SOLUTION OF THE OPTIMAL PLANT LOCATION AND SIZING PROBLEM USING SIMULATED ANNEALING AND GENETIC ALGORITHMS

Ramana Rao, Vijaykumar Hanagandi* and Kevin Buescher
MS C312, Los Alamos National Laboratory
Los Alamos, NM 87545
and
Michael Nikolaou
Department of Chemical Engineering
Texas A&M University
College Station, TX 77843-3122

Abstract. In the optimal plant location and sizing problem it is desired to optimize a cost function involving plant sizes, locations, and production schedules in the face of supply-demand and plant capacity constraints. We will use simulated annealing (SA) and a genetic algorithm (GA) to solve this problem. We will compare these techniques with respect to computational expenses, constraint handling capabilities, and the quality of the solution obtained in general.

Simulated Annealing is a combinatorial stochastic optimization technique which has been shown to be effective in obtaining fast suboptimal solutions for computationally hard problems. The technique is especially attractive since solutions are obtained in polynomial time for problems where an exhaustive search for the global optimum would require exponential time.

We propose a synergy between the cluster analysis technique, popular in classical stochastic global optimization, and the GA to accomplish global optimization. This synergy minimizes redundant searches around local optima and enhances the capability of the GA to explore new areas in the search space.

*Author to whom all correspondence should be addressed. Tel.: (505) 665 0570, FAX: (505) 665 4479, E-mail: vmh@lanl.gov

The Plant Location and Sizing Problem

This is a classic problem (Klein and Klimpel, 1967) in the area of linearly constrained nonlinear optimization. The problem involves maximizing the *net present value* (NPV) of a multi-plant enterprise subject to a set of linear constraints. The objective function is highly nonlinear, and the nonlinearities arise from the setup capital required for constructing a plant, and economies of scale relating to capital and annual operational cost. The objective function has a very large number of local maxima making gradient descent type approaches to optimization ineffective.

We have chosen to study a stochastic approach to this maximization problem using the simulated annealing algorithm (Kirkpatrick et al., 1983). The primary motivation for using simulated annealing is its ability to seek out sub-optimal solutions within reasonable amounts of computational time. Simulated annealing is more effective than gradient descent since it allows *down-hill* transitions, thus preventing the system from getting trapped in local maxima. Further, the quality of the final solution found by the annealing algorithm does not depend on the initial configuration. However, as in the gradient projection algorithm (Rosen, 1960), no guarantee can be made that the solution found by the annealing algorithm is indeed the global optimum.

The objective function for the plant location problem,

$$\begin{aligned}
J \;=\; & -.5753\bar{S}_1 - 1.0313S_1^{.6} - .0685P_{101} - .0597P_{102} - .0522P_{103} + .0135S_1 \\
& -.0674S_1^{.45} + .0025P_{101}S_1 + .002P_{102}S_1 + .0017P_{103}S_1 \\
& -.1979P_{101}S_1^{-.55} - .1702P_{102}S_1^{-.55} - .1463P_{103}S_1^{-.55} - .396P_{121} \\
& -.247P_{131} - .313P_{122} - .202P_{132} - .243P_{123} - .162P_{133} - .3428\bar{S}_2 \\
& -.892S_2^{.6} - .0685P_{201} - .0597P_{202} - .0522P_{203} + .0135S_2 - .0809S_2^{.45} \\
& +.0025P_{201}S_2 + .002P_{202}S_2 + .0017P_{203}S_2 - .02227P_{201}S_2^{-.55} \\
& -.1914P_{202}S_2^{-.55} - .1645P_{203}S_2^{-.55} - .396P_{211} - .495P_{231} - .313P_{212} \\
& -.448P_{232} - .243P_{213} - .405P_{233} - .3164\bar{S}_3 - 1.2987S_3^{.6} - .0942P_{301} \\
& -.0819P_{302} - .0712P_{303} - .0539S_3^{.45} + .003P_{301}S_3 + .0026P_{302}S_3 \\
& +.0022P_{303}S_3 - .2227P_{301}S_3^{-.55} - .1914P_{302}S_3^{-.55} - .1645P_{303}S_3^{-.55} \\
& -.247P_{311} - .495P_{321} - .202P_{312} - .448P_{322} - .162P_{313} - .405P_{323} \\
& -.2441\bar{S}_4 - 1.3707S_4^{.6} - .0577P_{401} - .0504P_{402} - .044P_{403} \\
& +.002P_{401}S_4 + .0017P_{402}S_4 + .0015P_{403}S_4 - .1484P_{401}S_4^{-.55} \\
& -.1276P_{402}S_4^{-.55} - .1097P_{403}S_4^{-.55} - .495P_{411} - .099P_{421} - .04P_{431} \\
& -.448P_{412} - .09P_{422} - .04P_{432} - .405P_{413} - .088P_{423} - .041P_{433}\ ,
\end{aligned} \tag{1}$$

where $S_i;\ i \in \{1,2,3,4\}$ is the size of the plant at location i and $P_{ijk};\ j \in \{0,1,2,3\}$, $k \in \{1,2,3\}$ is the amount of the product shipped plant location i to market j in year k, is to be maximized subject to the constraints

$$\begin{aligned}
& \sum_i S_i = 10;\ \sum_i P_{i11} = 1;\ \sum_i P_{i12} = 4;\ \sum_i P_{i13} = 5;\ \sum_i P_{i21} = 2;\ \sum_i P_{i22} = 3; \\
& \sum_i P_{i23} = 2;\ \sum_i P_{i31} = 4;\ \sum_i P_{i32} = 3;\ \sum_i P_{i33} = 2;\ P_{i0.} \le S_i
\end{aligned} \tag{2}$$

where $P_{i0k} = \sum_j P_{ijk}$ denotes the total production at Plant i during the Year k. It is instructive to rewrite the constraint set (Eq. 2) as shown in Table 1. The inequality constraints in Eq. 2 are obtained by going along the columns of Table 1. From the table, we

Table 1: Constraint Set

				Year 1				
P_{111}	$+$	P_{211}	$+$	P_{311}	$+$	P_{411}	$=$	1
$+$		$+$		$+$		$+$		
P_{121}	$+$	P_{221}	$+$	P_{321}	$+$	P_{421}	$=$	2
$+$		$+$		$+$		$+$		
P_{131}	$+$	P_{231}	$+$	P_{331}	$+$	P_{431}	$=$	4
$\leq$		$\leq$		$\leq$		$\leq$		
S_1	$+$	S_2	$+$	S_3	$+$	S_4	$=$	10
				Year 2				
P_{112}	$+$	P_{212}	$+$	P_{312}	$+$	P_{412}	$=$	4
$+$		$+$		$+$		$+$		
P_{122}	$+$	P_{222}	$+$	P_{322}	$+$	P_{422}	$=$	3
$+$		$+$		$+$		$+$		
P_{132}	$+$	P_{232}	$+$	P_{332}	$+$	P_{432}	$=$	3
$\leq$		$\leq$		$\leq$		$\leq$		
S_1	$+$	S_2	$+$	S_3	$+$	S_4	$=$	10
				Year 3				
P_{113}	$+$	P_{213}	$+$	P_{313}	$+$	P_{413}	$=$	5
$+$		$+$		$+$		$+$		
P_{123}	$+$	P_{223}	$+$	P_{323}	$+$	P_{423}	$=$	2
$+$		$+$		$+$		$+$		
P_{133}	$+$	P_{233}	$+$	P_{333}	$+$	P_{433}	$=$	2
$\leq$		$\leq$		$\leq$		$\leq$		
S_1	$+$	S_2	$+$	S_3	$+$	S_4	$=$	10

observe that in Year 2, the sum of the amounts of product shipped off from each plant site exactly equals the sum of the plant sizes. Further, we see that the production is less than the plant size in at least some of the plants during the Years 1 and 3.

If the permissible values for P_{ijk} and S_i are restricted to integers, each equality constraint now admits only a finite number of combinations for P_{ijk} and S_i. It is easy to see that the problem now has about 3.14×10^{13} different combinations[1]. Of these, nearly 10^7 combinations yield feasible solutions. An exhaustive search over this feasible space, yields the globally optimal solution (over integers)

$$\begin{aligned} &S_1 = 7, S_2 = 3, P_{111} = 1, P_{131} = 4, P_{112} = 4, P_{132} = 3, P_{222} = 3, \\ &P_{113} = 5, P_{133} = 2, P_{221} = 2, P_{223} = 2, J = -11.105 \end{aligned} \tag{3}$$

This solution was reported by Klein and Klimpel (1967) as the best solution found by the gradient projection method. Klein and Klimpel (1967) also hinted that this solution could be the global maximum even when P_{ijk} and S_i are allowed to assume noninteger values. The second best solution reported by the same authors is

$$S_1 = 10, P_{111} = 1, P_{121} = 2, P_{131} = 4, P_{112} = 4, P_{122} = 3, P_{132} = 3,$$

[1] obtained by multiplying the number of partitions of each equality constraint

$$P_{113} = 5, P_{123} = 2, P_{133} = 2, J = -11.131 \ . \tag{4}$$

As we show in the next section, this above solution is easily found by the annealing algorithm.

Simulated Annealing

Simulated Annealing, a combinatorial optimization technique, has been shown to be effective in obtaining fast suboptimal solutions for computationally hard problems. The technique is especially attractive since solutions are obtained in polynomial time for problems where an exhaustive search for the global optimum would require exponential time. As a general purpose combinatorial optimization technique, simulated annealing was proposed by Kirkpatrick et al. (1983). It is a generalization of the *Monte Carlo* method developed earlier by Metropolis et al. (1953). In the past, simulated annealing has been used to solve such optimization problems as the *Traveling Salesman Problem* (Černy, 1982) and the *Wire Length Minimization Problem* in VLSI circuits (Lam and Delosme, 1988), and the *Bin-packing Problem* (Rao and Iyengar, 1994).

The simulated annealing algorithm provides a probabilistic method for choosing between feasible solutions of a problem which ensures that the choice is made from a set containing increasingly better solutions as the computational time expended on the algorithm increases. A control parameter, the *temperature*, of the combinatorial system determines the termination criterion of the algorithm. This control parameter is the analogue of the thermodynamic temperature in a physical system. Just as a physical system settles into into its least-energy states when heated to a large temperature and then cooled gradually, a combinatorial system seeks out minima in the solution space when the annealing algorithm is applied, starting with a large value of the control parameter and gradually decreasing this value. These ideas are made more precise in the following.

Let $\{s_i\};\ i \in Z$ be the feasible solution space of an optimization problem $\mathcal{P}$ with an associated objective function $f(x_i);\ i \in 1, \ldots, M$. Without loss of generality, we may assume that the objective function is to be *minimized* with respect to a set of constraints $\{c_i\};\ i \in 1, \ldots, N$. The simulated annealing algorithm starts by generating a random (feasible) solution to the problem. The algorithm proceeds by generating other solutions that satisfy the constraint set and accepting/rejecting these solutions based on the so-called *Metropolis* criterion (Metropolis et al., 1953). The algorithm terminates when the target value for the control parameter is reached and the current solution is accepted as the solution for $\mathcal{P}$. The annealing algorithm is shown (in pseudo-code) in Fig. 1.

In Step 2, T_∞ is a high value for the control parameter T. T_0 in Step 3 is the algorithm termination criterion. It corresponds to a very small temperature in a physical system. T_∞ ensures that almost all generated feasible solutions are accepted, even if they do not lead to a solution which is better. This is done to avoid getting stuck in local minima of the solution space. The inner loop (Steps 4–9) is iterated a fixed number of times at the same value of T. This number bears a linear relation to the size of the problem. In the minimization problem, it is chosen to be a small multiple of the number of parameters (M) in the problem. Step 10 readjusts the value of T according to a monotonically decreasing function $\mathcal{F}$. The profile of this function is critical to the performance of the algorithm. Too rapid a decrease leads to *quenching* where the system gets stuck in a local minimum while too slow a decrease is computationally wasteful. Best results are obtained by choosing an exponentially decreasing profile for $\mathcal{F}$.

Table 2: Results: Simulated Annealing

#	S_1	S_2	S_3	S_4	Obj. Fun.
1	10.0	0.0	0.0	0.0	11.130454
2	10.0	0.0	0.0	0.0	11.130454
3	6.886355	3.113645	0.0	0.0	13.683241
4	10.0	0.0	0.0	0.0	11.130454
5	7.033023	2.966977	0.0	0.0	13.212274
6	10.0	0.0	0.0	0.0	11.130454
7	10.0	0.0	0.0	0.0	11.130454
8	10.0	0.0	0.0	0.0	11.130454

Annealing the Plant Location Problem

Simulated annealing is a combinatorial optimization technique; thus it is best suited for optimization problems where the feasible solution space is a collection of discrete, countable points. This space is explored by the algorithm in a stochastic fashion for increasingly better solutions to the problem. Though the plant location problem displays an objective function whose solution space is uncountable, it is still possible to apply the basic annealing strategy — stochastic search for better and better suboptimal solutions — to the problem.

An *annealing schedule* is described by three choices — the starting value for the control parameter T, the stopping value for T, and the temperature decrement function $d(t)$. The configuration space of the problem is explored before the actual annealing process in order to obtain information about the fundamental statistical quantities of the system. This phase is used to estimate the high and low temperature scales of the system being optimized. Based on this statistical information gathered from our initial experiments, we chose to implement the inhomogeneous version (van Laarhoven and Aarts, 1987) of the simulated annealing algorithm. The homogeneous version of the algorithm proceeds by agitating the system a number of times at each value of the control parameter. Each set of such iterations (at a fixed temperature) is called a *chain*. Successive chains in the annealing process are carried out at decreasing values of the temperature parameter. The inhomogeneous algorithm performs only one iteration at each value of the temperature parameter but the temperature is reduced by an infinitesimal amount between iterations. As in the homogeneous algorithm, choosing too rapid a decrease for the temperature results in *quenching* and leads to local minima which are analogous to metastable states of a physical system.

Table 2. summarizes the results of 8 different annealing schedules for the inhomogeneous algorithm. It is relevant to mention here that the random starting values for S_i were assigned to allow one or more them to have a zero value. Without this modification, the simulated annealing algorithm is unable to reach the best or the second best solution. This is to be expected since the probability that three of the four plant sizes are zero is extremely small if S_i are randomly assigned values in the range (0,10).

From Table 2 it is reasonable to infer that the second best solution reported by Klein and Klimpel (1967) is easily found by the annealing algorithm. In our experiments we have also tried a two level annealing approach. For instance, experiments 3 and 5 in Table 2 do lead to the best solution reported by Klein and Klimpel (1967) if a local search is carried out in the neighborhood of the solution found by the annealing method. However, we caution that this is an *ad hoc* modification to the basic algorithm. In general, a coarse-to-fine refinement approach where the values of S_i are changed by large amounts towards the beginning of the annealing process and smaller amounts at lower temperatures will perform favorably when compared to the *ad hoc* two level annealing method. We are currently studying a refinement-based approach to the problem.

Background on Genetic Algorithms

GAs accomplish the task of optimization by starting with a random "population" of values for the parameters of an optimization problem, and thereafter producing new "generations" of improved values that combine the best "parts" of values from previous populations. In order to achieve this, it is very crucial to represent the objective variables in a non ambiguous way. In our case studies we use the binary representation of real numbers and integers and manipulate these strings to achieve reproduction, crossover and mutation. Parent representations which are used to produce offspring are selected in a random manner but with a bias toward selecting candidates with high fitness values. This step is called *reproduction.* Offspring are produced by randomly cutting and swapping parts of the two parent representations. This is called the *crossover* operation. *Mutation* is achieved by screening the bits of the offspring and changing each binary bit to the opposite kind (0 to 1 or 1 to 0) with a very low probability. Mutation helps to introduce diversity in the population thus preventing the GA from getting saturated with solutions in local optima. Figs. and . illustrate the crossover and mutation operations using binary representations of the real and integer optimization variables. In each generation a limited number of candidates get to reproduce and have offspring which go into the next generation. When the problem is of objective function maximization the fitness $\bar{f}_i$ of candidate i in a given population is the normalized objective function given by

$$\bar{f}_i = \frac{f_i - f_{min}}{f_{max} - f_{min}}$$

where f_i is the objective function value of candidate i and f_{max} and f_{min} are the maximum and minimum values respectively of f_i encountered in that population. In a minimization problem the same procedure is followed but with a sign reversal. *Elitism,* in which the best candidate encountered thus far is reintroduced in each generation, can also be employed. Tables 3, 4 and 5 give the pseudocode for a simple GA (SGA). In the pseudocode, the array `fitness` contains the fitness values of all the members of the current population. `sumfitness` is the sum of the `fitness` values of all members of the population and the function `random()` returns a uniform random number on the real interval [0,1]. One run of the roulette wheel selection routine gives two parents which can now be used to perform crossover and mutation as demonstrated in Figs. 1 and 2. For further details about the theory and implementation of GAs we refer the reader to well known literature (Goldberg, 1989; Davis, 1991).

The following points summarize the advantages of using GAs:

- As a GA proceeds randomly (yet systematically) in its search, it does not require smoothness, derivability, continuity etc. in the objective function. The only requirement is that for a set of values for the optimization parameters, one must assign a fitness value.

Table 3: The SGA pseudocode - Main algorithm.

```
START
    generation count=0
    initialize population by random strings
    repeat
        generation count=generation count+1
        create next generation
    until (generation count ≥ max gen)
STOP
```

Table 4: The SGA pseudocode - Refinement of create next generation.

```
START
    j=0
    repeat
        select two members by roulette wheel selection
        perform crossover
        perform mutation
        introduce offspring into the new population
        j=j+2
    until(j ≥ population size)
STOP
```

Table 5: The SGA pseudocode - Refinement of select two members by roulette wheel selection.

```
START
    partial sum=0.0
    j=0
    rand=random()×sumfitness
    repeat
        j=j+1
        partial sum=partial sum+fitness(j)
    until(partial sum > rand or j=population size)
    selected individual=j
STOP
```

- GAs can efficiently handle highly nonlinear and noisy objective functions encountered in stochastic processes where traditional gradient based methods are inefficient.

- GAs are amenable to parallel processing. Unlike gradient search algorithms, in GA the objective function evaluation for one parameter set is independent of that for all others in the same generation. This facilitates the use of parallel computers for the search procedure.

Motivation for Using a Clustering Algorithm in Search

Many valuable lessons are learned by observing the nature closely. Holland (1975) made arguments concerning the distribution of individuals in a natural environment. He argued that, in nature, when like individuals begin to dominate a niche, increased competition for limited resources decreases life expectancy and birth rates among the members of that niche. Less crowded niches experience less pressure and achieve life expectancy and birth rates much closer to their actual potential. In other words, each individual has a certain survival probability and reproduction capability associated with its fitness in its environment. If a particular individual belongs to a niche which is dominated by like individuals then its survival and reproduction chances are diminished because it is more likely to compete with the members of its own community (niche) for limited resources whereas for an individual belonging to a less crowded niche these rates are not altered. To incorporate these aspects of nature in artificial genetics, De Jong used the so called *crowding model.* In this model, whenever an individual is born, it is compared with a fixed number of individuals in the current population and it replaces an individual which is closest to itself. He called the fixed number of individuals compared as the *crowding factor.* This helps prevent the duplication of like individuals and helps to maintain diversity to some extent. When multi-modal function optimization is of concern, the issue of niches being formed has yet another dimension.

Consider a situation where we start with a random initial population of parameters and do a few steps of gradient search starting from each one of these points. In this case there is a high probability that elements of a particular niche (which is nothing but a group of like individuals) tend toward the same local optimum if the gradient-based search is continued. Instead, it makes perfect sense to select the optimal member of this niche and do a gradient-based search starting from only this member. A similar concept was used by Törn (1977; 1978) in conjunction with the *multistart* algorithm- a stochastic optimization technique. Törn's algorithm, known as Algorithm LC (multiple local searches with clustering), is given below:

1. Choose uniformly random points.

2. Use a local search algorithm for a few steps.

3. Find clusters by using a cluster analysis technique.

4. Take a sample point from each cluster. Return to step 2.

We propose to use this idea in the context of the genetic search procedure. We use Törn's clustering technique to identify and dissolve clusters in the parameter space so that redundant function evaluations of candidates leading to the same local optimum are minimized and diversity is introduced in the population. In other words, by the cluster dissolving idea we explicitly impose the natural pressure on crowded niches and cause them to perish. When the performance of a simple genetic algorithm (SGA) and that of a simple genetic algorithm with De Jong's crowding model (SGA-CM) are compared we noticed that niches are not effectively dissolved for any value of the crowding factor. On the other hand our methodology explicitly identifies and breaks the clusters.

Methodology and Results

Although it eliminates redundant calculations, Törn's algorithm (see previous section) lacks the ability to search in new directions which is crucial for multi-modal function optimization. To remedy this problem we propose a hybrid methodology which relies on the strength of GAs to search in new directions and on the ability of the clustering technique to reduce redundant calculations. The steps in the proposed GA with clustering (SGA-CL) are as follows:

1. Start with an initial population.
2. Evaluate the fitness of each member of the population.
3. Apply a clustering algorithm to identify clusters.
4. If any clusters are identified, from each cluster select the best member and randomize the rest.
5. Evaluate fitness of the new members whose fitness has not been evaluated in step 2.
6. Apply reproduction, crossover and mutation.
7. Generate new population.
8. Stop if the number of generations exceeds the maximum allowable generations or if convergence is satisfactory. Else, go to step 2.

The breaking of clusters reduces redundant objective function evaluations and the randomization introduces diversity in the population. In the GA we use here, we employ elitism. Elitism accelerates local search. Using elitism in between cluster dissolving operations is analogous to doing a local search between cluster dissolving operations as suggested by Törn in his algorithm LC. The clustering algorithm we use here aims at selecting a seed point and growing the cluster by enlarging the hypersphere around the seed point as long as the density of the cluster remains greater than the average density of the space in which all the points are distributed. This method does not require the user to specify the number of clusters *a priori*. Details of this algorithm are given in APPENDIX A.

In the optimal plant location and sizing problem, the SGA and gradient-based technique gave inferior solutions. Constraint handling was done as explained in the simulated annealing section. The SGA with clustering approach gave the second best known solution with a success probability of unity (i.e. it gave this solution from any initial population of parameter settings). Details are summarized in Table 6. Table 6 gives details of the best solutions obtained using the indicated techniques. Of the three techniques listed, the SGA-CL techniques gave the best solution. However, none of these techniques gave the best solution known so far (Klein and Klimpel, 1967) which has an objective function of -11.105 MM$.

Table 6: The best objective function values reached by SGA, gradient and SGA-CL techniques

SGA	Gradient Technique (FSQP)	SGA-CL
-12.09 MM$	-16.80 MM$	-11.131 MM$

Conclusions

The simulated annealing algorithm was successful in identifying the second best solution known so far in most of the trials. In the experiments we have conducted, this algorithm needed augmentation (in the form of a local search in the neighborhood of the solution) before it could identify the best solution known so far. We are currently studying different techniques for enhancing the effectiveness of simulated annealing based methods for solving linearly constrained nonlinear optimization problems.

In population-based stochastic search techniques like GAs, formation of niches around local optima is detrimental to the convergence to global optimum. In this paper we presented a modification to the simple genetic search which is based on a clustering algorithm and demonstrated its efficiency in solving a plant location and sizing problem. This algorithm aims at identifying and dissolving the clusters so that diversity is preserved in the population. The superiority of the proposed methodology over the simple genetic search and a gradient-based technique is clear although it failed to attain the best solution known so far.

Our methodologies were able to find the next best solution in almost all the trials. The advantages of using the SA and SGA-CL stem from the fact that the objective function can be an arbitrary black box and nice analytic properties are not required (i.e., smoothness and/or differentiability of the objective function is not assumed).

References

Boender, C. G. E., Kan, A. H. G. R., and Timmer, G. T. (1982). A Stochastic Method for Global Optimization, *Mathematical Programming,* vol. 22, pp 125-130.

Černy, V. (1982). A Thermodynamic Approach to the Traveling Salesman Problem: an efficient simulation algorithm. *J. Optimization Theory and Applications,* vol. 45, pp. 41–51.

Davis, L. (1991), *Handbook of Genetic Algorithms*, Van Nostrand, New York.

De Jong, K. A. (1975). *An Analysis of the Behavior of a Class of Genetic Adaptive Systems,* Ph.D. Dissertation, University of Michigan, Ann Arbor, MI.

Dixon, L. C. W. and Szegö, G. P. (1978). *Towards Global Optimisation 2,* North-Holland, Amsterdam.

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

Gomulka, J. (1978a). Deterministic Versus Probabilistic Approaches to Global Optimization. In L. C. W. Dixon and G. P. Szegö (Eds) *Towards Global Optimisation 2,* , North-Holland, Amsterdam.

Gomulka, J. (1978b). A User's Experience with Törn's Algorithm. In L. C. W. Dixon and G. P. Szegö (Eds) *Towards Global Optimisation 2,* , North-Holland, Amsterdam.

Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems,* The University of Michigan Press, Ann Arbor, MI.

Kan, A. H. G. R. and Timmer, G. T. (1984). A Stochastic Approach to Global Optimization. In *Numerical Optimization,* SIAM, Philadelphia, PA (1984).

Kirkpatrick S., Gelatt Jr., C.D., and Vecchi, M.P. (1983). Optimization by Simulated Annealing. *Science,* vol. 220, pp. 671–680.

Klein M., and Klimpel, R.R. (1967). Application of Linearly Constrained Nonlinear Optimization to Plant Location and Sizing *J. Industrial Engineering,* Vol. XVIII, No. 1, pp. 90–95.

Lam, J., and Delosme, J-M. (1988). Performance of a New Annealing Schedule. *Proc. 25th ACM/IEEE Design Automation Conference,* pp. 306–311.

Metropolis, M., Rosenbluth, A., Rosenbluth, M., Teller, A., and Teller, E. (1953). Equation of State Calculations by Fast Computing Machines. *J. Chem. Phys.,* vol. 21, pp. 1087–1092.

Rao, R.L. and Iyengar, S.S. (1994). Bin-packing by simulated annealing. *Int. J. Comput. and Math. Appl.,* Vol. 27, No. 5.

Rosen, J.B. (1960). The Gradient Projection Method for Nonlinear Programming—Part I, Linear Constraints. *J. SIAM,* Vol. 8, No. 1.

Törn, A. A. (1978). A Search-Clustering Approach to Global Optimisation. In L. C. W. Dixon and G. P. Szegö (Eds) *Towards Global Optimisation 2,* North-Holland, Amsterdam.

Törn, A. A. (1977). Cluster Analysis Using Seed Points and Density-Determined Hyperspheres as an Aid to Global Optimization. *IEEE Trans. Sys. Man. Cyber.,* vol 7, pp 610.

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

algorithm *anneal*
begin
generate a random initial solution $\mathbf{s}$ in $\{s_i\}$
$T \leftarrow T_\infty$;
while $T > T_0$ **do**
  repeat
    generate new solution $\mathbf{s}'$ in $\{s_i\}$
    $\Delta C = \mathcal{C}(\mathbf{s}') - \mathcal{C}(\mathbf{s})$
    if $\Delta C < 0$ **or** $\eta < e^{-\frac{\Delta C}{T}}$
      $\mathbf{s} \leftarrow \mathbf{s}'$
  until thermal equilibrium is reached
  $T \leftarrow \mathcal{F}(T)$
end do
output $\mathbf{s}$ $\{\mathbf{s}$ is the best solution$\}$
end.

Figure 1: The Simulated Annealing Algorithm for Minimization

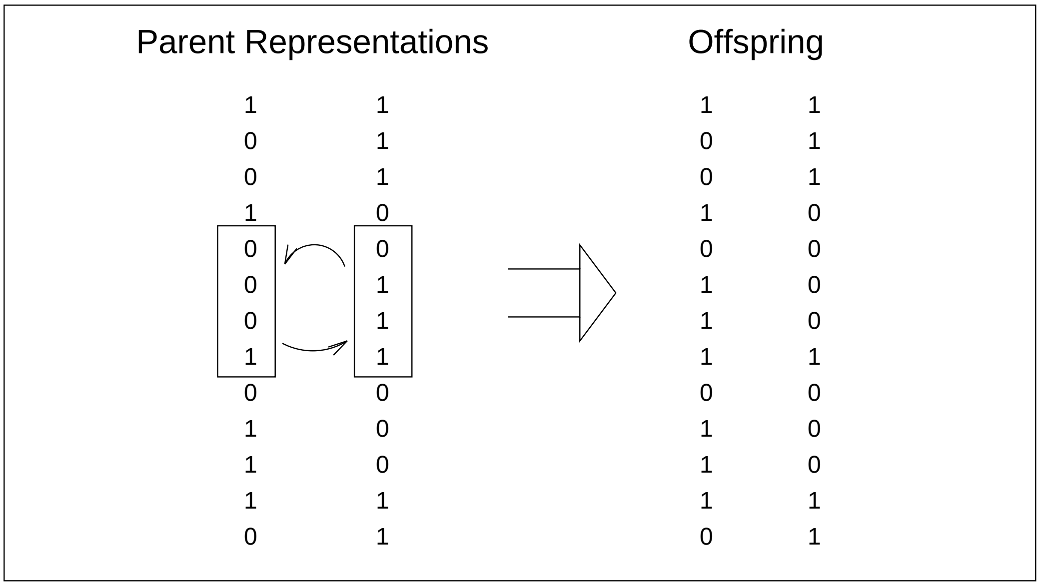

Figure 2: Illustration of the crossover operation. The offspring are a result of cutting and swapping parts indicated between the arrows in the parent representations.

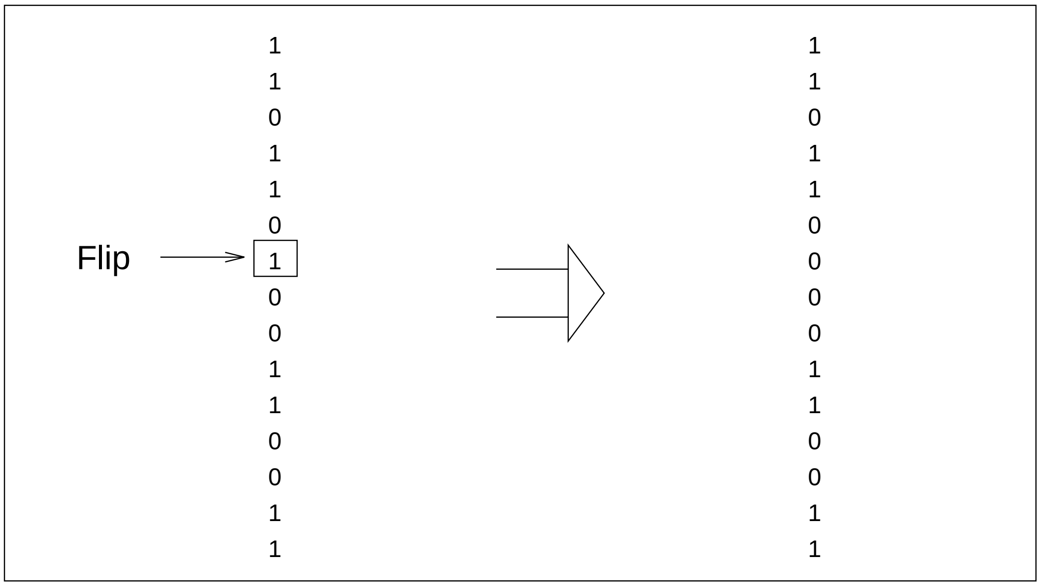

Figure 3: Illustration of mutation. Each bit is flipped to the opposite kind with a very low probability.

APPENDIX A

TÖRN'S CLUSTERING ALGORITHM

In what follows we describe briefly the clustering algorithm used by Törn in his algorithm LC. The basic idea is to grow clusters from seed points to include all the neighboring points. There are two main issues in identifying clusters:

- Location of the seed points.
- When to stop the growth of a cluster. In other words, deciding about a threshold distance r for stopping the growth of a cluster.

The choice of a threshold distance is critical because if r is large, all points will be assigned to a single cluster and if r is small, each point will be form an isolated cluster.

In a multi-modal objective function minimization problem, consider a collection of N points in which we want to identify clusters. The assumption is made that the local minima would be good seed points to use and that the point with the smallest function value may be used as an estimate of the local minimum. The cluster growing is started from this point. When this cluster stops growing and a new seed point is needed, the point, among those not yet classified, with the least function value is chosen. This procedure is repeated until all the N points are classified.

A cluster is grown by enlarging a hypersphere around the seed point as long as the point density in the hypersphere remains greater than the uniform point density $\rho = \frac{N}{V}$ where V is the volume of the space containing all the N points and is given by

$$V = \Pi_{i=1}^{N} 4\lambda_i^{\frac{1}{2}}.$$

λ_i are the eigenvalues of the covariance matrix S and

$$s_{ij} = \frac{1}{N-1}\left(\sum_{k=1}^{N} x_{ik}x_{jk} - \frac{1}{n}\sum_{k=1}^{n} x_{ik}\sum_{k=1}^{n} x_{jk}\right)$$

In the above equation, $\mathbf{x}_k$ is the kth parameter vector and x_{ik} is the ith component of $\mathbf{x}_k$. Törn's original clustering algorithm includes only the parameters of optimization in the vectors $\mathbf{x}_k$ for cluster identification, but we include the objective function value along with the parameters of optimization in the vectors $\mathbf{x}_k$. This modification gives better classification (cluster identification) particularly for functions with multiple local minima, each having different objective function values, which are crowded in a particular search space.

List of Figures

List of Tables