ELSEVIER

Operations Research Letters 16 (1994) 101–113

operations research letters

A new adaptive multi-start technique for combinatorial global optimizations†

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

UCLA Computer Science Department, Los Angeles, CA 90024-1596, USA

(Received 8 June 1993; revised 1 December 1993)

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 for the traveling salesman problem and obtain very significant speedups over previous multi-start implementations.

Key words: Global optimization; Heuristic search; 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 $f: S \to \Re$. Global optimization seeks a solution $s^* \in S$ with $f(s^*) \leqslant f(s')\ \forall s' \in S$. 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)

* Corresponding author.

† 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.

SSDI 0167-6377(93)00007-S

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* [14] generates a random $s' \in N(s_i)$ in Step 1, while Step 2 sets $s_{i+1} = s'$ with probability one if $f(s') \leqslant f(s_i)$, 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} iteration. Other heuristics are *greedy* with $s_{i+1} = s'$ in Step 2 only if $f(s') < f(s_i)$. Of specific interest to us is the nondeterministic Greedy_Descent procedure, which iteratively tests solutions $s' \in N(s_i)$ in *random* order until an improvement $s_{i+1} = s'$ with $f(s_{i+1} < f(s_i)$ can be made; the procedure terminates if no improving $s' \in N(s_i)$ exists. With greedy search, progress stops when the first local minimum is encountered. Simulated annealing can escape from local minima and has gained wide popularity because it is guaranteed theoretically to return a globally optimum solution (given infinite time), and in practice yields better solutions than most other methods. On the other hand, annealing usually requires large amounts of CPU time to be successful.

Because greed returns a good solution relatively quickly, one alternative to simulated annealing is to apply greed repeatedly and return the best result. Several studies have shown "greedy multi-start" superior to simulated annealing in terms of both solution quality and run time. Johnson [10] 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 [23] show that a multi-start approach is superior to standard simulated annealing on a class of fractal cost surfaces. Boese and Kahng [5] 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 to zero and infinite temperatures). Multi-start is also attractive for its trivial parallelizability on distributed architectures.

Nevertheless, the multi-start approach has its weaknesses. Recent analyses of optimization 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" (e.g., [3, 13]). Moreover, other work on graph partitioning indicates that the number of greedy descents needed to achieve stable, good solutions (where stability means a low standard deviation in solution cost) can grow rapidly with problem size [11, 26].

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; for example, Sorkin [22] and Weinberger [27] have fitted fractal and AR(1) processes, respectively, to real-world optimization cost surfaces. For our purposes, the leading study is due to Kirkpatrick and Toulouse [15], who attempt to confirm an ultrametric relationship between local minima for the traveling salesman problem (TSP). They observe only inconclusive evidence for ultrametricity, but do find that the distances between random pairs of local minima satisfy a normal distribution with surprisingly low average. (Related studies are due to Mezard and Parisi [17] and Sourlas [24]; 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 [2] and one-dimensional circuit placement [21]. Both [24, 17] discuss correlations of TSP tour costs with distances between tours, much as we do in Section 2.1 below; however, they fail to make any of the enabling observations that we present here.)

We also study relationships among local minima, but in a different way: we consider the set of local minima *from the perspective of the best local minimum.* As we describe in the remainder of this section, our results indicate that many problem spaces exhibit a "globally convex" [9] structure, suggesting improved

multi-start strategies which derive starting points from the best previously found local minima. Section 3 will develop this new class of *adaptive multi-start* (AMS) methods. AMS bears some similarities to "genetic local search" algorithms [1, 18, 19, 25], although the latter generally form new starting solutions from only two "parents", rather than from many local minima. Moreover, AMS does not depend on any evolutionary analogy for its motivation. (Note that Mühlenbein [18] and Ackley [1, p. 35] do mention multi-parent, voting approaches for forming new solutions and that Mühlenbein et al. [19] also analyze the distribution of local minima in a manner similar to ours. Note also that while "iterated greed" [10] and tabu search in some sense use information about local minima, such methods do not follow a "multi-start" paradigm.)

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 cities in the tour. We use the Lin 2-opt neighborhood operator that is usual in studies of the TSP [16]: a 2-opt deletes two nonadjacent 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 [15] measure the similarity between t_1 and t_2 according to the number of edges, or "bonds", common to both tours. We will use the term *bond distance*, denoted $b(t_1, t_2)$, to equal n minus the number of edges that are present in both t_1 and t_2 (disregarding edge direction). No previous results directly link bond distance to the 2-opt or any other TSP neighborhood structure. We have partially addressed this gap through the following result (see Appendix A 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, $b(t_1, t_2)/2 \leqslant d(t_1, t_2) \leqslant b(t_1, t_2)$, with the lower bound being tight.*

Recall that our new approach to multi-start will be motivated by examining the set of local minima from the perspective of the *best* local minimum. From each of 2500 random locally minimum tours for a 100-city random Euclidean TSP instance, Fig. 1(a) plots the tour's cost versus its average bond distance to all (2499) other local minima. All TSP instances that we discuss are chosen randomly from a uniform distribution over the unit square. A "random local minimum" is found by starting at a random initial solution and executing Greedy_Descent. In the figure we see a clear correlation: the best local minimum appears to be "central" to all other local minima, and indeed a "big valley" structure [6] can be said to govern the set of locally minimum tours.

Further insight is gained from Fig. 1(b), which plots the costs of the same 2500 local minima against their distances from the best local minimum found. Note that all local minima are within bond distance 48. In Appendix B, we show that the average distance between two random n-city tours is just under $n - 2$, slightly sharpening an observation in [15]. Appendix B also gives the first efficient enumeration of tours at each bond distance; for a 100-city TSP instance, this calculation indicates that less than $1/10^{59}$ of the solution space lies within a "ball" of radius equal to 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. Finally, Fig. 2 gives analogous plots for a random 500-city Euclidean TSP instance (for which a ball of radius 243 corresponds to less than $1/10^{468}$ of the solution space). In [6], we have obtained similar results for random symmetric TSPs (with edge weights uniform in [0, 1]), which are studied in [15] and elsewhere.

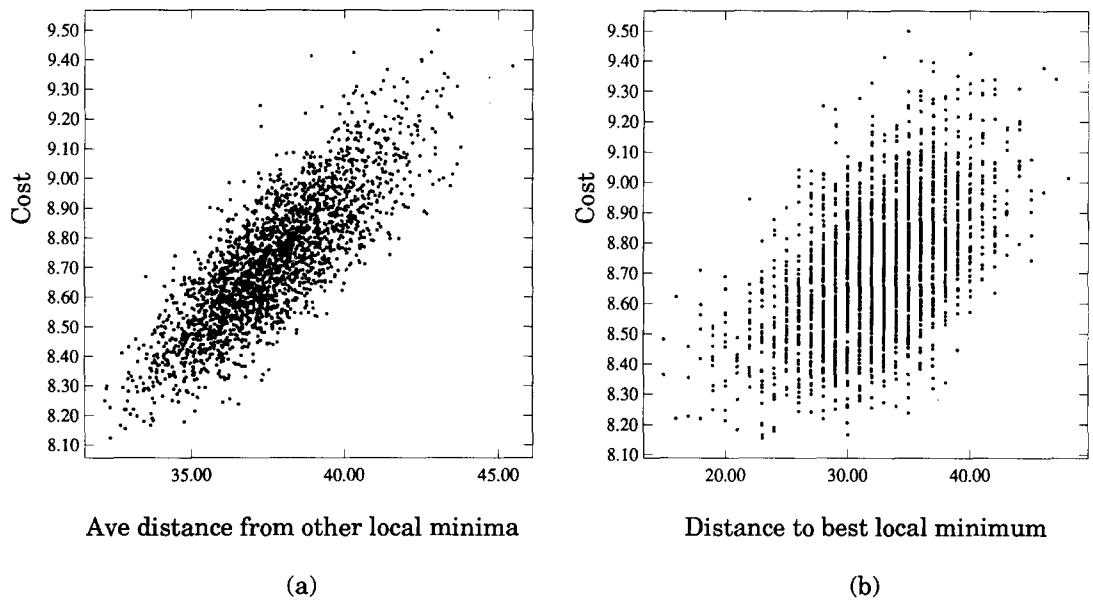

Fig. 1. Analysis of 2500 random local minima for a 100-city Euclidean TSP instance. Tour cost (vertical axis) is plotted against (a) average distance from the other 2499 local minima and (b) distance from the local minimum with lowest cost. All 2500 local minima are distinct.

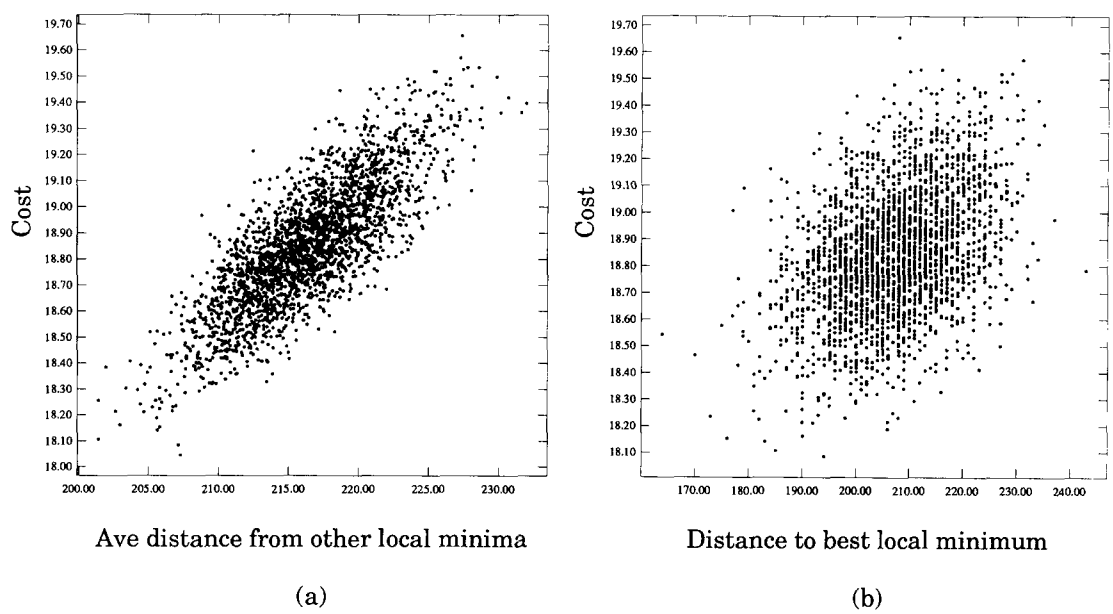

Fig. 2. Analysis of 2500 random local minima from a 500-city Euclidean TSP instance. All 2500 local minima are distinct. In (b), we do not show the best local minimum, which is at distance zero.

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 graph bisection problem seeks a partition of V into disjoint subsets U and

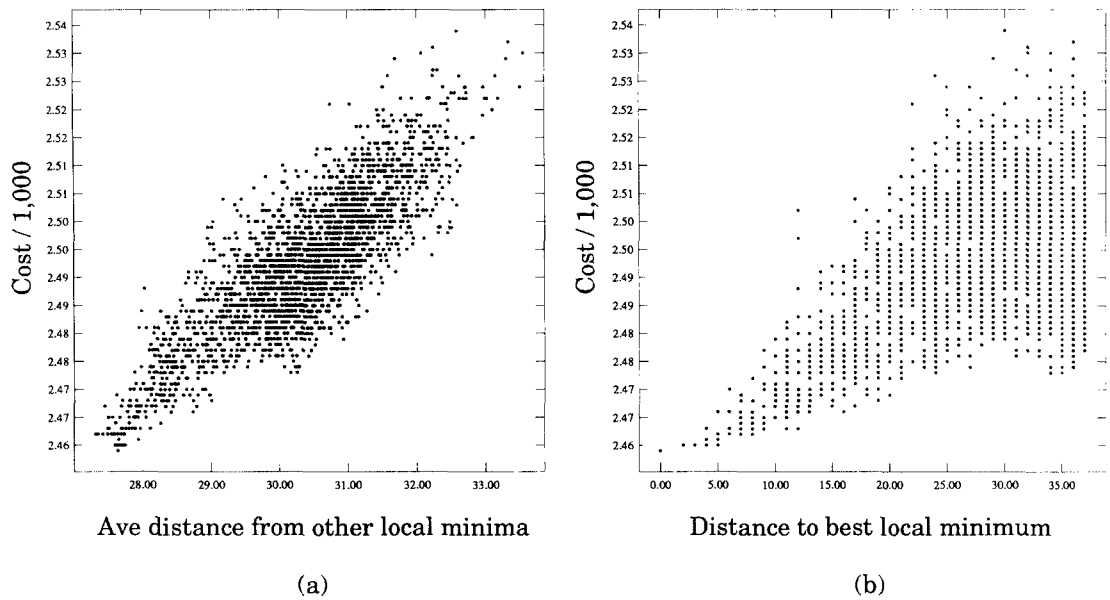

Fig. 3. Analysis of 2500 random locally minimum bisections for graph in $G(150, 0.5)$. The data represent 2399 distinct local minima.

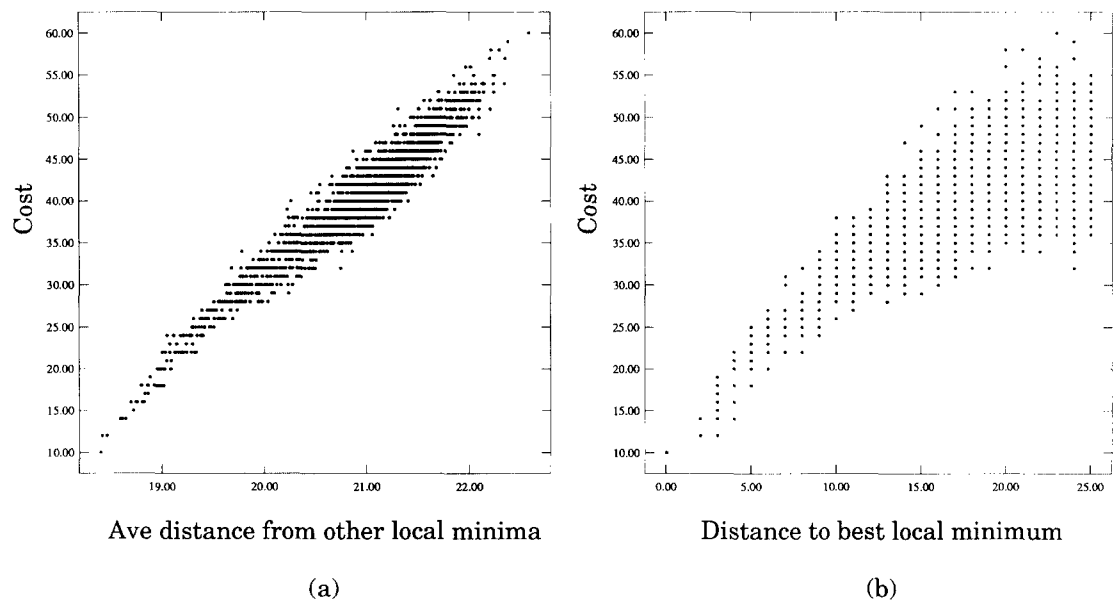

Fig. 4. Analysis of 2500 random locally minimum bisections for graph in $G_{\text{Bui}}(100, 4, 10)$. The data represent 2343 distinct local minima.

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 first study a standard class of random graphs $G(n, p)$, i.e., graphs having n vertices and each possible edge present independently with probability p (see [7]). Because graphs in $G(n, p)$ have expected minimum

bisection cost within a constant factor of the expected random bisection cost [8], more "difficult", structured models have been proposed. In particular, we also study the class $G_{\text{Bui}}(n, d, b)$ of random graphs proposed by Bui et al. [8], which have n nodes, are d-regular, and have minimum bisection cost almost certainly equal to b. Typical results are shown in Figs. 3 and 4. The "big valley" correlation is again clearly apparent, although local minima are not as strongly confined to a small region of the solution space. (Some local minima have maximum distance from the best local minimum. Note that the expected distance between random bisections is easily computed to be 23.02 for $n = 100$ and 35.04 for $n = 150$.)

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 assume that a "big valley" structure holds 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 adaptive multi-start (AMS) heuristic

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

Phase One: Generate R random starting solutions and run Greedy_Descent from each to determine a set of corresponding *random local minima.*

Phase Two: Based on the local minima obtained so far, construct *adaptive* starting solutions and run Greedy_Descent A times from each one 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 Fig. 5. (Note that henceforth our discussion is couched with respect to the symmetric TSP.)

In the Fig. 5 template, the term *descent* denotes a single execution of Greedy_Descent. We use D to denote the total number of calls to Greedy_Descent. The number of passes through Phase Two (Lines 3–6) is determined by the relationship $passes = \lceil (D - R)/A \rceil$. In obtaining the results of Section 4, we uniformly use $R = D/2$ (i.e., we spend exactly half our CPU budget in Phase One) and $A = 10$. When $D - R$ is not an exact multiple of A we truncate the final pass in Line 5. The subroutine Construct_Adaptive_Starting_Tour always constructs an adaptive starting tour from the set of k best local minima found so far; we use $k = 10$ in our experiments. In the description of Construct_Adaptive_Starting_Tour, a *partial tour* is 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* wtih respect to t iff $t \cup \{e\}$ is a partial tour. The experimental results in Section 3.2 were obtained using the following additional implementation details:

1. In Line S4, $w(e_i)$ is the sum of the inverse tour costs for tours in which e_i appears, i.e.,

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

2. In Line S6, we use

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

where

$$W = \sum_{j=1}^{|M|} \frac{1}{\text{cost}(M_j)}$$

is the weight of an edge that is contained in all tours in M.

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

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 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 ← 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 locally 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 = ∅.
S3.  Assign weight w(e_i) to each edge e_i ∈ In
S4.  for e_i ∈ 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 ∪ {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 \ t
```

Fig. 5. Adaptive–Multi-Start template.

(In Line S4, our definition of $w(e_i)$ gives greater weight to edges that are present in many short tours; however, we have obtained similar results using uniform edge weights. In Line S6, while we have also tried other probabilistic weighting methods, our choice allows the probability that e_i is included in t to approach 1 as $w(e_i)$ approaches its maximum possible value. In Lines S7 and S8, we have found 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.)

Given these implementation decisions, Construct_Adaptive_Starting_Tour has $O(kn \log n + n^2)$ worst-case run time. Of course, this is dominated by the known exponential worst-case run time of Greedy_Descent for 2-opt in the TSP [20].

3.2. Experimental results

Table 1 compares results of our AMS implementation 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).

NN tours are suggested in [10] for obtaining initial tours for 2-opt descents; although Bentley [4] has since shown that the greedy TSP tour is a slightly better starting point for a single descent, it may be less appropriate for multi-start because there is only one greedy tour of a given instance versus up to n different NN tours. As problem size and the CPU budget 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 5700 descents to achieve. A more detailed portrait of this data is given in Fig. 6.

Table 2 shows the stability of our AMS method, where we define stability to be the standard deviation of cost(t^*). Measured over 50 separate executions on a single 100-city instance, the stability of AMS is very good, with a standard deviation of only 0.030. Although other multi-start methods can have greater stability, AMS maintains its superiority because of its low average solution cost. AMS is also "stably better" than the other methods. For example, Table 1, AMS with $D = 200$ gave a superior solution to random multi-start in all instances and to NN multi-start in 47 out of 50 instances.

Table 1
Experimental results for AMS executions on 50 random 100-city Euclidean TSP instances in the unit square

# Descents used in AMS	Equivalent # Descents: Random strategy	Equivalent # Descents: NN strategy	Average Tour cost (AMS)	% above Held-Karp Lower bound
1	1	1	8.5404	10.91
50	442	50	7.9835	3.68
100	1508	176	7.9266	2.94
150	>4100	426	7.8967	2.55
200	>5300	826	7.8806	2.34
400	>7200	> 3000	7.8555	2.02
600	>8600	> 4000	7.8367	1.78
1000	>9300	> 4500	7.8275	1.66

Results 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 3000 descents and nearest-neighbor multi-start for 2000 descents on each of the 50 instances. Entries with > are conservative estimates based on linear extrapolation. Average tour costs and their relationship to the expected Held-Karp lower bound on TSP cost (provided by Johnson and Rothberg [12]) are shown in order to facilitate comparison with other work, e.g., [10].

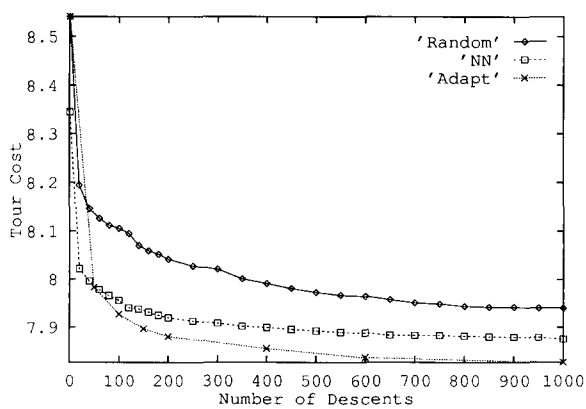

Fig. 6. Graphical comparison of AMS, random multi-start, and nearest-neighbor multi-start approaches averaged over 50 random 100-city Euclidean TSP instances.

Table 2
Stability comparison of different multi-start strategies on a *single* random 100-city TSP instance

	Adaptive		NN		Random	
	Mean	Std. dev.	Mean	Std. dev.	Mean	Std. dev.
$D = 1$	–	–	7.713	0.171	8.015	0.265
D = 100	7.412	0.030	7.492	0.020	7.597	0.011
D = 200	7.393	0.020	7.456	0.019	7.552	0.009

Mean and standard deviation of solution cost are computed for 50 executions of each strategy.

Table 3
Mean (standard deviation) of bond distance between 50 starting tours and the corresponding locally 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

	50-city instance			100-city instance		
Starting tours	Random	NN	Adaptive	Random	NN	Adaptive
Distance to local minima	47.52 (1.32)	13.20 (1.66)	5.18 (1.93)	97.90 (1.34)	27.16 (3.92)	9.69 (5.25)
Average distance to 100 local random minima	47.58 (1.13)	22.99 (0.82)	13.51 (0.88)	98.08 (1.01)	39.01 (1.14)	31.27 (1.52)

Sample size is 50 for all data; single random Euclidean instances are used for both $n = 50$ and for $n = 100$.

Finally, Table 3 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 locally minimum tour it returns. It 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 locally minimum tour found by Greedy_Descent; in fact, this local minimum is no closer to the starting tour than a "random local minimum". By contrast, the NN and adaptive local minima are not only much closer to their starting tours, but are also significantly closer than other "random" local minima. (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 50 separate times with $D = 20$; this yields slightly higher means and lower standard deviations 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 in practical applications, but the traditional random multi-start implementation suffers from a "central-limit catastrophe" when problem size grows large. In this paper we address this difficulty with an *adaptive multi-start* methodology that is 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 best-found local

minimum). Our analyses show evidence of a "big valley" governing local minima in the optimization cost surface, and motivate the *adaptive multi-start* methodology. In other words, our evidence suggests a *globally convex* [9] structure for the set of local minima (we may make an analogy 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, tabu search, iterated greed, 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 locally minimum solutions to generate starting points for subsequent greedy descents. Experimental evidence for TSP instances shows significant improvement in run time 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 other 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 A: 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_1 and t_2 of a given TSP instance, $b(t_1, t_2)/2 \leqslant d(t_1, t_2) \leqslant b(t_1, t_2)$, with the lower bound being tight.*

Proof. The lower bound $d(t, t') \geqslant b(t_1, t_2)/2$ 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 bound is tight, since a sequence of j 2-opts from t ($1 \leqslant j \leqslant \lfloor n/2 \rfloor$), each of which is applied to two edges remaining from t, will yield t' with $d(t, t') = j$ and $b(t, t') = 2j$.

We depict tours as permutations $\langle 1, \ldots \rangle$ with city 1 listed first, followed by its *lower-index* neighbor. In proving the upper bound, we use a canonical *t-subtour* representation to express t' in terms of t. A *t-subtour* of t' is a sequence of contiguous cities in t that are 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 defined to be *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 $t = \langle 1, 2, 3, 4, 5, 6 \rangle$ and $t' = \langle 1, 2, 4, 3, 5, 6 \rangle$, then the t-subtour representation of t' is $A\bar{B}C$, where $A = \langle 1, 2 \rangle$, $B = \langle 3, 4 \rangle$, and $C = \langle 5, 6 \rangle$. Any 2-opt in t' which preserves t-subtours will reverse a sequence of contiguous t-subtours in t', e.g., a 2-opt involving the two edges separating $A|\bar{B}$ and $C|A$ in $t' = A\bar{B}C$ will yield the tour $A\bar{C}B = \langle 1, 2, 6, 5, 3, 4 \rangle$.

We will prove the following three facts. (Note that $b(t, t') = 1$ is impossible.)

Fact A.1. *If $b(t, t') = 2$, then $d(t, t') = 1$.*

Fact A.2. *If the t-subtour representation of t' contains a reversed t-subtour, then there is a 2-opt which transforms t' into t'' such that $b(t'', t) < b(t', t)$.*

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

The following recipe transforms t' to t using at most $b(t, t')$ 2-opts; the recipe relies directly on Facts A.1 and A.3. (Fact A.2 is used in the proof of Fact A.3.) Each 2-opt used in the recipe reduces the bond distance

from t' to t by either 0, 1, or 2; and every move that leaves the bond distance unchanged can be paired with a move that decreases the bond distance by 2. Thus, we are left only with proving Facts A.1–A.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 (Fact A.3).
3. If $b(t, t') = 2$ perform a 2-opt which transforms t' into t (Fact A.1).
4. Repeat Lines 1–3 until $t' = t$.

Proof of Fact A.1. If $b(t, t') = 2$, then t' can be represented using t-subtours as either $A\bar{B}$ or $A\bar{B}C$. In either case, a single 2-opt reversing B will transform t' into t. □

Proof of Fact A.2. Let β represent the lowest t-subtour that is reversed in t', i.e., it appears as $\bar{\beta}$, and let α be the (nonreversed) t-subtour immediately preceding β alphabetically. The basic idea is to reduce the bond distance by bringing α and β next to each other using a single 2-opt. If α occurs before $\bar{\beta}$ in t', then a 2-opt which places β directly after α will reduce the bond distance to t by at least one, i.e., $t' = A \ldots \alpha \mathscr{Z} \bar{\beta} \ldots$ becomes $t'' = A \ldots \alpha\beta\bar{\mathscr{Z}} \ldots$, where $\mathscr{Z}$ represents the sequence of t-subtours between α and β in t'. (For example, if $t' = AC\bar{B}$, then $\beta = B$, $\alpha = A$, and $t'' = AB\bar{C}$.) If α occurs after $\bar{\beta}$ in t', then the 2-opt which places $\bar{\alpha}$ directly after $\bar{\beta}$ will reduce the bond distance to t; i.e., $A \ldots \bar{\beta}\mathscr{Z}\alpha \ldots$ becomes $A \ldots \overline{\beta\alpha\mathscr{Z}} \ldots$ □

Proof of Fact A.3. Suppose t' contains a reversed t-subtour, 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 $\mathscr{X}$ be the first maximal contiguous sequence of reversed t-subtours in t'. Suppose t' contains a t-subtour outside $\mathscr{X}$. Then any 2-opt will leave a reversed subtour. Since by Fact A.2, there must be some 2-opt move reducing the bond distance to t, it must be that all reversed t-subtours in t' are contained in $\mathscr{X}$. As before, let β be the lowest reversed t-subtour in t', let α be the t-subtour preceding β alphabetically, and let $\mathscr{Z}$ be the sequence of t-subtours between α and β, with η denoting another t-subtour which may be present in t'. The template below shows that the following three conditions must hold when there is no 2-opt move which both reduces the bond distance to t and leaves a reversed t-subtour: (a) α appears before $\bar{\beta}$; (b) $\bar{\beta}$ is located at the end of block $\mathscr{X}$; and (c) α is located immediately before block $\mathscr{X}$. If we 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 $\bar{\gamma}$ must be located at the beginning of $\mathscr{X}$, and δ must lie immediately after $\mathscr{X}$. Hence, the resulting tour structure allows a 2-opt from $t' = \ldots \alpha\gamma\mathscr{Z}\beta\delta \ldots$ to $t'' = \ldots \alpha\beta\mathscr{Z}\gamma\delta \ldots$, which reduces the bond distance to t by 2. Thus, if t' contains a reversed t-subtour and (i) is false, then (ii) must be true.

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

□.

Appendix B: Distribution of tours in the TSP

In [15], an analysis of the distribution of tours at each bond distance from a given tour is attributed to D. Gross and M. Mezard. By symmetry, this distribution is the same as the distribution of $b(t_1, t_2)$ between random tours t_1 and t_2. The cited result is that the number of common edges between two random tours t_1 and t_2, i.e., $n - b(t_1, t_2)$, approaches a Poisson distribution with mean = 2 as $n \to \infty$. We have studied this distribution somewhat more precisely. In what follows, we show that the number of edges in common has mean $= 2 * n/(n-1)$. We then describe an efficient method for exactly calculating the $b(t_1, t_2)$ distribution.

Fact B.1. *The expected number of edges in common between two random tours on n cities is* $2 * n/(n-1)$.

Proof. Each edge between cities occurs in exactly $(n-2)!$ different tours. Since each tour has n edges, the expected number of overlaps between a given tour and a random tour in the solution space S is equal to

$$\frac{n(n-2)!}{|S|} = \frac{n(n-2)!}{(n-1)!/2} = 2 * \frac{n}{n-1}. \qquad \square$$

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* $\sigma(n) = \langle 1, 2, \ldots, n-1, n \rangle$. Each $I(n+1, k)$ 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 $0 \leqslant k \leqslant n$ we can compute all the values $I(n+1, k)$ for $0 \leqslant k \leqslant n+1$. For a given n, the entire computation of all $I(n, k)$ values requires only $\Theta(n^3)$ time and $\Theta(n^2)$ space. We use the term *bond* to denote any adjacency in an n-city tour t that is also contained 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 in t when we insert city $n+1$ between the two cities forming a bond in t. One readily sees that breaking a bond will usually decrement by one the number of bonds remaining in t'; similarly, inserting $n+1$ next to city 1 or n will generally increment the number of bonds by one, while other positions for city $n+1$ will generally leave the number of bonds unchanged. The analysis considers eight cases for each k, $0 \leqslant k \leqslant n$, corresponding to the possible presence or absence of each of three "special" bonds $(1, 2)$, $(1, n)$ and $(n-1, n)$. The reader is referred to [6] for details.

References

[1] D.H. Ackley (1987), *A Connectionist Machine for Genetic Hillclimbing*, Kluwer, Dordrecht.
[2] S. Bacci and N. Parga (1989), "Ultrametricity, frustration and the graph colouring problem", *J. Phy. A: Math. General* **22**, 3023–3032.
[3] E.B. Baum (1986), "Toward practical 'neural' computation for combinatorial optimization problems", in: J. Denker (ed.), *Neural Networks for Computing*, American Institute of Physics.
[4] J.L. Bentley (1992), "Fast algorithms for geometric traveling salesman problems", *ORSA J. Comput.* **4**, 387–411.
[5] K.D. Boese and A.B. Kahng (1994), "Best-so-far vs. where-you-are: implications for optimal finite-time annealing", *Systems Control Lett.*, to appear.
[6] K.D. Boese, A.B. Kahng and S. Muddu (1993), "On the big valley and adaptive multi-start for discrete global optimizations", *Technical Report TR-930015*, UCLA CS Department.
[7] B. Bollobas (1985), *Random Graphs*, Academic Press, New York.
[8] T.N. Bui, S. Chaudhuri, F.T. Leighton and T. Sipser (1987), "Graph bisection algorithms with good average case behavior", *Combinatorica* **2**, 171–191.
[9] T.C. Hu, V. Klee and D. Larman (1989), "Optimization of globally convex functions", *SIAM J. Control Optim.* **27**, 1026–1047.
[10] D.S. Johnson (1990), "Local optimization and the traveling salesman problem, in: *Proc. 17th Internat. Coll. on Automata, Languages and Programming*, pp. 446–460.

[11] D.S. Johnson, C.R. Aragon, L.A. McGeoch and C. Schevon (1989), "Optimization by simulated annealing: an experimental evaluation, Part I, graph partitioning", *Oper. Res.* **37**, 865–892.
[12] D.S. Johnson and E. E. Rothberg, "Asymptotic experimental analysis of the Held–Karp lower bound for the traveling salesman problem", (to appear).
[13] S. Kauffman and S. Levin (1987), "Toward a general theory of adaptive walks on rugged landscapes", *J. Theoret. Biol.* **128**, 11–45.
[14] S. Kirkpatrick, C.D. Gelatt and M. Vecchi (1983), "Optimization by simulated annealing", *Science* **220**, 671–680.
[15] S. Kirkpatrick and G. Toulouse (1985), "Configuration space analysis of traveling salesman problems", *J. de Phys.* **46**, 1277–1292.
[16] E.L. Lawler, J.K. Lenstra, A. Rinnooy-Kan and D. Shmoys (1985), *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*, Wiley, New York.
[17] M. Mezard and G. Parisi (1986), "A replica analysis of the traveling salesman problem", J. Phys. **47**, 1285–1296.
[18] H. Mühlenbein (1989), "Parallel genetic algorithms, population genetics and combinatorial optimization", in: *Proc. Internat. Conf. on Genetic Algorithms*, pp. 416–421.
[19] H. Mühlenbein, M. Georges-Schleuter and O. Krämer (1988), "Evolution algorithms in combinatorial optimization", *Parallel Comput.* **7**, pp. 65–85.
[20] C. Papadimitriou (1992), "The complexity of the Lin–Kernighan heuristic for the traveling salesman problem", *SIAM J. Comput.* **21**, 450–465.
[21] S.A. Solla, G.B. Sorkin and S.R. White (1986), "Configuration space analysis for optimization problems", in: E. Bienenstock et al. (eds.), *Disordered Systems and Biological Organization*, Springer, Berlin, pp. 283–292.
[22] G.B. Sorkin (1991), "Efficient simulated annealing on fractal energy landscapes", *Algorithmica* **6**, 367–418.
[23] G.B. Sorkin (1993), personal communication.
[24] N. Sourlas (1986), "Statistical mechanics and the traveling salesman problem", *Europhys. Lett.* **2**, 919–923.
[25] N.L.J. Ulder, E.H.L. Aarts, H.-J. Bandelt, P.J.M. van Laarhoven and E. Pesch (1990), "Genetic local search algorithms for the traveling salesman problem," in: H.-P. Schwefel and R. Männer, (eds.), *Parallel Problem Solving from Nature*, Springer, Berlin, pp. 109–116.
[26] Y.C. Wei and C.K. Cheng (1990), "A two-level two-way-partitioning algorithm", in: *Proc. IEEE Internat. Conf. on Computer-Aided Design*, pp. 516–519.
[27] E. Weinberger (1990), "Correlated and uncorrelated fitness landscapes and how to tell the difference", *Biol. Cybernet.* **63**, 325–336.