**Session 2520**

# Optimization Using the Simulated Annealing Algorithm

Edgar N. Reyes, Dennis I. Merino, Carl Steidley
Southeastern Louisiana University/Texas A&M University - Corpus Christi
Hammond, LA 70402/Corpus Christi, TX 78412

**Abstract**

In this paper we will briefly review the simulated annealing algorithm, an algorithm with applications in optimization and pattern recognition used extensively in artificial intelligence. In earlier papers the authors analyzed a simulation of the annealing of a solid, a dodecahedron in particular. Our use of this algorithm, which is based in the field of combinatorial optimization, reflects properties of Boltzman machines - a neural network characterized by massive parallelism.

We will demonstrate two implementations of this algorithm in simulated annealing. Each of the implementations depends upon a neighborhood structure and a transition mechanism. In the first implementation our neighborhood structure is a linear transformation of the vector space of all configurations and the transition probability is deterministic. In this case, we will use techniques from character theory of finite groups to analyze simulated annealing. In the second implementation, a special case of which includes the first implementation, our neighborhood structure is a set-valued function and the transition mechanism is stochastic in nature. In this case, we use techniques from matrix analysis, in particular properties of doubly stochastic matrices, to analyze simulated annealing modeled and based on a class of Boltzman machines.

For pattern recognition, we use the simulated annealing algorithm to solve the classic *seven-segment display problem*. This is a classification problem which we will solve by choosing an appropriate Boltzmann machine.

## 1. Introduction.

Annealing is the physical process of heating up a solid and following it by a specified slow cooling process. We shall use the simulated annealing algorithm, a method based in the field of combinatorial optimization, to describe simulated controlled cooling processes. In the annealing process, one can interpret the states (and free energy) of the solid in the cooling process as solutions (and cost function, respectively) of a combinatorial optimization problem [1]. Our use of the simulated annealing algorithm reflects properties of Boltzman machines, a neural network model belonging to a class of connectionists models and which has massive parallelism as a feature, amongst others.

Also, we will use an appropriate Boltzmann machine to solve a pattern recognition problem, namely, *the seven-segment display problem*. The display of the decimal digits in a hand-watch for instance uses a seven-segment display. In identifying the digit displayed, we will maximize an overall measurement of desirability of the Boltzmann machine.

We shall briefly review some aspects of Boltzman machines and the simulated annealing algorithm. Let *(U, C)* be a network consisting of units, $U = \{u_i : i = 1, \ldots, n\}$, and a set of connections, *C,* consisting of unordered pairs $\{u_i, u_j\}$. A connection $\{u_i, u_j\}$ in *C* is said to join $u_i$ to $u_j$. Intrinsic to Boltzman machines are the notions of a connection strength *s* and a

configuration *k* of the network *(U, C)*; respectively, they are real-valued functions defined on *C* and *U*, respectively. The values $s\{u_i, u_j\}$ and $k(u_i)$ give us the strength of the connection $\{u_i, u_j\}$ and the state of the unit $u_i$, respectively. Thus, a Boltzman machine is a network *(U,C)* with a given connection strength s [1, chapter 8]. An objective of a Boltzman machine is to find an optimal configuration $k_{opt}$ in the space *S* of all configurations *k* that minimizes the consensus function defined by

$$C(k) = \sum_{\{u_i, u_j\} \in C} s\{u_i, u_j\} k(u_i) k(u_j) . \qquad (1.1)$$

The values of the consensus function provide an overall measurement of desirability of the connections and the states of the units. The function in (1.1) is usually called the cost function in combinatorial optimization.

To optimize (1.1), we will use the simulated annealing algorithm. There are several ways to implement this algorithm, moreover, each way depends on a neighborhood structure and a transition mechanism. A neighborhood structure is a function *N* from *S* into *P( S)*, the family of all subsets of *S*. A configuration *l* in *N(k)* is called a neighbor of *k*. To optimize the consensus (1.1), we need a mechanism which allows a configuration to change. Given a configuration *k*, we shall randomly generate a neighbor *l* in *N(k)*, with the neighborhood structure being defined at the outset, and then it will be determined whether *l* will replace *k*. Specifically, let *X(m)* be the configuration on the *m*th trial and let $P_{k,l}(m) = P(X(m) = l \mid X(m-1) = k)$ be the probability of accepting configuration *l* on the *m*th trial given that the configuration of the *(m-1)*th trial is *k*. Under certain conditions, such as those discussed in [1, page 18, 42, or 46], the sequence of configurations generated by the simulated annealing algorithm asymptotically converges to an optimal configuration.

The controlled cooling process is represented by a sequence $\{c_m \mid m=0,1,2,3,...\}$ of real numbers. Following [5], the simulated annealing is described by the algorithm below

```
Begin Simulated Annealing Algorithm
     Initialize:
          k_o ; an initial configuration
          m = 0; a counter
     Do: generate k in N(k_o)
          if C(k)≤C(k_o), then k_o = k

          else if P_{k_o,k}(m)> Random(0,1), then k_o = k
          m:=m+1;
     Until:    Calculate Stop Criterion (c_m);
end;
```

## 2. A Special Case : Cooling of a Dodecahedron

Our first implementation is the simulated annealing of a dodecahedron, i.e., a regular solid with *12* faces which we choose because of its inherent symmetry.. We start with a network $(U_d, C_d)$ consisting of units that are connected in some way. The set $U_d = \{u_i \mid i = 1,..,12\}$ represents the set of faces of a dodecahedron. A connection $\{u_i, u_j\}$ in $C_d$ is said to join $u_i$ to $u_j$. For computational purposes, let us suppose the following pairs of faces are opposite of each other

$u_1$ and $u_2$, $u_3$ and $u_{10}$, $u_4$ and $u_{11}$,
$u_5$ and $u_{12}$, $u_6$ and $u_8$, $u_7$ and $u_9$ (2.2)

In describing a cooling process for the dodecahedron, we label the *12* faces of the dodecahedron with real numbers. These numbers play the role of the temperatures of the faces. Let us suppose in this cooling process, that the numbers (or temperatures) change every minute. At the first minute , initially, assume for each *i* in *{1,...,12}* that face $u_i$ is labeled with the number *i* representing its temperature. On the second minute, the numbers on each of the faces changes according to a specified schedule. That is, on the second minute the number on face $u_i$ becomes the average of the first minute's numbers except those that were on $u_i$ and on the face opposite $u_i$. For example, on the second minute, the number on face $u_1$ is $\frac{1}{10}\sum_{i=3}^{12} i$ ; since $u_2$ is the face opposite $u_1$ then the numbers on the second minute on $u_1$ and $u_2$ must be equal. Also, the number on the third and tenth faces on the second minute is given by $\frac{1}{10}\sum_{i=1, i\neq 3,10}^{12} i$ . We continue with this process and on the third minute, the number on face $u_i$ becomes the average of the second minute's numbers except those that were on $u_i$ and on the face opposite $u_i$. Repeating this process for four minutes, we find that the numbers on the faces are given by (rounded to 1 decimal place)

| Min | $u_1$ | $u_2$ | $u_3$ | $u_4$ | $u_5$ | $u_6$ | $u_7$ | $u_8$ | $u_9$ | $u_{10}$ | $u_{11}$ | $u_{12}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| 2 | 7.5 | 7.5 | 6.5 | 6.3 | 6.1 | 6.4 | 6.2 | 6.4 | 6.2 | 6.5 | 6.3 | 6.1 |
| 3 | 6.3 | 6.3 | 6.5 | 6.5 | 6.6 | 6.5 | 6.6 | 6.5 | 6.6 | 6.5 | 6.5 | 6.6 |
| 4 | 6.5 | 6.5 | 6.5 | 6.5 | 6.5 | 6.5 | 6.5 | 6.5 | 6.5 | 6.5 | 6.5 | 6.5 |

(2.3)

In particular, the numbers (or temperatures) for the first four minutes on face $u_1$ are *1, 7.5, 6.3,* and *6.5*. For each face, note that the number on the fourth minute is approximately *6.5*, which is the average of the integers *1,...,12.* Then as the number of minutes increases (or as more trials or iterations are done) the numbers on the faces shall all be approximately equal to *6.5.*

To find the `equilibrium or optimal state' of the dodecahedron, i.e. the configuration when the temperatures on the faces are all equal, we shall seek to minimize

$$\sum_{i=1}^{12}\left(k(u_i)-m(k)\right)^2 \tag{2.4}$$

where $m(k)=\frac{1}{12}\sum_{i=1}^{12}k(u_i)$ over all configurations *k*. To implement the simulated annealing algorithm, we need to interpret (2.4) as the consensus of *k*. To do this, we shall need to define an appropriate connection strength. Simplifying (2.4), we obtain $\sum_{i=1}^{12}\left(k(u_i)-m(k)\right)^2=\frac{11}{12}\sum_{i=1}^{12}k(u_i)^2-\frac{1}{6}\sum_{i<j}k(u_i)k(u_j)$ and by defining the connection strength *s* on *C* according to

$$s(\{u,v\})=\begin{cases}\frac{11}{12} & if \quad u=v\\ -\frac{1}{6} & otherwise\end{cases} \tag{2.5}$$

we find that the consensus *C(k)* in (1.1) and cost function in (2.4) become equal. Next, we provide a neighborhood structure *N* for the cooling process given in (2.3). In this first implementation, the values of the neighborhood structure *N* shall be singletons instead of being set-valued in general. Given a configuration *k*, let *N(k)* be the configuration defined by

$$N(k)(u_i)=\frac{1}{10}\sum k(u_j) \tag{2.6}$$

where the sum is taken over all faces $u_j$, except $u_i$ and the face opposite $u_i$. If we define the transition probabilities by

$$P_{k,l}(m)=\begin{cases}1 & if\ N(k)=l\\ 0 & otherwise\end{cases} \tag{2.7}$$

for all trials *m*, then together with the program for the simulated annealing algorithm given at the end of the introduction we have described a cooling process of the dodecahedron.

The neighborhood structure *N* in (2.6) completely describes the simulated annealing of the dodecahedron as described in [6]. In particular, the equation in (2.6) defines a linear transformation *N* from the complex vector space *V* (consisting of all configurations $\{u_1,\ldots,u_{12}\}$ where $u_i$ is a complex number) into itself. Certain invariant subspaces and eigenvalues of the linear transformation are identified by using techniques of character theory (for instance); which in turn lead us to conclude that iterates of this operator converges to a multiple of the identity operator. In particular, $\lim_{m\to\infty}N^m=\frac{1}{12}I_{12}$ where $I_{12}$ is the identity operator on *V*. Thus, as we have shown in [6], if the initial numbering (or set of temperatures) of the *12* faces is given by the sequence $(B^o{}_1, B^o{}_2, \ldots, B^o{}_{12})$ of real numbers, i.e., face $u_i$ is labeled (or has the temperature)

$B_i^o$ , then after several minutes or iterations each number (or temperature) on the face shall be approximately $\frac{1}{12}\sum_{i=1}^{12} B_i^0$ .

## 3. Using Matrices in Simulated Annealing

In this section, we generalize the results (as derived in [4]) of the previous section.

Let $U = \{u_i \mid i = 1,\ldots,n\}$ and let $C = \left\{\{u_i, u_j\} \mid 1 \le i, j \le n\right\}$ be a network representing a Boltzman machine, where $n \ge 2$ . For the connection strength *s*, we consider

$$s\{u_i, u_j\} = \begin{cases} \dfrac{n-1}{n} & \text{if} \quad i = j \\ -\dfrac{2}{n} & \text{otherwise} \end{cases} \tag{3.8}$$

An optimal configuration $k_{opt}$ minimizing the consensus (1.1) with connection strength (3.8) necessarily and sufficiently minimizes

$$\sum_{i=1}^{n} \left(k(u_i) - m(k)\right)^2 \tag{3.9}$$

over all configurations *k* where $m(k) = \frac{1}{n}\sum_{i=1}^{n} k(u_i)$ .

To generalize the results in section 2, a few preliminaries are necessary. Let $N_n = \begin{pmatrix} 1 & \Lambda & 1 \\ \mathbf{M} & \mathbf{O} & \mathbf{M} \\ 1 & \Lambda & 1 \end{pmatrix}$ be the *n*-by-*n* matrix whose entries are all *1*'s. For instance $N_{12}$ is a *12*-by-*12* matrix. One says that an *n*-by-*n* matrix, *Q*, is *doubly stochastic* if the entries of *Q* are nonnegative, and the sum of the entries in each row and column is *1*. In the previous section, the neighborhood structure in (2.6) when thought of as a linear transformation can be represented by the matrix $N_d = \frac{1}{10}\left(N_{12} - 2Q\right)$ where *Q* is the doubly stochastic matrix given by

$$\begin{pmatrix} .5 & .5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ .5 & .5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & .5 & .5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & .5 & .5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & .5 & .5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & .5. & .5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & .5 & .5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & .5 & .5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & .5 & .5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & .5 & .5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & .5 & .5 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & .5 & .5 \end{pmatrix} \tag{3.10}$$

To enlarge the neighborhood structure in section 2, let

$$N(k) = \left\{ \frac{1}{n-c}(N_n - cQ) : Q \text{ is a doubly stochastic matrix and } 0 \le c < \frac{n}{2} \right\} \tag{3.11}$$

Once again in utilizing the simulated annealing algorithm, we need to define a mechanism for generating a neighbor *l* in *N(k)* of *k*. If $\{E_r : r = 1, \ldots, n!\}$ is the set of all *n*-by-*n* permutation matrices, then each doubly stochastic matrix *Q* can be expressed in the form

$$Q = \sum_{r=1}^{n!} \alpha_r E_r \tag{3.12}$$

where $\sum_{r=1}^{n!} \alpha_r = 1$, $\alpha_r \ge 0$ [3, Theorem 8.7.1]. This implies that given $c < \frac{n}{2}$, we can stochastically generate a double stochastic matrix *Q* by stochastically choosing an *n!*-tuple $(\alpha_r)$ and then constructing the matrix in (3.12). Furthermore, to implement the simulated annealing algorithm we shall have to define a transition probability. For one, let $P_{k,l}(m) = \alpha_1$ where $l \in N(k)$ is given by $l = \frac{1}{n-c}\left(N_n - c\sum_{r=1}^{n!} \alpha_r E_r\right)$

Suppose *k* is an initial configuration of a Boltzman machine *(U, C )*. By realizing k as a column vector, then the configuration in the second trial is $\frac{1}{n-c_1}(N_n - c_1 Q_1)k$ and the configuration

in the third trial is $\left(\frac{1}{n-c_2}\left(N_n - c_2 Q_2\right)\right)\left(\frac{1}{n-c_1}\left(N_n - c_1 Q_1\right)\right)k$ for some $0 \le c_1, c_2 < \frac{n}{2}$ and doubly stochastic matrices *Q_1, Q_2*. In [4], provided each $c_i$ satisfies $0 \le c_i < \frac{n}{2} - \varepsilon$ for some $\varepsilon > 0$, we have shown that after several trials the *m*th configuration shall be approximately $\frac{1}{n}N_n(k)$, i.e., $\lim_{m\to\infty}\prod_{i=1}^{m}\frac{1}{n-c_i}(N_n - c_i Q_i)(k) = \frac{1}{n}N_n(k)$. As a special case, when *n=12*, $c_i = 2$, and each $Q_i$ is the matrix in (3.10) for all *i*, we obtain the simulated annealing of the dodecahedron in section 2; that is, $\lim_{m\to\infty}\prod_{i=1}^{m}\frac{1}{n-c_i}(N_n - c_i Q_i)(k) = \lim_{m\to\infty} N_d^m(k) = \frac{1}{12}I_{12}$.

## 4. Classification: Identifying Digits

A class of optimization problems that can be easily solved by human beings but very difficult for computers are the so-called classification problems - these are problems on associating objects with subsets. Classification problems have their origins in pattern recognition. A specific pattern recognition problem we will solve with combinatorial optimization and Boltzman machines is *the seven-segment display problem*. This particular problem is extensively discussed in [1,Section 10.3].

The display of the digits 0,1,2,3,...,9 often uses a seven-segment display. (Imagine two equal squares where one sits on top of the other square). Each of the segments can be independently assigned 0 (for `off') or 1 (for `on'). We will choose a Boltzman machine which will identify any digit displayed. It is possible that the figure shown in the seven-segment display is not a number but we still would like the Boltzmann machine to assign a digit to the seven-segment display.

We will consider a neural network whose set of units, C, is the union of $U_i = \{u_1, \ldots, u_7\}$, the input units, and $U_o = \{v_1, \ldots, v_{10}\}$, the output units. The state of a unit is either 0 or 1. In particular, a configuration is a *17*-tuple consisting of *0*'s and *1*'s. For each of the ten digits, we assign a configuration according to the table below.

| digit | u1 | u2 | u3 | u4 | u5 | u6 | u7 | v1 | v2 | v3 | v4 | v5 | v6 | v7 | v8 | v9 | v10 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 4 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 5 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |

| digit | u1 | u2 | u3 | u4 | u5 | u6 | u7 | v1 | v2 | v3 | v4 | v5 | v6 | v7 | v8 | v9 | v10 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 6 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 7 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 8 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 9 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

We will call the given set of configurations above as the classification set $V'$. If each of the seven input units are assigned a state, we would like the Boltzmann machine to maximize its overall desirability by assigning states to the remaining units, the outputs units. Observe that each of the configurations in $V'$ assigns the state 1 to exactly one output unit. For us, an acceptable Boltzmann machine will assign exactly one digit to a given configuration of the input units.

For the set of connections, we have the union of $C_{i,o} = \left\{\left\{u_i, v_j\right\} : 1 \le i \le 7, 1 \le j \le 10\right\}$ and $C_{o,o} = \left\{\left\{v_i, v_j\right\} : 1 \le i < j \le 10\right\}$. Note, no connection exists between input units and there are no bias connections. Following [1, Section 10.3], a connection $\left\{u_i, v_j\right\} \in C_{i,o}$ is said to be excitatory if there exist $k \in V'$ such that $k(u_i)k(v_j) = 1$; otherwise, $\left\{u_i, v_j\right\}$ is said to be inhibitory. To each $v_j \in U_o$, let $N^+_{v_j} = \left\{u_i : \left\{u_i, v_j\right\} \textit{ is excitatory}\right\}$ and let $N^-_{v_j} = \left\{u_i : \left\{u_i, v_j\right\} \textit{ is inhibitory}\right\}$.

For the connection strength *s*, we first choose a positive constant $\gamma$ and let $\delta = \underset{v_j \in C_{o,o}}{Min}\left\{\frac{\gamma}{\left|N^+_{v_j}\right|}\right\}$.

Note, $-\gamma + \delta < 0$. For each $\left\{u_i, v_j\right\} \in C_{i,o}$, let

$$s\left\{u_i, v_j\right\} = \begin{cases} \dfrac{\gamma}{\left|N^+_{v_j}\right|} & \textit{if} \quad u_i \in N^+_{v_j} \\ -\dfrac{\gamma}{\left|N^-_{v_j}\right|} & \textit{if} \quad u_i \in N^-_{v_j} \end{cases} \tag{4.15}$$

and if $\left\{v_i, v_j\right\} \in C_{o,o}$, let $s\left\{v_i, v_j\right\}$ be any negative constant number satisfying $-\gamma < s\left\{v_i, v_j\right\} < -\gamma + \delta$.

As shown in [1, page 186], if the given set of states of the input units represents one of the digits *0,1,2,3,...,9* then the optimal configuration maximizing the consensus function (obtained by choosing the optimal configuration of the remaining units, i.e., of the output units) represents and identifies the correct input digit.

## 5. Summary

In this paper, we have simulated annealing processes and solved a pattern recognition problem by using the simulated annealing algorithm and Boltzman machines. These are two types of fields, combinatorial optimization and pattern recognition (a part of artificial intelligence) amongst others, where Boltzman machines can be used.

We have interpreted the simulated annealing processes in sections 2 and 3 as minimization problems; in particular the minimization of the consensus function over all configurations of an appropriate Boltzman machine. The states of the units represented the temperatures of the units. The simulated annealing algorithm has close connections with statistical mechanics. In the annealing process, one can interpret the states (respectively, free energy) of the solid in the cooling process as configurations (respectively, consensus function) of a Boltzman machine [1,2].

Pattern recognition, briefly, finds a correct output for a given input. As a combinatorial optimization problem, the seven-segment display problem is construed as a constrained optimization problem. An appropriate Boltzman machine with a set of units which can be separated into two subsets, namely, the subset of input units and the subset of output units. Giving an input is equivalent to giving a configuration of the input units. Once an input is given, the objective then is to find a configuration of the remaining units (which are the output units) which maximizes the consensus function of the Boltzman machine.

### 5. References

1. AARTS, E. and KORST, J., *Simulated Annealing and Boltzman Machines*, Wiley-Interscience Series in Discrete Mathematics and Optimization, John Wiley and Sons, 1989.
2. BONOMI, E. and LUTTON, J.-L., *The N-City Travelling Salesman Problem: Statistical Mechanics and the Metropolis Algorithm*, Siam Review 26, No. 4, October 1984.
3. HORN, R.A. and JOHNSON, C.R., *Matrix Analysis*, Cambridge University Press, 1990.
4. MERINO, D.I., REYES, E.N., and STEIDLEY, C., Using Matrix Analysis to Approach the Simulated Annealing Algorithm, accepted for publication in Computers in Education Journal of the American Society for Engineering Education
5. REYES, E.N. and STEIDLEY, C., *A GAP Approach to the Simulated Annealing Algorithm*, Computers in Education Journal of the American Society for Engineering Education, Vol. 7 (1997), No. 3, 43-47.
6. REYES, E.N. and STEIDLEY, C., A Theoretical Discussion of the Mathematics Underlying ``A GAP Approach to the Simulated Annealing Algorithm", Computers in Education Journal of the American Society for Engineering Education, Vol. 7 (1997), No. 4, 50-57.
7. SCHONERT, M et al., *GAP-Groups, Algorithms, and Programming*, Version 3 Release 4, Lehrstuhl D fur Mathematik, RWTH, Aachen, Germany, 1994.