Computer Science Department Technical Report
University of California
Los Angeles, CA 90024-1596

OLD BACHELOR ACCEPTANCE: A NEW CLASS OF NON-MONOTONE THRESHOLD ACCEPTING METHODS

A. B. Kahng
C.-W. A. Tsao

September 1992
CSD-920040

Old Bachelor Acceptance: A New Class of Non-Monotone Threshold Accepting Methods*

Andrew B. Kahng and Chung-Wen Albert Tsao
UCLA Dept. of Computer Science, Los Angeles, CA 90024-1596

September 25, 1992

Abstract

Stochastic hill-climbing algorithms, particularly simulated annealing (SA) and threshold acceptance (TA), have become very popular for global optimization applications. Typical implementations of SA or TA use *monotone* temperature or threshold schedules, and moreover are not formulated to accommodate practical time limits. We present a new threshold acceptance strategy called *Old Bachelor Acceptance* (OBA) which has three distinguishing features: (i) it is specifically motivated by the practical requirement of optimization within a prescribed time bound, (ii) the threshold schedule is *self-tuning*, and (iii) the threshold schedule is *non-monotone*, with threshold values allowed to become negative if necessary. The original TA method of Dueck and Scheuer is a special case of OBA. Experiments using several classes of symmetric traveling salesman problem instances show that OBA outperforms previous hill-climbing methods for time-critical optimizations. A number of directions for future work are suggested.

1 Preliminaries: Global Optimization Heuristics

Given a set S of feasible solutions and a real-valued cost function $f : S \rightarrow \Re$, global optimization may without loss of generality be formulated as the search for a global minimizer $s \in S$ such that $f(s) \leq f(s') \; \forall s' \in S$. Typically, $|S|$ is very large compared to the number of solutions that can be examined in practice. For small instances of certain global optimizations, implicit enumeration (e.g., branch-and-bound) or polyhedral approaches can prune the solution space and afford solutions within practical time limits; other problem formulations may be tractable to problem-specific methods. However, many important global optimization formulations (both discrete and continuous) are NP-complete, with heuristics therefore being of interest.

1.1 Iterative Methods

We are interested in heuristics which iteratively apply the following two rules (Figure 1):

Rule 1 is *memoryless*, with generation of s' based only on the current solution s_i.[1] Rule 1 also induces the notion of a *neighborhood structure* over S, where the neighborhood $N(s_i)$ of the current

*This work was supported in part by NSF MIP-9110696, NSF Young Investigator Award MIP-9257982, ARO DAAK-70-92-K-0001 and ARO DAAL-03-92-G-0050.

[1] A history-dependent Rule 1′ might be, for example: Given the history of solutions evaluated thus far, generate a new trial solution s'. Rule 1′ accommodates such methods as iterated descent [3] and tabu search [8]. These latter

Iterative Global Optimization

Rule 1: Given the current solution s_i, generate a new trial solution s'
Rule 2: Decide whether to set $s_{i+1} = s_i$ or $s_{i+1} = s'$.

Figure 1: High-level template for iterative global optimization.

solution $s_i \in S$ is the set of possible trial solutions s' that can be generated from s_i. The quality of solutions defines a *cost surface* over the neighborhood structure, and optimization is search for a global minimum in this cost surface. Typically, the set $N(s)$ consists of slight perturbations of the current solution s, for example, via the 2-interchange operator for the traveling salesman problem [15] or the pair-swap operator for graph bisection [17]. When the size of $N(s)$ is constant for all $s \in S$, we denote the neighborhood size by $|N|$. In practice, Rule 1 simply picks a random $s' \in N(s_i)$ from within "obvious" neighborhood structures such as those noted for the TSP and graph bisection problems [15]. Therefore, it is Rule 2 which determines the nature of an optimization heuristic as it traverses the cost surface.

A simple instance of Rule 2 is, "Replace s_i by s' if $f(s') < f(s_i)$," which corresponds to greedy optimization. Greed has been widely employed because of its simplicity and its acceptable success in a variety of implementations, e.g., Johnson et al. [16] [17] have documented the utility of greed for several hard combinatorial problems. However, the performance of greedy methods is erratic, and achieving "stable" – i.e., predictable – performance requires multiple random initial starting solutions. Johnson et al. [16] have determined that several thousand initial random starting configurations are necessary for greed to afford stable solution quality for graph bisection instances of size $n = 500$; this number grows rapidly with n and becomes hopeless fo instance sizes of, e.g., $n = 100,000$ which arise in arenas such as VLSI circuit partitioning. Moreover, central limit phenomena in the cost surface [3] imply that as problems grow large, random local minima are almost surely of "average" quality[2] so that simple "multi-start" heuristics [28] fail. In view of these factors, global optimization heuristics must escape from local minima to adequately explore the solution space of large problems.

1.2 Stochastic Hill-Climbing: SA and TA

Stochastic hill-climbing methods probabilistically escape from local minima in the cost surface. The first such method, *simulated annealing* (SA), was proposed independently by Kirkpatrick et al. [23]

methods systematically explore solution paths emanating from several best-known solutions; see also the heuristic search techniques used in artificial intelligence [26].

[2]See discussions by Baum [3] and Kirkpatrick and Toulouse [24] on traveling salesman structures; by Kauffman and Levin [21] on evolutionary optimization for "adaptive landscapes"; and by Bui et al. [5] and Hagen and Kahng [9]) for graph partitioning.

and Cerny [6]. Motivated by analogies between the solution space of an optimization instance and microstates of a statistical thermodynamical ensemble, the idea of simulated annealing is summarized in Figure 2. SA uses the following criteria for Rule 2. If $f(s') < f(s_i)$, then $s_{i+1} = s'$, i.e., the new solution is adopted. If $f(s') \geq f(s)$, the "hill-climbing" disimprovement to $s_{i+1} = s'$ still has a nonzero probability of being adopted, determined by both the magnitude of the disimprovement and the current value of a *temperature* parameter T_i. This probability is given by the "Boltzmann acceptance" criterion in Line 6 of Figure 2.

Algorithm SA(M)

$M \equiv$ limit on number of Rule 1 iterations

1. Choose (random) initial solution s_0;
2. Choose initial *temperature* T_0;
3. for $i = 0$ to $M - 1$
4. Choose (random) neighbor solution $s' \in N(s_i)$;
5. if $f(s') < f(s_i)$ then $s_{i+1} = s'$
6. else $s_{i+1} = s'$ with $Pr = exp((f(s_i) - f(s'))/T_i)$;
7. $T_{i+1} = next(T_i)$;
8. Return s_i, $0 \leq i \leq M$, such that $f(s_i)$ is minimum.

Figure 2: Bounded-time SA template.

Another stochastic hill-climbing heuristic called *threshold accepting* (TA), which uses a different Rule 2 criterion (Figure 3), has recently been proposed by Dueck and Scheuer [7]. TA relies on a *threshold*, T_i, which defines the maximum disimprovement $f(s') - f(s_i)$ that is acceptable at the current iteration. All disimprovements greater than T_i are rejected, while all that are less than T_i are accepted. Thus, in contrast to the Boltzmann acceptance rule of annealing, TA offers a deterministic Rule 2.

Algorithm TA(M)

$M \equiv$ limit on number of Rule 1 iterations

1. Choose (random) initial solution s_0;
2. Choose initial *threshold* T_0;
3. for $i = 0$ to $M - 1$
4. Choose random neighbor solution $s' \in N(s_i)$;
5. if $f(s') < f(s_i) + T_i$ then $s_{i+1} = s'$
6. else $s_{i+1} = s_i$
7. $T_{i+1} = next(T_i)$;
8. Return s_i, $0 \leq i \leq M$, such that $f(s_i)$ is minimum.

Figure 3: Bounded-time TA template.

At timestep i, the SA temperature T_i allows hill-climbing by establishing a nonzero probability of accepting a disimprovement, while the TA threshold T_i allows hill-climbing by specifying a permissible amount of disimprovement. In practice, SA will use a large initial temperature and a final temperature of zero.[3] The monotone decrease in T_i is accomplished by $next(T_i)$, which is a heuristic function of the T_i value and the number of iterations since the last cost function improvement. (Typically, $next(T_i)$ tries to maintain "thermodynamic equilibration" at each temperature value.) Similarly, implementations of TA [7] begin with a large initial threshold T_0 which decreases monotonically to $T_M = 0$. It should be noted that both SA and TA will in practice return the "best-so-far energy" (BSFE), which is the minimum-cost solution among $s_0, s_1, \ldots, s_M$; this is reflected in the templates of Figures 2 and 3 and in the experimental comparisons of Section 3 below.

The SA and TA algorithms both enjoy certain theoretical attractions. By using Markov chain arguments and basic aspects of Gibbs-Boltzmann statistics one can show that with an appropriate $next(T_i)$ function, $Pr(s_M \in R) \rightarrow 1$ as $M \rightarrow \infty$, where $R = \{s \in S | f(s) \leq f(s') \; \forall s' \in S\}$ (R denotes the set of all global minimum solutions). In other words, SA is optimal in the limit of infinite time [1]. Althofer and Koschnick [2] argue that each execution of SA lies in some sense within the convex hull of a set of TA executions, and that TA is therefore also provably good. However, this convergence result is slightly weaker than those established for SA.

Finally, the practical utility of stochastic hill-climbing is well-documented, with the SA algorithm now being one of the most widely used heuristics for global optimization [1]. Thus, it is noteworthy that Dueck and Scheuer [7] claim that their TA method "yields better results than SA" with respect to both CPU time and the number of "new state choice steps" (i.e., applications of Rule 1), a standard measure of runtime complexity). Experimental results are presented in [7] which support this claim. A further practical advantage of TA is its greater simplicity of Rule 2, with no exponentiation or random number generation being required. With this in mind, our experimental results below compare OBA variants against the TA algorithm.

2 Non-Monotone Threshold Schedules: The OBA Approach

2.1 Motivations

Recall from the above discussion that SA and TA are traditionally implemented with *monotone* temperature or threshold schedules. For SA, the thermodynamic analogy, as well as the convergence proof based on Gibbs-Boltzmann statistics, together motivate the following intuition [10]: monotone temperature schedules allow annealing to explore "large features" of the cost surface at high T, and then perform finer optimization at lower T. For TA, the authors of [7] state that the "trivial" threshold schedule (linear in i, with $T_i = T_0 * (1 - \frac{i}{M})$) is "essentially best", and suggest that the

[3]Note that $T = \infty$ accepts all moves (i.e., a random walk in the cost surface); $T = 0$ accepts only improving moves (i.e., greed).

performance of TA is basically insensitive to the threshold schedule. Indeed, the successful results reported in [7] were obtained using monotone threshold schedules. However, despite the tremendous success of both SA and TA, certain observations motivate the study of alternative hill-climbing strategies.

First, standard implementations of SA and TA are not amenable to *a priori* specification of CPU limits. With respect to the templates of Figures 2 and 3, common practice will use $M = \infty$ and test for a stopping criterion (e.g., "equilibration" in SA) to terminate the algorithm. A finite time limit M will obviate the theoretical convergence results, but in practice a finite-time requirement in optimization is very real. Experimental results [20] [18] for large discrete and continuous global optimizations show that optimal annealing schedules vary strongly with the time limit M,[4] but it is not clear how $next(T_i)$ should be defined to accommodate finite M.

Second, current SA and TA implementations are "blind" to the specific features of the cost surface in any given optimization instance. Previous work [20] [18] [19] [29] has shown that large, real-world cost surfaces exhibit strong fits to models of self-similar random structure (e.g., VLSI placement problems have hierarchical scaling properties which resemble high-dimensional fractional Brownian motions [29]). The parameters of such fitted models vary with the individual problem instances, and again, evidence suggests that optimal annealing schedules should be tuned to these parameters [19].

These two observations prompt a variety of questions and simple thought experiments. Consider the BSFE performance of TA from random starting solutions in the one-dimensional cost surface of Figure 4 (six solutions s_i, each with $|N| = 2$). For any value of M, note that the last value T_{M-1} in the threshold schedule does not affect the best-so-far solution value at time M. One can readily see that if $M = 2$, the optimal schedule should have $T_0 = 0$, with all values of T_1 yielding optimal length-two schedules of form $\{0, X\}$. We have exhaustively enumerated threshold schedules for $M = 2, 3, \ldots, 10$ for the cost surface of Figure 4. Optimal schedules which maximize the probability of finding the optimal solution D within prescribed time bounds (starting from a random solution) include: $\{0, 3, 0, X\}$ for $M = 4$ ($Pr = 0.6250$); $\{0, 3, 0, 0, X\}$ for $M = 5$ ($Pr = 0.6875$); $\{0, 0, 3, 0, 0, X\}$ for $M = 6$ ($Pr = 0.7396$); $\{0, 3, 0, 0, 3, 0, 0, X\}$ for $M = 8$ ($Pr = 0.8047$); and so on. In fact, *all* of the optimal schedules for $M \geq 4$ are clearly non-monotone.

A slightly different cost surface (Figure 5) points out that optimal threshold sequences can easily have *negative* values, e.g., $\{-3, X\}$ and $\{-2, X\}$ are optimal ($Pr = 0.4167$) for $M = 2$; other optimal schedules include $\{-3, 1, X\}$ ($Pr = 0.5$) for $M = 3$ and $\{-3, 3, 1, X\}$ ($Pr = 0.5625$) for $M = 4$.

Recently, we have found that a number of authors have also touched on the issue of non-monotonicity in annealing. In particular, Strenski and Kirkpatrick [30] have shown that "locally optimal" annealing schedules can be non-monotone for a small instance of the graph bisection problem that is highly

[4] In [20] and [18], single-temperature annealing schedules were used in order to reduce the number of degrees of freedom in the experiment. Recent work by Boese, Kahng and Tsao [4] has confirmed these results for general annealing schedules; see also the work of Strenski and Kirkpatrick [30] and Althofer and Koschnick [2], discussed later in this section.

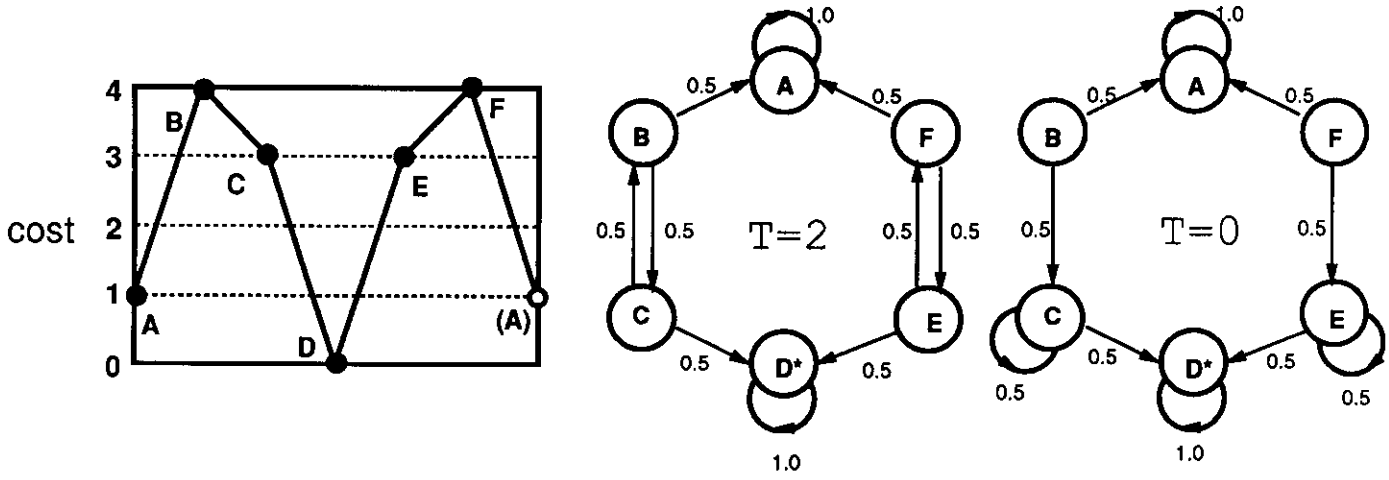

Figure 4: A simple cost surface, along with transition probabilities for $T = 1$ and $T = 3$.

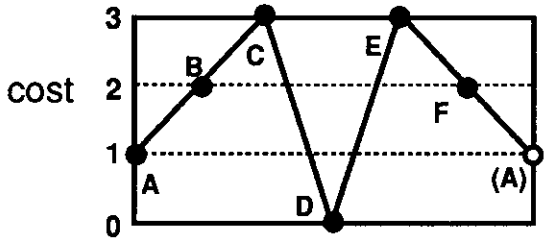

Figure 5: Cost surface for which optimal schedules (M small) contain negative T values.

structured to reduce the size of the solution space. Hajek and Sasaki [11] show that a class of cost surfaces exists for which optimal schedules are non-monotone. Finally, Althofer and Koschnick [2] enumerate optimal TA schedules for a small cost surface and find clear evidence (Table 4.1 in [2]) of non-monotonicity; however, the authors surprisingly make no comment on this data. Given these motivations, we have investigated a class of threshold accepting methods which use non-monotone threshold sequences.

2.2 The OBA Algorithm

Old Bachelor Acceptance uses a threshold criterion in Rule 2, but the threshold changes dynamically – up or down – based on the perceived likelihood of being near a local minimum. Observe that if the current solution s_i has lower cost than most of its neighbors, it will be hard to move to a neighboring

solution; in such a situation, standard TA will repeatedly generate a trial solution s' and fail to accept it. OBA uses a principle of "dwindling expectations": after each failure, the criterion for "acceptability" is relaxed by slightly increasing the threshold T_i (this motivates the name "Old Bachelor Acceptance"). After sufficiently many consecutive failures, the threshold will become large enough for OBA to escape the current local minimum. The converse of "dwindling expectations" is what we call *ambition*, whereby after each acceptance of s', the threshold is *lowered* so that OBA becomes more aggressive in moving toward a local minimum. With this in mind, the basic OBA template is as shown in Figure 6.

Algorithm OBA(M)

$M \equiv$ limit on number of Rule 1 iterations

Choose (random) initial solution s_0;
Choose initial threshold T_0;
for $i = 0$ to $M - 1$ do
  Choose (random) neighbor solution $s' \in N(s_i)$;
  if $f(s') < f(s_i) + T_i$ then
    $s_{i+1} = s'$;
    $T_{i+1} = T_i - decr(T_i)$;
  else
    $s_{i+1} = s_i$;
    $T_{i+1} = T_i + incr(T_i)$;
  endif.

Figure 6: High-level OBA description.

Notice that if we use constant update functions $decr(T_i) = -incr(T_i) = \frac{T_0}{M}$, OBA corresponds to the TA method of Dueck and Scheuer. Thus, special cases of OBA enjoy the same convergence properties shown for TA in [2].

2.3 OBA Variants

Via the threshold update functions $incr(T_i)$ and $decr(T_i)$, the template of Figure 6 captures many possible strategies. We have typically based *decr* and *incr* on the following factors:

1. The *neighborhood size*, $|N|$, along with the *age* of the current iteration, which is the number of Rule 1 applications since the last move acceptance. The value of $|N|$ affects "reachability" between solutions, i.e., the diameter of and multiplicity of paths within the neighborhood structure. Intuitively, *age* reflects the OBA algorithm's current perception of local structure in the cost surface: increasing *age* implies greater likelihood that s_i is a local minimum, and that the threshold should increase faster.

2. The amount of time remaining, $M - i$. Since previous work [18] [19] has observed strong dependence of optimal hill-climbing strategies on the time bound M, we may allow *decr* and *incr* to depend on the proportion of time used, i/M.

3. The current threshold value T_i. We may allow different update rules depending on whether T_i is highly positive, highly negative, close to zero, etc.

Initially, we tried variants that were based on the "obvious" choice of $decr(T_i) = \Delta_1$ and $incr(T_i) = \Delta_2$ both being constant functions. This was in part motivated by the above observation that $\Delta_1 = -\Delta_2$ yields the TA algorithm. In Section 3 below, we give experimental results for five slightly more sophisticated strategies, which we call OBA1 - OBA5.

OBA1. The OBA1 variant (Figure 7) heuristically "accelerates" the incrementing of T_i as *age* increases.[5] Second, OBA1 "damps" the magnitude of T_i as $i \rightarrow M$; this does not have any thermodynamic motivations as with annealing schedules that go to zero, but is rather to ensure that at some point during the optimization, the appropriate "granularity" in the threshold update is applied. Third, we note that following the intuition provided by the example of Figure 5, the OBA1 variant allows threshold values to become negative; the algorithm may thereby prefer a *good* improving move over a random improving move. These considerations yield the following core threshold update rule:

$$T_{i+1} = \begin{cases} (T_i - \Delta) * (1 - i/M) & if \quad age = 0(decr(T_i)) \\ (T_i + \Delta * \frac{age}{|N|}) * (1 - i/M) & if \quad age > 0(incr(T_i)) \end{cases}$$

OBA2. The second variant, OBA2 (Figure 8), has parameters M, Δ, a, b, and c, along with a core threshold update strategy of form

$$T_{i+1} = ((\frac{age}{a})^b - 1) * (\Delta) * (1 - \frac{i}{M})^c.$$

The parameters a, b and c afford the ability to fine-tune the growth rate $incr(T_i)$ as follows: a tunes the threshold growth rate by a multiplicative factor; b allows a power-law growth rate; and c allows tuning of a "damping" heuristic similar to that used in the OBA1 approach. Notice that whenever $age = 0$ (Line 9 of the Figure 8 template), OBA2 immediately sets the threshold to the most negative value allowable, thus giving the algorithm the "ambition" to improve rapidly. The threshold then rises from this negative value until the next move acceptance occurs. Thus, there is a clear contrast between OBA1 and OBA2, with the latter reflecting a "maximally ambitious" approach.

OBA3. The OBA3 variant (Figure 9) is identical to OBA1, except that we do not explicitly damp the threshold magnitude (i.e., the final threshold value T_M can be far from zero). The notion

[5] In some sense, this is a "milder descent, steeper ascent" strategy; cf. the "steepest descent, mildest ascent" approach proposed by Hansen [12].

Algorithm variant OBA1(M,Δ)

M $\equiv$ limit on number of Rule 1 iterations
Δ $\equiv$ threshold update granularity
$|N|$ $\equiv$ neighborhood size

1. $T_0 = 0$;
2. $age = 0$;
3. Choose (random) initial solution s_0;
4. for $i = 0$ to $M - 1$ do
5. Choose random neighbor solution $s' \in N(s_i)$;
6. if $f(s') < f(s_i) + T_i$ then
7. $s_{i+1} = s'$;
8. $T_{i+1} = T_i - \Delta$;
9. $age = 0$;
10. else
11. $s_{i+1} = s_i$;
12. $T_{i+1} = T_i + \Delta * \frac{age}{|N|}$;
13. $age = age + 1$;
14. endif
15. $T_{i+1} = T_{i+1} * (1 - \frac{i}{M})$
16. endfor.

Figure 7: OBA1 variant: threshold magnitude is damped to zero, $decr(T_i)$ affords linear decrease in threshold value, and $incr(T_i)$ accelerates quadratically with increasing age.

of "appropriate granularity" in the threshold update is retained by the $(1 - \frac{i}{M})$ multiplicative factors in lines 8 and 12.

OBA4. The OBA4 variant (Figure 10) is designed to provide the converse of the OBA1 strategy. OBA4 uses a quadratically growing decrement function $decr(T_i)$ which affords greater ambition, and it uses a linear $incr(T_i)$ function so that expectations do not "dwindle" as rapidly as in OBA1. (Observe that this recalls the "steepest descent, mildest ascent" strategy proposed by Hansen and described in [12].) In lines 9-10 of the algorithm template, we establish the criterion for an "easy" move acceptance, namely, that fewer than $\sqrt{2 * |N|}$ move generations have elapsed since the last acceptance of s' (the quantity $\sqrt{2 * |N|}$ is actually another parameter of the OBA4 variant; note that in the traveling salesman problem that we study below, $\sqrt{2 * |N|}$ is the number of cities in the problem instance).

OBA5. Finally, our OBA5 variant (Figure 11) captures the traditional multistart approach [28] [25] by using a "timeout" parameter. Intuitively, when sufficiently many move generations have been made without finding a strictly improving move, we may assume that we are in a local minimum and that hill-climbing should be initiated. OBA5 uses the neighborhood size as the criterion for "sufficiently many", and uses a quadratically increasing $incr(T_i)$ function in performing the hill-climbing. Note that after a new state is accepted, the threshold is reset to zero.

Algorithm variant OBA2(M,Δ,a,b,c)

M $\equiv$ limit on number of Rule 1 iterations
Δ $\equiv$ threshold update granularity
a $\equiv$ multiplicative factor in growth rate
b $\equiv$ power-law in growth rate
c $\equiv$ damping coefficient for threshold magnitude
$|N|$ $\equiv$ neighborhood size

1. $T_0 = 0$;
2. $age = 0$;
3. Choose (random) initial solution s_0;
4. for $i = 0$ to $M - 1$ do
5. Choose random neighbor solution $s' \in N(s_i)$;
6. if $f(s') < f(s_i) + T_i$ then
7. $s_{i+1} = s'$;
8. $age = 0$;
9. else
10. $s_{i+1} = s_i$;
11. $age = age + 1$
12. endif
13. $T_{i+1} = ((\frac{age}{a*\sqrt{2*|N|}})^b - 1) * (\Delta) * (1 - \frac{i}{M})^c$;
15. endfor.

Figure 8: OBA2 variant, incorporating finer tuning of *incr* threshold update function, along with a maximally ambitious *decr* threshold update strategy.

Finally, we also report results for additional variants OBA1_N, OBA2_N, ..., OBA5_N. Each OBAx_N result is obtained by executing the OBAx algorithm, with the only difference being that the Rule 2 acceptance criterion treats negative threshold values $T_i < 0$ as if they are equal to zero. This affords some indication as to whether "ambition" is practically useful, the example of Figure 5 notwithstanding. Since the OBA5 algorithm can never have $T_i < 0$, there is no difference between OBA5 and OBA5_N and we report results for OBA5 only.

We close this section with a depiction of the qualitative differences in threshold schedules for the four variants of OBA1 and OBA2. The differing behaviors of these algorithms are shown in Figures 12 and 13. Figure 12 shows the qualitative nature of 400,000-step sequences for each of the four algorithms, executed on a single random 50-city Euclidean planar TSP instance. The threshold sequences are superposed against the linearly decreasing TA threshold sequence and sampled at every 400 time steps. Figure 13 shows a detailed plot of the 500-step threshold sequence from $i = 100,000$ to $i = 100,500$ for each of the same runs. The threshold sequences for OBA3, OBA4 and OBA5 similar reflect their respective motivations.

Algorithm variant OBA3(M,Δ)

M $\equiv$ limit on number of Rule 1 iterations
Δ $\equiv$ threshold update granularity
$|N|$ $\equiv$ neighborhood size

1. $T_0 = 0$;
2. $age = 0$;
3. Choose (random) initial solution s_0;
4. for $i = 0$ to $M - 1$ do
5. Choose random neighbor solution $s' \in N(s_i)$;
6. if $f(s') < f(s_i) + T_i$ then
7. $s_{i+1} = s'$;
8. $T_{i+1} = T_i - \Delta * (1 - \frac{i}{M})$;
9. $age = 0$;
10. else
11. $s_{i+1} = s_i$;
12. $T_{i+1} = T_i + \Delta * \frac{age}{|N|} * (1 - \frac{i}{M})$;
13. $age = age + 1$;
14. endif
16. endfor.

Figure 9: OBA3 variant: As in OBA1, $decr(T_i)$ is "linear" and $incr(T_i)$ accelerates quadratically with increasing age. However, the threshold is not damped to zero; rather, granularity of the updates is shifted by lines 8 and 12 of the algorithm.

3 Experimental Results

We tested OBA1 - OBA5, their OBAx_N variants, and TA on instances of the traveling salesman problem (TSP). The TSP is a well-studied NP-hard problem as well as a historically ubiquitous testbed for both SA and TA. Our experimental protocol was as follows:

1. Two classes of TSP instances were used: (i) *Euclidean* planar instances corresponding to random pointsets drawn from a uniform distribution in the Euclidean unit square; and (ii) *random* instances with symmetric distance matrices, i.e., each intercity distance drawn from a uniform distribution in $[0, 1]$. These are the two most commonly treated classes of TSP, and in some sense represent limiting cases with respect to "metricity" of the TSP instance.[6]

2. Instance sizes ranged from 50 to 200; for these sizes, we considered CPU limits of between 0.4×10^6 and 1.0×10^6 applications of Rule 1, following the studies of Rossier et al. [27].

3. Our Rule 1 corresponds to the popular Lin 2-opt neighborhood structure [22], wherein a neighbor solution s' is generated by deleting a random pair of edges in s_i and then reconnecting

[6]Note: we also studied a class of *hierarchical* TSP instances, which have a clustered, non-uniform distribution of points in the Euclidean plane. The results for these instances were qualitatively similar to those for Euclidean instances which had n equal to the number of clusters in the hierarchical instances.

Algorithm variant OBA4(M,Δ)

M $\equiv$ limit on number of Rule 1 iterations
Δ $\equiv$ threshold update granularity
$|N|$ $\equiv$ neighborhood size
count $\equiv$ number of consecutive "easy" move acceptances

1. $T_0 = 0$;
2. $count = 1$;
3. Choose (random) initial solution s_0;
4. for $i = 0$ to $M - 1$ do
5. Choose random neighbor solution $s' \in N(s_i)$;
6. if $f(s') < f(s_i) + T_i$ then
7. $s_{i+1} = s'$;
8. $age = 0$;
9. if $prev\_age < \sqrt{2 * |N|}$ then
10. $count = count + 1$;
11. else
12. $count = 1$;
14. endif
13. $T_{i+1} = T_i - count * \Delta * (1 - \frac{i}{M})$;
15. else
16. $s_{i+1} = s_i$;
17. $age = age + 1$;
18. $T_{i+1} = T_i + \frac{\Delta}{\sqrt{2*|N|}} * (1 - \frac{i}{M})$;
19. endif
20. $prev\_age = age$.
21. endfor.

Figure 10: OBA4 variant, embodying steeper descent and milder ascent, along with criterion for consecutive "easy" move acceptances. Note that OBA4 provides the converse of the OBA1 strategy.

the two paths to achieve the other possible tour. The size of the neighborhood structure is $|N| = \frac{n(n-1)}{2}$, where n is the number of cities in the TSP instance.

4. We report the best solution quality encountered during the execution of the algorithm, i.e., the minimum value among $f(s_0), f(s_1), \ldots, f(s_M)$, as indicated in line 8 of the template in Figure 3. The solution quality is normalized to the Held-Karp one-tree lower bound on optimal TSP tour cost [13]. This lower bound is given by the total edge length of a minimum spanning tree over the n cities plus the n^{th}-smallest edge length among the points of the instance.

Tables 1 - 6 compare all of the OBA variants against TA for Euclidean and random instances of size 50 ($M = 0.4 \times 10^6$), 100 ($M = 0.7 \times 10^6$) and 200 ($M = 1.0 \times 10^6$). These are exactly the same time bounds used by Rossier et al. ([27], p. 162, Table 4) in their studies of the SA algorithm. In the tables, we measure the relative performance of the algorithms versus TA at intervals of $M/5$ move generations; $f(x)$ denotes the solution quality at step x, normalized to the TA solution quality. Thus

Algorithm variant OBA5(M,Δ)

$M \equiv$ limit on number of Rule 1 iterations
$\Delta \equiv$ threshold update granularity
$|N| \equiv$ neighborhood size

1. $T_0 = 0$;
2. $age = 0$;
3. Choose (random) initial solution s_0;
4. for $i = 0$ to $M - 1$ do
5. Choose random neighbor solution $s' \in N(s_i)$;
6. if $f(s') < f(s_i) + T_i$ then
7. $s_{i+1} = s'$;
9. $age = 0$;
10. else
11. $s_{i+1} = s_i$;
13. $age = age + 1$;
14. endif
15. if $age > |N|$ then
16. $T_{i+1} = T_i + \frac{\Delta}{2*\sqrt{|N|}}$;
17. else
18. $T_{i+1} = 0$;
19. endif
20. endfor.

Figure 11: OBA5 variant, which emulates traditional multistart heuristics. The "timeout" parameter is equal to the neighborhood size, and a quadratic $incr(T_i)$ function is used.

$f(x) \equiv 1.000$ for the TA algorithm. Each of our results represents a geometric average of single runs for each of 100 randomly generated instances. In most cases, the OBA variants find significantly better solutions within the early stages of the optimization. The OBA4 variant seems particularly promising: it uniformly outperforms TA on the random instances and is very competitive, if not better, on the Euclidean instances.

Finally, Figure 14 gives a more detailed portrayal of the OBA1 and OBA2 solution quality versus the TA solution quality; again, we use the geometric average of performance ratio, averaged at each time step (100 Euclidean instances and one run per instance, using $n = 50$). Figure 15 similarly portrays the same OBA variants over random symmetric instances with $n = 50$.

4 Discussion and Conclusions

As a result of our detailed experiments, we believe that there is a strong case for non-monotone threshold and temperature schedules in hill-climbing approaches to global optimization. We also believe that the OBA paradigm provides a powerful and general template for exploration of such non-

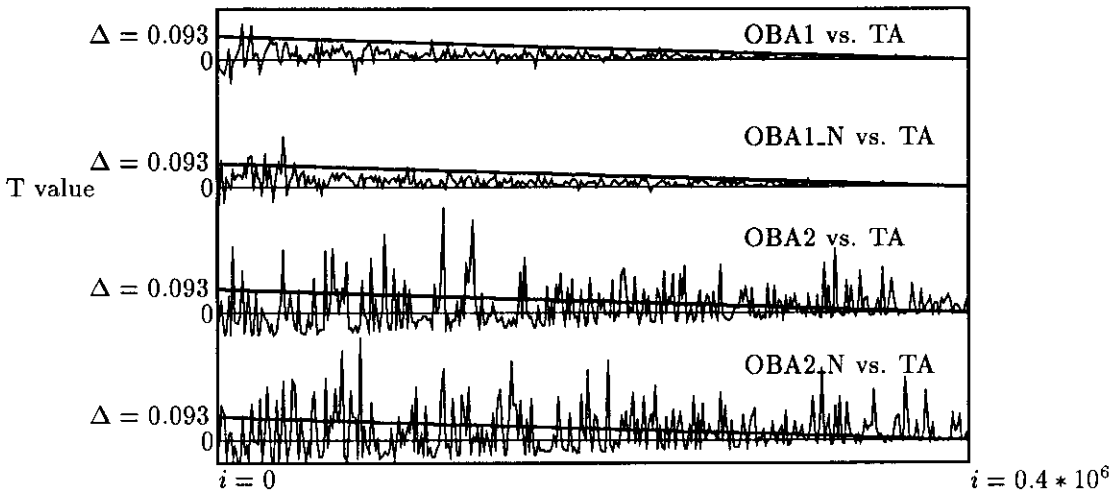

Figure 12: Sampled threshold sequences for four OBA variants on random Euclidean planar TSP instance with $n = 50$. The sampling interval is every 400 steps, and the trivial TA sequence is shown by dashed lines. Specific parameterizations: $M = 0.4 \times 10^6$, $\Delta = 0.093244$, $a = 1$, $b = 2$ and $c = 0.5$.

Alg.	f(i)	f(2i)	f(3i)	f(4i)	f(5i)
TA	1.000	1.000	1.000	1.000	1.000
OBA1	0.974	1.010	1.025	1.028	1.027
OBA1_N	0.969	1.005	1.021	1.024	1.024
OBA2	1.002	1.016	1.017	1.007	1.004
OBA2_N	0.962	0.987	0.999	1.000	0.998
OBA3	0.971	0.996	1.008	1.009	1.007
OBA3_N	0.959	0.988	1.001	1.003	1.002
OBA4	0.971	0.995	1.004	1.004	1.002
OBA4_N*	0.958	0.986	0.999	1.001	1.000
OBA5	1.010	1.040	1.052	1.052	1.047

Table 1: Results for n50_400k (average taken over 100 TSP instances ($n = 50$) with Euclidean distance matrix, $M = 0.4 \times 10^6$.

monotone heuristics. Since the OBA variants (particularly OBA4_N) perform well on both random and Euclidean TSP instances, which are extremal with respect to "geometricness" or "metricity" of symmetric TSP instances, we conclude that the variants reported here are fairly robust. The poor performance of OBA5 may indicate a weakness in traditional multistart approaches, while

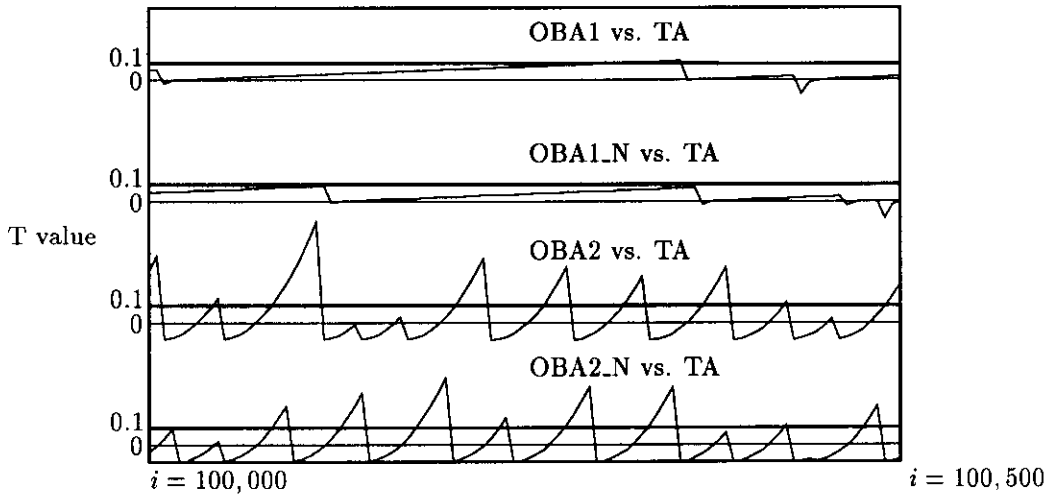

Figure 13: Detailed 500-step intervals, $i = 100,000$ to $i = 100,500$, taken from each of the above runs. The non-monotone nature of the OBA threshold sequences is clearly shown (again, relevant portion of trivial TA sequence shown by dashed lines.

Alg.	f(i)	f(2i)	f(3i)	f(4i)	f(5i)
OBA1	0.986	1.046	1.073	1.078	1.072
OBA1_N	0.969	1.030	1.058	1.068	1.070
OBA2	1.001	1.025	1.033	1.029	1.023
OBA2_N	0.947	0.991	1.012	1.017	1.017
OBA3	0.960	1.005	1.026	1.033	1.033
OBA3_N	0.942	0.993	1.017	1.025	1.026
OBA4	0.968	1.003	1.016	1.018	1.016
OBA4_N*	0.939	0.986	1.006	1.013	1.012
OBA5	0.982	1.043	1.070	1.079	1.080

Table 2: Results for n100_700k (average taken over 100 TSP instances ($n = 100$) with Euclidean distance matrix, $M = 0.7 \times 10^6$.

the excellent performance of OBA4 may suggest revisiting the "steepest descent, mildest ascent" strategy proposed by Hansen [12] (see also the discussions in [3] [14]). Beyond these implications, the most far-reaching consequence of this research seems to stem from the motivations of Section 2.1 – specifically, the "BSFE" studies described for the cost surfaces of Figures 4 and 5. Our related work in this area is reported in [4].

Alg.	f(i)	f(2i)	f(3i)	f(4i)	f(5i)
OBA1	1.004	1.066	1.093	1.093	1.082
OBA1_N	0.946	1.018	1.064	1.081	1.086
OBA2	1.010	1.044	1.060	1.055	1.048
OBA2_N	0.942	0.997	1.033	1.044	1.047
OBA3	0.971	1.007	1.041	1.053	1.055
OBA3_N	0.936	0.991	1.030	1.044	1.048
OBA4	1.103	1.017	1.032	1.036	1.030
OBA4_N*	0.931	0.981	1.014	1.025	1.026
OBA5	0.948	1.020	1.066	1.083	1.088

Table 3: Results for n200_1000k (average taken over 100 TSP instances ($n = 200$) with Euclidean distance matrix, $M = 1.0 \times 10^6$.

Alg.	f(i)	f(2i)	f(3i)	f(4i)	f(5i)
TA	1.000	1.000	1.000	1.000	1.000
OBA1	0.883	0.881	0.881	0.881	0.881
OBA1_N	0.885	0.884	0.884	0.884	0.884
OBA2	0.966	0.928	0.908	0.893	0.889
OBA2_N	0.941	0.915	0.899	0.893	0.887
OBA3	1.090	1.073	1.065	1.061	1.058
OBA3_N	1.098	1.078	1.069	1.067	1.061
OBA4	0.925	0.895	0.884	0.876	0.869
OBA4_N*	0.914	0.889	0.879	0.871	0.867
OBA5	1.119	1.110	1.100	1.090	1.081

Table 4: Results for r50_400k (average taken over 100 TSP instances ($n = 50$) with random symmetric distance matrix, $M = 0.4 \times 10^6$.

We conclude by listing some additional, ongoing research directions.

Alg.	f(i)	f(2i)	f(3i)	f(4i)	f(5i)
OBA1	0.968	0.966	0.967	0.967	0.967
OBA1_N	0.968	0.969	0.970	0.970	0.970
OBA2	1.055	0.998	0.967	0.952	0.945
OBA2_N	1.011	0.971	0.949	0.939	0.935
OBA3	1.249	1.230	1.223	1.218	1.214
OBA3_N	1.251	1.228	1.223	1.214	1.209
OBA4	0.985	0.943	0.926	0.920	0.914
OBA4_N*	0.978	0.937	0.922	0.916	0.911
OBA5	1.097	1.100	1.098	1.097	1.095

Table 5: Results for r100_700k (average taken over 100 TSP instances ($n = 100$) with random symmetric distance matrix, $M = 0.7 \times 10^6$.

Alg.	f(i)	f(2i)	f(3i)	f(4i)	f(5i)
OBA1	1.039	1.060	1.064	1.065	1.063
OBA1_N	1.024	1.054	1.058	1.059	1.059
OBA2	1.066	1.075	1.055	1.041	1.030
OBA2_N	1.026	1.034	1.023	1.014	1.009
OBA3	1.351	1.381	1.374	1.365	1.359
OBA3_N	1.340	1.369	1.367	1.361	1.356
OBA4	1.131	1.031	1.005	0.983	0.972
OBA4_N*	1.035	1.017	0.990	0.976	0.968
OBA5	1.027	1.057	1.062	1.063	1.062

Table 6: Results for r200_1000k (average taken over 100 TSP instances ($n = 200$) with random symmetric distance matrix, $M = 1.0 \times 10^6$.

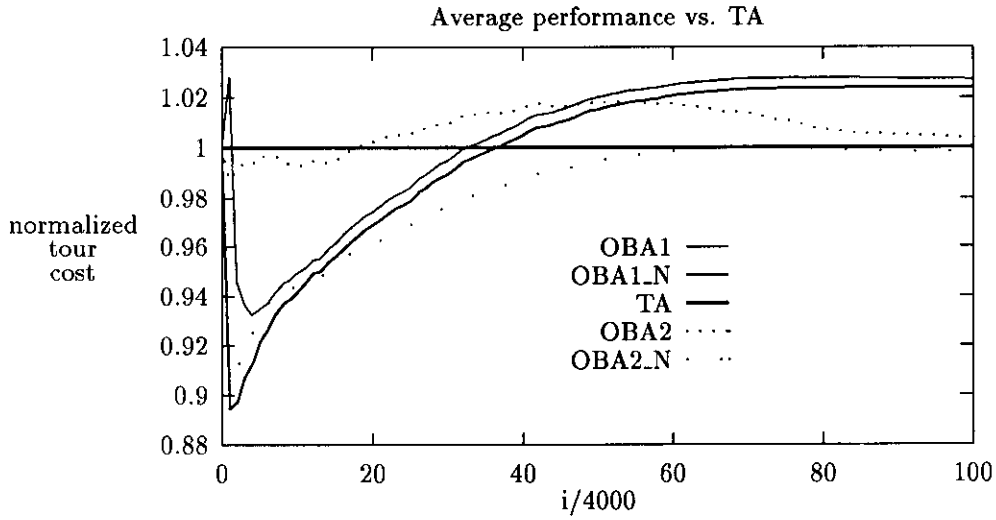

Figure 14: Ratios of OBA solution quality to TA solution quality, geometrically averaged at each time step over 100 instances.

1. Extension of the non-monotone OBA approaches to simulated annealing (temperature T) instead of threshold accepting (threshold T).

2. Further tests of the robustness of OBA, e.g., on asymmetric TSP instances.

3. Examination of the conjecture that OBAx_N will always outperform the OBAx variant for "real" cost surfaces, the example of Figure 5 notwithstanding.

4. Further investigation of the OBA4_N strategy, since it is clearly the best among the OBA variants we report here.

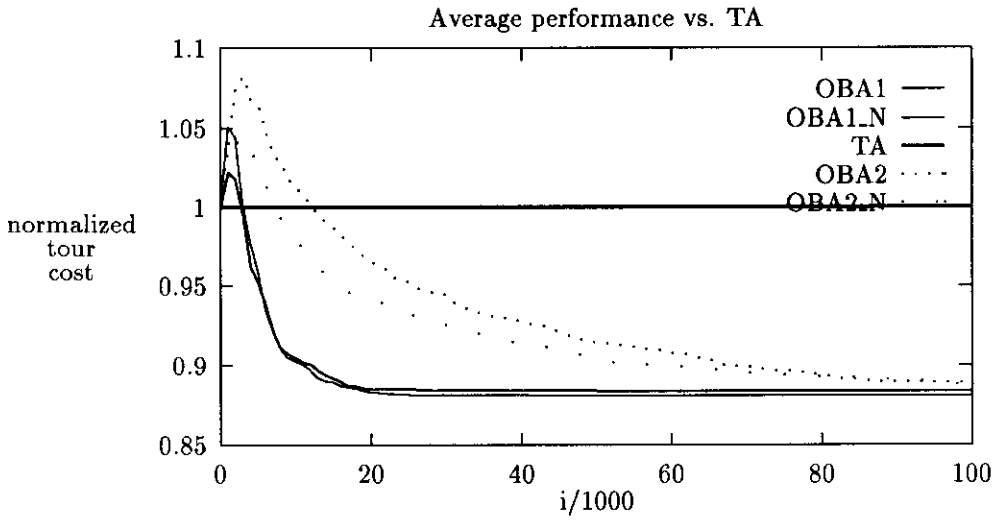

Figure 15: Ratios of OBA solution quality to TA solution quality, geometrically averaged at each time step over 100 instances.

5. Further investigation of the practical significance of "update granularity" and the damping of threshold values to zero, e.g., as in the contrast between OBA1 and OBA3.

6. Extension of the hill-climbing optimization template to include non-degenerate history (i.e., memory) in the Rule 1 generation of s'. We believe that this, in combination with our existing OBA template, would provide a very powerful characterization of available hill-climbing approaches.

7. Finally, we are investigating the performance of the time-bounded OBA strategy on much larger combinatorial instances, with the goal of deriving natural relationships between the parameters of the OBA variants, the available CPU limit M, and the size n of the problem instance.

References

[1] E. H. L. Aarts and J. Korst. *"Simulated Annealing and Boltzmann Machines: a Stochastic Approach to Combinatorial Optimization and Neural Computing"*. Wiley, 1989.

[2] I. Althofer and K. U. Koschnick. "On the Convergence of Threshold Accepting". *Applied Mathematics and Optimization*, 24:183–195, 1991.

[3] E. B. Baum. "Iterated Descent: a Better Algorithm for Local Search in Combinatorial Optimization Problems". Technical Report 164-30, Crellin Laboratory, California Institute of Technology, Pasadena, CA 91125.

[4] K. Boese, A. B. Kahng, and C. W. Tsao. *manuscript*, 1992.

[5] T. N. Bui, S. Chauduri, F. T. Leighton, and M. Sipser. "Graph Bisection Algorithms with Good Average Case Behavior". Combinatorica, 7(2):171–191, 1987.

[6] V. Cerny. "Thermodynamical Approach to the Traveling Salesman Problem: an Efficient Simulation Algorithm". *J. Optimization Theory and Applications*, 45(1):41–51, January 1985.

[7] G. Dueck and T. Scheuer. "Threshold Accepting: a General Purpose Optimization Algorithm Appearing Superior to Simulated Annealing". *Journal Of Computational Physics*, 90:161–175, 1990.

[8] F. Glover. "Tabu Search - part I". *ORSA J. on Computing*, 1:190–206, 1989.

[9] L. Hagen and A. B. Kahng. "New Spectral Methods for Ratio Cut Partitioning and Clustering". *IEEE Trans. on CAD*, pages 1074–1085, September 1992.

[10] B. Hajek. "Cooling Schedules for Optimal Annealing". *Mathematics of Operations Research*, 13:311–329, 1988.

[11] B. Hajek and G. Sasaki. "Simulated Annealing - to Cool or Not". *Systems and Control Letters*, 12:443–447, 1989.

[12] Pierre Hansen and Brigitte Jaumard. "Algorithms for the Maximum Satisfiability Problem". *Computing*, 44:279–303, 1990.

[13] M. Held and R. M. Karp. "The Traveling-Salesman Problem and Minimum Spanning Trees". *Operations Research*, 18:1138–1162, 1970.

[14] D. S. Johnson. "Local Optimization and the Traveling Salesman Problem". In *Proceedings of the 17th International Colloquium on Automata, Languages and Programming*, pages 446–460, England, July 16-20 1990.

[15] D. S. Johnson, C. R. Aragon, L. A. McGeoch, and C. Schevon. "Optimization by Simulated Annealing: an Experimental Evaluation; Part III, The Traveling Salesman Problem". *Operations Research.*

[16] D. S. Johnson, C. R. Aragon, L. A. McGeoch, and C. Schevon. "Optimization by Simulated Annealing: an Experimental Evaluation; Part I, Graph Partitioning". *Operations Research*, 37(6):865–892, November-December 1989.

[17] D. S. Johnson, C. R. Aragon, L. A. McGeoch, and C. Schevon. "Optimization by Simulated Annealing: an Experimental Evaluation; Part II, Graph Coloring and Number Partioning". *Operations Research*, 39(3):378–406, May-June 1991.

[18] A. B. Kahng. "Exploiting Fractalness in Error Surfaces: New Methods for Neural Network Learning". In *Proc. IEEE Intl. Symp. on Circuits and Systems*, pages 41–44, San Diego, May 1992.

[19] A. B. Kahng. "Random Structure of Error Surfaces: New Stochastic Learning Methods". In *Proc. SPIE Conf. on Neural Networks and Optimization* , Orlando, April 1992. invited paper.

[20] A. B. Kahng and G. Robins. "On Structure and Randomness in Practical Optimization". In *UCLA Computer Science Department 1990-1991 Annual*, pages 23–38. UCLA Computer Science Department, 1990.

[21] S. Kauffman and S. Levin. "Towards a General Theory of Adaptive Walks on Rugged Landscapes". *Journal of Theoretical Biology*, 128:11–45, 1987.

[22] B. Kernighan and S. Lin. "An Efficient Heuristic Procedure for Partitioning Graphs". *The Bell System Tech. Journal*, 49(2):671–680, May 1983.

[23] S. Kirkpatrick, Jr. C. D. Gelatt, and M. Vecchi. "Optimization by Simulated Annealing". *Science*, 220(4598):671–680, May 1983.

[24] S. Kirkpatrick and G. Toulouse. "Configuration Space Analysis of Traveling Salesman Problems". *Journal de Physique*, 46:1277–1292, 1985.

[25] J. B. Lasserre, P. P. Varaiya, and J. Walrand. "Simulated Annealing, Random Search, Multi-Start or SAD?". *Systems and Control Letters*, 8:297–301, 1987.

[26] J. Pearl. *"Heuristics: Intelligent Search Strategies for Computer Problem Solving"*. Addison-Wesley, Reading, MA, 1984.

[27] Y. Rossier, M. Troyon, and T. M. Liebling. "Probabilistic Exchange Algorithms and Euclidean Traveling Salesman Problems". *OR Spektrum*, 8(3):151–164, 1986.

[28] F. Schoen. "Stochastic Techniques for Global Optimization: A Survey of Recent Advances". *J. Global Optimization*, pages 207–228, 1991.

[29] G. Sorkin. "Efficient Simulated Annealing on Fractal Energy Landscapes". *Algorithmica*, 6:367–418, 1991.

[30] P. Strenski and S. Kirkpatrick. "Analysis of Finite Length Annealing Schedules". *Algorithmica*, pages 346–366, 1991.