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

ON THE "BIG VALLEY" AND ADAPTIVE MULTI-START FOR DISCRETE GLOBAL OPTIMIZATIONS

K. D. Boese
A. B. Kahng
S. Muddu

July 1993
CSD-930015

On the "Big Valley" and Adaptive Multi-Start for Discrete Global Optimizations*

Kenneth D. Boese, Andrew B. Kahng and Sudhakar Muddu

UCLA Computer Science Dept., Los Angeles, CA 90024-1596 USA

Abstract

We analyze relationships among local minima for the traveling salesman and graph bisection problems under standard neighborhood structures. Our work reveals surprising correlations that suggest a *globally convex*, or "big valley" structure in these optimization cost surfaces. In conjunction with combinatorial results that sharpen previous analyses, our analysis directly motivates a new *adaptive multi-start* paradigm for heuristic global optimization, wherein starting points for greedy descent are adaptively derived from the best previously-found local minima. We test a simple instance of this method and obtain very significant speedups over previous multi-start implementations.

Keywords: Heuristic global optimization, stochastic hill-climbing, multi-start, traveling salesman problem, graph bisection.

1 Introduction

A combinatorial problem has a finite solution set S and a real-valued cost function $f : S \rightarrow \Re$. Global optimization seeks a solution $s^* \in S$ with $f(s^*) \leq f(s') \; \forall s' \in S$. Because many formulations are intractable, heuristic methods are employed, which can often be described by the following template:

```
Iterative Global Optimization

for (i = 0; ; i++)
   Step 1: Given the current solution $s_i$, generate a new trial solution $s'$
   Step 2: Decide whether to set $s_{i+1} = s_i$ or $s_{i+1} = s'$
   (When stopping condition is satisfied, Return best solution found)
```

Typically, s' is a slight perturbation of s_i, i.e., $s' \in N(s_i)$ where $N(s_i)$ is the *neighborhood*, or set of all possible "neighbor" solutions, of s_i. The function f then defines a *cost surface* over the neighborhood topology.

This template is quite general. For example, *simulated annealing* [13] generates a random $s' \in N(s_i)$ in Step 1, and in Step 2 sets $s_{i+1} = s'$ with probability one if $f(s') \leq f(s_i)$, and probability $exp((f(s_i) - f(s'))/T_i)$ if $f(s') > f(s_i)$, where T_i is the "temperature" parameter at the i^{th} step. Other heuristics are *greedy*, with $s_{i+1} = s'$ in Step 2 if and only if $f(s') < f(s_i)$. Of specific interest to us is the non-deterministic **Greedy_Descent** procedure, which iteratively tests solutions $s' \in N(s_i)$ in *random* order until an improvement $s_{i+1} = s'$ with

*Partial support for this work was provided by ARO DAAK-70-92-K-0001, ARO DAAL-03-92-G-0050, NSF MIP-9110696, and NSF Young Investigator Award MIP-9257982.

$f(s_{i+1}) < f(s_i)$ can be made; the procedure terminates when no improving $s' \in N(s_i)$ exists. The well-known weakness of greedy search is that progress stops when the first local minimum is encountered. Simulated annealing can escape from local minima and has gained wide popularity because it yields superior solution quality, and because it is theoretically guaranteed to return a global optimum solution in the limit of infinite time. On the other hand, successful annealing generally requires very large amounts of CPU.

Because greed returns a good solution relatively quickly, one alternative to simulated annealing is to apply greed repeatedly and return the best result. Indeed, several studies have shown "greedy multi-start" superior to simulated annealing in terms of both solution quality and runtime. Johnson [9] describes extensive empirical studies of the traveling salesman problem (TSP) and indicates that multiple runs of various greedy methods can outperform simulated annealing. Recent results of Sorkin [21] show that a multi-start approach is superior to standard simulated annealing on a class of fractal cost surfaces. Boese and Kahng [3] have computed *optimal* annealing temperature schedules for small combinatorial problems; these schedules can resemble multi-start, with alternating periods of greedy descent and randomization (corresponding to annealing at zero and infinite temperatures). Multi-start is also attractive for its trivial parallelizability.

Nonetheless, the multi-start approach has its weaknesses. For such problems as graph partitioning, the number of greedy descents needed to achieve a stable methodology (i.e., having predictable solution quality) grows rapidly with problem size [10] [23]. Recent analyses of cost surfaces show that as problems grow large, random local minima are almost surely of "average" quality, implying that current **random** multi-start heuristics which rely on random starting solutions are doomed to a "central limit catastrophe".[1]

2 Global Structure of Optimization Cost Surfaces

Our motivating hypothesis is that multi-start heuristics can remain successful for large problem instances only by exploiting *global structure* in the cost surface. Several general structural models have been proposed, e.g., Sorkin [20] and Weinberger [24] have fitted fractal and AR(1) processes to real-world optimization cost surfaces. For our purposes, the leading study is due to Kirkpatrick and Toulouse [14], who attempt to confirm an ultrametric relationship between local minima of TSP instances. These authors find that the distances between random pairs of local minima satisfy a normal distribution with surprisingly low average. Kirkpatrick and Toulouse find only inconclusive evidence for ultrametricity. Related studies are due to Mezard and Parisi [17] and Sourlas [22]; the latter fails to find evidence for ultrametricity in the TSP, and goes on to propose a modified simulated annealing heuristic which eliminates edges from consideration if they appear infrequently in good solutions. Similar studies of ultrametricity have been made for other combinatorial problems such as graph coloring [1] and one-dimensional circuit placement [19]. Note that [22] and [17] both discuss correlations of TSP tour costs with distances between tours, much as we do in Subsection 2.1 below; however, they fail to

[1] See, e.g., discussions by Baum [2] on traveling salesman structures and by Kauffman and Levin [12] on "adaptive landscapes" in evolutionary optimization.

make any of the enabling observations that we present here.

In this work, we also study relationships among local minima, but in a novel way: we consider the set of local optima *from the perspective of the best local optimum.* As we describe in the remainder of this section, our results indicate that many solution spaces exhibit a "globally convex" [7] or what we call a ***big valley*** structure. Figure 1 gives an intuitive picture of the big valley, in which the set of local minima appears convex with one central global minimum. This general structure suggests improved multi-start strategies that derive starting points from the best previously-found local minima. Section 3 will describe our initial investigations of this new class of *adaptive multi-start* methods.

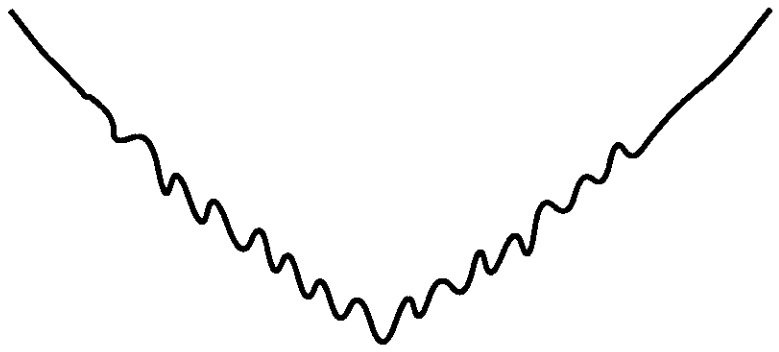


Figure 1: Intuitive picture of the "big valley" solution space structure.

2.1 The Symmetric Traveling Salesman Problem

The symmetric TSP is perhaps the most well-studied of all NP-hard combinatorial problems [16]. Given n cities with symmetric intercity distances, the TSP seeks a minimum-cost *tour*, i.e., a (cyclic) permutation of the cities which minimizes the sum of the n distances between adjacent city pairs of the tour. A given TSP instance is represented by a complete weighted graph $G = (V, E)$ with $|V| = n$ and $|E| = \frac{n(n-1)}{2}$, so that any tour is a Hamiltonian circuit in G. We use the Lin 2-opt neighborhood operator that is usual in studies of the TSP [16]: a 2-opt deletes two non-adjacent edges of the current tour and then reconnects the two resulting paths into a new tour.

To study the structure of the TSP solution space, we require a measure of distance between two tours t_1 and t_2. A natural definition of distance is the minimum number of 2-opts needed to transform t_1 into t_2; we call this the *2-opt distance*, denoted $d(t_1, t_2)$. Since no polynomial method for computing $d(t_1, t_2)$ is known, Kirkpatrick and Toulouse [14] study a distance measure based on the number of edges, or ***bonds***, common to both t_1 and t_2. This *bond distance*, denoted $b(t_1, t_2)$, is equal to n minus the number of edges that are present in both t_1 and t_2 (disregarding edge direction). No previous results link bond distance directly to the 2-opt or any other TSP neighborhood structure. However, we have partially addressed this gap through the following theorem (see Appendix 1 for proof), which supports the existing practice of measuring bond distance even in a 2-opt neighborhood structure.

Theorem 1. For any two tours t_1 and t_2 of a given TSP instance, $\frac{b(t_1,t_2)}{2} \leq d(t_1,t_2) \leq b(t_1,t_2)$. $\square$

Recall that our new approach to multi-start will be motivated by our novel examination of the set of local optima, from the perspective of the *best* local optimum. For each of 2,500 unique random local minimum tours in a 100-city Euclidean TSP instance, Figure 2(a) plots the tour's cost versus its average bond distance to all (2,499) other local minima. [2] There is a clear correlation: the best local minimum appears to be "central" to all other local minima, and indeed a "big valley" structure can be said to govern the set of locally minimum tours.

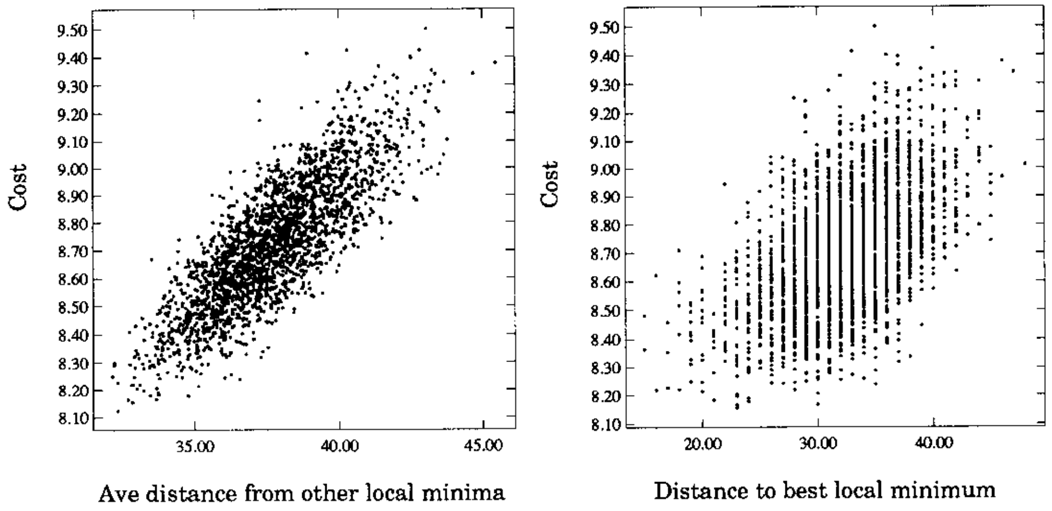


Figure 2: Analysis of 2,500 random local minima for a 100-city 2-D Euclidean TSP instance. Tour cost (vertical axis) is plotted against (a) average distance from the other 2,499 local minima and (b) distance from the local minimum with lowest cost. In (b), we do not show the best local minimum, which is at distance zero.

Further insight is gained from Figure 2(b), which plots the costs of the same 2,500 local optima against their distances from the best local optimum found. Note that all the local optima (2,499 plus the best one) are at bond distance ≤ 48. In Appendix 2 we show that the expected distance between two random n-city tours is just under $n-2$, slightly sharpening an observation in [14]. Appendix 2 also gives the first efficient enumeration of tours at each bond distance; this calculation indicates that less than $1/10^{60}$ of the solution space lies within a "ball" of radius $= 48$. Thus, the set of local minima not only has a "big valley" structure, but is also confined to a tiny portion of the solution space S. (Such intuitions are clearly suggestive vis-a-vis multi-start strategies.)

[2]Our Euclidean TSP instances are chosen randomly from a uniform distribution over the unit square. A "random local minimum" solution is found by starting at a random initial solution and executing procedure **Greedy_Descent**.

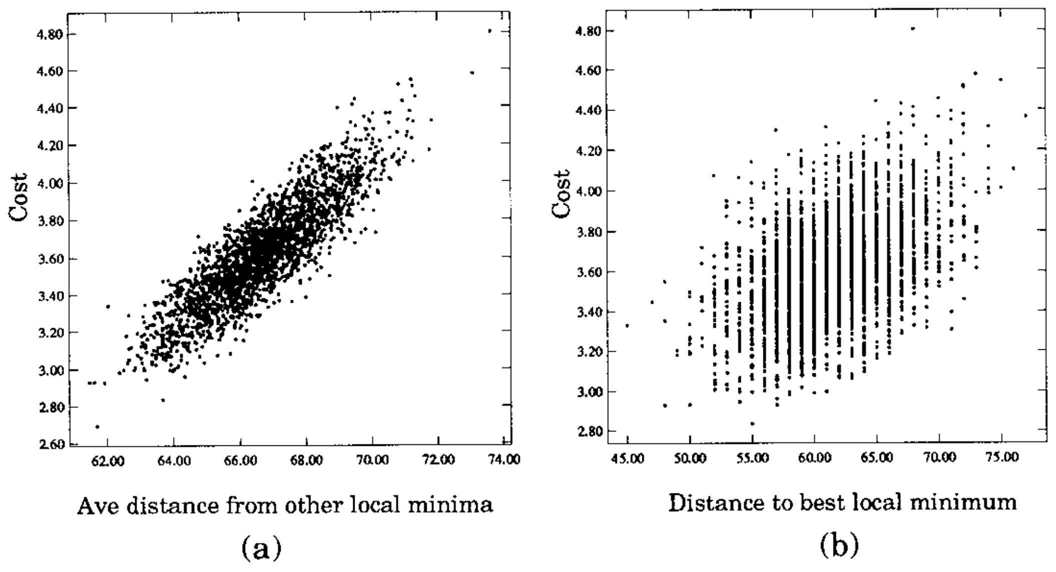


Figure 3: Analysis of 2,500 random local minima for a 100-city random symmetric TSP instance.

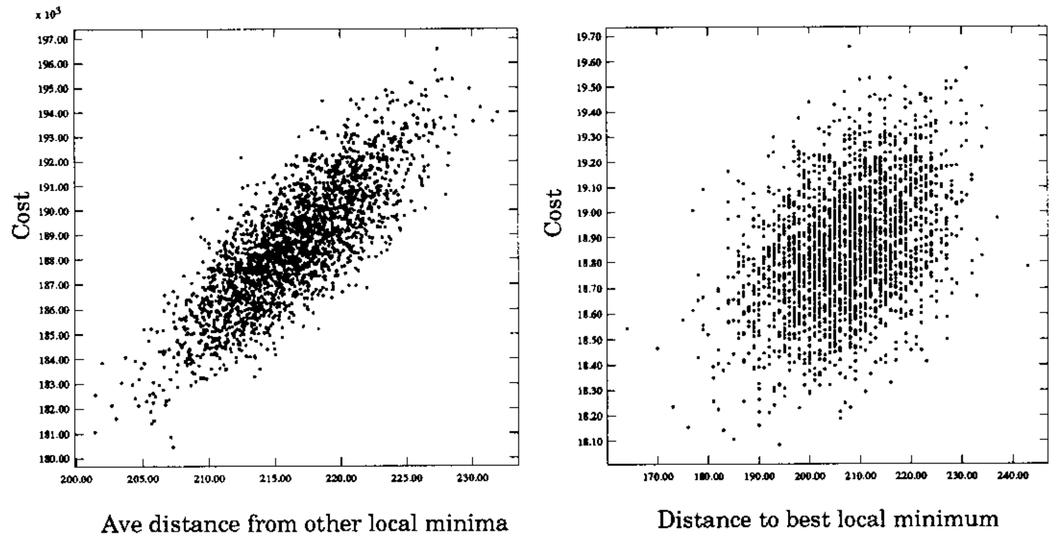


Figure 4: Analysis of 2,500 random local minima from a 500-city Euclidean TSP instance.

In Figure 2 we report data for the Euclidean TSP because its metricity is reflective of many practical TSP instances. Very similar results are shown in Figure 3 for a 100-city instance of the random symmetric TSP[3] which is studied in [14] and elsewhere. Finally, Figure 4 gives analogous plots for a random 500-city Euclidean TSP instance (note that a ball of radius = 245 corresponds to less than $1/10^{463}$ of the solution space).

[3]In the random symmetric TSP, distances between pairs of cities are distributed uniformly and independently over the interval [0,1].

2.2 The Graph Bisection Problem

We have found that a similar structure governs the local minima for ***graph bisection*** instances. Given an unweighted graph $G = (V, E)$, the (unweighted) graph bisection problem is to find a partition of V into disjoint subsets U and W, with $|U| = |W|$, such that number of edges $(u, w) \in E$ with $u \in U$, $w \in W$ is minimized. We adopt the standard 2-interchange neighborhood structure, where a 2-interchange swaps a pair of vertices $u \in U$ and $w \in W$. The distance between solutions s_1 and s_2 is the number of 2-interchanges required to transform s_1 into s_2, and can be at most $|V|/4$.

We study a standard class of random graphs $G(n,p)$, i.e., graphs having n vertices and each edge present independently with probability p [4]. Because graphs in $G(n,p)$ have expected optimum bisection cost within a constant factor of the random bisection cost [5], more "difficult", structured models have been proposed. In particular, we also study the class $G_{Bui}(n,d,b)$, proposed by Bui et al. [5], which have n nodes, are d-regular, and are constructed to have optimum bisection cost almost certainly equal to b. Typical results are shown in Figures 5 and 6. The "big valley" correlations are again clearly apparent, although local minima can be up to the maximum distance away from the best local minimum. Note that the average distance between random bisections is 23.02 for $n = 100$ and 35.04 for $n = 150$.

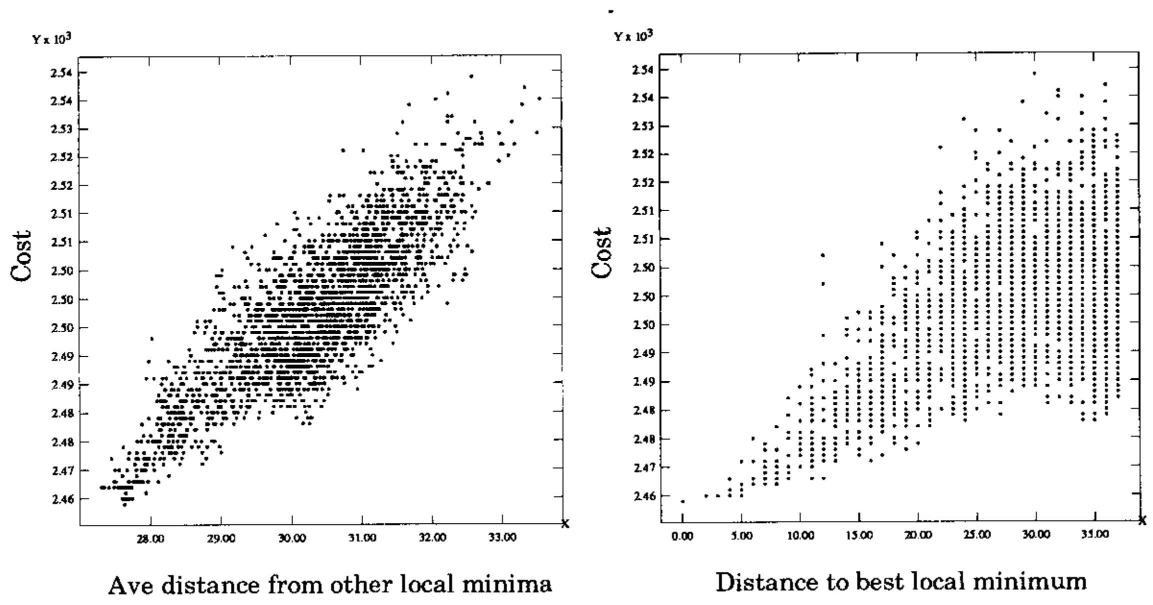
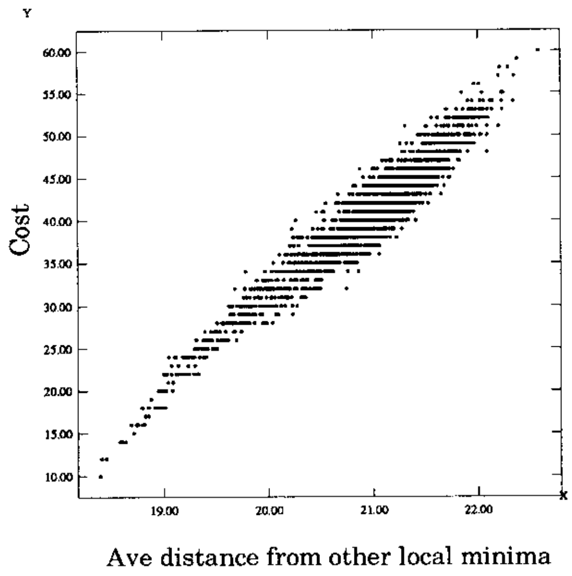


Figure 5: Analysis of 2,500 random local minimum bisections for graph in $G(150, 0.5)$.



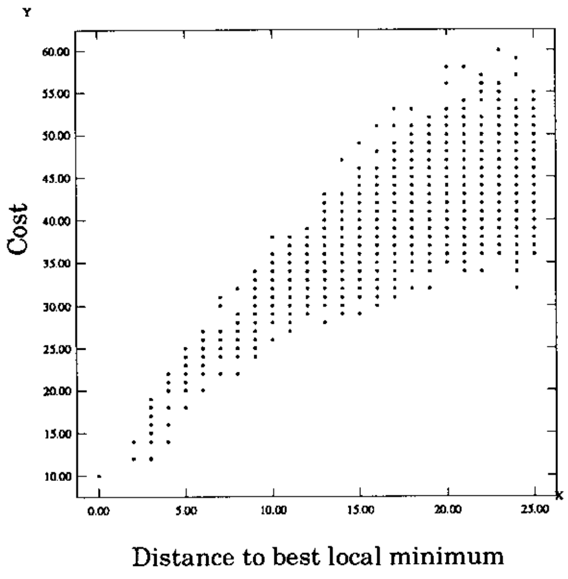


Figure 6: Analysis of 2,500 random local minimum bisections for graph in $G_{Bui}(100, 4, 10)$.

3 Exploiting Global Structure: Adaptive Multi-Start

It is natural to wonder whether more effective starting solutions for **Greedy_Descent** can be derived if we rely on a "big valley" structure holding for the set of local minima. In this section, we consider a simple instance of such an *Adaptive Multi-Start* (AMS) methodology and demonstrate its effectiveness in practice.

3.1 A Simple AMS Heuristic

We have implemented a simple AMS heuristic consisting of two phases:

1. **Phase One:** Generate R random starting solutions and run **Greedy_Descent** from each to determine a set of corresponding *random local minima*.
2. **Phase Two:** Based on the local minima obtained so far, compose *adaptive* starting solutions and run **Greedy_Descent** A times from them to yield corresponding *adaptive local minima*.

Intuitively, the two phases respectively develop, then exploit, a structural picture of the cost surface. Our AMS heuristic is more precisely described in Figure 3.1. (Note that our discussion is henceforth couched with respect to the symmetric TSP.)

In the Figure 3.1 template, the term *descent* denotes a single execution of **Greedy_Descent**. The number of passes through Phase Two (Lines 3-6) is determined by the relationship $passes = \lceil \frac{D-R}{A} \rceil$. In obtaining the results of Section 4, we uniformly use $A = 10$ and $R = D/2$ (i.e., we spend half our CPU budget

```
Adaptive_Multi-Start (G,D,R,k,A)
Input: TSP instance $G = (V, E)$ with $|V| = n$
        $D \equiv$ limit on number of descents $\equiv$ CPU budget
        $R \equiv$ number of descents from random starting tours
        $k \equiv$ number of local minima used to construct each adaptive tour
        $A \equiv$ number of descents made from each adaptive starting tour
Output: $t^* \equiv$ best local minimum tour found after $D$ descents
Local Variables:
        $M \equiv$ set of $k$ best local minimum tours so far
/*   (Phase One)   */
1.  Generate $R$ random local minima
/*   (Phase Two)   */
2.  repeat
3.      Update $M$
4.      $t \leftarrow$ Construct_Adaptive_Starting_Tour($M$)
5.      Run Greedy_Descent($t$) $A$ times
6.      Update $t^*$
7.  until $D$ total descents.
```

```
Subroutine Construct_Adaptive_Starting_Tour(M)
Input: Set $M = \{M_1, \ldots, M_k\}$ of local minimum tours
Output: Set $t$ of edges forming a new starting tour
S1.  In = union of all edges in set of tours $M$
S2.  $t = \emptyset$.
S3.  Assign weight $w(e_i)$ to each edge $e_i \in$ In
S4.  for $e_i \in$ In in order of decreasing $w(e_i)$ do
S5.      if $e_i$ is a valid tour edge with respect to $t$ then
S6.          $t = t \cup \{e_i\}$ with probability $Pr(e_i)$
S7.  while $|t| < n$ (i.e., $t$ is not yet a tour) do
S8.      Add a randomly chosen valid tour edge from $E \setminus t$
```

Figure 7: **Adaptive_Multi-Start** template.

in Phase One); when $D - R$ is not an exact multiple of A we truncate the final pass in Line 5. The subroutine **Construct_Adaptive_Tour** called during Phase Two always constructs an adaptive starting tour from the set $M = \{M_1, \ldots, M_k\}$ of the best k local minima found so far; k is a heuristic parameter set to $k = 10$ in our experiments. To explain **Construct_Adaptive_Tour**, we use *partial tour* to denote a set of edges that is a subset of the edges in some tour. Given a partial tour t, edge e is a *valid tour edge* with respect to t, iff $t \cup \{e\}$ is a partial tour. Our experimental results below were obtained using the following additional specific choices:[4]

1. In Step S4, $w(e_i)$ is the sum of the inverses of the tour costs over those tours in which e_i appears, i.e.,

$$w(e_i) = \sum_{M_j \ni e_i} \frac{1}{cost(M_j)} \tag{1}$$

[4]In Step S4, our choice gives greater weight to edges that are present in many short tours; however, we have obtained similar results using uniform edge weights. In Step S6, while we have also tried other probabilistic weighting methods, our choice allows the probability of e_i being included into t to approach 1 as $w(e_i)$ approaches its maximum possible value. In Steps S7-S8, we again find that a number of other strategies yield very similar results, e.g., adding the shortest valid edge, testing edges according to a fixed order, or following a nearest-neighbor heuristic.

2. In Step S6, we use

$$Pr(e_i) = \exp(-(1 - \frac{w(e_i)}{W})) \qquad (2)$$

where $W = \sum_{j=1}^{|M|} \frac{1}{cost(M_j)}$ is the weight an edge would have it were contained in all tours in M.

3. In Steps S7-S8, we simply insert random valid edges until t becomes a tour.

Given these implementation decisions, **Construct_Adaptive_Tour** has $O(nk\log(n))$ worst-case runtime.[5] Of course, this is dominated by the known exponential worst-case runtime of **Greedy_Descent** for 2-opt in the TSP [18]. We also observe that our particular choice of Phase Two, with its multiple passes, is reminiscent of a genetic algorithm [8].

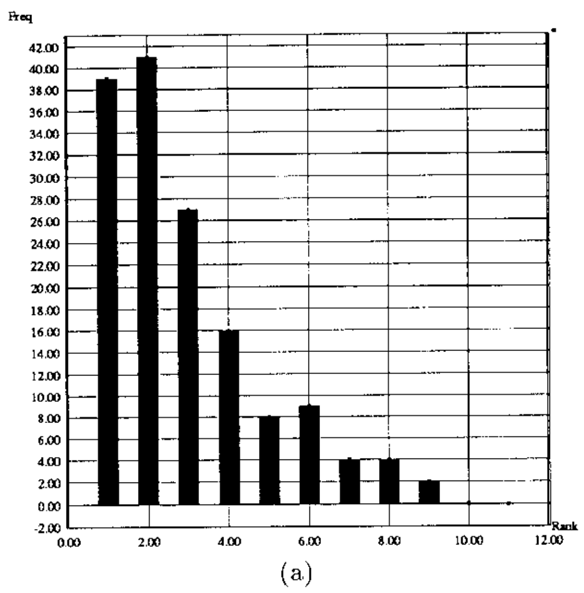
3.2 Experimental Results

Figure 8 gives some intuition into why the AMS heuristic is promising: it shows the distribution of the rank of an eleventh descent compared to ten previous random descents when the eleventh descent is (a) an adaptive descent based on the previous ten descents and (b) another random descent. The adaptive local minimum is not always superior to the minima on which it is based; however, its distribution is much better than a random descent, providing a clear improvement upon the use of multiple random descents.

Table 1 compares results of the AMS implementation above with results of Random multi-start and Nearest Neighbor (NN) multi-start (i.e., multi-start from initial tours obtained by the nearest-neighbor TSP heuristic; note that this is the method suggested in [9] for obtaining initial tours for 2-opt descents). As problem size and the CPU budgets D increase, we obtain significant improvements over the current methods, e.g., for 100-city TSP instances our adaptive strategy achieves in only 200 descents what random multi-start would require over 5,700 descents to achieve. A more detailed portrait of this same comparison data is given in Figure 9.

We emphasize that the stability of our method is very good: for a single 100-city instance, 50 separate executions of the AMS heuristic with D = 100 yielded mean $cost(t^*)$ = 7.4207, with standard deviation = .03329. For a single 50-city instance, 50 separate executions of the AMS heuristic with D = 100 yielded mean $cost(t^*)$ = 5.3840, with standard deviation = .00277.

[5] To analyze its time complexity, **Construct_Adaptive_Tour** can be divided into four parts: i) assigning weights in Step S3; ii) sorting the edges by weight for Step S4; iii) checking whether edges are valid for inclusion in t in Step S5; iv) adding valid edges in Step S8. i) Weights can be assigned in $O(kn)$. ii) The $O(nk)$ edges in all k locally minimum tours can be sorted by weight in $O(nk\log n)$ time. (We assume that $k = O(n)$.) iii) We note that an edge (i, j) can be added to t if and only if its endpoints i and j each have degree less than two and are each contained in a different connected components of t; i.e., adding edge (i, j) will not give us a city of degree > 2 or a tour subcycle. (An exception occurs when the partial tour has $n - 1$ edges and the last edge is forced.) We can determine whether the kn candidate edges are valid for inclusion in t in $O(nk + n\log n)$ time by using the weighted union heuristic described in ?? to test whether the two cities in an edge are contained in the same connected component of t. iv) Finally, we can fill out the tour in Step S8 using $O(n\log n)$ time by starting from an arbitrary degree-1 city and continuing the tour from that city, adding a random valid city at each step when needed. $O(n)$ new cities will be added with a cost of $O(\log n)$ to maintain the list of valid cities each time a new city is added to the partial tour. Thus, sorting the edge for Step 4 dominates the time complexity of **Construct_Adaptive_Tour** giving it an overall complexity of $O(nk\log n)$.



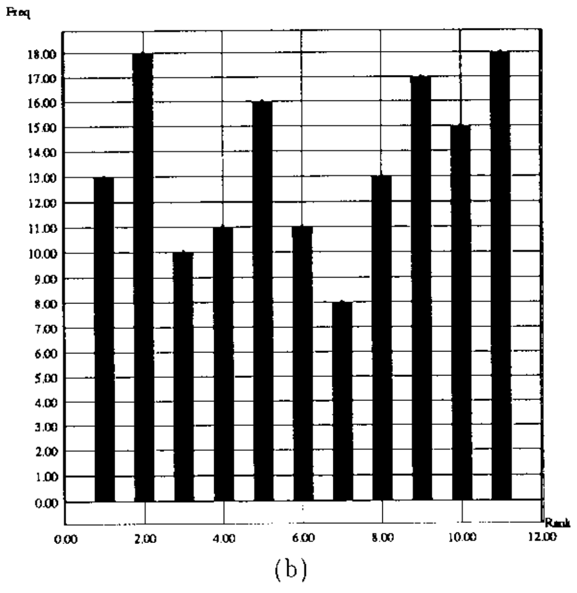


Figure 8: For 50-city Euclidean TSPs, distribution of the rank of an eleventh local minimum compared with respect to ten previous random local minima when the eleventh tour is (a) adaptive (average rank = 2.92) and (b) random (average rank = 6.19; expected rank = 6.00). Each distribution is calculated using eleven descents on each of 150 different randomly generated TSP instances.

# Descents used in AMS	Equivalent # Descents		Average Soln. Cost (AMS)	% Above Held-Karp Lower Bound
	Random strategy	NN strategy		
1	1	1	8.5404	10.91
50	442	50	7.9835	3.68
100	1508	176	7.9266	2.94
150	> 4500	426	7.8967	2.55
200	> 5700	826	7.8806	2.34
400	> 7800	> 3000	7.8555	2.02
600	> 9100	> 4000	7.8367	1.78
1000	> 10000	> 4500	7.8275	1.66

Table 1: Sample experimental results for AMS implementation on 100-city Euclidean TSP instances. Results are averaged over 50 separate random instances in the unit square, and show the number of descents needed for Random multi-start or Nearest-Neighbor-based multi-start to achieve the same solution quality. Random multi-start was run for 3,000 descents and Nearest-Neighbor multi-start for 2,000 descents on each of the 50 instances. Entries with ">" are conservative estimates based on linear extrapolation. Absolute tour costs and relationship to Held-Karp lower bound on TSP cost (provided by D. S. Johnson [11]) are shown to facilitate comparison with other work, e.g., [9].

Finally, Table 2 shows the strong relationship between our adaptive starting tours and the corresponding local minima. For the three types of starting tours (random, NN, adaptive), the Table shows average bond distance between the starting tour used by **Greedy_Descent** and the local minimum tour it returns. The

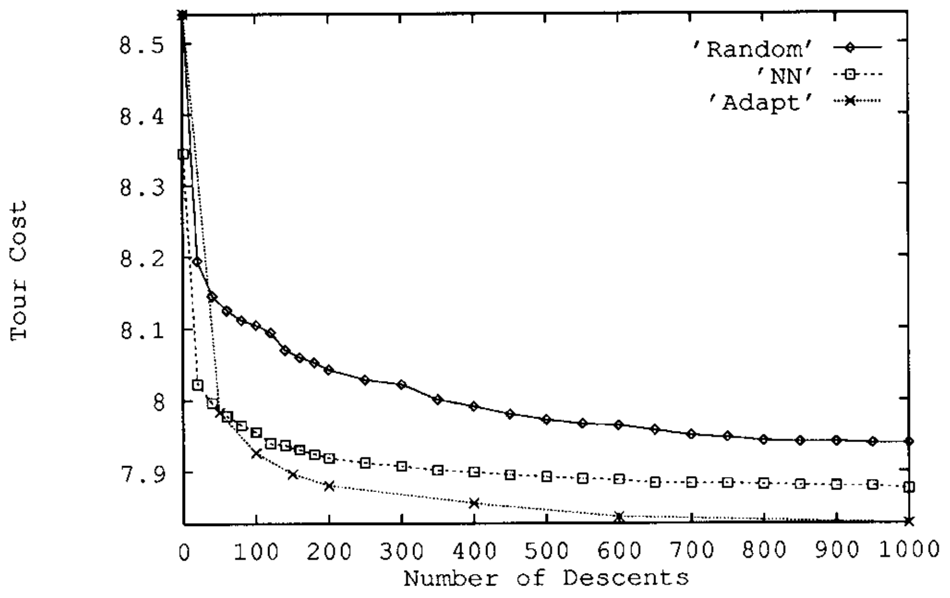


Figure 9: Graphical comparison of AMS, Random multi-start, and Nearest-Neighbor multi-start approaches averaged over 50 random 100-city Euclidean TSPs instances.

	50-city instance			100 city instance		
Starting Tours	**Random**	NN	**Adaptive**	**Random**	NN	**Adaptive**
Distance to Local Min	47.52 (1.32)	13.20 (1.66)	5.18 (1.93)	97.90 (1.34)	27.16 (3.92)	9.69 (5.25)
Ave Dist to Random Mins	47.58 (1.13)	22.99 (0.82)	13.51 (0.88)	98.08 (1.01)	39.01 (1.14)	31.27 (1.52)

Table 2: Mean (standard deviation) of bond distance between 50 starting tours and the corresponding local minimum tours found by **Greedy_Descent**. Also given are the mean (standard deviation) of the average bond distance between each starting tour and 100 "random" local minima. Sample size is 50 for all data; single random Euclidean instances are used for both $n = 50$ and for $n = 100$.

Table also compares these distances with the average bond distance between starting tours and 100 "random" local minima obtained by **Greedy_Descent** from 100 different random starting tours. We see that the position of a *random* starting tour has little effect on the position of the local minimum tour found by **Greedy_Descent**: in fact, the starting tour is no closer to the resulting local minimum than to a "random local minimum". By contrast, the NN and adaptive starting tours are not only much closer to their resulting local minima, but are also significantly closer to other "random" local minima.[6]

[6]The high standard deviation of bond distance from the adaptive starting tour to the corresponding local minimum tour is due to our data collection methodology: to obtain 50 distinct adaptive starting tours using the AMS heuristic with the usual $R = D/2$, $A = 10$, etc. we must use $D \approx 1000$, and the quality of the adaptive initial tours increases significantly over the course of this process. We could also obtain the adaptive initial tours by executing AMS with $D = 20$ a total of 50 separate times; this yields slightly higher mean, and lower (std), values of 7.46 (1.766) for $n = 50$, and 15.20 (3.339) for $n = 100$.

4 Conclusions

Multi-start greedy optimization has shown much promise for a number of practical applications, but the traditional random multi-start implementation suffers from a "central-limit catastrophe" when problem size grows large. In this paper we have developed new *adaptive multi-start* methods, based on new insights into global structure of optimization cost surfaces.

For instances of the symmetric TSP and graph bisection, we study correlations between the cost of a local minimum and its average distance to all other local minima (as well as its distance to the global minimum). Our analyses show evidence of a "big valley" governing local minima in the optimization cost surface, and motivate the *adaptive multi-start* methodology. Specifically, our evidence suggests a *globally convex* [7] structure for the set of local minima (we may draw a simile to the structure of an integer polytope, which "viewed from afar" may appear to have a single minimum point, but which up close has many local minima). Our results may explain why simulated annealing and other hill-climbing heuristics have been so successful in practice: very good solutions are located near other good solutions.

Based on these insights, our Adaptive Multi-Start (AMS) heuristic uses best-known local minimum solutions to generate starting points for subsequent greedy descents. Experimental evidence for TSP instances shows significant improvement in runtime and solution quality over previous multi-start methods (random- and nearest-neighbor-based). Future work should apply our adaptive approach to other greedy methods (e.g., 3-opt and Lin-Kernighan for TSP) and combinatorial formulations (e.g., graph partitioning and VLSI circuit placement). Our current study also motivates consideration of alternate neighborhood structures which can induce a "big valley" over the solution space.

Appendix 1. Relation between $d(t,t')$ and $b(t,t')$

Recall from Section 2.1 that $d(t,t')$ is the 2-opt distance and $b(t,t')$ is the bond distance between TSP tours.

Theorem 1: For any two tours t and t' of a given n-city TSP instance, $\frac{b(t,t')}{2} \leq d(t,t') \leq b(t,t')$.

Proof: The lower bound follows easily by noting that a single 2-opt affects only two edges in the tour, and can therefore increase the bond distance by at most two. This lower bound is tight: a sequence of j 2-opts ($1 \leq j \leq \lfloor \frac{n}{2} \rfloor$), each of which is applied to two edges remaining from t, will yield t' with $d(t,t') = j$ and $b(t,t') = 2j$.

In proving the upper bound, we will use a canonical **t-subtour** representation to express t' in terms of t. We depict tours as permutations $< 1, \ldots >$ with city 1 listed first, followed by its *lower-index* neighbor. A *t-subtour* of t' is a contiguous sequence of cities in t that is also contiguous in t'; we use capital letters to label the t-subtours, with these labels assigned in alphabetic order according to their positions in t. A t-subtour is *lower* (*higher*) than another t-subtour if its label is closer to the beginning (end) of the alphabet (e.g., B is lower than C). Because all tours are notated as permutations beginning with city 1, the canonical t-subtour representation always begins with A. A t-subtour that appears in t' in reverse order is denoted by a bar above its label. For example, if $t =< 1,2,3,4,5,6 >$ and $t' =< 1,2,4,3,5,6 >$, then the t-subtour representation of t' is $A\overline{B}C$, where $A =< 1,2 >$, $B =< 3,4 >$, and $C =< 5,6 >$. Note that reversing a sequence of t-subtours corresponds to a single 2-opt, e.g., a 2-opt involving the two edges in $t' = A\overline{B}C$ which separate $A\overline{B}$ and CA will yield the tour $A\overline{C}B =< 1,2,6,5,3,4 >$.

We will prove the following three facts:

- **Fact 1:** If $b(t,t') = 2$, then $d(t,t') = 1$.
- **Fact 2:** If the t-subtour representation of t' contains a reversed t-subtour, then there is a 2-opt which transforms t' to t'' such that $b(t'',t) < b(t',t)$.
- **Fact 3:** If the t-subtour representation of t' contains a reversed t-subtour, then there is 2-opt which transforms t' into t'' such that either (i) $b(t'',t) \leq b(t',t) - 1$ and t'' contains a reversed t-subtour, or (ii) $b(t'',t) = b(t',t) - 2$ and t'' contains no reversed subtour.

The following recipe then transforms t' to t using at most $b(t,t')$ 2-opts; the recipe relies directly on Facts 1 and 3 (Fact 2 is used to prove Fact 3).

```
1.  If $t'$ contains no reversed $t$-subtours
        arbitrarily reverse a $t$-subtour, leaving the bond distance to $t$ unchanged.
2.  If $t'$ contains at least one reversed $t$-subtour, perform a 2-opt that either
        reduces $b(t,t')$ by 1 or 2 and leaves a reversed $t$-subtour; or
        reduces $b(t,t')$ by 2 and leaves no reversed $t$-subtours.
3.  If $b(t,t') = 2$ perform a 2-opt which transforms $t'$ into $t$.
4.  Repeat Steps 1 - 3 until $t' = t$.
```

Proof of Fact 1: If $b(t,t') = 2$, then the t-subtour representation of t' must be either $X\overline{Y}$ or $X\overline{Y}Z$. Here, X, Y and Z may denote sequences of t-subtours. In either case, a single 2-opt transforms t' into t; note that $b(t,t') = 1$ is impossible.

Proof of Fact 2: Let β represent the lowest t-subtour that is reversed in t', i.e., it appears as $\overline{\beta}$, and let α be the (non-reversed) t-subtour immediately preceding β alphabetically. The basic idea is to reduce the bond distance by bringing the two t-subtours α and β next to each other using a single 2-opt. If α occurs before $\overline{\beta}$ in t', then a 2-opt which places β directly after α will reduce the bond distance to t by at least one. This is shown in Line (a) of the template below (in the template, $\mathcal{Z}$ represents the sequence of t-subtours between α and β in t'). If α occurs after $\overline{\beta}$ in t', then the 2-opt move which places $\overline{\alpha}$ directly after $\overline{\beta}$ will reduce the bond distance to t (see Line (b)).[7]

(a)	$A \ldots \alpha \mathcal{Z} \overline{\beta} \ldots$	$\Rightarrow$	$A \ldots \alpha \beta \overline{\mathcal{Z}} \ldots$
(b)	$A \ldots \overline{\beta} \mathcal{Z} \alpha \ldots$	$\Rightarrow$	$A \ldots \overline{\beta} \overline{\alpha} \overline{\mathcal{Z}} \ldots$

[7]For an example of Line (a): if $t' = AC\overline{B}$, then $\beta = B$, $\alpha = A$, and $t'' = AB\overline{C}$. An example of Line (b): $t' = A\overline{C}DB$, $\beta = C$, $\alpha = B$, and $t'' = A\overline{C}\overline{B}\overline{D}$.

Proof of Fact 3: Suppose t' contains a reversed (sequence of) t-subtour(s), and that (i) is not true, i.e., there is no 2-opt which transforms t' to t'', with t'' containing a reversed t-subtour and having $b(t'', t) < b(t', t)$. Let $\mathcal{X}$ be the first maximal contiguous sequence of reversed t-subtours in t'. If t' contains any t-subtours outside $\mathcal{X}$, then any 2-opt will leave a reversed subtour, and by Fact 2, there will exist some 2-opt move that will reduce the bond distance to t. Thus, all reversed t-subtours in t' must be contained in $\mathcal{X}$. As in the proof of Fact 2, let β be the lowest reversed t-subtour in t', let α be the t-subtour preceding β alphabetically, and let $\mathcal{Z}$ be the sequence of t-subtours between α and β, with η denoting another t-subtour in t'. The template below shows that the following must hold if no 2-opt move exists which reduces the bond distance to t and leaves a reduced t-subtour: (a) α appears before $\overline{\beta}$; (b) $\overline{\beta}$ is located at the end of block $\mathcal{X}$; and (c) α is located immediately before block $\mathcal{X}$. Let γ denote the highest reversed t-subtour in t' and let δ denote the t-subtour after γ alphabetically (we use $\delta = A$ if γ is the highest t-subtour.) A similar argument shows that $\overline{\gamma}$ must be located at the beginning of $\mathcal{X}$ and δ must lie immediately after $\mathcal{X}$. Consequently, the tour structure forced by (a), (b), and (c) allows a 2-opt from $t' = \ldots\alpha\overline{\gamma\mathcal{Z}\beta}\delta\ldots$ to $\ldots\alpha\beta\mathcal{Z}\gamma\delta\ldots$, reducing the bond distance to t by 2.

Case	If () holds	Then $\exists$ 2-opt satisfying (i)
(a)	(α not before $\overline{\beta}$ in t')	$\ldots\overline{\beta}\mathcal{Z}\alpha\ldots \quad \Rightarrow \quad \ldots\overline{\beta\alpha\mathcal{Z}}\ldots$
(b)	($\overline{\beta}$ not at end of $\mathcal{X}$ in t')	$\ldots\alpha\overline{\mathcal{Z}\beta\eta}\ldots \quad \Rightarrow \quad \ldots\alpha\beta\mathcal{Z}\overline{\eta}\ldots$
(c)	(α not next to $\mathcal{X}$ in t')	$\ldots\alpha\eta\overline{\mathcal{Z}\beta}\ldots \quad \Rightarrow \quad \ldots\alpha\beta\mathcal{Z}\overline{\eta}\ldots$

□

Appendix 2. Distribution of Tours in the TSP

Analysis of the distribution of tours at each bond distance from a given tour, which by symmetry is the same as the distribution of $b(t_1, t_2)$ between random tours t_1 and t_2, has been attributed to D. Gross and M. Mezard in [14]. The cited result is that the number of edges in common between any two random tours t_1 and t_2, i.e., $n - b(t_1, t_2)$, approaches a Poisson distribution as $n \to \infty$, and has mean = 2. We have studied this distribution more precisely. In what follows, we show that the number of edges in common has mean $= 2 * \frac{n}{n-1}$. We then describe an exact method for efficiently calculating the $b(t_1, t_2)$ distribution.

Fact 4: The mean number of edges in common between two random tours on n cities is $2 * \frac{n}{n-1}$.

Proof: Each edge in E occurs in exactly $(n-2)!$ tours. Since each tour has n edges, the average number of overlaps between a given tour and all tours in solution space S is equal to $\frac{n(n-2)!}{|S|} = \frac{n(n-2)!}{(n-1)!/2} = 2 * \frac{n}{n-1}$. □

To compute the $b(t_1, t_2)$ distribution, without loss of generality we compute the distribution of $I(n, k)$, where $I(n, k)$ denotes the number of tours on n cities that have k edges in common with the *identity tour*, $\sigma(n) =< 1, 2, \ldots, n-1, n >$. $I(n+1, k)$ can be calculated exactly based on the observation that any tour t' of $n+1$ cities is uniquely expressible as the insertion of city $n+1$ into a tour t of n cities.

The basic methodology is that of dynamic programming: given the values $I(n,k)$ for $0 \leq k \leq n$ we can compute all the $I(n+1,k)$ values for $0 \leq k \leq n+1$. For a given n, the entire computation of all $I(n,k)$ values requires only $\Theta(n^2)$ time and $\Theta(n)$ space. We use the term *bond* to denote any adjacency in the n-city tour t that is also an adjacency in $\sigma(n)$, i.e., t has k bonds if $b(t,\sigma(n)) = n-k$, or equivalently if there are k edges in common between t and $\sigma(n)$. We say that we *break* a bond when we insert city $n+1$ between the two cities which form that bond in t. One readily sees that breaking a bond will usually reduce the number of bonds that remain in t' to $k-1$; similarly, inserting $n+1$ next to city 1 or n will generally increment the number of bonds, while other positions for city $n+1$ will generally leave the number of bonds unchanged.

To be more precise: when placing city $n+1$ into a tour t of n cities with k bonds, we must consider eight cases corresponding to the possible presence or absence of each of three "special" bonds in t:

- $(1,n)$: breaking this bond *increases* the number of bonds by 1 because the new bonds $(1,n+1)$ and $(n,n+1)$ in $\sigma(n+1)$ are created. If t has this bond and the placement of city $n+1$ does not break it, then the number of bonds decreases by 1 because $(1,n)$ is no longer a bond in a tour of $n+1$ cities.
- $(1,2)$: breaking this bond will leave k constant unless t also contains bond $(1,n)$.
- $(n-1,n)$: breaking this bond will also leave k constant in the absence of bond $(1,n)$.

Figure 10 illustrates the ways in which city $n+1$ can be placed into a 6-city tour $t =< 1,2,4,3,5,6 >$, which contains all three special bonds.

New Tour t'	Effect of Placement	Effect on k
$< 1,\ \mathbf{7}\ ,2,4,3,5,6 >$	break bond $(1,2)$	$k \leftarrow k-1$
$< 1,2,\ \mathbf{7}\ ,4,3,5,6 >$	not break any bond	$k \leftarrow k-1$
$< 1,2,4,\ \mathbf{7}\ ,3,5,6 >$	break non-special bond $(3,4)$	$k \leftarrow k-2$
$< 1,2,4,3,\ \mathbf{7}\ ,5,6 >$	not break any bond	$k \leftarrow k-1$
$< 1,2,4,3,5,\ \mathbf{7}\ ,6 >$	break bond $(n-1,n)$	$k \leftarrow k-1$
$< 1,2,4,3,5,6,\ \mathbf{7}\ >$	break bond $(1,n)$	$k \leftarrow k+1$

Figure 10: Illustration of the six ways to insert city $n+1 = 7$ into tour $t =< 1,2,4,3,5,6 >$. Because t contains bonds $(1,2)$, $(3,4)$, $(5,6)$, and $(1,6)$, $k = 4$ for t. Note that t contains all three special bonds, $(1,2)$, $(1,n)$, and $(n-1,n)$.

Figure 11 gives details on how the placement of city $n+1$ affects the number of overlaps with $\sigma(n+1)$. (Although not shown in the table, it is also necessary to keep track of how the placement of $n+1$ affects which special bonds occur in the resulting tour t'.) Thus, if we know the number of tours of length n containing k bonds for each k and each combination of the three special bonds, we can compute the number of tours of length $n+1$ for each k and special bond combination. Summing up the eight cases for special bonds for n and k gives us the value for $I(n,k)$. This overall computation of $I(n,k)$, $0 \leq k \leq n$, can be accomplished in time $\Theta(n^2)$ and space $\Theta(n)$.

Special Bonds in t	Number of Placements	Effect of Placement	Effect on k
–	k	break non-special bond	$k \leftarrow k-1$
	4	create new bond (next to 1 or n)	$k \leftarrow k+1$
	$n-4-k$	other	$k \leftarrow k$
$(1,2)$	1	break bond $(1,2)$	$k \leftarrow k$
	$k-1$	break non-special bond	$k \leftarrow k-1$
	3	create new bond (next to 1 or n)	$k \leftarrow k+1$
	$n-k-3$	other	$k \leftarrow k$
$(1,n)$	1	break bond $(1,n)$	$k \leftarrow k+1$
	$k-1$	break non-special bond	$k \leftarrow k-2$
	2	create new bond (next to 1 or n)	$k \leftarrow k+1$
	$n-k-2$	other	$k \leftarrow k$
$(n-1,n)$	1	break bond $(n-1,n)$	$k \leftarrow k$
	$k-1$	break non-special bond	$k \leftarrow k-1$
	3	create new bond (next to 1 or n)	$k \leftarrow k+1$
	$n-k-3$	other	$k \leftarrow k$
$(1,2)(1,n)$	1	break bond $(1,2)$	$k \leftarrow k-1$
	1	break bond $(1,n)$	$k \leftarrow k+1$
	$k-2$	break non-special bond	$k \leftarrow k-2$
	1	create new bond (next to n)	$k \leftarrow k$
	$n-k-1$	other	$k \leftarrow k-1$
$(1,2)(n-1,n)$	1	break bond $(1,2)$	$k \leftarrow k$
	1	break bond $(n-1,n)$	$k \leftarrow k$
	$k-2$	break non-special bond	$k \leftarrow k-1$
	2	create new bond (next to 1 or n)	$k \leftarrow k$
	$n-k-2$	other	$k \leftarrow k$
$(1,n)(n-1,n)$	1	break bond $(1,n)$	$k \leftarrow k+1$
	1	break bond $(n-1,n)$	$k \leftarrow k-1$
	$k-2$	break non-special bond	$k \leftarrow k-2$
	1	create new bond (next to 1)	$k \leftarrow k$
	$n-k-1$	other	$k \leftarrow k-1$
$(1,2)(1,n)(n-1,n)$	1	break bond $(1,n)$	$k \leftarrow k+1$
	1	break bond $(n-1,n)$	$k \leftarrow k-1$
	1	break bond $(1,2)$	$k \leftarrow k-1$
	$k-3$	break non-special bond	$k \leftarrow k-2$
	$n-k$	other	$k \leftarrow k-1$

Figure 11: Details on the effect on the number edges in common with the identity tour from placing city $n+1$ in a tour t of n cities.

Figure 12 shows the resulting cumulative distribution of $b(t_1, t_2)$ for 100-city and 500-city TSPs. Note that all 2,500 local optima shown in Figure 2(b) for a 100-city instance are confined to a ball of radius = 48 containing less than $1/10^{60}$ of the entire solution space; similarly, all 2,500 local optima in Figure 4(b) are contained in a ball of radius = 245 corresponding to less than $1/10^{463}$ of the solution space.

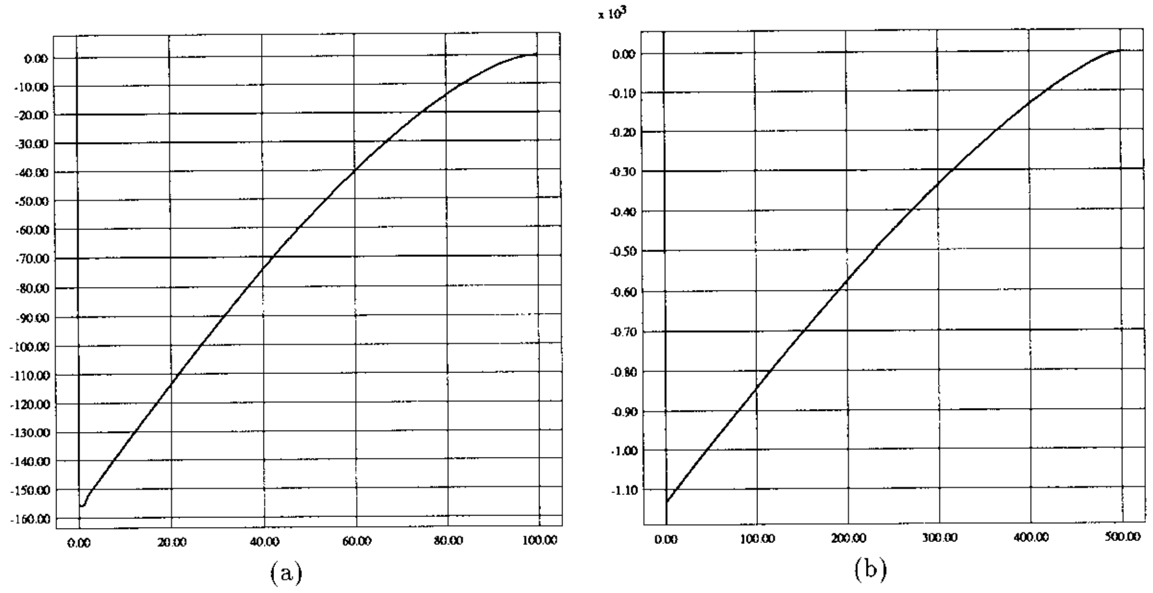


Figure 12: Cumulative distribution of the fraction of all tours within a given bond distance (a) $n = 100$ and (b) $n = 500$. (Log scale base 10 used on vertical axis).

References

[1] S. Bacci and N. Parga, "Ultrametricity, Frustration and the Graph Colouring Problem," *J. Physics A: Math. and General* **22**, 3023-3032 (1989).

[2] E. B. Baum, "Towards Practical 'Neural' Computation for Combinatorial Optimization Problems", in J. Denker, ed., *Neural Networks for Computing*, American Institute of Physics, 1986.

[3] K. D. Boese and A. B. Kahng, "Best-So-Far vs. Where-You-Are: Implications for Optimal Finite-Time Annealing", to appear in *Systems and Control Letters*, 1993.

[4] B. Bollobas, *Random Graphs*, Academic Press, 1985.

[5] T. N. Bui, S. Chaudhuri, F. T. Leighton, and T. Sipser, "Graph Bisection Algorithms with Good Average Case Behavior", *Combinatorica*, **2** 171-191 (1987).

[6] T. H. Cormen, C. E. Leiserson and R. L. Rivest, *Introduction to Algorithms*, MIT Press, 1990.

[7] T. C. Hu, V. Klee and D. Larman, "Optimization of Globally Convex Functions", *SIAM J. on Control and Optimization* **27** (5), 1026-1047 (1989).

[8] J. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, 1975.

[9] D. S. Johnson, "Local Optimization and the Traveling Salesman Problem, in *Proceedings of the 17th International Colloquium on Automata, Languages and Programming*, July 1990, 446-460.

[10] 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** 865-892 (1989).

[11] D. S. Johnson and E. E. Rothberg, "Asymptotic Experimental Analysis of the Held-Karp Lower Bound for the Traveling Salesman Problem" (to appear).

[12] S. Kauffman and S. Levin. "Toward a General Theory of Adaptive Walks on Rugged Landscapes", *Journal of Theoretical Biology* **128**, 11-45 (1987).

[13] S. Kirkpatrick, C. D. Gelatt, and M. Vecchi, "Optimization by Simulated Annealing", *Science*, **220**, 671-680 (1983).

[14] S. Kirkpatrick and G. Toulouse, "Configuration Space Analysis of Traveling Salesman Problems", *Journal de Physique* **46**, 1277-1292 (1985).

[15] J. B. Lasserre, P. P. Varaiya, and J. Walrand, "Simulated Annealing, Random Search, Multistart or SAD?", *Systems and Control Letters* **8**, 297-301 (1987).

[16] E. L. Lawler, J. K. Lenstra, A. Rinnooy-Kan and D. Shmoys, *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*, Wiley, 1985.

[17] M. Mezard and G. Parisi, "A Replica Analysis of the Travelling Salesman Problem", J. Physique **47** 1285-1296 (1986).

[18] C. Papadimitriou, "The Complexity of the Lin-Kernighan Heuristic for the Traveling Saleman Problem", *SIAM J. Computing* **21** (3), 450-465 (1992).

[19] S. A. Solla, G. B. Sorkin and S. R. White, "Configuration Space Analysis for Optimization Problems", in E. Bienenstock et al., eds., *Disordered Systems and Biological Organization*, Springer-Verlag, 1986, 283-292.

[20] G. B. Sorkin, "Efficient Simulated Annealing on Fractal Energy Landscapes", *Algorithmica* **6**, 367-418 (1991).

[21] G. B. Sorkin, *personal communication*, May 1993.

[22] N. Sourlas, "Statistical Mechanics and the Travelling Salesman Problem", *Europhysics Letters* **2** (12), 919-923 (1986).

[23] Y. C. Wei and C. K. Cheng, "A Two-Level Two-Way Partitioning Algorithm", in *Proc. IEEE Intl. Conf. on Computer-Aided Design*, Nov. 1990, 516-519.

[24] E. Weinberger, "Correlated and Uncorrelated Fitness Landscapes and How to Tell the Difference", *Biological Cybernetics* **63**, 325-336 (1990).