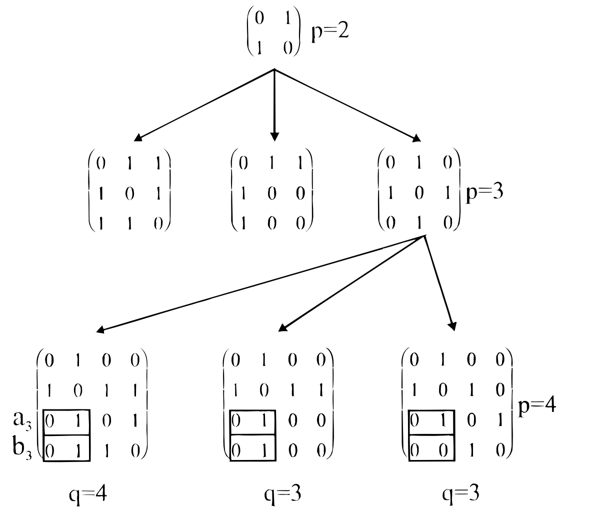

**Figure 4.** Random construction of semicanonic adjacency matrices assigned to molecular graphs composed of *p* vertices and *q* edges. Each level of this scheme is specified by the current number of vertices (i.e., dimension of adjacency matrices). In the third level two rows from the lower-triangle part are boxed; they correspond to row vectors $\boldsymbol{a}_3$ and $\boldsymbol{b}_3$ that must satisfy relation $\boldsymbol{a}_3 \geq \boldsymbol{b}_3$.

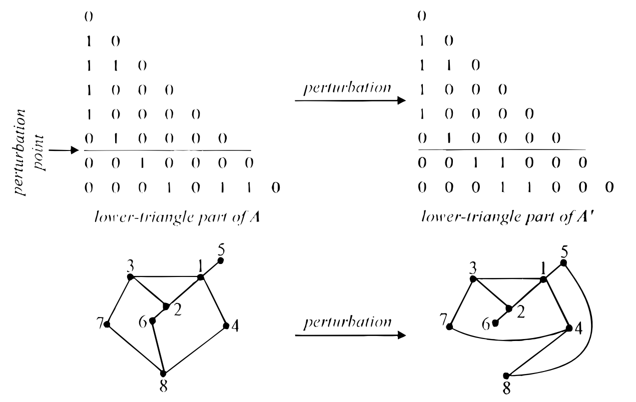

**Figure 5.** The perturbation process of a semicanonic adjacency matrix $\boldsymbol{A}$ onto another semicanonic adjacency matrix $\boldsymbol{A'}$. Starting from a randomly selected perturbation point, rows are randomly generated so that the resulting matrix is semicanonic and corresponds to a graph with *p* vertices and *q* edges.

stochastic perturbation transformation is realized so that starting from a randomly selected row the adjacency matrix $\boldsymbol{A}$ is again randomly extended to *p* rows and *q* “1” entries, so that the resulting perturbed matrix $\mathbf{A'}$ is semicanonic, see Figure 5.

Let us consider graphs that are composed of *p* vertices and $q = p$ edges and are simple cycles (i.e., all vertices have valences equal to 2). The Randić and Wiener topological indices of those graphs are simply determined as follows

$$\chi_R = \frac{p}{2} \tag{28a}$$

$$\chi_W = \begin{cases} \dfrac{p^3}{8} & \text{(p is even)} \\ \dfrac{p(p^2-1)}{8} & \text{(p is odd)} \end{cases} \tag{28b}$$

Then the required value of property (17) expressed as the convex combination of Randić and Wiener topological indices is determined by

$$\chi_{\text{req}} = \begin{cases} \omega\dfrac{p}{2} + (1-\omega)\dfrac{p^3}{8} & \text{(p is even)} \\ \omega\dfrac{p}{2} + (1-\omega)\dfrac{p(p^2-1)}{8} & \text{(p is odd)} \end{cases} \tag{29}$$

Introducing this result into (12) we get the objective function $f_\omega(G)$ determined for simple cycles.

Finally, few notes should be said about the dimension of search space composed of semicanonic adjacency matrices that correspond to graphs with *p* vertices and *q* edges. Since these matrices are symmetric and with zero values of diagonal entries, it is sufficient to consider only their lower-triangle parts without diagonal entries. They are composed of $(p-1)(p-2)/2$ elements with *q* “1” entries, it means that there exist

$$\begin{pmatrix} \dfrac{(p-1)(p-2)}{2} \\ q \end{pmatrix} \tag{30}$$

different adjacency matrices. These matrices correspond to general graphs composed of *p* vertices and *q* edges. The graphs may be connected as well as disconnected, and their vertex valences are not bounded from above by a small positive integer. In order to restrict this general (exact) result (30) to semicanonic adjacency matrices of molecular graphs (i.e., with only a few cycles and vertex valences bounded from above by four) the following rough estimation of the number of such matrices can be used: Since the unity entries are in semicanonic adjacency matrices mainly localized near the zero diagonal entries, we shall assume that each of $(p-1)$ rows contains *a* positions that can be occupied by *b* “1” elements, where $b < a$. Then, a rough evaluation of the number of different semicanonic adjacency matrices that correspond to molecular graphs is given by

$$\begin{pmatrix} a \\ b \end{pmatrix}^{p-1} \tag{31}$$

If we set $a = 5$ and $b = 2$ (i.e., rows contain four free positions that are occupied by two unit entries), then we get $10^{p-1}$ different semicanonic matrices of graphs composed of *p* vertices. For instance, for $p = 10$ roughly $10^9$ semicanonic adjacency matrices form the search space of the simulated annealing when applied to the construction problem.

## MODEL SIMULATED ANNEALING CALCULATIONS

In order to evaluate the effectiveness of the proposed simulated annealing construction of molecular graphs with required property represented by convex combination of Randić and Wiener topological indices we have carried out two types of model calculations. The first type was done for trees with number of vertices $p = 25$ and the required property specified by (23), whereas the second type was done for general graphs with number of vertices and edges determined by $p = q = 10$ and the required property specified by (17) and (28), where $\omega = 0.5$. Both types of simulated annealing calculations were based on the following values of control parameters $k_{\max} = 10^4$ and $\alpha = 0.9$. The used maximal and minimal temperatures were different for the first and second type of calculations, $T_{\max} = 100$, $T_{\min} = 0.1$ and $T_{\max} = 5$, $T_{\min} = 0.001$, respectively.

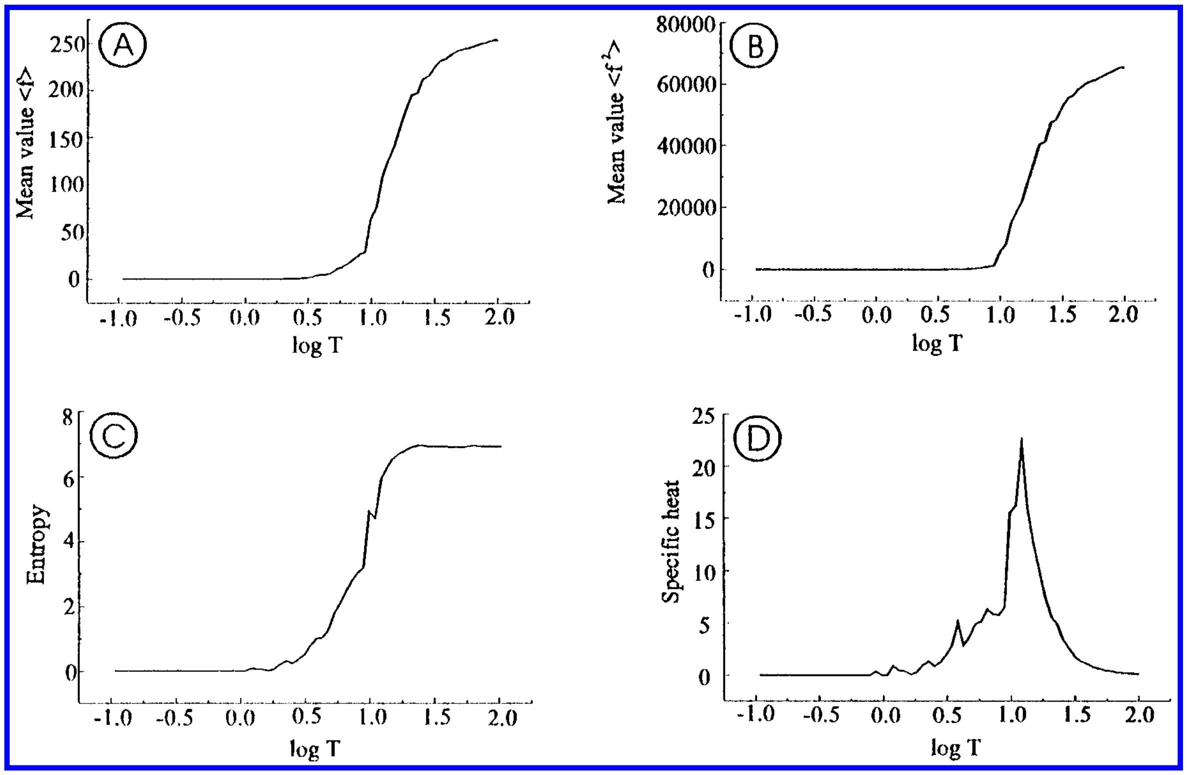

**Figure 6.** Plots of all four macroscopic quantities $\langle f \rangle_T$, $\langle f^2 \rangle_T$, *S(T)*, and *C(T)* for simulated annealing construction of the linear tree composed of 25 vertices, for parameters specified by $k_{max} = 10^4$, $T_{max} = 100$, $T_{min} = 0.1$, and $\alpha = 0.9$. Plots A to C are monotonously decreasing as the temperature tends to zero. Plot D of heat capacity manifests "giant" maximum, where according to concepts of statistical physics, transition processes are running.

These calculations were done for a number of randomly generated trees and connected graphs with cycles, for which the value of property was calculated and this value was supplied to the simulated annealing algorithm as the required one. The simulated annealing method gave correct results that corresponded to molecular graphs isomorphic with those ones used for the evaluation of required properties.

Plots of macroscopic quantities (6−9) are displayed in Figures 6 and 7. The calculations for these examples were done for trees with 25 vertices and for connected graphs with 1 cycle and 10 vertices. For trees the linear tree and for general graphs the simple cycle have been achieved by the simulated annealing. Moreover, we have recorded in both calculations the best solutions obtained so far in the course of simulated annealing, and it was observed that they were equal (after four or five iterations of Metropolis algorithm in algorithm 2) to the correct results. The plots of heat capacities manifest for some temperature region sharp maxima. For this temperature range the so-called state transitions appear and it is recommended[1−3] here to decrease temperature steps (i.e., the constant $0 \ll \alpha < 1$ should be increased). The plots of $\langle f \rangle_T$, $\langle f^2 \rangle_T$, and *S(T)* manifest correct behavior, i.e., they are monotonously decreasing as the temperature *T* tends to zero. It means, that large enough control parameter $k_{max}$ was selected. In Figures 8 and 9 are displayed trees and graphs, respectively, produced by the simulated annealing method for different temperatures. We see that as the temperature is decreasing the trees and graphs are more and more similar to the required linear trees and simple cycles. For sufficiently small temperature they are identical to the linear tree and the simple cycle. To summarize the above considerations, the method of simulated annealing is able to find correct solutions from huge search spaces, where their dimensions are roughly determined for the trees with $p = 25$ by $10^7$ and for the general molecular graphs with $p = q = 10$ by $10^9$.

## ILLUSTRATIVE EXAMPLE—ASSIGNING A PARTICULAR $^{13}C$ NMR CHEMICAL SHIFT TO THE CORRESPONDING ALKANE CARBON ATOM FROM A SET OF ALL $C_2$−$C_9$ ALKANES

Another illustrative example of the present simulated annealing theory of a construction of molecular graphs with a required property is its simple application to the construction of "rooted" alkanes $C_2$−$C_9$, where the "root" $^{13}C$ carbon atoms have the desired value of the NMR chemical shift. Experimental values of 326 $^{13}C$ NMR chemical shifts of topologically nonequivalent carbon atoms of 63 alkanes were taken from the literature.[30,31]

Carbon atoms in alkanes were determined by 13 descriptors that correspond to the so called embedding frequencies of rooted subtrees.[32] These descriptors are equal to numbers of appearance of smaller structural skeletons composed of two to five carbon atoms, with one distinguished carbon atom, which NMR shift is computed. All the rooted subtrees through five vertices are considered, with exception of "star" rooted subtrees with 2, 3, and 4 edges going from the root. Those subtrees can be determined by a number of subtrees consisting from a single edge going from the root. Embedding frequency does not uniquely determine a root carbon atom for larger alkanes, but it determines the neighborhood of the root carbon atom up to $\delta$-effects.

In our recent work[33] we have found a linear regression between these embedding frequencies as descriptors and chemical shifts of the corresponding $^{13}C$ carbon atoms in alkanes.

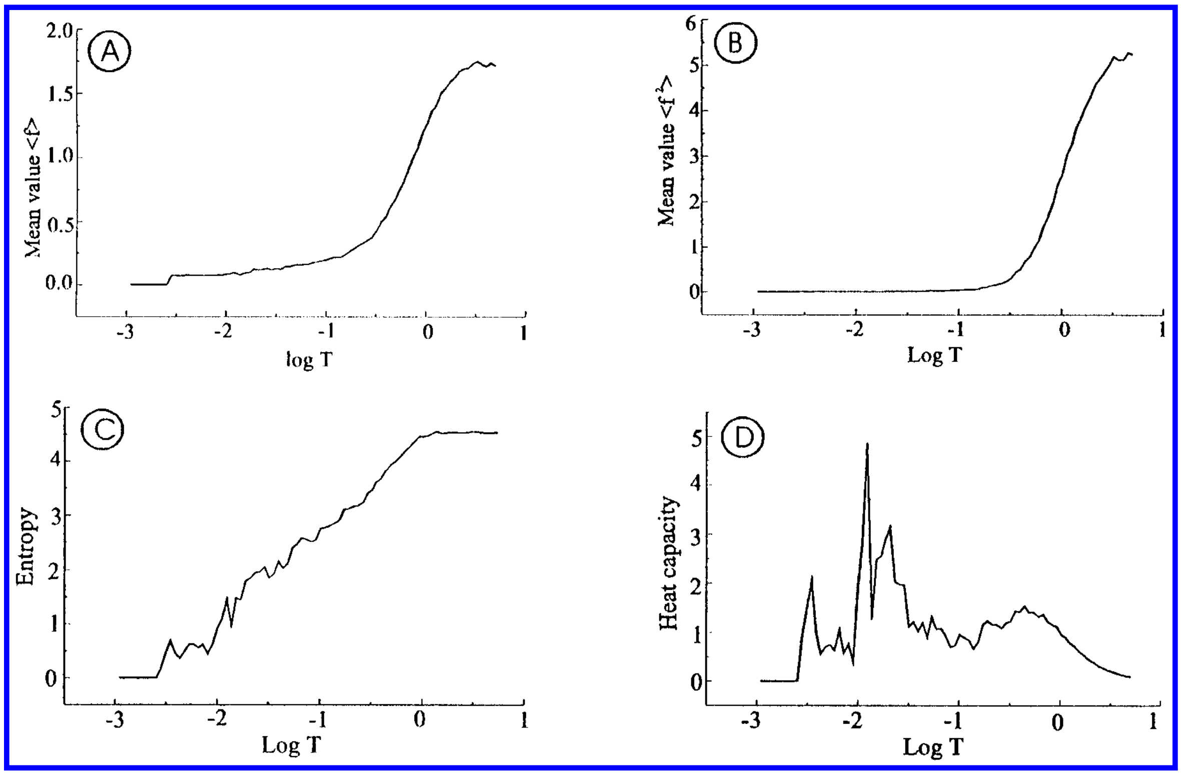

**Figure 7.** Plots of all four macroscopic quantities $\langle f\rangle_T$, $\langle f^2\rangle_T$, $S(T)$, and $C(T)$ for simulated annealing construction of the cycle composed of 10 vertices and 10 edges, for parameters specified by $k_{max} = 10^4$, $T_{max} = 5$, $T_{min} = 0.001$, and $\alpha = 0.9$. Plots A to B are nicely monotonously decreasing as the temperature approaches zero. Plot C of the entropy $S(T)$ manifests many small local maxima, though overall it is decreasing as the temperature is also decreasing. Plot D of the heat capacity manifests many local maxima in broad range of temperatures, indicating that the control parameter $k_{max}$ is not sufficiently large.

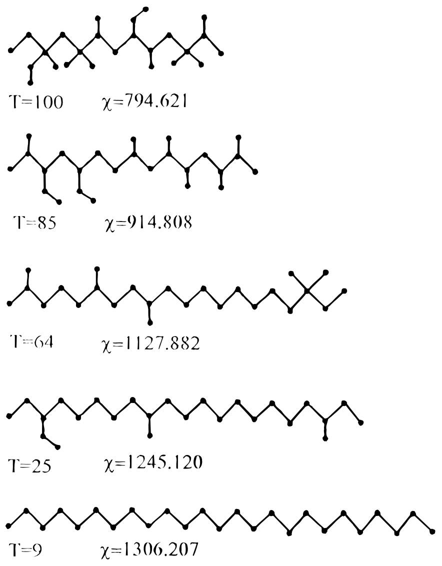

**Figure 8.** Some illustrative trees that appeared in the course of simulated annealing construction of the linear tree with 25 vertices for five different temperatures. As the temperature decreases the number of branches in trees is also decreased. Starting for the temperatures smaller than 9 only linear trees appear in the simulated annealing.

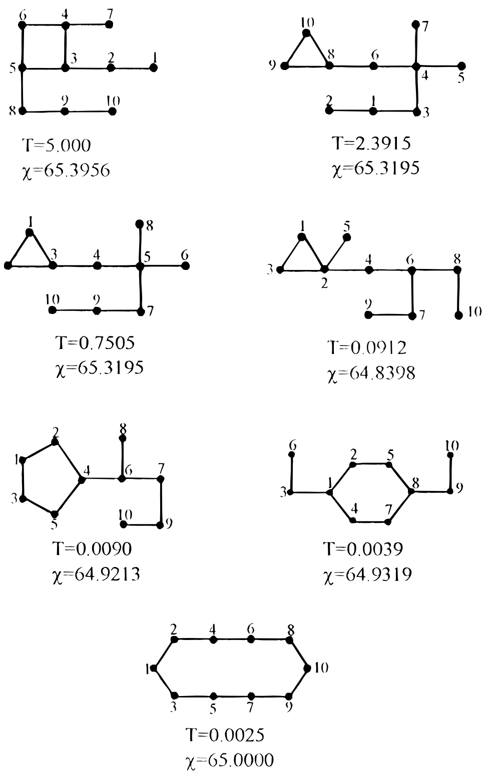

**Figure 9.** Some illustrative graphs for different temperatures that appeared in the course of simulated annealing construction of simple cycle composed of 10 vertices and 10 edges, where all vertices are of the same valence equal to 2. As temperature is decreasing to zero, the graphs have progressively larger cycles. Starting from the temperatures smaller than 0.0025 only correct cycles appear in the simulated annealing.

In the present model calculations we have picked up a chemical shift of a concrete carbon atom of an alkane. This shift was taken as a required value, and corresponding rooted carbon atom chosen from all carbon atoms from all alkanes with prescribed number of vertices was searched. We have recorded best solutions from parallel simulated annealing reconstruction. The carbon atom indexed by "1" was always taken as the root, and each constructed rooted tree was analyzed, so that the embedding frequencies were found and corresponding shift was calculated by the linear regression.[33] We have found that in all trial cases the right carbon atom

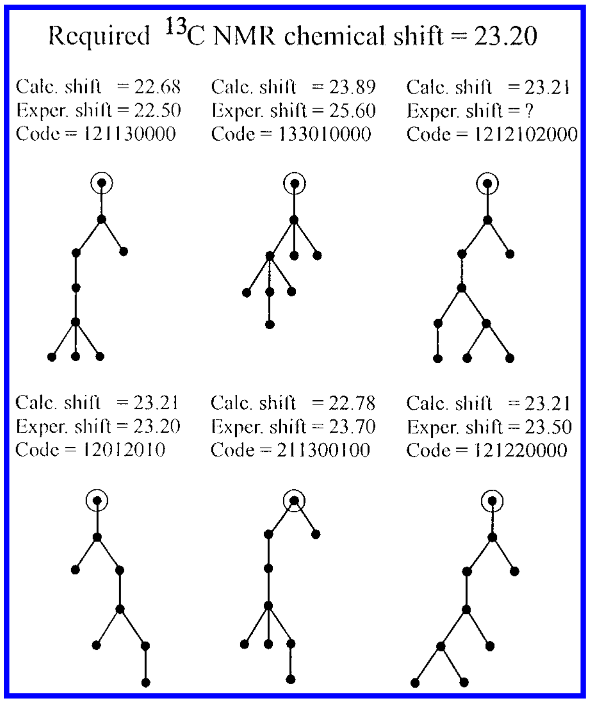

**Figure 10.** Results of illustrative construction of rooted alkanes with the required $^{13}$C NMR chemical shift equal to 23.20 taken from literature, the property function $\chi(G)$ is determined by linear regression[33] from embedding frequencies.[32] The six resulting rooted alkanes have $^{13}$C atom depicted by an encircled dot. Both calculated and experimental $^{13}$C NMR chemical shift are given (for the third alkane in the row we had no experimental value in a database). The correct answer is the fourth alkane.

of the right alkane was among the best solutions. The main problem represents not the quality of construction, but the quality of regression. In order to record the correct solution, where prediction by regression differs more substantially from the measured value, more solutions within a broader range of values have to be recorded. Therefore the annealing procedure does not have to run to the low temperature, because its precision is higher than precision of regression, even at relatively high annealing temperature. An example of construction of a rooted alkane with the required (measured) $^{13}$C NMR chemical shift of a "root" carbon quite close to its value calculated by regression is displayed in Figure 10.

## DISCUSSION

The simulated annealing construction of molecular graphs with required property provides a simple and effective numerical tool for solution of an inverse problem to those typical in the research of structure vs activity/property relationships. In particular, the usual approach in this recently very active part of computational chemistry consists of regression analysis of data from the training set. An analytic expression is constructed, where molecular descriptors (often determined by physico-chemical parameters and/or topological indices) serve as independent variables that are mapped onto the required property. This functional relationship is used for fast prediction of the required property of compounds outside the training set. The presented simulated annealing method offers a new possibility to solve effectively the inverse problem, where the numerical value of the required property is prescribed, and we look for those graphs that have property values closely related (or even equal) to the prescribed one.

The computational times were around 1 min on PC 486/33MHz, for generation of trees and cycles with required value of combination of topological indices as well as for prediction of alkanes with required $^{13}$C NMR chemical shift. The computational time heavily depends on chosen parameters, and more computational time usually means higher probability to find exact optimum. However, the error of the used regression function (representing the quality of predicted $^{13}$C NMR chemical shift) was relatively high comparing with precision of optimum, which can be obtained for this already chosen evaluation function by construction of graphs controlled by simulated annealing. It is not reasonable to get the perfect optimum of a function, which does not mirror so precisely the facts existing in reality. The stress should be put to get a better evaluation function first, before getting its perfect optimum.

The crucial moment of each simulated annealing method is the perturbation of a current "state" onto another "state", where both "states" belong to the same set of objects (e.g., restricted by the same class of conditions). In the present paper we start from a random perturbation point and the rest of the code (linear code for trees or adjacency matrix for general graphs) is again randomly generated (see Figures 2 and 5). Of course, there exist many other ways of how to define a perturbation. For instance, we have experimented with a perturbation that is much more "gentle" than that one being used, so that a randomly selected edge is moved to a randomly selected empty position. For trees it should be specified so that the moved edge is marginal (a leaf of tree), in the opposite case, when an arbitrary edge is moved, it may happen that a disconnected "tree" is produced. For general graphs the similar situation may appear. Therefore, an application of this simple idea for general graphs must be accompanied by a check whether the resulting graph is connected or not. This and other problems were main reasons why we have decided to use the form of perturbation as was presented in the previous section.

The generalization of perturbations to operations on molecular graphs, which can increase or decrease the number of vertices and/or edges, should not present any problem. Concrete results should be presented in further publications.

The method of simulated annealing is usually considered as a stochastic method almost compatible with the genetic algorithms.[14] Therefore, in order to ensure this compatibility, it is necessary to introduce in the framework of genetic algorithms the so-called *crossover operation,* which combines two numerical graph codes into other two numerical codes. Unfortunately, a straightforward realization of the crossover operation may give rise to some difficulties connected with the fact that produced codes do not satisfy conditions required for graphs. In particular, since linear codes of trees must satisfy restriction conditions (19a,b), it is necessary to apply some *repair process* in order that the produced codes will again satisfy these condition. A similar situation exists in the traveling salesman problem (TSP) solved by genetic algorithms[14] where a permutation representation of the traveling salesman path is used. The crossover operation has to be accompanied by a repair process[14] which ensures that the produced new chromosomes correspond again to permutations.

Future strategy for the design of molecules with required properties may switch from simulated annealing to evolutionary programming,[34] which uses the same type of operations as simulated annealing. In contrast to simulated annealing, it works with a set of possible solutions-codes, instead of a single one, and all the possible solutions are subjected to mutation (perturbation in simulated annealing). Then the whole set is randomly divided into nonoverlapping subsets, and then the selection takes place, choosing the fittest of each group to survive into the next generation. This approach is recurrently repeated. Evolutionary programming represents evolution of different kinds of species, which do not mate; therefore it has no need for crossover as genetic algorithms, which represent population of one species. This algorithm, together with parallel simulated annealing,[5] can give several different results close to the correct one, so that the user can have more choice. The same goal can be reached by saving several different best solutions during a run of a simple simulated annealing; however, such solutions would probably tend to be not so divergent as those ones given by evolutionary programming or parallel simulated annealing.

## REFERENCES AND NOTES

(1) Van Laarhoven, P. J. M.; Aarts, E. H. L. *Simulated Annealing: Theory and Applications*; D. Reidel Publishing Company: Dordrecht, 1987.

(2) Otten, R. H. J. M.; van Ginneken, L. P. P. P. *The Annealing Algorithm*; Kluwer Academic Publishers: Dordrecht, 1989.

(3) Kirkpatrick, S.; Gelatt, C. D.; Vecchi, M. P. Optimization by Simulated Annealing. *Science* **1983**, *220*, 671−680.

(4) Černý, V. Thermodynamic Approach to the Traveling Salesman Problem: An Efficient Simulation Algorithm. *J. Opt. Theory Appl.* **1985**, *45*, 41−51.

(5) Pospíchal, J.; Kvasnička, V. Fast Evaluation of Chemical Distance by Simulated Annealing Algorithm. *J. Chem. Inf. Comput. Sci.* **1993**, *33*, 879−885.

(6) Snow, M. E. Powerful Simulated Annealing Algorithm Locates Global Minimum of Protein-Folding Potentials From Multiple Starting Conformations. *J. Comput. Chem.* **1992**, *13*, 579−584.

(7) Sutter, J. M.; Dixon, S. L.; Jurs, P. C. Automated Descriptor Selection for Quantitative Structure−Activity Relationships Using Generalized Simulated Annealing. *J. Chem. Inf. Comput. Sci.* **1995**, *35*, 77−84.

(8) Michalewicz, Z. *Genetic Algorithms+Data Structures=Evolution Programs*; Springer Verlag: Berlin, 1992.

(9) Kvasnička, V.; Pospíchal, J. Canonical Indexing and Constructive Enumeration of Molecular Graphs. *J. Chem. Inf. Comput. Sci.* **1990**, *30*, 99−105.

(10) Skvortsova, M. I.; Baskin, I. I.; Slovokhotova, O. L.; Palulin, V. A.; Zefirov, N. S. Inverse Problem in QSAR/QSPR Studies for the Case of Topological Indices Characterizing Molecular Shape (Kier Indices). *J. Chem. Inf. Comput. Sci.* **1993**, *33*, 630−634.

(11) Venkatasubramanian, V.; Chan, K.; Caruthers, J. M. Evolutionary Design of Molecules with Desired Properties Using the Genetic Algorithm. *J. Chem. Inf. Comp. Sci.* **1995**, *35*, 188−195.

(12) Sheridan, R. P.; Kearsley, S. K. Using a Genetic Algorithm To Suggest Combinatorial Libraries. *J. Chem. Inf. Comput. Sci.* **1995**, *35*, 310−320.

(13) Derringer, G. C.; Markham, R. L. A Computer-Based Methodology for Matching Polymer Structure with Required Properties. *J. Appl. Polymer. Sci.* **1985**, *30*, 4609−4617.

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

(15) Randić, M. On Characterization of Molecular Branching. *J. Am. Chem. Soc.* **1975**, *97*, 6609−6615.

(16) Wiener, H. Structural Determination of Paraffin Boiling Points. *J. Am. Chem. Soc.* **1947**, *69*, 17−20.

(17) Read, R. C. The Enumeration of Acyclic Chemical Compounds. In *Chemical Application of Graph Theory*; Balaban, A. T., Ed.; Academic Press: London, 1976; pp 25−61.

(18) Knop, J. V.; Muller, W. R.; Szymanski, K.; Trinajstić, N. *Computer Generation of Certain Classes of Molecules*; SKTH: Zagreb, Croatia, 1985.

(19) Knop, J. V.; Muller, W. R.; Jercevic, Z.; Trinajstić, N. Computer Enumeration and Generation of Trees and Rooted Trees. *J. Chem. Inf. Comput. Sci.* **1981**, *21*, 91−99.

(20) Kvasnička, V.; Pospíchal, J. Constructive Enumeration of Acyclic Molecules. *Collect. Czech. Chem. Comm.* **1991**, *56*, 1777−1802.

(21) Kvasnička, V.; Pospíchal, J. Constructive Enumeration of Molecular Graphs with Prescribed Valence States. *Chemometrics Intell. Lab. Syst.* **1991**, *11*, 137−147.

(22) Kvasnička, V.; Pospíchal, J. An Improved Method for Constructive Enumeration of Graphs. *J. Math. Chem.* **1992**, *9*, 181−196.

(23) Pospíchal, J.; Kvasnička, V. Alternative Approach for Constructive Enumeration of Graphs. *Collect. Czech. Chem. Comm.* **1993**, *58*, 754−774.

(24) Metropolis, M.; Rosenbluth, A. W.; Rosenbluth, M. N.; Teller, A. H.; Teller, E. Equation of State Calculations for Fast Computing Machines. *J. Chem. Phys.* **1953**, *21*, 1087−1092.

(25) Toda, M.; Kubo. R.; Saito, N. *Statistical Physics*; Springer Verlag: Berlin, 1983.

(26) Rouvray, D. H. Topological Indices as Chemical Behavior Descriptors. *Congr. Numerantium* **1985**, *49*, 161−179.

(27) Rouvray, D. H. The Modeling of Chemical Phenomena Using Topological Indices. *J. Comput. Chem.* **1987**, *8*, 470−480.

(28) Read, R. C.; Corneil, D. G. The Graph Isomorphism Disease. J. *Graph Theory* **1977**, *1*, 339−352.

(29) Garey, M. R.; Johnson, D. S. *Computers and Intractability - A Guide to the Theory of NP-Completeness*. Freeman and Co.: San Francisco, 1979.

(30) Lindeman, L. P.; Adams, J. Q. Carbon-13 Nuclear Magnetic Resonance Spectroscopy: Chemical Shifts for the Paraffins though $C_9$. *Anal. Chem.* **1971**, *43*, 1245−1252.

(31) Kalinowski, H. O.; Berger, S.; Braun, S. *$^{13}$C NMR Spektroskopie;* G. Thieme Verlag: Stuttgart, 1984.

(32) Kvasnička, V.; Pospíchal, J. Simple Construction of Embedding Frequencies of Trees and Rooted Trees. *J. Chem. Inf. Comput. Sci.* **1995**, *35*, 121−128.

(33) Svozil, D.; Pospíchal, J; Kvasnička, V. Neural-Network Prediction of Carbon-13 NMR Chemical Shifts of Alkanes. *J. Chem. Inf. Comput. Sci.* **1995**, *35*, 924−928.

(34) Fogel, D. *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence*; IEEE Press: Piscataway, NJ, 1995.

CI9500703