

# Simulated Annealing Articles summaries

William Bezuidenhout

Tuesday, 25 May 2010

## **An Unsupervised Intrusion Detection Method Combined Clustering with Chaos Simulated Annealing**

```
@INPROCEEDINGS{4370702,  
  author={Lin Ni and Hong-Ying Zheng},  
  booktitle={Machine Learning and Cybernetics, 2007 International Conference on},  
  title={An Unsupervised Intrusion Detection Method Combined Clustering with Chaos  
  Simulated Annealing},  
  year={2007},  
  month={19-22},  
  volume={6},  
  number={},  
  pages={3217 -3222},  
  keywords={chaos simulated annealing algorithm;computer networks security;  
  hardware failures;malicious attacks; near-optimal partitioning clustering;  
  optimization technique;software flaws;tentative probing; unsupervised  
  clustering intrusion detection method;chaos;computer networks;pattern  
  clustering; security of data;simulated annealing;telecommunication security;},  
  doi={10.1109/ICMLC.2007.4370702},  
  ISSN={},}
```

Simulated Annealing is a powerful optimization technique that attempts to find a global minimum of a function using concepts borrowed from Statistical Mechanics. The algorithm was originally intended for simulating the evolution of a solid in a heat bath to thermal equilibrium. As it was first described, the algorithm starts with a “substance” composed of many interacting individual molecules arranged in a random fashion. Then, small random perturbations to the structure of the molecules are attempted, and each perturbation is accepted with a probability based on the associated

“energy” increase,  $\Delta E$ . If  $\Delta E$  is at least 0, then the perturbation is accepted with probability  $\exp(\frac{\Delta E}{T})$ . If  $\Delta E$  is less than 0, then the perturbation is accepted with probability 1. Eventually, after a large number of trial perturbations, the energy settles to equilibrium for the temperature. SA uses both the high- and the low-temperature properties of the Metropolis algorithm to find low energy, regardless of the initial structure. The cooling is made slow to overcome the high dependence of low temperature equilibrium energies on the initial state. Simulated annealing exploits the obvious analogy between process annealing and combinatorial optimization problems, where the “molecules” are the variables in the data structure and the “energy” function is the objective function.

## A Simulated Annealing Algorithm with Constant Temperature for Discrete Stochastic Optimization

A Simulated Annealing Algorithm with Constant Temperature for Discrete Stochastic Optimization

Author(s): Mahmoud H. Alrefaei and Sigrn Andradttir

Source: Management Science, Vol. 45, No. 5 (May, 1999), pp. 748-764

Published by: INFORMS

Stable URL: <http://www.jstor.org/stable/2634729>

Accessed: 09/03/2010 05:10

Kirkpatrick et. al (1983) and Cerny (1985) introduced the idea of simulated annealing to the field of combinatorial optimization. Their procedure resembles the model of Metropolis et al. (1953) for how in the (physical) annealing process, particles of a solid arrange themselves into thermal equilibrium at a given temperature. More specifically, consider the optimization problem (1). Suppose that exact objective function values are available and that the objective function may have multiple local optimal solutions. Simulated Annealing allows “hill-climbing” moves in order to avoid local optimal solutions. This method starts with an arbitrary state in  $\varphi$  is the estimated optimal solution. If  $x \in \varphi$  is the estimated optimal solution in iteration  $k$ , then a new state  $x'$  is randomly selected from a neighborhood of  $x$ ,  $N(x)$ . if  $f(x') \leq f(x)$ , so that  $x'$  is better state than  $x$ , then the method accepts the move from  $x$  to  $x'$ , and  $x'$  becomes the new estimated optimal solution. On the other hand, if  $f(x') > f(x)$ , then there is a chance that  $x'$  will be accepted even though it is a worse state than  $x$ ; this is because good points might be “hidden behind”  $x'$ . More specifically, a uniform random number

$U_k \sim U[0, 1]$  is generated (i.e,  $U_k$  is uniformly distributed on the interval  $[0, 1]$ ). If  $U_k \leq \exp[-[f(x') - f(x)]^+/T_k]$ , where  $T_k > 0$  is a controlling parameter called “temperature” and  $y^+ = \max\{0, y\}$  for all  $y \in \mathbb{R}$ , the state  $x'$  is accepted as the new estimate of the optimal solution; otherwise  $x$  remains the estimate of the optimal solution. Note that the higher the temperature  $T_k$ , the more likely it is that a “hill-climbing” move from  $x$  to  $x'$  with  $f(x') > f(x)$  is accepted. The simulated annealing algorithm assumes  $T_k > 0$  for all  $k$  and that  $T_k \rightarrow 0$  as  $k \rightarrow \infty$ . The initial temperature  $T_0$  and the rate at which the sequence  $\{T_k\}$  (the annealing schedule) is decreased play important roles in the convergence of the simulated annealing algorithm and the quality of the final estimate of the optimal solution.

Most authors studying simulated annealing have focused on determining an appropriate annealing schedule in order to obtain convergence in probability to the set of global optimal solutions  $\varphi^*$ . Hajek (1988), Theorem 1, shows that for annealing schedules  $\{T_k\}$  of the form

$$T_k = \frac{C}{\ln(1+k)}, \forall k \in \mathbb{N} \quad (1)$$

a necessary and sufficient condition on  $C$  for the algorithm to converge in probability to the set  $\varphi^*$  is that  $C \geq d^*$ , where  $d^*$  is the maximum depth of the local optimal solutions that are not globally optimal. Earlier results on the convergence of simulated annealing are given by Geman and Geman (1984) and by Mitra et al. (1986).

The original simulated annealing algorithm assumes that the objective function values can be evaluated exactly. However, in my practical, it is impossible to evaluate the objective function exactly, and instead the valuation of the objective function values may include some noise. This is, for example, the case when values of the objective function are the expected performance of a complex stochastic system under different system configurations and are estimated using simulation. Despite the wide use of simulated annealing procedure can be viewed as an ordinary hill-descent method with artificial noise added.

## **A simulated annealing approach for curve fitting in automated manufacturing systems**

Title: A simulated annealing approach for curve fitting in automated manufacturing

Author(s): Hsien-Yu Tseng, Chang-Ching Lin

Journal: Journal of Manufacturing Technology Management

SA is a stochastic search technique. It is designed to lead jumping out local optimum during the search process. SA is a very effective combinatorial optimization method, which is successfully applied to VLSI design, schedule, plant layout and etc. and the combinatorial optimization problem related to production while extending its application to the continuous variable optimization problem.

Similar to statics mechanism, the search process of SA is also operated in accordance with transition probability. This transition probability depends on control temperature and the changing amount of objective function. Since, SA possesses stochastic search policy, it can be “uphill” move by using the solution of a larger objective function as the present solution. Moving the present solution to an inferior solution under a controlled probability may allow SA jump out local optimum and may be a better “downhill” path will be found, which further obtains a better solution. The “uphill” move is controlled carefully by the temperature. When temperature is too high, “uphill” move probability will increase; when temperature decreases gradually, “uphill” move probability will decrease accordingly. Kirkpatrick demonstrated that SA is capable of obtaining the real optimum under a very long search time. In actual execution, this real optimum is not obtainable due to the restriction of calculation time, but an approximation of the real optimum can be obtained.

When SA is applied to one problem, four basic components should be defined. These are:

- *Configuration* — represents the possible solution of problem.
- *Move set* — Is an allowable move, which can let us achieve all feasible configuration, this set have to be easy for calculation.
- *Cost Function* — Is used to measure the quality of configuration
- *Cooling schedule* — Sets the initial temperature and the cooling regulation to determine when and how many degrees to decrease the present temperature, and when to end the annealing.

The SA method that solves the continuous function problem proposed by Corana et al. (1987) faces a tough problem that needs long copious calculation. The curve fitting optimization model mentioned in the previous

section is immense and complicated. The method mentioned by Corana et al. is not suitable to apply to the optimization model solving. So to decrease the SA calculation demand, the PSSA optimization method is proposed in this current paper uses PS as the move generating function to accelerate the search process.

### 0.0.1 Patter search

The PSSA optimization method recommended in this paper uses Pattern Search(PS) as move generation mechanism. The operation of PS includes two kinds of move: exploratory move and pattern move. Exploratory move examines the local behaviour of function, and finds out the direction of “down-hill” path. Pattern move utilizes the information generated by exploratory move to reach the valley rapidly.

Exploratory move starts search from initial point and moves along, the axial direction of each variable according to step size, and decides whether neighboring move is set as present solution in accordance with objective function. If setting neighboring solution as present solution is unacceptable (no improvement in objective function), it will search the opposite direction. During exploratory move, each move only refers to one variable, and explores each variable in proper order to check whether there is improvement in the objective function. If there is an improvement, it means that a direction “downhill” path (or pattern direction) exists. After an exploratory move ends, it will check whether a pattern direction exists. If it does exist, make the pattern move according to this pattern direction and this move can rapidly reach the minimum of the function (the valley) and continue executing the pattern move until no objective function improves anymore.

Changing step size or not depends on pattern move existence after each exploratory move ends. If a pattern direction does not exist, then change the step size. PS uses exploratory move and pattern move repeatedly until the result condition is satisfied.

There are many cooling schedules discussed in the literature, and geometric cooling schedule is very fast and very effective

## A Survey of Simulated Annealing as a Tool for Single and Multiobjective Optimization

@article{2006,  
jstor\_articletype = {primary\_article},

```

title = {A Survey of Simulated Annealing as a Tool for Single and Multiobject
author = {Suman, B. and Kumar, P.},
journal = {The Journal of the Operational Research Society},
jstor_issuetitle = {},
volume = {57},
number = {10},
jstor_formatteddate = {Oct., 2006},
pages = {1143--1160},
url = {http://www.jstor.org/stable/4102365},
ISSN = {01605682},
language = {},
year = {2006},
publisher = {Palgrave Macmillan Journals on behalf of the Operational Research
copyright = {Copyright 2006 Operational Research Society},
}

```

Simulated Annealing (SA) is a compact and robust technique, which provides excellent solutions to single and multiple objective optimization problems with a substantial reduction in computation time. It is a method to obtain an optimal solution of a single objective optimization problem and to obtain a Pareto set of solutions for a multi-objective optimization problem. It is based on an analogy of thermodynamics with the way metals cool and anneal. If a liquid metal is cooled slowly, its atoms form a pure crystal corresponding to the state of minimum energy for the metal. The metal reaches a state with higher energy if it is cooled quickly. SA has received significant attention in the last two decades to solve optimization problems, where a desired global minimum/maximum is hidden among many poorer local minima/maxima. Kirkpatrick et al (1983) and Cerny (1985) showed that a model for simulating the annealing of solids, proposed by Metropolis et al (1953), could be used for optimization problems, where the objective function to be minimized corresponds to the energy of states of the metal. These days SA has become one of the many heuristic approaches designed to give a good, not necessarily optimal solution. It is simple to formulate and it can handle mixed discrete and continuous problem with ease. It is also efficient and has low memory requirement. SA takes less CPU time than genetic algorithm (GA) when used to solve optimization problems, because it finds the optimal solution using point-by-point iteration rather than a search over a population of individuals.

Initially, SA has been used with combinatorial optimization problems. Maffioli (1987) showed that SA can be considered as one type of randomized

heuristic approaches for combinatorial optimization problems.

In SA, instead of this strategy, the algorithm attempts to avoid being trapped in a local minimum by sometimes accepting even the worse move. The acceptance and rejection of the worse move is controlled by a probability function. The probability of accepting a move, which causes an increase  $\delta$  in  $f$ , is called the acceptance function. It is normally set to  $\exp(\frac{-\delta}{T})$ , where  $T$  is a control parameter, which corresponds to the temperature in analogy with the physical annealing. The acceptance function implies that the small increase in  $f$  is more likely to be accepted than a large increase in  $f$ . When  $T$  is high most uphill moves will be rejected. Therefore, SA starts with a high temperature to avoid being trapped in local minimum. The algorithm proceeds by attempting a certain number of moves at each temperature and decreasing the temperature.

Like other heuristic optimization techniques, there is a chance of revisiting a solution multiple times in SA as well. It leads to extra computational time without any improvement in the optimal solution. Mingjun and Huanwen (2004) have proposed chaos simulated annealing (CSA) by introducing chaotic systems to SA. The CSA is different from SA as chaotic initialization and chaotic sequences replace the Gaussian distribution. CSA is more likely to converge to the global optimum solution because it is random, stochastic and sensitive to the initial conditions. It has been shown that CSA improves the convergence and is efficient, applicable and easy to implement.

### 0.0.2 Annealing Schedule

The setting of the parameters for the SA-based algorithm determines the generation of the new solution. The precise rate of cooling is an essential part of SA as it determines the performance of the SA-based algorithm. A high cooling rate leads to poor results because of lack of the representative states, while a low cooling rate requires high computation time to get the result. The following choices must be made for any implementation of SA and they constitute the annealing schedule: initial value of temperature ( $T$ ), cooling schedule, number of iterations to be performed at each temperature and stopping criterion to terminate the algorithm.

**Initial value of temperature ( $T$ )** Initial temperature is chosen such that it can capture the entire solution space. One choice is a very high initial temperature as it increases the solution space. However, at a high initial temperature, SA performs a large number of iterations, which may be without giving better results. Therefore, the initial temperature is chosen by

experimentation depending upon the nature of the problem. van Laarhoven et al (1988) have proposed a method to select the initial temperature based on the initial acceptance ratio  $\chi_0$ , and the average increase in the objective function,  $\Delta f_0 : T = -\frac{\Delta f_0}{\ln(\chi_0)}$  where  $\chi_0$  is defined as the number of accepted bad moves divided by the number of attempted bad moves. A similar formula has been proposed by Sait and Youssef (1999) (with the only difference being the definition of  $\chi_0$ ). They have defined  $\chi_0$  as the number of accepted moves divided by the number of attempted moves. A simple way of selecting initial temperature has been proposed by Kouvelis and Chiang (1992). They have proposed to select the initial temperature by the formula  $P = \exp(-\frac{\Delta s}{T})$  where  $P$  is the initial average probability of acceptance and is taken in the range of 0.50-0.95.

**Cooling Schedule** Cooling schedule determines functional form of the change in temperature required in SA. The earliest annealing schedules have been based on the analogy with physical annealing. Therefore, they set initial temperature high enough to accept all transitions, which mean heating up substance till all the molecules are randomly arranged in liquid. A proportional temperature is used, that is  $T(i+1) = \alpha T(i)$ , where  $\alpha$  is a constant known as the cooling factor and it varies from 0.80 to 0.90. Finally, temperature becomes very small and it does not search any smaller energy. It is called the frozen state. Three important cooling schedules are logarithmic, Cauchy and exponential. SA converges to the global minimum of the cost function if temperature change is governed by a logarithmic schedule in which the  $T(i)$  at step  $i$  is given by  $T(i) = \frac{T_0}{\log i}$ . This schedule requires the move to be drawn from a Gaussian distribution. A faster schedule is the Cauchy schedule in which  $T(i) = \frac{T_0}{i}$  converges to the global minimum when moves are drawn from a Cauchy distribution. It is sometimes called ‘fast simulated annealing’. The fastest is exponential or geometric schedule in which  $T(i) = T_0 \exp(-C_i)$  where  $C$  is a constant. There is no rigorous proof of the convergence of this schedule to the global optimum, although good heuristic arguments for its convergence have been made for a system in which annealing state variables are bounded. The logarithmic schedule is the best cooling schedule because it ensures convergence towards the set of optima with probability one.

### 0.0.3 Other detail

It has been shown that Tabu and Evolutionary Algorithms outperform SA. The SA algorithm for single objective problems sometimes accepts solutions,



which are worse than the current solution. Therefore, it is possible that the final solution can be worse than the best solution. It is suggested to store the best solution to improve the performance of SA.

A very important way of improving a heuristic technique is to mix two algorithms. Attempts have been made to improve the performance of SA by combining it with another algorithm. There are two ways of using SA with another algorithm. In the first way, a good initial guess is provided to the SA algorithm which is improved by SA. In the second way, SA is implemented by using a parallel version of the algorithm. SA is difficult to parallelize due to its intrinsic serial nature and direct parallelization of the Metropolis algorithm.

Although the SA method can find the optimal solution to most single and multi-objective optimization problems, the algorithm always requires numerous numerical iterations to yield a good solution.

Annealing schedule has been a topic of research in SA since it has been used as an optimization technique. It is used at the probability step in SA and therefore, it governs the move. A wise choice of annealing schedule can save computational time and can improve the quality of solution. Attention is needed to select the initial temperature and to decide the number of iterations to be performed at each temperature.

The search space in SA is explored one by one and in a sequential manner. Therefore, there is a chance of revisiting a solution multiple times, which leads to extra computation time without improving the quality of the solution. Recently, researchers have attempted to use tabu search with SA to avoid revisiting. However, avoidance of revisiting a solution by a tabu search depends on the size of the tabu list. A large tabu list is also computationally intensive. Additionally, it has been used only in single objective optimization problems. SA with tabu search should be more effective in multi-objective problems since the management of a Pareto set is computationally intensive.

SA generates the new solution vector randomly; it is inefficient for search for the optimal solution of large parameter optimization problems.

## Clustering based simulated annealing for standard cell placement

```
@INPROCEEDINGS{14776,
author={Mallela, S. and Grover, L.K.},
booktitle={Design Automation Conference, 1988. Proceedings., 25th ACM/IEEE},
title={Clustering based simulated annealing for standard cell placement },
```

```

year={1988},
month={12-15},
volume={},
number={},
pages={312 -317},
keywords={CPU time;LTX2 VLSI layout system;clustering based simulated annealing;in
doi={10.1109/DAC.1988.14776},
ISSN={},}

```

Simulated Annealing is a general purpose combinatorial optimisation technique to determine the global minimum of an objective function, and is based on the annealing phenomenon in crystallization. Its characteristic feature is the ability to explore the configuration space via configurations that actually increase the cost function being minimized. A parameter called *temperature* is defined, which is of the same dimension as the cost function. A set of *moves* is selected, with which one state of the configuration space can be generated from another. Moves that reduce the cost are called *downhill* moves, and those that increase the cost are called uphill moves, and those that increase the cost are called uphill moves. The Metropolis algorithm from statistical physics is used to simulate the system at a given temperature. The system is simulated starting from a high temperature, and the temperature is gradually lowered. Moves are generated at random; all downhill moves are accepted, and uphill moves are accepted with a probability of  $\exp(-\frac{\Delta E}{T})$ , where  $\Delta E$  is the increase in cost and  $T$  is the temperature. At very high temperatures almost all the moves are accepted, and the system moves about freely between different areas of the configuration space. At very low temperatures it accepts only downhill moves, and behaves like a greedy algorithm. It has been proven that simulated annealing will asymptotically produce the global minimum, if certain conditions are satisfied on the moves generated at each temperature.

In this paper we present a novel technique to reduce the computation time by reducing the size of the problem that simulated annealing has to handle. We do this by splitting the simulated annealing process into two stages. We form *clusters* of cells based on the strength of their interconnections, and place them first using the conventional simulated annealing. Then we break up the clusters, and refine the placement by another stage of simulated annealing at the cell level, but by starting at a lower temperature and greatly limiting the range of cell moves. The original problem is thus divided into two smaller problems, each with much less computational requirement than the original problem.

While reducing the computation time is our principal goal, we want to ensure that the cost of the final placement is no greater than what can be achieved with the conventional simulated annealing. In the final placement, it is therefore important not to place any constraints on the relative locations of cells that were in the same cluster during the top-down placement phase. An inexpensive version of the simulated annealing technique is used to achieve this goal and determine the locations of the cells. Thus, for standard cell placement we use 2-stage simulated annealing process.

In the first stage the conventional simulated annealing algorithm is used to place the various clusters into rows. Each cluster is a composite of the individual cells it contains, and has a width equal to the sum of the individual cell widths. We do not attempt to determine the precise locations of the cells within a cluster. So all the terminals of the cluster are assumed to be at the center of the cluster. Thus the simulated annealing program tries to solve a problem with fewer but larger entities, and fewer nets – only those between clusters. The temperature scheduling and the move-selection function are left unchanged from the conventional version. The aim is to still start with a high value for the temperature and gradually decrease it to the freezing point. At the end of this stage the cells are placed within a small neighborhood of their "best" locations.

In the second stage, we determine the precise location for each cell. First, we break up all the clusters and convert the problem back to the one involving the placement of individual cells. Then we use an inexpensive simulated annealing technique, which limits the range of cell moves to a small neighborhood of their location, and anneals quickly from low temperature. The number of moves per cell is also less than in the conventional process. The effect is to permit some amount of hill-climbing, while ensuring that the final configuration does not stray too far from the starting state. After this stage we obtain the final placement, which is a refinement of the placement obtained from the first stage.

*In article "Global Planning of 3G Networks using Simulated Annealing" nice flow chart of SA algorithm*

## Simulated annealing, its parameter settings and the longest common subsequence problem

```
@inproceedings{1389253,
  author = {Weyland, Dennis},
  title = {Simulated annealing, its parameter settings and the longest common subsequence problem}}
```

```

booktitle = {GECCO '08: Proceedings of the 10th annual conference on Genetic and
computation},
year = {2008},
isbn = {978-1-60558-130-9},
pages = {803--810},
location = {Atlanta, GA, USA},
doi = {http://doi.acm.org/10.1145/1389095.1389253},
publisher = {ACM},
address = {New York, NY, USA},
}

```

The algorithm we investigate is a kind of standard Simulated Annealing Algorithm operating on fixed length bit strings. This algorithm has, besides the fitness function and the input length, two different parameters, namely the initial temperature and the cooldown factor. These two parameters do not change during the execution of the algorithm.

At the beginning the temperature is set to the value of the initial temperature and an initial solution is created uniformly at random. In each iteration the current solution is modified by mean of 1-bit mutation. If the fitness value of the new solution is better or equal than the fitness value of the current solution, the new solution will replace the current solution for the next iteration, otherwise the new solution will replace the current solution only with a probability of  $\exp(-\frac{\Delta f}{T})$ , where  $\Delta f$  is the (positive) difference of the fitness values and  $T$  is the current temperature. If the current solution after that step fulfills the stopping criterion, the algorithm stops, otherwise the temperature will be updated by multiplying the current temperature with the cooldown factor and the algorithm continues with the next iteration.

For theoretical analyses these general suggestions are not very helpful, since the probability of accepting a worse solution is independent of the input size and so it decrease too fast below an exponentially small value. That means, the number of iterations to leave a local optimum is too small with constant values for the initial temperature and the cooldown factor.

A low initial temperature and small colldown factor lead to a small time window for leaving local optima, high initial temperature and a cooldown factor very near to 1 leavd to a huge time window for leaving local optima, On the other hand, once the basin of attraction of a flobal optima is reac-head, a low initial temperature and a small cooldown factor lead to a faster convergence to this optimum in general, whereas an algorithm with a high iniital temperature and a cooldown factor near to 1 needs more iterations

to reach that optimum. So there is a kind of tradeoff between the number of iterations, within local optima can be left, and the number of iterations the algorithm needs to reach a global optimum once there is a solution in its basin of attraction.

## Comparative Study of Various Cooling Schedules for Location Area Planning in Cellular Networks Using Simulated Annealing

```
@ARTICLE{10.1109/NetCoM.2009.60,
  author = {Nilesh B. Prajapati and Rupal R. Agravat and Mosin I. Hasan},
  title = {Comparative Study of Various Cooling Schedules for Location Area
Planning in Cellular Networks Using Simulated Annealing},
  journal = {Networks \& Communications, International Conference on},
  year = {2009},
  volume = {0},
  pages = {146-150},
  address = {Los Alamitos, CA, USA},
  doi = {http://doi.ieeecomputersociety.org/10.1109/NetCoM.2009.60},
  isbn = {978-0-7695-3924-9},
  publisher = {IEEE Computer Society}
}
```

Simulated Annealing (SA) is a random-search technique which exploits an analogy between the way in which a metal cools and freezes into a minimum energy crystalline structure (the annealing process) and the search for a minimum in a more general system; it forms the basis of an optimization technique for combinatorial and other problem. In 1983 SA was developed for dealing with highly nonlinear problems. An important characteristic of the SA algorithm is that it does not require specialist knowledge about how to solve a particular problem. This makes the algorithm generic in the sense that it can be used in a variety of optimization problem without change the basic structure of the computation. There are two important criteria for SA: Selection of Neighbor and Cooling Schemes. Accuracy and optimization of SA depends on these two criteria.

In Neighboring Solution, collection of all the optimal and non-optimal solutions is called Solution Space. During annealing process, algorithm randomly selects any solution from these Solution space.

## 0.1 Cooling Schedule

A cooling schedule consists of three important components: setting the starting (initial) temperature, the decision of when and how to decrease the temperature, and the decision of when to stop the algorithm.

Initial temperature calculation is very important, because if it is too high, then it may take too long to reach the result. And if the initial temperature is too small, then the algorithm does not have the freedom of making sufficient number of random moves to visit different local minima and stay in relatively deep one.

## 0.2 Conclusion

SA is one of the best heuristic methods for finding optimal or near-optimal solution in cellular network in tolerable run-times. SA gives the better LA planning in cellular network so total location management cost is reduced. The main aim of the experiment of better LA planning is fulfilled.

## Adaptive Simulated Annealing for maximum temperature

```
@INPROCEEDINGS{1243786,  
  author={Miki, M. and Hiroyasu, T. and Jitta, T.},  
  booktitle={Systems, Man and Cybernetics, 2003. IEEE International Conference on},  
  title={Adaptive Simulated Annealing for maximum temperature},  
  year={2003},  
  month={5-8},  
  volume={1},  
  number={},  
  pages={ 20 - 25 vol.1},  
  keywords={ acceptance probability; adaptive simulated annealing; heating process;  
  doi={10.1109/ICSMC.2003.1243786},  
  ISSN={1062-922X},}
```

It was Kirkpatrick et. al. who first proposed simulated annealing, SA, as a method for solving combinatorial problems. It is reported that Sa is very useful for several types of combinatorial optimization problems. However, the most remarkable disadvantages are that it needs a lot of time to find the optimum solution and it is very difficult to determine the proper cooling schedule.

For continuous optimization problems, reduced computation time can be realized by using sophisticated generating functions such as the very fast simulated reannealing, but we can not use such approaches for discrete optimization problems. For discrete optimization, the generation of a new solution is determined by defining the operation for changing the current solution. Therefore, we can control only the temperature schedule.

The most appropriate temperature schedule has the following characteristics: sufficiently high maximum temperature, sufficiently low minimum temperature, and sufficiently slow cooling. However, this type of temperature schedule has one drawback, that is, a long computation time. There are three approaches to reduce the computation time. They are: 1) to decrease the maximum temperature, 2) to increase the minimum temperature, 3) to increase the cooling rate. Among those, the minimum temperature is generally determined by the acceptance ratio during the SA process, that is, the temperature is decreased until the system is ‘frozen’. Therefore, the first and last approaches are promising, Klebsch et al. proposed a new method for estimating the maximum temperature by using equilibrium dynamics, and Romeo et al. proposed an efficient cooling method but these methods use experimental parameters and some tuning of these parameters is necessary.

The most important characteristic of SA is the probabilistic acceptance of a bad solution. The probability of acceptance of a newly generated,  $P_{AC}$ , is as follows.

$$P_{AC} = \begin{cases} 1, & \text{if } \Delta E < 0 \\ \exp(-\frac{\Delta E}{T}), & \text{otherwise} \end{cases} \quad (2)$$

This type of criterion is called the Metropolis’s Criterion, and  $\Delta E$  is the change in the energy,  $T$  is the temperature.

In SA, a particular temperature region that can be performed very effectively exists. It is called the important temperature region in this paper. In order to find the important temperature region, some traveling salesman problems (TSPs) are solved by a SA with constant temperature, where the neighbor is generated by using the 2-change method.

In SA, the maximum temperature should be sufficiently high in order to escape local optima, but an excessively high maximum temperature wastes computational time. The relation between the important temperature region and the maximum temperature is investigated for some TSPs.

It is clearly found that the maximum temperature that is higher than the important temperature region provides good solutions, and the maximum

temperature lower than the important temperature does not. Therefore, it can be concluded that the effective maximum temperature must be higher than the important temperature region and the lowest maximum temperature should be the upper bound of the important temperature region. If we can find this lowest maximum temperature before or at the beginning of searching an optimum, the number of annealing steps can be reduced to roughly half. That is, the maximum temperature determined by a conventional method is excessively high, and massive computation is consumed ineffectively.

### 0.3 SA with an adaptive maximum

From the above result, the lowest maximum temperature is found to be the upper bound of the important temperature. However, each problem has its own important temperature regions, and a lot of experiments have to be performed to find them. Therefore, we have to find a characteristic movement of the solutions during the annealing. The histories of the solutions to some typical TSPs are investigated here. The temperature is conventionally determined as follows.

We conduct an experiment on SA with reversed temperature schedule, that is, heating, not cooling. At the beginning, the temperature is very low and hill-climbing local search is performed, and a local minimum is obtained. And, as the temperature increases the solution remains at the local minimum, but the solution escapes the local minimum and it becomes better as the temperature enters the important temperature region. The solution becomes worse as the temperature increases beyond the important temperature region.

The proposed method, which is called the Adaptive SA for Maximum Temperature is based on this method. At first, the temperature is the lowest, and it is increased until the preliminary maximum temperature determined by a conventional method. The histories of the energy can be classified into three types. At  $T = 0$ , the local hill-climbing search is performed and a local minimum is obtained. After that, the temperature is increased to the maximum temperature. During this process, the energy decreases below the local minimum once (Type 1) or more than once (Type 2), or the energy does not decrease (Type 3), and the energy increases as the temperature increases. The actual maximum temperature is determined as the temperature where the energy finally increases across local minimum. When the actual maximum temperature is found, a conventional SA is carried out with this maximum temperature. The heating rate is 10 times as fast as



the cooling rate, and therefore, the total computation time can be reduced considerably. If the energy history during the heating has no decrease below the local minimum as shown as Type, this local minimum is the global minimum, and the optimum solution has been obtained already. In this case, a conventional Sa is not necessary. Some easy problem show such behavior and the probabilistic hill-climbing provides the global optimum.

This method is called the Adaptive Simulated Annealing for Maximum Temperature, and this method determines the effective maximum temperature in SA adaptively. There is no parameter in the method, and therefore the method can be considered to be easy to use for many combinatorial problems.