ELSEVIER Physica A 243 (1997) 77–112

PHYSICA A

Search-space smoothing for combinatorial optimization problems

Johannes Schneider[a,*], Markus Dankesreiter[a], Werner Fettes[a], Ingo Morgenstern[a], Martin Schmid[a], Johannes Maria Singer[b]

[a] *Fakultät Physik, Universität Regensburg, Universitätsstr. 31, D-93053 Regensburg, Germany*

[b] *Physikinstitut, Universität Zürich, Winterthurerstrasse 190, CH-8057 Zürich, Switzerland*

Received 5 February 1997

Abstract

Commonly there are two types of local search approaches known to treat combinatorial optimization problems with very complex search-space structure: One is to introduce very complicated types of local move classes, allowing a bypass of high energetic barriers separating different minima. The second is introducing a control-parameter (i.e. temperature in physics terminology) dependent state space walker, which is – depending on this control parameter – more or less easily able to climb over barriers. A third, less well-known, but very obvious approach is to smooth the search space, i.e. to eliminate barriers between low-energy configurations and therefore to allow a fast and easy approach to the global optimum. This procedure will be discussed in depth in the following work.

PACS: 02.50.-r; 02.50.Ga; 02.70.Lq; 89.20.+a; 89.80.+h; 89.90.+n

Keywords: Optimization; Monte Carlo; Traveling salesman; Great deluge; Smoothing; Local search

1. Introduction

Given N "cities" and the distances between them, the Traveling Salesman Problem (TSP) consists of finding the length of the shortest "tour" (path) visiting every city exactly once, where the tour length (i.e. the objective of the optimization problem) is the sum of the city-to-city distances along the tour [1–3]. This problem, while simple to state, is very difficult to solve, but perhaps of its straightforward formulation, the TSP has attracted wide interest in computer science, in operations research and, in the last two decades, in physics. In computer science, most of the work has concentrated

* Corresponding author. Tel: +49/941-943-2042; fax: +49/941-943-1968; e-mail: johannes.schneider@physik.uni-regensburg.de.

on determing the algorithmic complexity of the problem and finding cases of easily optimizable instances (i.e. configurations of cities). In operations research, the focus has been on developing efficient algorithms for finding the optimum tour; with the current state of the art, for instances, with a few hundred cities very close to the optimum or even optimal tours can be found on a workstation within a couple of minutes. In physics, much effort has concentrated on transfering physical concepts into algorithms for finding good approximate or optimal solutions. One very successful such algorithm is Simulated Annealing [4], a close relative is Threshold Accepting [5], and both physicists and mathematicians have studied issues pertaining to properties of these algorithms.

A large class of combinatorial optimization problems are structurally quite similar to the TSP, and usually a simple mapping onto a TSP-like formulation can be found. A common property is that the set of solutions or configurations is finite. This set together with an appropriate metric leads to a discrete phase space. To link configurations in state space, i.e. to transform a particular configuration σ in a given way into another configuration τ, we use a mapping procedure called moves $m : \sigma \rightarrow \tau$. The moves define the state-space neighborhood of every configuration, and the state space together with this neighborhood structure results in a search space. In a physicists' terminology the hyper surface built by the points of the search space and the corresponding energy is called the "energy landscape" (or "fitness landscape" for evolutionary biology). This name refers to the corrugated valley–hill–structure, with valleys corresponding to local minima of the objective (energy) and hills to maxima [2]. Although few mathematical problems have attracted as much attention as the TSP, very little is known on the structure and properties of its landscape. Local Search Algorithms are moving like a walker through this terrain, they proceed forward by making small changes (moves) in the configurations and use only simple types of moves. The result of every step forward is only slightly different from its neighbors. This causes a severe drawback for local algorithms: In a severly corrugated energy landscape with steep valleys and high mountains the procedure (our walker searching for the global minimum) gets easily stuck in one of the deep canyons, unable to climb out and to reach the global energy minimum [1,3].

From the point of view of a walker in this energy terrain there are generally three possibilities to avoid getting stuck: The first one is to introduce more moves to generate a bigger neighborhood, which means that more possible ways to walk around energy barriers exist. Literally speaking, even paths through the energy mountains are possible, which may be associated in physics terminology with a "barrier tunneling" (in the sense that certain barriers are eliminated in the new search space due to the introduction of the new moves). For example, if we want to optimize a complex atomic cluster of crystal structure [6], this would be equal to direct tunneling moves between certain known energetically low-lying configurations, like e.g. from orthorhombic to tetragonal without climbing over energetic barriers in terms of local atomic displacement moves. But an increasing number of possibilities leading through the landscape increases the probability to miss the optimum if we use a constant number of move attempts.

The second "brute force" approach is to intermediately introduce energy to the walker (i.e. to feed and strengthen our friend), which is successively withdrawn in a more or less complicated schedule. Again in physics language this means introducing a system temperature for the energy landscape walker, which allows us to lift the walker up or lower down the path and therefore to help the search process over high barriers separating local optima from the global one. This is the concept used in algorithms like Simulated Annealing or Threshold Accepting. And finally, the third way, which we want to discuss in this paper, is simply to smooth the landscape and make it easier to jump over energy barriers. "Smoothing" ideally means to reduce the number of minima in the landscape to only one, the global minimum. $\mathscr{H}(\sigma)$ denotes the energy (length, "Hamiltonian" or more commonly the optimization objective) of the system in the state σ. There are four possibilities to achieve a smoothing of the energy landscape:

- We change $\mathscr{H}$ for each state and smooth with a function f. Each state gets a new energy $f(\mathscr{H}(\sigma))$.
- We may smooth $\Delta\mathscr{H} = \mathscr{H}(\tau) - \mathscr{H}(\sigma)$, the energy difference between two neighbor solutions σ and τ, which can be connected by a member of the move class, with a function $f(\Delta\mathscr{H})$ instead of the energies themselves.
- If the Hamiltonian of the system is more complicated, and $\mathscr{H}(\sigma) = \sum_i \mathscr{H}_i(\sigma)$, i.e. the total energy $\mathscr{H}$ consists of a number of different contributions (like penalty and correlation terms), it might be necessary to smooth the particular contributions to the Hamiltonian with different functions, leading to an energy $\sum_i f_i(\mathscr{H}_i(\sigma))$ for the smoothed system.
- Similarly, we can smooth $\Delta\mathscr{H}$ for complicated systems by $\sum_i f_i(\Delta\mathscr{H}_i)$.

In the following discussion we will extend the basic idea originating from Jun Gu's [7] work and show the application of a number of realizations for this smoothing idea for the traveling salesman problem; as indicated at the beginning, this procedure can be easily adapted for any combinatorial optimization problem with similar search space considerations.

2. Example: the Traveling Salesman Problem

2.1. Traveling Salesman Benchmark Problems

We want to present our results for two special symmetric instances (i.e. energy/costs d_{ij} for travelling from i to j is equal to d_{ji}) of the Traveling Salesman Problem, the 442 drilling holes on a printed circuit board (PCB442, Fig. 1) by Grötschel et al. [8] and the 532 US cities with AT&T regional offices (ATT532, Fig. 2) by Padberg and Rinaldi [9]. The data for both problems can be received from the TSP-Library by Reinelt [10]. Besides being important for operations research (e.g. tour planning) itself this class of symmetric TSP problems has extended applications in crystallography, electronics, protein folding, etc.

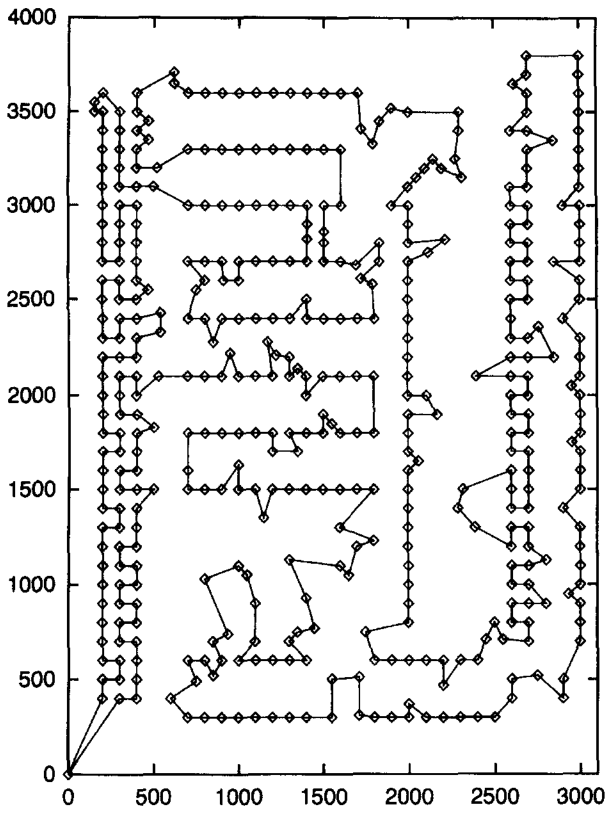


Fig. 1. The figure shows a solution for the PCB442 printed circuit board instance [1] with the known optimal length of 50783.547513735... (REAL*8) [3,10].

In the following discussion we use the term "solution" for a closed path obeying the TSP definition, i.e. touching every city exactly once, and "optimal solution" then denotes the shortest (i.e. energetically lowest) of all solutions. Only solutions are part of our state space, the terms "tour" and "solution" are equivalent; technically the obvious choice for its representation is an ordered list of cities.

2.1.1. The Grötschel–Jünger–Reinelt Problem (PCB442)

Looking at Fig. 1 we see that some parts of the problem of the 442 drilling holes appear optimally solvable at a first glance, especially some left–right connections in the left-half of the board, whereas in other parts, e.g. in the upper-half of the left three columns, it is not quite clear how to connect the points. This problem has a highly degenerate ground state [1,3] and there is a very large number of nearly optimal solutions quite close to the global optima. Moreover, there are many optimization algorithms which have severe difficulties with this problem. For the 442-problem we use the two-dimensional Euclidean REAL*4-metric.

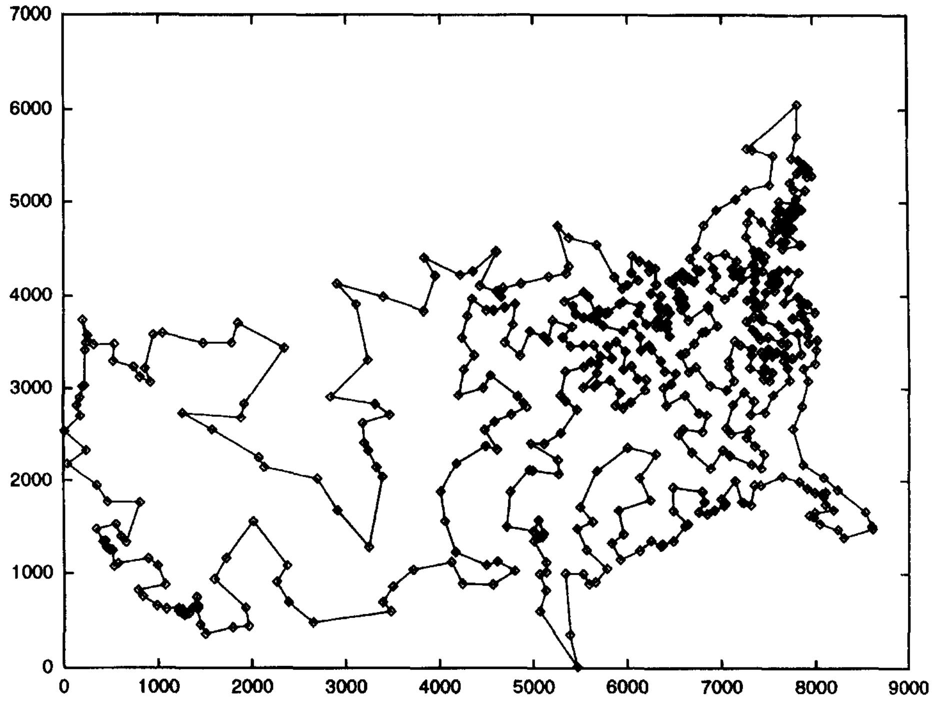


Fig. 2. Here is one of our shortest solutions for the ATT532 instance with a length of 27 735, slightly longer than the known optimal solution of length 27 686.

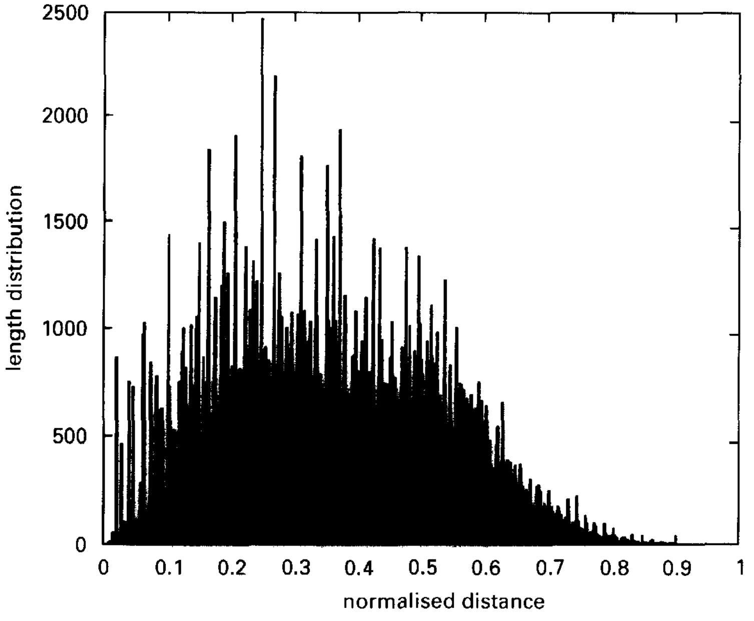
2.1.2. *The Padberg–Rinaldi Problem (ATT532)*

The TSP through 532 cities of the USA is also very well known because of the difficulty that the distances to the next node are very short at the east coast, whereas in other regions of the USA they are very long. This leads to a kind of separation of the problem into subproblems with different length scales as shown in Fig. 2. For this problem another distance definition is commonly in use, the pseudo-Euclidean ATT-distance matrix, as described in [10].

The central difference between the two problems is shown in the two length distribution histograms, (Fig. 3). The PCB442 length histogram is quite noisy, but otherwise uniform over the whole range. In contrast, the ATT532 histogram is smoother, but has two clearly distinct, overlapping distributions, reflecting the above-described, different length scales in the US east coast region and the western parts of the country. Therefore, we use the PCB442 instance as a standard example to test our algorithm, and cross check the results on the ATT532 instance to exclude artifacts of a particular instance and length distribution.

2.2. *Moves*

We start the optimization process with a randomly generated solution, i.e. a random starting point in state space, and change it with moves in order to get a new route. Fig. 4



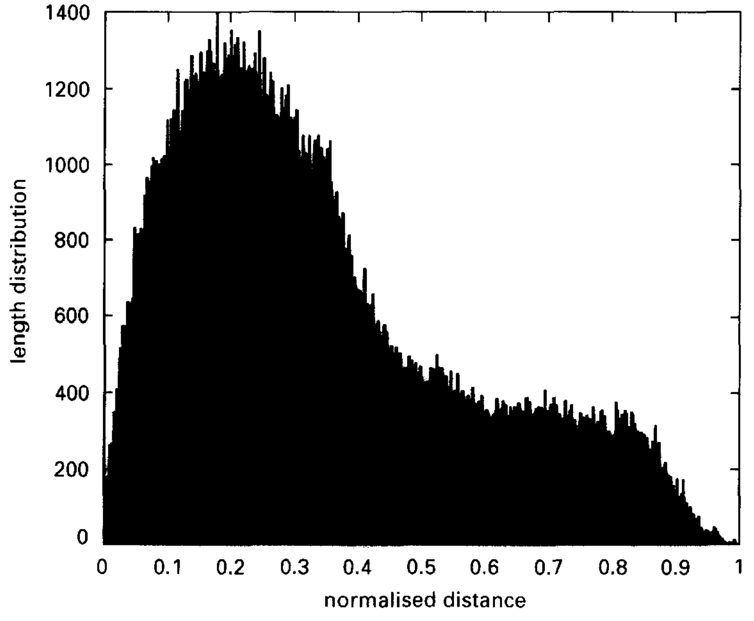


Fig. 3. Here we show histograms of the length distribution in the distance matrices of PCB442 (top) and of ATT532 (bottom). The maximum distance in the 442-problem is 4841.487..., in the 532-problem 2790, the mean distance $\bar{d}$ for the 442-problem is 1747.91..., for the 532-problem $\bar{d} = 962.6\ldots$ We have rescaled the distance matrix of both problems so that the maximum distance is 1, therefore we get $\bar{d}(442) = 0.361\ldots$ and $\bar{d}(532) = 0.345\ldots$

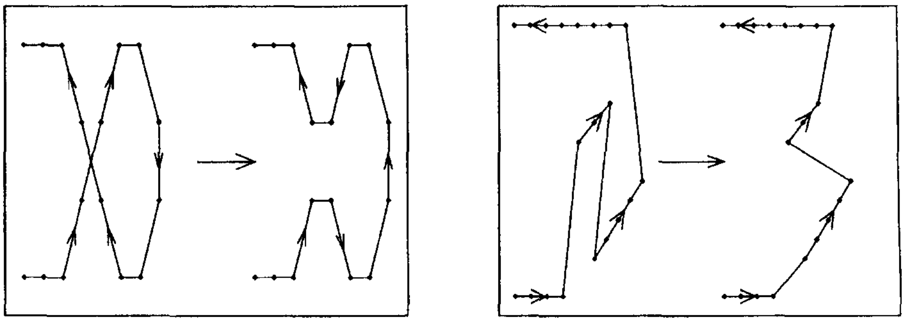


Fig. 4. We use two moves, the lin-2-opt shown on the left and the lin-3-opt shown on the right half of the figure.

illustrates the Lin-type moves to transform one solution into another by local displacements used in this paper, the lin-2-opt and one of four possibilities for the lin-3-opt.

The lin-2-opt changes the direction of a part of the tour. Using the lin-2-opt one usually obtains better results than with an exchange of two points ("transposition") [11]. For the symmetric TSP it is a simpler move than the exchange of two points because only two connections are cut. For the asymmetric TSP it is much more difficult since the directions between nodes in the part to be turned have to be considered.

The used lin-3-opt exchanges a part of the tour with its succeeding part without changing their direction. This is the one out of the four possibilities for the lin-3-opt which can only be constructed with three lin-2-opts, the others can be constructed with two lin-2-opts. This is the reason why we used this one. The lin-3-opt is introduced because of our experience with the peculiar situation in the 442-Problem, especially with the high degeneracy of the distance matrix; in particular, for several nodes the distance to the next two or four nodes is the same. A speciality of this move is that it just displaces a point in the tour if one of the two succeeding parts of the tour to be exchanged consists only of one point. We felt no need to introduce further high-order lin-n-opts: a lin-4-opt exchanging two not succeeding parts of the tour without changing their direction had no extensive influence on the results. One usually obtains better solutions with the lin-3-opt than using just the lin-2-opt because the neighborhood structure of the lin-3-opt is much larger [11,12].

We use both types (the ratio is 5 times as many lin-3-opts as lin-2-opts), since a combination, taking advantages of both moves, has proven to be a good basis [1,3]. In addition, we do not use higher moves, like lin-4-opts, in order to avoid an interference with the local search effect; this effect is reduced with an increasing number of neighboring states.

3. Local Search Algorithms

Local Search is very efficient to determine extrema in combinatorial optimization problems. Metropolis et al. advanced the idea not to choose simply successive states

$\{\sigma_i\}$ along a random walk in state space independently of each other, but to construct a Markov chain where each state σ_{i+1} is generated with a probability depending on its thermodynamic weight from a previous state σ_i. The suggested moves for our example, the lin-2-opt and the lin-3-opt, do not change a configuration much, there are only two or three new edges. Therefore, we perform only small steps in a phase space which consists of the set of configurations and the Hamming distance d_H as metric. $d_H(\sigma, \tau)$ is defined as the number of edges which are not common to both configurations. The various Local Search Algorithms all use this principle to change the configurations only very slightly, they differ in the choice of a suitable transition probability $p(\sigma_i \to \sigma_{i+1})$.

3.1. *Greedy Algorithm* (*GA*)

The simplest choice for this transition probability is the following: let $l(\sigma_i)$ be the length of the configuration σ_i, $\Delta l = l(\sigma_{i+1}) - l(\sigma_i)$, then the transition probability can be chosen as

$$p(\sigma_i \to \sigma_{i+1}) = \begin{cases} 1 & \text{if } \Delta l \leqslant 0\,, \\ 0 & \text{otherwise}\,. \end{cases}$$

This means that every improvement (i.e. energy/cost decrease) is accepted. If a move leads to an increase of the objective then it is not accepted.

Using the GA we run from a start solution into a local optimum usually in the state-space neighborhood and get trapped because the algorithm provides no concept to escape. Therefore, we have to use more elaborate definitions for the transition probability allowing also under some conditions an energy increase.

3.2. *Simulated Annealing* (*SA*)

Simulated Annealing combines the Boltzmann equilibrium distribution function

$$\pi_{\text{equ}}(\sigma_i) = \frac{1}{Z}\exp(-\beta l(\sigma_i)) \quad \text{with } Z = \sum_i \exp(-\beta l(\sigma_i)) \quad \text{and} \quad \beta = \frac{1}{T}$$

with the condition of detailed balance

$$\pi_{\text{equ}}(\sigma_i)p(\sigma_i \to \sigma_j) = \pi_{\text{equ}}(\sigma_j)p(\sigma_j \to \sigma_i)\,.$$

The obvious choice for the transition probability, according to these conditions, is the Metropolis criterion [13]

$$p(\sigma_i \to \sigma_{i+1}) = \begin{cases} 1 & \text{if } \Delta l \leqslant 0\,, \\ \exp(-\beta \Delta l) & \text{otherwise}\,. \end{cases}$$

Furthermore, it fulfills the condition of ergodicity, that means each state can be reached with a positive finite probability.

3.3. The great deluge algorithm (GDA)

The transition probability of the Great Deluge Algorithm [14] is given by

$$p(\sigma_i \to \sigma_{i+1}) = \begin{cases} 1 & \text{if } l(\sigma_{i+1}) \leqslant \mathcal{T}, \\ 0 & \text{otherwise}. \end{cases}$$

Therefore, the Great Deluge Algorithm fulfills the condition of detailed balance because it performs a random walk in a subset $\Gamma_{\mathcal{T}}$ of the state space Γ, $\Gamma_{\mathcal{T}} = \{\sigma \mid l(\sigma) \leqslant \mathcal{T}\}$, so that each $\sigma \in \Gamma_{\mathcal{T}}$ has the same probability $\pi(\sigma) = 1/|\Gamma_{\mathcal{T}}|$ and the transition probability is given by 1. States lying above the level $\mathcal{T}$ have the probability 0 and are not accepted from states under the level, i.e. the ergodicity condition is violated according to the whole canonical system. But if we interpret the GDA as a microcanonical algorithm (as described in [2]) we see that from the microcanonical point of view the ergodicity criterion is not violated, because the system can reach all states in a energy shell below $\mathcal{T}$.

3.4. Computational results

3.4.1. Results for the Greedy Algorithm

The only interesting observable in case of the Greedy is the length of solutions at the end. Because of the relatively large number of moves leading to $\Delta l = 0$ for the PCB442 problem, we vary the transition probability slightly so that a move with $\Delta l = 0$ is accepted with probabilities $p = 0, 0.5$ or 1, which leads to different results:

transition probability p	0	0.5	1
mean length l ($\times 10^3$)	53.8	53.5	53.3

Because each corresponding error bar is smaller than 100 for an ensemble average over 50 runs these results imply that a transition probability of 1 for moves with $\Delta l = 0$ is the best choice. This seems to contradict the results for the Edwards–Anderson Ising spin glass, that a choice of $p = 0.5$ for $\Delta l = 0$ should be superior, however not all results from the spin glass theory can be transferred on the Traveling Salesman Problem. Because of this result we also used $p = 1$ for $\Delta l = 0$ working with Simulated Annealing and for $l(\sigma_{i+1}) = \mathcal{T}$ using the Great Deluge Algorithm. In Section 4.5 more results for the Greedy Algorithm will be provided.

3.4.2. Results for Simulated Annealing

To compare the concept of search space smoothing we first show a result of a "conventional" Simulated Annealing optimization for the PCB442 problem (Fig. 5). When we discuss SA algorithms we want to mention and include the whole conceptual apparatus of SA-like methods; this explicitly takes related and hybrid methods into account.

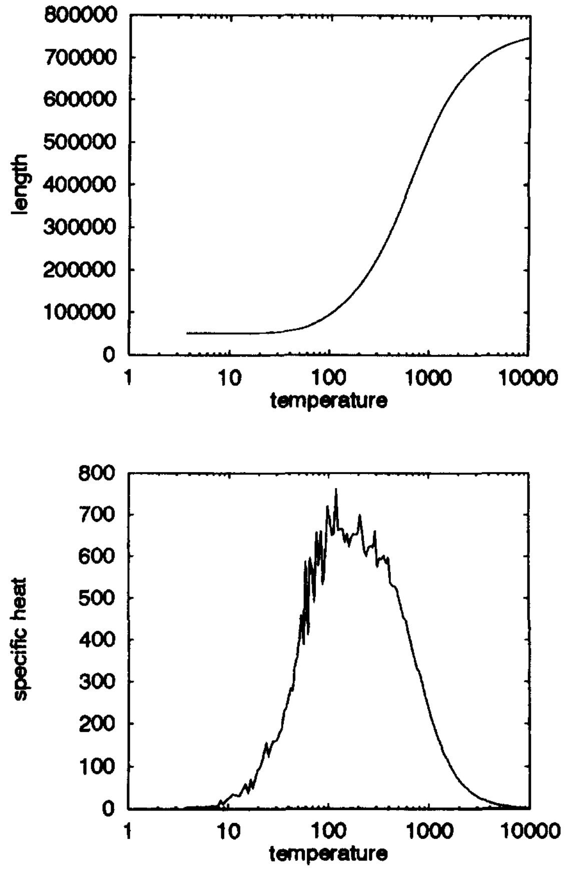


Fig. 5. Here we show results for an optimization of the problem of the 442 drilling holes (PCB442) with Simulated Annealing, the decrease of the length with decreasing temperature (top) and the specific heat of the problem (bottom).

The upper part shows the length $\mathscr{L}$ (energy) of the tour as a function of the "temperature" T (i.e. control parameter), the lower part the corresponding "specific heat",

$$C = \frac{\langle \mathscr{L}^2 \rangle - \langle \mathscr{L} \rangle^2}{T^2} .$$

The specific heat has a pronounced maximum around $T \approx 180$, indicating the temperature range of maximum "rearrangement activity" in terms of a transition between a disordered and an ordered regime. Conceptually very closely related to the Simulated Annealing approach is the theoretical physics concept of spin glasses [15].

3.4.3. Results for the Great Deluge Algorithm

In Fig. 6 we show the time development of the control parameter ("water level") $\mathscr{T}$. The condition to lower the level is as follows: usually, one obtains the beginning level $\mathscr{T}_0$ from a randomly generated initial configuration. We, however, decided to start at a level $\mathscr{T}_0$ which is the length of the worst (i.e. energetically highest) solution of the problem known to us, 1129442.0 for the 442-problem and 716 823 for the 532-problem [1]. This sets the initial level to the highest possible values. The new level $\mathscr{T}$ is calculated from the old level $\mathscr{T}_{\text{old}}$ and the length $\mathscr{L}$ of the actual solution as

$$\mathscr{T} = \begin{cases} \mathscr{L} & \text{if } \mathscr{L} \geqslant 0.99\mathscr{T}_{\text{old}}\,, \\ 0.99\mathscr{T}_{\text{old}} & \text{otherwise}\,. \end{cases}$$

At the beginning $\mathscr{T}$ follows $0.99\mathscr{T}_{\text{old}}$, then there is a range where both choices exist and after that it is dominated by $\mathscr{L}$. If we have a look at the first two graphs we see that $\mathscr{T}$ is lowered exponentially when $\mathscr{T}$ is chosen equal to $\mathscr{L}$. In the middle of Fig. 6 the same graph as above is shown again with a logarithmic y-axis; a fit shows the dependency of the level of the temperature step t by $5.5 \times 10^5 \exp(-1.2 \times 10^{-3}t)$ for the PCB442, an equivalent analysis gives $4.1 \times 10^5 \exp(-1.5 \times 10^{-3}t)$ for the ATT532-problem in a certain range of the run. The prefactors differ slightly for different runs but we could reproduce the exponential decrease itself. There are only small deviations from the exponential function at high levels where the transition between the two choices happens and larger ones at low levels for the Grötschel problem.

Therefore, $\mathscr{T}$ is lowered only very slowly by this condition, and $\mathscr{L}$ is obviously very near to the actual level, corresponding to the mean value of the length $\langle\mathscr{L}\rangle$. The lower left part of Fig. 6 shows in a blow-up from the upper left corner of this figure the range where the level converges to the length (energy) of the system and starts to press it down.

The deviation of $\mathscr{T}$ from the exponential curve at small $\mathscr{T}$ can also be explained: because the distribution of the states goes obviously exponentially with the length in a wide range between the optimum and the solutions in the noisy range, each configuration must have exponential more neighbors energetically lying above of it than lying lower than it. Therefore, the system stays a longer time in the states which are quite close to the level because it randomly selects configurations lying energetically higher most of the time. For a wide time range this has no additional effect (except of the decrease of the acceptance rate) because the interval in which the system cannot accept every move is very small in comparison to the whole possible energy interval (The maximum distance is 4841.487... whereas the possible energy interval is two orders of magnitude wider, e.g. at a level of 500 000, because the optimum has a length of slightly above 50 000.). But for small levels these two length scales coincide so that the system really sticks at the level.

Let

$$\mathscr{A} = \frac{\mathscr{T} - \langle\mathscr{L}\rangle}{\mathscr{T}}$$

be the deviation of the mean value of the length to the level. Fig. 6 shows at the top right corner that $\mathscr{A}$ is of order 10^{-4}. Obviously, the number of states decreases with decreasing length of the TSP dramatically. The system performs a random-walk through the subset $\Gamma_{\mathscr{T}}$, nearly all measured states have a length $\mathscr{L} \approx \mathscr{T}$. Therefore,

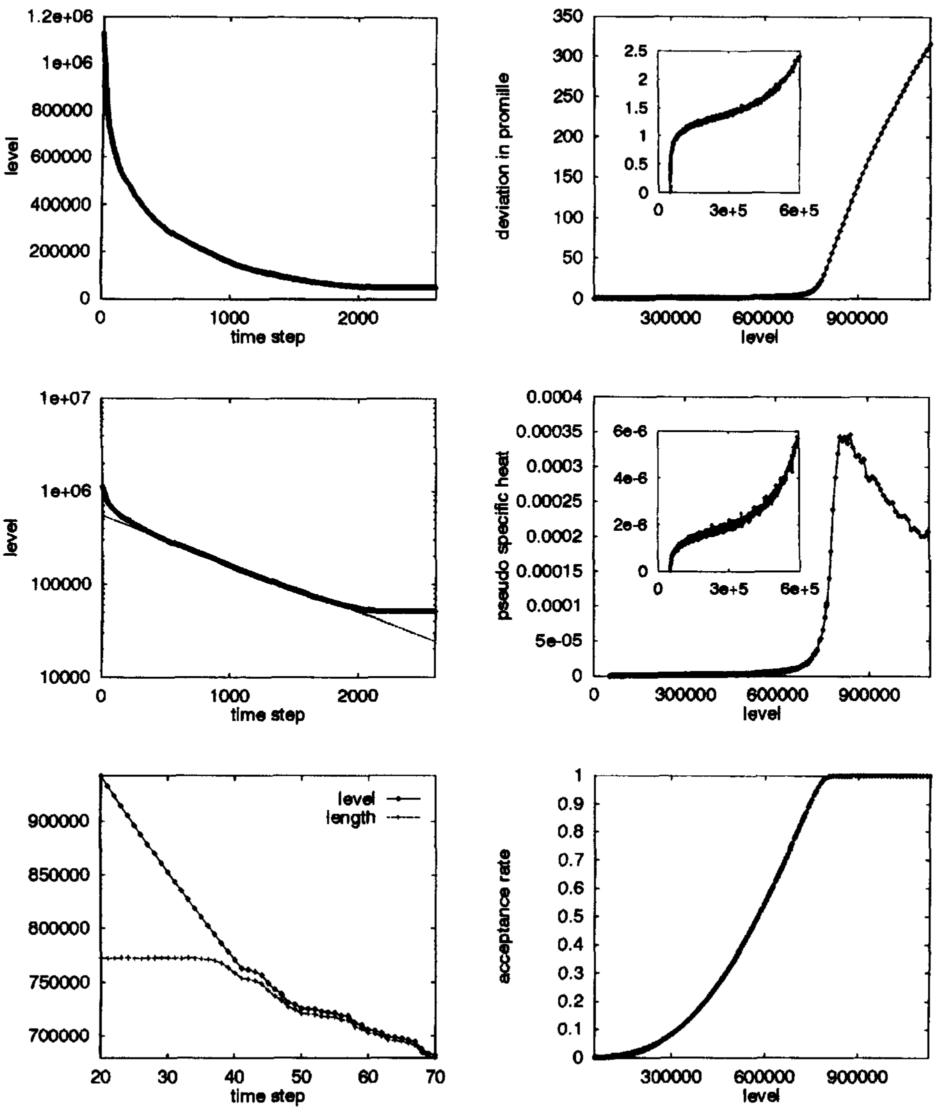


Fig. 6. In these graphs the properties of the Great Deluge Algorithm are documented very well: in the left column there is at the top the development of the level $\mathcal{T}$ as a function of simulation time, in the middle we show the same data in a logarithmic plot with a fit curve attached and at the bottom we give an enlarged plot of the range in which the level approaches the length (energy) of the system and begins to force down the energy. In the right part we see at the top the deviation $\mathcal{A}$ of the mean length from the level in promille, in the middle we show the variable $\mathrm{Var}(\mathcal{L})/\mathcal{T}^2$ which is a formal equivalent to the specific heat in case of Simulated Annealing, and which has for the GDA a similar behavior at a first glance. But a zoom-in at the range of smaller levels shows that it is similar to $\mathcal{A}$. Furthermore, it has a peak exactly in the range in which the level meets the mean length of the system. Finally, we show at the bottom of the right-half the acceptance rate. We see that the system obviously adheres to the level more and more intensely.

we conclude that there must be more states than in the range $\mathscr{L} \ll \mathscr{T}$. Because of the cooling behavior we may expect an exponential decrease of the number of states with decreasing length in wide ranges, and we may call the Great Deluge Algorithm a "beach-fan" because it seems to favorite the "coast regions".

Furthermore, we look at the variable

$$\mathscr{B} = \frac{\mathrm{Var}(\mathscr{L})}{\mathscr{T}^2}$$

(Fig. 6, right middle picture), which might be of interest because it is formally equivalent to the specific heat if we work with Simulated Annealing. $\mathscr{B}$ looks at a first glance like a specific heat: its peak is not in the control parameter range in which the system orders, but in the range in which the level reaches the lengths of the solutions. Furthermore, $\mathscr{B}$ shows a similar behavior as $\mathscr{A}$ for small levels, as can be seen in the insets.

Finally, we show the acceptance rate. It is 1 until the level begins to push down the length of the system and decreases quadratically with the "water level".

4. Search-space smoothing (SSS)

4.1. The idea

Due to the rough valley-hill landscape of the search space a local search algorithm usually gets trapped at a local suboptimal solution, as shown in Fig. 7. To avoid this obstacle one may ask the question whether there is a possibility to smooth the search space and to cut away all the local "traps", i.e. to deform the energy landscape in a way that most of the local structure disappears and ideally only one minimum, the global one, remains (see Fig. 7). This basic idea is simple: its realization, described in the following sections, is much more difficult because we have to introduce a new metric which differs from the previously used two-dimensional Euclidean metric in the sense that the optima including the ground state may be displaced. If we use, e.g. the logarithm of the distances instead of the original values we may construct the following example: let a, b be edges of a locally optimal solution σ with edge lengths $a = b = e^3$. There shall be a state τ neighbored to the optimum σ by one lin-2-opt. If we exchange the edges a, b with two new edges a', b' we get τ from σ. The lengths of these edges shall be $a' = e^4, b' = e$. We see that

$$e^4 + e^1 > e^3 + e^3 \quad \text{but} \quad 4 + 1 < 3 + 3\,,$$

the local optimum is transferred from σ to τ if we use the logarithm of the edge lengths instead of the original edge lengths. Therefore, we first have the problem to construct very good solutions for an altered metric, which produces a very smooth solution space, then to change back this metric gradually into the original one. "Besides" it is necessary to adapt the solution to the new metric so that one has finally – in terms of the original metric – an optimal solution.

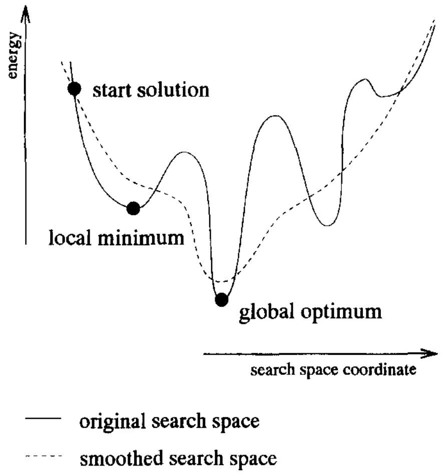


Fig. 7. Here we depict a schematic fraction of energy landscape of a simple problem. If we use the Greedy as local search algorithm in the original (i.e. unsmoothed) landscape, we would get stuck at the local optimum near the starting point. Even if we use, e.g. GDA we cannot be sure to reach the global optimum at the end. However, if we smooth the search space in a way that only one minimum, the global optimum, remains we reach the global optimum independently from our start configuration.

Every conventional local search algorithm can be used in connection with this smoothing concept for finding the minimum in the smoothed landscape [7], the simple Greedy Algorithm may be an obvious first choice.

4.2. A simple example

In order to demonstrate this general idea by a simple example we have a look at a small asymmetric TSP consisting of 4 edges on a square, the distance $d(2,1)$ however shall be $\sqrt{2}$ so that we get an asymmetry and the distance matrix shown in Table 1. The only allowed move shall be turning around one edge (nearest-neighbor exchange) so that the system has the ground state (1234) and a local minimum (4321), with total lengths 4 and 4.41, respectively, the other four states are neighboring both of them, as shown in Table 1. All possible paths through the corresponding valley hill landscape are shown in the upper part of Fig. 8. The mean length of the states is

$$\bar{d} = 4 \times \frac{7+5\sqrt{2}}{12} \approx 4.69\,.$$

The lower part gives an impression of the neighborhood structure and the corresponding energy landscape of this small problem. As one can see no connection between the two minima, configurations 1 and 6, exists.

We want to introduce a new simple rule for smoothing: first all states are supposed to be of length $\bar{d}$, in the second step the state which original length deviates most from $\bar{d}$ returns to its original length, then the state with the next largest absolute deviation,

Table 1
A small example for a TSP

	1	2	3	4
1	0	1	$\sqrt{2}$	1
2	$\sqrt{2}$	0	1	$\sqrt{2}$
3	$\sqrt{2}$	1	0	1
4	1	$\sqrt{2}$	1	0

$\longrightarrow$

State	Sequence	Length	Neighbors
1	(1234)	4	2, 3, 4, 5
2	(1243)	$2+2\sqrt{2}$	1, 3, 5, 6
3	(1324)	$2+2\sqrt{2}$	1, 2, 4, 6
4	(1342)	$1+3\sqrt{2}$	1, 3, 5, 6
5	(1423)	$2+2\sqrt{2}$	1, 2, 4, 6
6	(1432)	$3+\sqrt{2}$	2, 3, 4, 5

Note: It consists of 4 nodes on the edges on a square, has therefore 6 states. The distance $d(2,1)$ however shall be $\sqrt{2}$ so that we have an asymmetric TSP. On the left there is the asymmetric distance matrix for this problem, on the right the sequences, lengths and neighbors of all states.

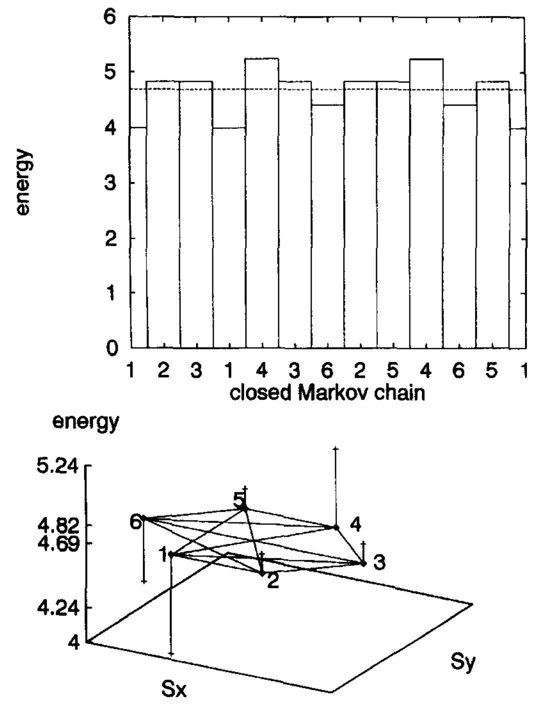


Fig. 8. This picture shows a closed Markov chain with minimal length through the search space of the toy problem of Table 1. Each connection between different points is exactly used once. We see that the state 1 is the global minimum of the system. The dotted line shows the mean length 4.69 of all states. At the bottom we see the two-dimensional projection of the search space (S_x, S_y). The lines between the states show the possible connections between them. Furthermore, we indicate the valley–hill–landscape of the problem by the vertical lines, which show the difference between the length of one state and the mean length of all states.

and so on, until finally all states have their original lengths. This leads to the following sequence:

1. We have a plateau in search space, i.e. all states have equal length.
2. The state # 1 gets back its original length, the other states remain on the plateau. $\rightarrow$ # 1 becomes the global minimum of the system.
3. The state # 4 returns to its original position in the energy landscape and becomes the global maximum of the system.
4. The state # 6 becomes a local minimum of the system.
5. The states # 2, # 3 and # 5 return to their original length.

Therefore, we already make use of the search-space smoothing at step 2 and jump into the global optimum.

Of course, this is only a simple example, but it shows the potential of this idea. In practical problems, we have to use more elaborate schedules for search-space smoothing because an exact enumeration of a real problem is usually impossible, since it costs too much computation time. Therefore, we have to find a rule to change the values in the distance matrix itself.

4.3. *The ansatz by Jun Gu*

Jun Gu provides an ansatz [7] for this problem: He suggests to alter the metric in dependence of an additionally introduced control parameter α such as

$$d_\alpha(i,j) = \begin{cases} \bar{d} + (d(i,j) - \bar{d})^\alpha & \text{if } d(i,j) \geqslant \bar{d}\,, \\ \bar{d} - (\bar{d} - d(i,j))^\alpha & \text{if } d(i,j) < \bar{d}\,, \end{cases}$$

with

$$\alpha \geqslant 1, \quad \bar{d} = \frac{1}{N(N-1)} \sum_{\substack{i,j=1 \\ i \neq j}}^{N} d(i,j)\,.$$

All distances $d(i,j)$ have to be normalized to the interval [0; 1]. If one has a problem with larger distances one has to divide all $d(i,j)$ by the maximum distance $d_{\max} = \max(d(i,j))$. This linear transformation rescales all lengths; however there is no real change of the phase space. Additionally, this has the effect that $0 < \bar{d} < 1$.

If $\alpha \gg 1$ then $d_\alpha(i,j) \approx \bar{d}$ because $|d(i,j) - \bar{d}| < 1$, and we are left with a nearly flat plateau. If α is decreased from a high positive value to 1 we get more and more corrugations in the plateau; finally, we return to the original landscape for $\alpha = 1$. This algorithm is based on the SSS idea that it is possible to transfer the good solution for high values of α into a good solution according to the original metric.

Jun Gu uses the Greedy algorithm as local search procedure.

4.4. *Further formulas for smoothing*

4.4.1. *Linear smoothing*

The simplest smoothing approach is linear smoothing:

$$d_\alpha(i,j) = \bar{d} + \alpha(d(i,j) - \bar{d}),$$

with $0 \leqslant \alpha \leqslant 1$. With this kind of smoothing the original search-space landscape is preserved, the local optima are unchanged because this formula provides only a linear transformation of the original distance matrix. Therefore, we expect that this formula leads to no improvement over a pure local search algorithm on the original landscape.

4.4.2. *Power-law smoothing*

Jun Gu's ansatz can be called power-law smoothing since the control parameter is the exponent α, which weighs the deviation of the distance $d(i,j)$ between the nodes i and j to $\bar{d}$.

In the following sections we provide further recipes to smooth the search space in analogy to the Jun Gu formula. They are also driven by the deviation and a control parameter α in order to construct a series of more or less smoothed search spaces:

4.4.3. *Exponential smoothing*

A good choice for search-space smoothing is exponential smoothing:

$$d_\alpha(i,j) = \begin{cases} \bar{d} + \dfrac{\alpha}{\exp\left(\dfrac{\alpha}{d(i,j) - \bar{d}}\right) - 1} & \text{if } d(i,j) \geqslant \bar{d}, \\[2ex] \bar{d} - \dfrac{\alpha}{\exp\left(\dfrac{\alpha}{\bar{d} - d(i,j)}\right) - 1} & \text{if } d(i,j) < \bar{d}, \end{cases}$$

with $\alpha \searrow 0$. We have $d_\alpha(i,j) \approx \bar{d}$ for large α and $d_0(i,j) = d(i,j)$. The transition between the totally smoothed landscape and the original landscape should be much sharper with this formula than with the ansatz of Jun Gu.

4.4.4. *Hyperbolic smoothing*

Another simple formula is hyperbolic smoothing

$$d_\alpha(i,j) = \bar{d} + \frac{1}{\alpha}(d(i,j) - \bar{d})$$

with $\alpha \geqslant 1$. Also this kind of smoothing provides only a linear transformation of the energy landscape. Furthermore, it is similar to linear smoothing because one has only to replace α by $1/\alpha$, but we have another distribution of the calculation time with this formula.

4.4.5. Sigmoidal smoothing

We use the tangens hyperbolicus as an example for a sigmoidal function

$$d_\alpha(i,j) = \bar{d} + \frac{\tanh(\alpha(d(i,j) - \bar{d}))}{\alpha}$$

with $\alpha \geqslant 0$. For small α the tangens hyperbolicus can be approximated linearly leading to no smoothing, for $\alpha \gg 1$ it is roughly a constant slightly smaller than 1.

4.4.6. Logarithmic smoothing

Looking for a wider range for this transition we suggest a logarithmic type of smoothing

$$d_\alpha(i,j) = \begin{cases} \bar{d} + \dfrac{\log(1 + \alpha(d(i,j) - \bar{d}))}{\alpha} & \text{if } d(i,j) \geqslant \bar{d}, \\ \bar{d} - \dfrac{\log(1 + \alpha(\bar{d} - d(i,j)))}{\alpha} & \text{if } d(i,j) < \bar{d}, \end{cases}$$

with $\alpha \searrow 0$. For small values of α we have $\log(1+x) \approx x$, and therefore we get finally the original distance matrix.

4.5. Computational results

4.5.1. General considerations

The maximum distance between two different cities (i.e. "drilling holes") of the PCB442-problem is 4841.487..., the mean value of the distances is $\bar{d} = 1747.91$. (Computing the mean value we do not take the zeros in the diagonal of the distance matrix into account, because these distances do not contribute to any solution.) If we normalize all distances to the interval [0; 1] we get $\bar{d} = 0.361028$. The maximum distance of the ATT532-problem is 2790, the mean value is 962.6275... and the normalized mean value $\bar{d}$ is 0.345... Note that both normalized mean values are smaller than 0.5. Obviously, our previous simple estimate was completely false: when we smooth the search space we first get a plateau, all distances are equal to $\bar{d}$. Reducing the smoothness control α causes first the rise of hills consisting of states which include the longest distances (they have the largest deviation to $\bar{d}$ because $\bar{d} < 0.5$), and we get finally a landscape similar to the famous US "Monument Valley" national park. The system will remain in the plateau and will not climb up the hills, we only have configurations which do not include the longest edges and the original lengths of the configurations decrease, while the lengths computed with the α-metric stay exactly constant. When finally the α-lengths decrease both metrices converge. They become equal, the system may be already frozen in suboptimal states and only singular improvements can be found when the landscape returns to the original one in the sense that there is again a real valley–hill–landscape.

Absolute results averaged over 50 runs with the unsmoothed and smoothed algorithms can be found in Tables 3–6. The runs took a few hours of CPU time on a

Table 2
Estimation and comparison of the smoothing formula

Formula	α	1	3	10	30	100	300	1000
	$\Delta = 0.1$	0.1	1E-3	1E-10	1E-30	—	—	—
Power law	$\Delta = 0.3$	0.3	0.027	5.9E-6	2.1E-16	—	—	—
	$\Delta = 0.6$	0.6	0.21	6.0E-3	2.2E-7	6.5E-23	—	—
	$\Delta = 0.1$	4.5E-5	2.8E-13	3.7E-43	—	—	—	—
Exponential	$\Delta = 0.3$	0.037	1.4E-4	3.3E-14	1.1E-42	—	—	—
	$\Delta = 0.6$	0.23	0.020	5.8E-7	5.8E-21	—	—	—
	$\Delta = 0.1$	0.1	0.033	0.01	3.3E-3	1E-3	3.3E-4	1E-4
Hyperbolic	$\Delta = 0.3$	0.3	0.1	0.03	0.01	3E-3	1E-3	3E-4
	$\Delta = 0.6$	0.6	0.2	0.06	0.02	6E-3	2E-3	6E-4
	$\Delta = 0.1$	0.10	0.097	0.076	0.033	0.01	3.3E-3	1E-3
Sigmoidal	$\Delta = 0.3$	0.29	0.24	0.10	0.033	0.01	3.3E-3	1E-3
	$\Delta = 0.6$	0.54	0.32	0.10	0.033	0.01	3.3E-3	1E-3
	$\Delta = 0.1$	0.095	0.087	0.069	0.046	0.023	0.011	4.6E-3
Logarithmic	$\Delta = 0.3$	0.26	0.21	0.14	0.077	0.034	0.015	5.7E-3
	$\Delta = 0.6$	0.47	0.34	0.19	0.098	0.041	0.017	6.4E-3

Note: We have put together three characteristic deviations of the original lengths to $\bar{d}$ and show the development of the deviations with increasing α for the different smoothing formula. A "–" sign shall say that the deviation is 0 because of the finite numerical precision of REAL*4-numbers.

medium-sized machine. These tables show especially the minimal reached length of each method and smoothing recipe. We see that each nonlinear smoothing method provides better results than Greedy and GDA only. Hyperbolic and logarithmic smoothing seem to be the best variants for our class of problems. Obviously, smoothing the search space provides better results but one has to choose the proper smoothing formula. Besides, we want to mention that we got solutions for the 442-problem with lengths of about 52 000 working with Simulated Annealing on the original unchanged landscape and investing roughly the same calculation time.

4.5.2. Comparison: Unsmoothed, linear, power law, exponential, hyperbolic and sigmoidal

In order to get a measure to what extent the landscape is smoothed with these formulas we do a small estimation and comparison (Table 2). The deviation of a distance to the mean length vanishes fast for exponential smoothing, followed by power law smoothing. For sigmoidal smoothing all deviations converge to one common value. Linear smoothing is not shown there because it has another range for α.

In Figs. 9 and 10 we show the results for unsmoothed, linearly smoothed and power-law-smoothed (Jun Gu-like) Greedy optimizations. Whereas Fig. 9 provides results typical for a Greedy algorithm optimization (after a short time the system is already frozen in a suboptimal state, sometimes a further improvement can be found, the acceptance rate is slightly above 0 because of the degeneracy of the 442-problem and we come after a short time in a high-lying local minimum. Here we operate on the

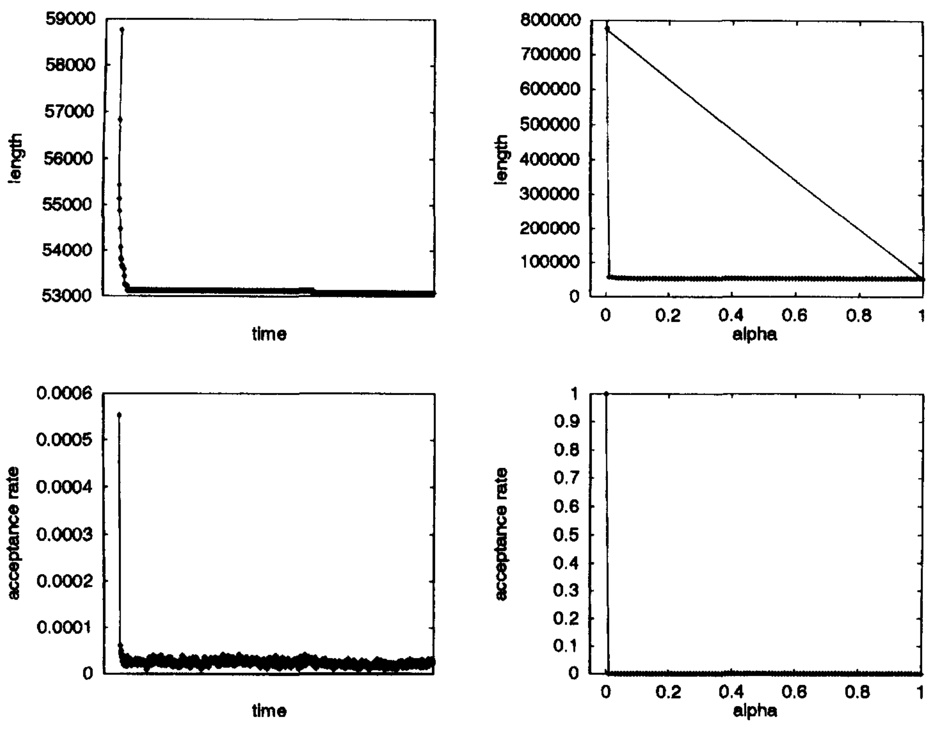


Fig. 9. Here we present data for the PCB442-problem. We performed 320 Greedy-steps for each α. The top row graphs "length versus α" show a different behavior for the different metrics. In the left column we give results for the Greedy without smoothing for comparison; the abscissa is here not α but the elapsed time (The interval between two points is 320 Greedy-steps.). On the right there are data for linear smoothing (α is increased from 0 to 1 in steps by 0.01), where only a linear transformation of the distance matrix is performed. The length development according to the original distance matrix is similar in both cases (top row), and we perform here only a Greedy in both cases. After a short time the system is "frozen" in high-lying suboptimal states, as the acceptance rates shown at the bottom indicate.

original landscape so that we can be quite sure that the system finds a local optimum of the smoothed landscape in one step most of the time), the power law smoothing of the search space opens a new aspect: depending on the value of the control parameter the objective function (length) performs a transition from oscillating around a high-energy plateau into the minimum; this transition is quite smooth in case of the ATT532 instance. Corresponding effects can be found in the acceptance rate. The total behavior remembers to a certain degree on the effects found for Simulated Annealing (Fig. 5). Since the Greedy takes only the next lower state, the transition must necessarily result from the change in the underlying search space, i.e. the α-dependent degree of "smoothness" in the particular search space. As previously mentioned, the analogue in SA is the control parameter dependent ability of the Markovian walker to climb over mountains and ridges. (Note that we start with a much higher α as Jun Gu. He considered in his paper the developments in the range of small α only [7].)

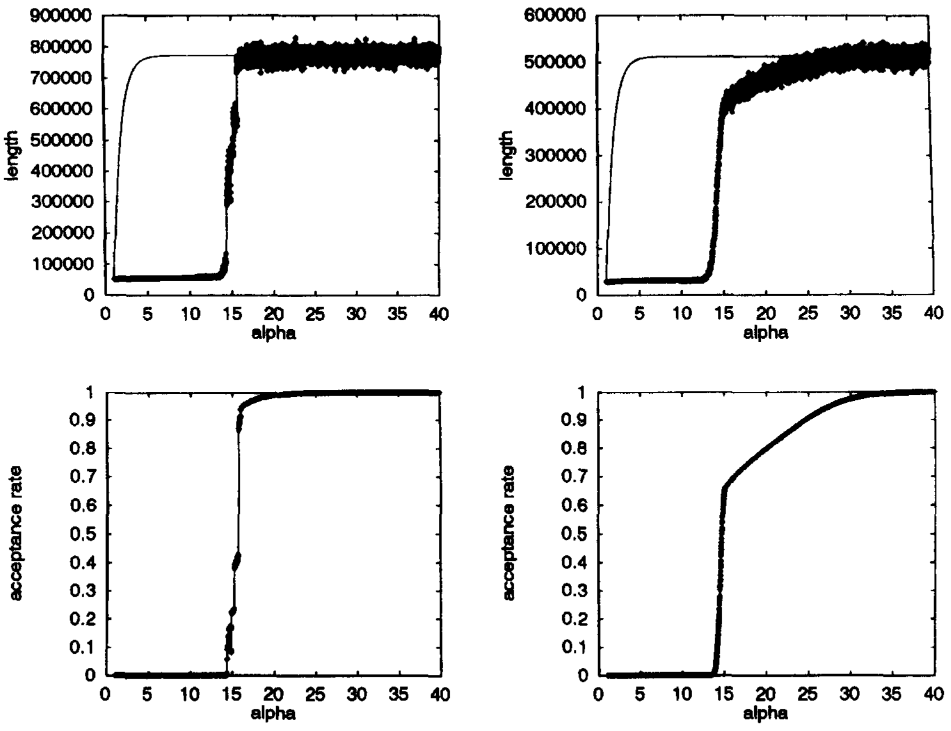


Fig. 10. On the left, results for smoothing the search space of the PCB442 with the original formula by Jun Gu are shown, the right column gives similar results for the 532-problem. α has been lowered from 40 to 1 in 0.01 decrements. Here we have a completely different behavior than in Fig. 9: first we are in the totally noisy range in which the system randomly oscillates around the length = number of points $\times \bar{d}$ at high α. Then there is in a small range a decrease of the length according to the original metric and of the acceptance rate of the moves while the length according to the α-metric remains constant. Finally, the length according to the α-metric decreases still, whereas the system is already frozen in a state, which can be seen looking at the length in the original metric as well as looking at the acceptance rate.

Fig. 11 compares data for exponential, hyperbolic and sigmoidal smoothing: whereas exponential smoothing is quite similar to power-law smoothing (we decreased α again in steps of 0.01.) – only the transition range is smaller in which the length according to the original length decreases – the other two smoothing techniques show a completely different behavior: here the transition range is so wide (more than one order of magnitude) that we felt impelled to change the decrease of α from a linear to an exponential way. We lowered α from 2.0×10^8 to 1 (hyperbolic) or 0 (sigmoidal) by a factor of 0.98. There are also differences between hyperbolic and sigmoidal smoothing: while hyperbolic smoothing has a similar effect as Jun Gu smoothing (except for the wide transition range), the sigmoidal smoothing formula obviously operates in a few distinct steps. First we are in the totally noisy range, the acceptance rate is 1, we perform a random walk on a plateau. Then the length according to the original distance matrix breaks down to a new value. Here again the system does not change much for several α-steps, the acceptance rate is constant, obviously the system

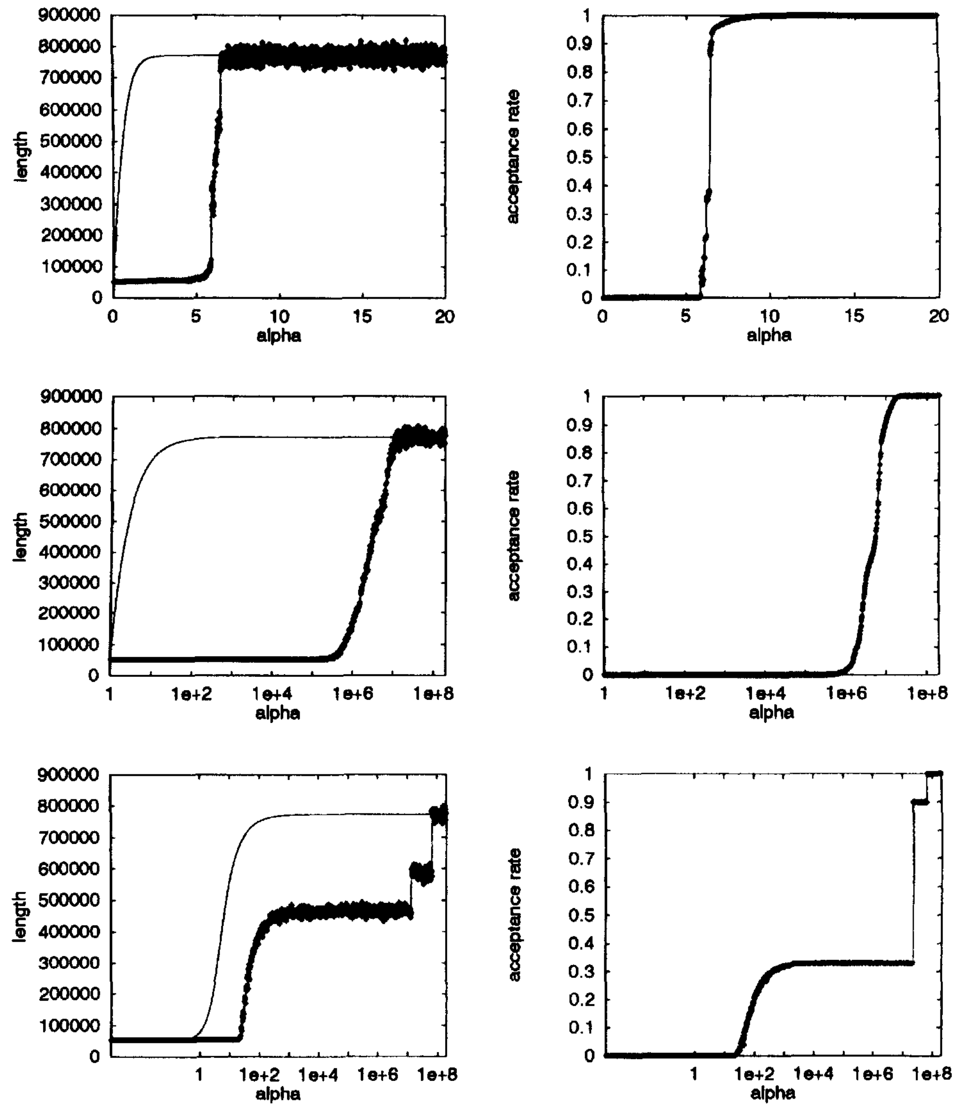


Fig. 11. On the left-half the length developments are shown, on the right-half the acceptance rate for the 442-problem, at the top for exponential smoothing, in the middle for hyperbolic smoothing and at the bottom for sigmoidal smoothing.

performs a random motion in a subset of the configuration space which is a plateau. We may call that a "hill-avoiding" pseudo-random walk. Afterwards, again a breakdown occurs, the system is now caught in a much smaller subset. After that also the length according to the smoothed distance matrix decreases, there arise new structures in the small plateau so that the length according to the original distance matrix and the acceptance rate decrease. Finally, both metrices converge (note that with this kind of

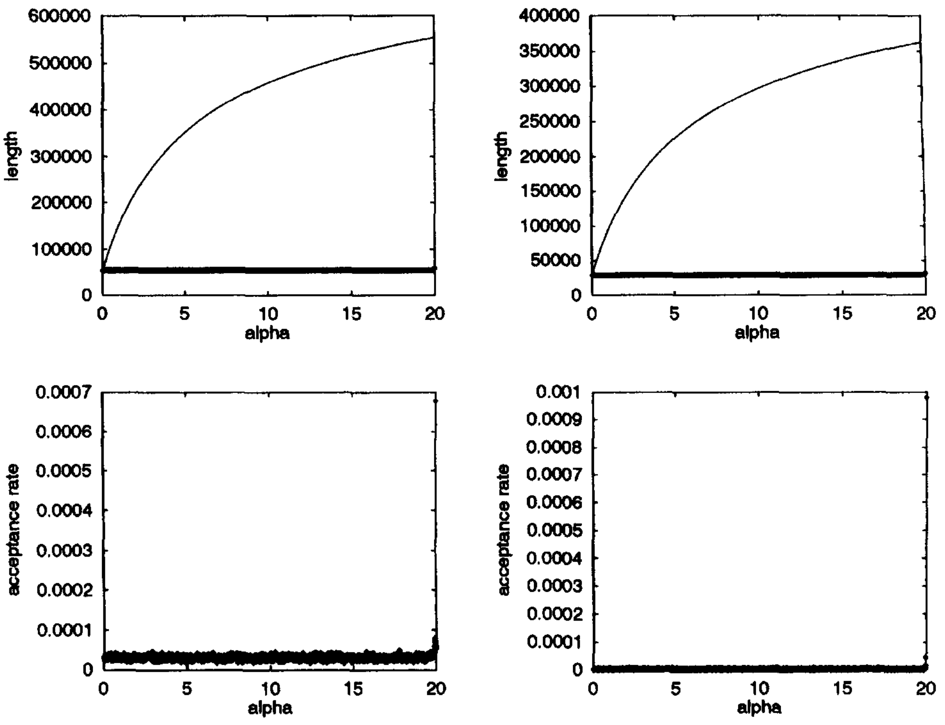


Fig. 12. Logarithmic smoothing, results for both the 442-problem on the left side and the 532-problem on the right. We have scanned the whole range performing a few runs with different starting α. α was decreased linearly in steps of 0.01. Here we see only the system performing a Greedy, the smoothed metric assimilates to the original metric only. The same composition of graphics are in the next figures to logarithmic smoothing.

smoothing the length according to the smoothed distance matrix is at the end going down quadratically).

4.5.3. Logarithmic smoothing

Finally, in this section we want to have an in-depth look on logarithmic smoothing: we show in Figs. 12–16 results for both, the PCB442-problem in the left column, and the ATT532-problem in the right one, respectively. We used an exponential decrease of α – similar to sigmoidal smoothing – for the results for logarithmic smoothing in the Tables 3–6. But it is impossible to show all details of the length development of this kind of smoothing with a single logarithmic abscissa. Therefore, we have scanned the whole range performing a number of runs with different initial α-values from 2×10^1 to 2×10^9. α has been decreased linearly in appropriate decremental steps, the finest one is as small as $\Delta\alpha = 0.01$. Fig. 12 shows the system performing only a Greedy, there is no noticeable effect of the smoothing, i.e. the smoothed metric assimilates totally with the original metric. Fig. 13 uses an initial control parameter value $\alpha = 2 \times 10^6$, and astonishingly some marginal structures in the objective and in the acceptance rate appear.

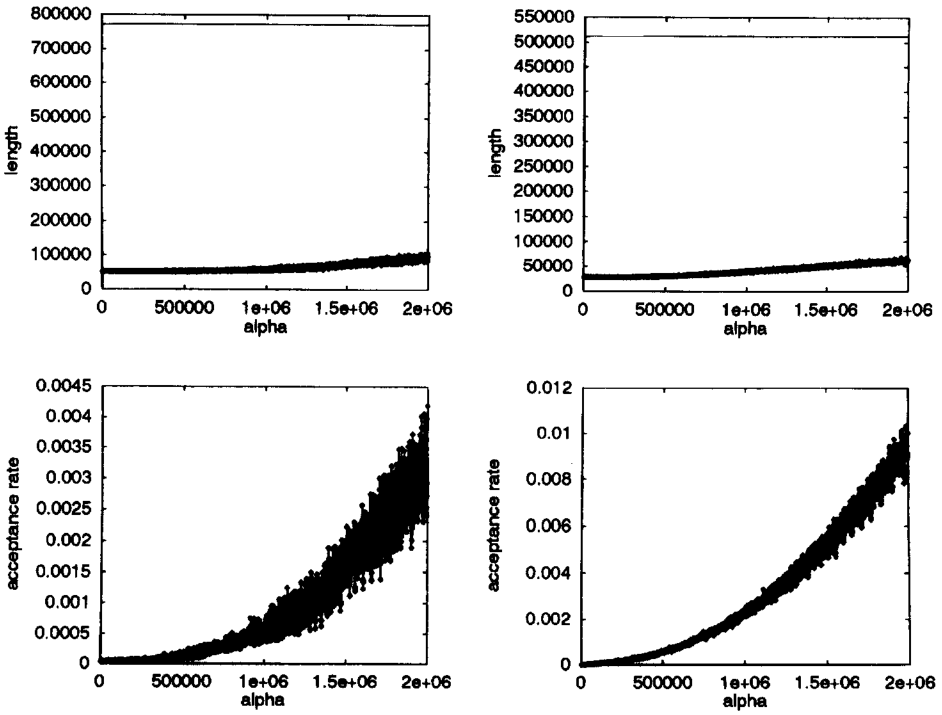


Fig. 13. Data similar to Fig. 12, initial $\alpha = 2 \times 10^6$, decrement $\Delta\alpha = 10^3$.

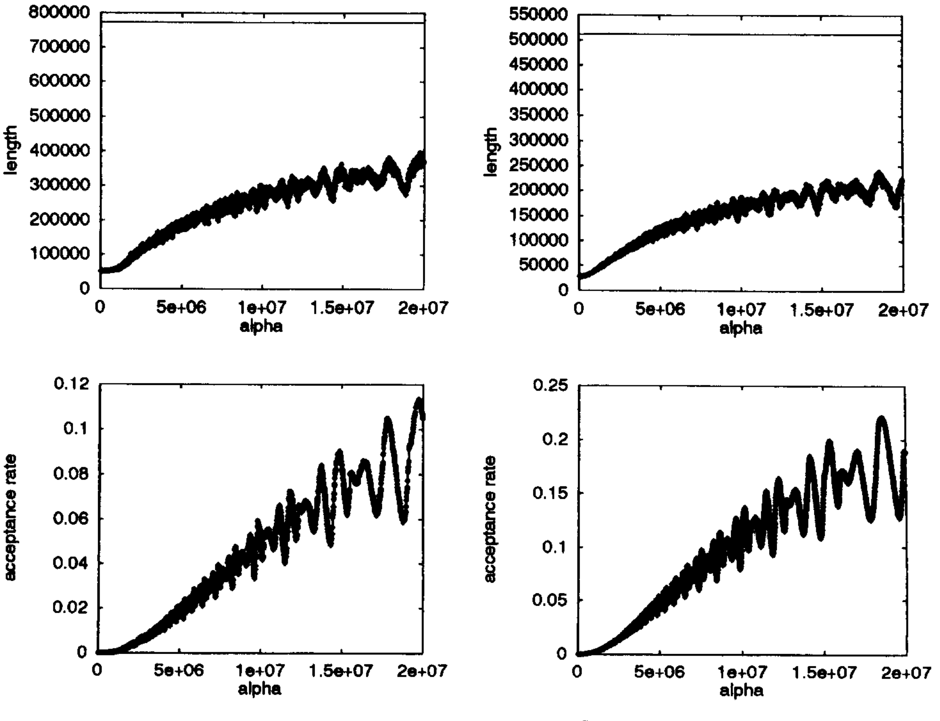


Fig. 14. Data similar to Fig. 12, initial $\alpha = 2 \times 10^7$, decrement $\Delta\alpha = 10^4$.

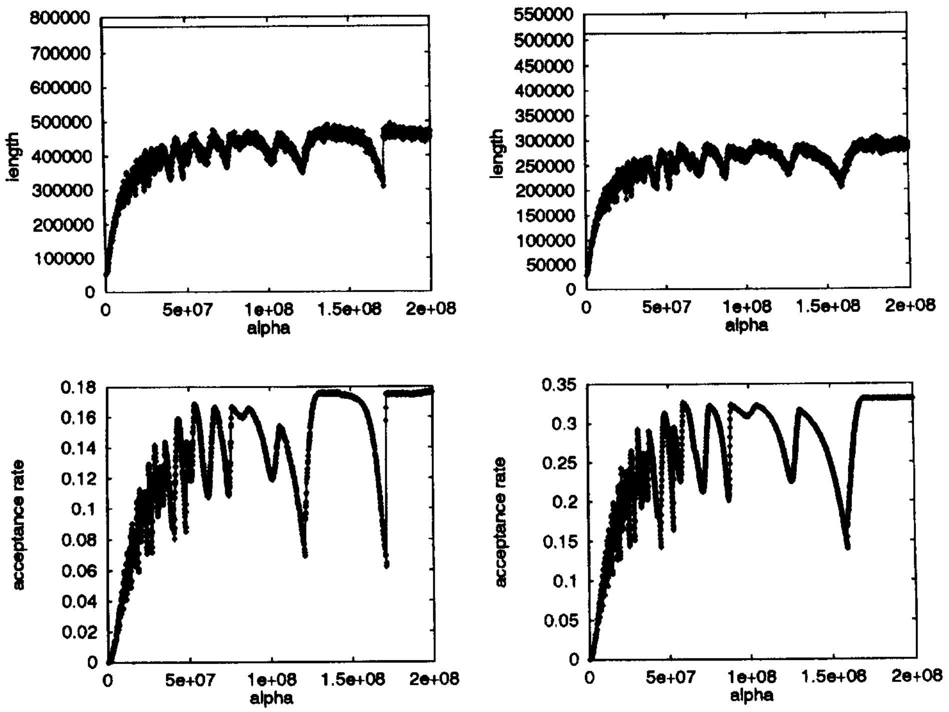


Fig. 15. Data similar to Fig. 12, initial $\alpha = 2 \times 10^8$, decrement $\Delta\alpha = 10^5$.

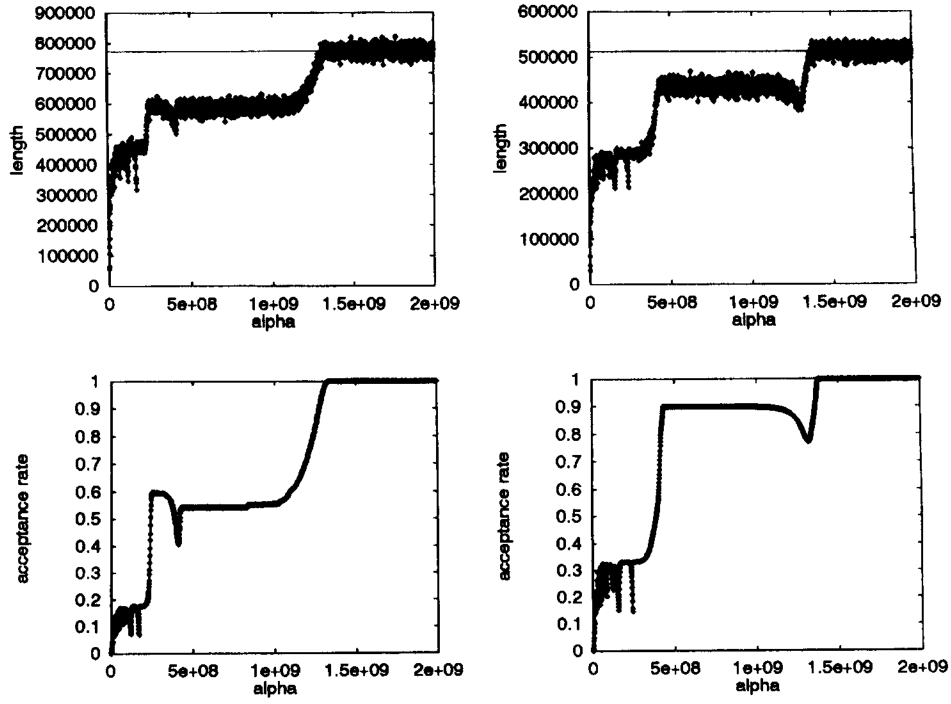


Fig. 16. Last picture for logarithmic smoothing, data similar to Fig. 12, now with a starting α of 2×10^9 and decrements $\Delta\alpha = 10^6$.

Table 3
Statistic of SSS with Greedy for the 442-problem

Formula	Minimum	Maximum	Mean value $\pm$ error	Median
Unsmoothed	52 500	54 312	53 284 $\pm$ 68	53 258
Linear	52 202	54 637	53 579 $\pm$ 84	53 467
Power law	51 221	52 312	51 689 $\pm$ 30	51 706
Exponential	51 172	52 309	51 711 $\pm$ 38	51 741
Hyperbolic	50 996	51 857	51 362 $\pm$ 28	51 363
Sigmoidal	51 222	53 198	52 409 $\pm$ 60	52 471
Logarithmic	50 945	52 118	51 426 $\pm$ 29	51 409

Table 4
Statistic of SSS with Greedy for the 532-problem

Formula	Minimum	Maximum	Mean value $\pm$ error	Median
Unsmoothed	28 537	29 790	29 149 $\pm$ 36	29 132
Linear	28 646	29 677	29 117 $\pm$ 38	29 095
Power law	27 910	29 250	28 562 $\pm$ 40	28 611
Exponential	27 867	29 389	28 606 $\pm$ 42	28 579
Hyperbolic	27 798	28 301	28 024 $\pm$ 16	28 033
Sigmoidal	28 390	29 114	28 727 $\pm$ 26	28 725
Logarithmic	27 835	28 377	28 099 $\pm$ 17	28 087

Table 5
Statistic of SSS with GDA for the 442-problem

Formula	Minimum	Maximum	Mean value $\pm$ error	Median
Unsmoothed	51 834	53 846	52 642 $\pm$ 64	52 633
Linear	51 427	53 577	52 548 $\pm$ 67	52 501
Power law	50 969	52 644	51 691 $\pm$ 51	51 651
Exponential	51 185	52 633	51 765 $\pm$ 44	51 729
Hyperbolic	51 091	52 588	51 694 $\pm$ 44	51 678
Sigmoidal	51 609	53 297	52 548 $\pm$ 63	52 621
Logarithmic	51 059	52 217	51 690 $\pm$ 37	51 637

Increasing the starting α further by a factor of 10 we see the first really surprising diagram (Fig. 14) for logarithmic smoothing: the system obviously performs more or less periodic oscillations here, the length development according to the original distance matrix is in phase with the acceptance rate. When the system finds more paths down into a better solution, it is simultaneously lifted up (in energy) by larger changes of the smoothed valley hill landscape. Moreover, here we recognize also a kind of self-similarity on different α-scales, the structures are repeated again and again when decreasing α, and they become smaller and smaller.

Another increase of α (Fig. 15) by a factor of 10, leads again to oscillations, but they appear to be not as regular as in Fig. 14. The length development according to the original metric is still in phase with the acceptance rate, whereas the length according to the smoothed metric stays constant.

Table 6
Statistic of SSS with GDA for the 532-problem

Formula	Minimum	Maximum	Mean value ± error	Median
Unsmoothed	28 313	29 311	28 758 ± 25	28 739
Linear	28 297	29 327	28 748 ± 31	28 681
Power law	27 978	29 289	28 580 ± 45	28 598
Exponential	28 065	29 446	28 638 ± 51	28 662
Hyperbolic	27 807	28 515	28 225 ± 23	28 237
Sigmoidal	28 221	29 135	28 624 ± 28	28 602
Logarithmic	27 944	28 590	28 254 ± 19	28 244

The last picture for logarithmic smoothing in combination with the greedy method with a starting $\alpha = 2 \times 10^9$ and decrement size $\Delta\alpha = 10^6$ shows for huge α values that the system performs a kind of random walk around 772577... for PCB442 and 512177.78... for the ATT532-problem. Then the system is closed in a subset of the search space, the acceptance rate stays constant for a wide range. Afterwards, again many rearrangements happen in a relatively small α-window. This, finally, gives the proper explanation of the oscillating behavior described earlier in this paragraph.

Summarizing, for logarithmic and sigmoidal smoothing we have different results than for power law, exponential and hyperbolic smoothing. Therefore, we cannot speak of a general behavior of the SSS method.

5. Search-space smoothing: Combination with various local search algorithms

Till now we only used the Greedy as a simple local search heuristic; to improve this method one may combine the idea of search-space smoothing (SSS) with all local search algorithms as Jun Gu suggested [7].

5.1. Combination of SSS with SA

The combination of search-space smoothing with Simulated Annealing, however, would counteract the spirit of this method, since SSS is based on the idea that a good (i.e. low-lying) solution of the simplified smoothed problem is guided through a series of search-space UN-smoothing operations to a good solution of the original problem. Simulated Annealing allows to leave the suboptimal "trap" where the good solution of the simplified problem lies and therefore destroys the guidance effect of SSS. Again we want to point to the pronounced difference which distinguishes the search space smoothing concept from a method like SA: In SA the Markovian walker has due to the control parameter "temperature" the ability to "climb" over energetic barriers. In SSS approaches the walker is not necessarily able to climb over the barriers; instead, the barriers can be "broken down" via the "smoothness control" handle α.

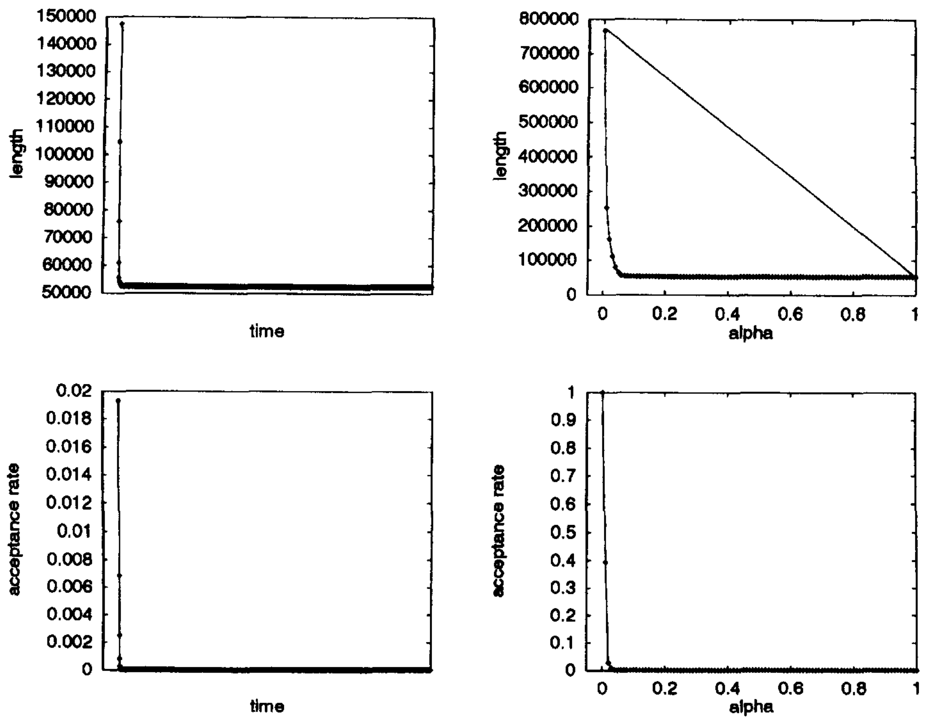


Fig. 17. Unsmoothed and linearly smoothed, combined with GDA: results for the PCB442 instance on the left side and for the ATT532-problem on the right-half, length and acceptance rate, similar to Fig. 9, Greedy Algorithm.

5.2. Combination of SSS with GDA

On the other hand, especially the combination of SSS and the Great Deluge Algorithm ("GDA-SSS") makes sense: if we reduce the smoothness control parameter the state in which the system lies will be in most cases no longer a local optimum, frequently it is now at a crooked wall. If we use the greedy procedure we would fastly run again into the next local optimum, as shown in Fig. 7. But reducing the α handle will mostly lead to a locally more complicated structure, and the local optima will be not only displaced; moreover, new hills and valleys develop and the single local optimum may split up in a few new optima in this way. The Great Deluge Algorithm seems to be a clever choice here: the starting level shall be the length (in the new α-metric) of the actual solution which was the former local optimum in the old α-metric. In the valley where the initial configuration lies the GDA searches for the best optimum below the starting level.

Therefore, we combine the advantages of search-space smoothing and the Great Deluge Algorithm: we use the guidance effect of SSS which leads a good solution of the previous α-metric to a good solution of the new metric, which again in a countermove provides us with a default valley to search for a good solution. In this

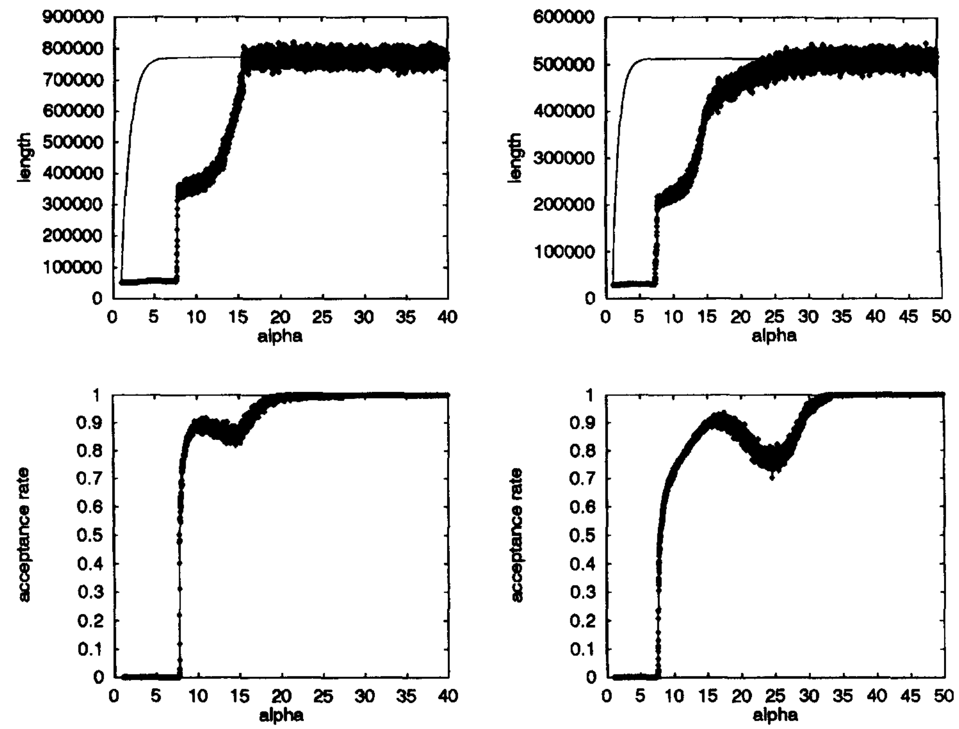


Fig. 18. Power-law smoothing (Jun-Gu-like) combined with Great Deluge Algorithm, length (top) and acceptance rate (bottom), PCB442 (left column) and ATT532 (right column).

valley we search with the Great Deluge Algorithm for the best optimum, whereas we would get stuck in the next local optimum if we used the Greedy only.

The following results (Fig. 17) for the combination of search-space smoothing with the Great Deluge Algorithm (GDA) are shown in a similar sequence as in the case of the Greedy-smoothing combination.

5.2.1. *Computational results: Linear, power law, exponential, hyperbolic, sigmoidal*

Fig. 17 provides results for GDA on the original unsmoothed landscape and in combination with a linear smoothing. Since the used GDA (the run for Fig. 6 needs about a week of CPU time on a medium-sized machine) takes too much computing power we decided to perform only 320 GDA-steps with the described lowering condition of the level $\mathcal{T}$. Furthermore, we changed the factor from 0.99 to 0.9 and started with a $\mathcal{T}$ value which was the length of an initial random configuration. Additionally, we perform one Greedy step at the end of each α-step in order to surely reach at least a local optimum most of the time, so that we can really use the guidance effect of search-space smoothing. With these conditions the system again shows a Greedy-like behavior in both cases, for an unsmoothed landscape and for linear search-space smoothing. The graphs are similar to Fig. 9 and can be described in the same way. Note that the

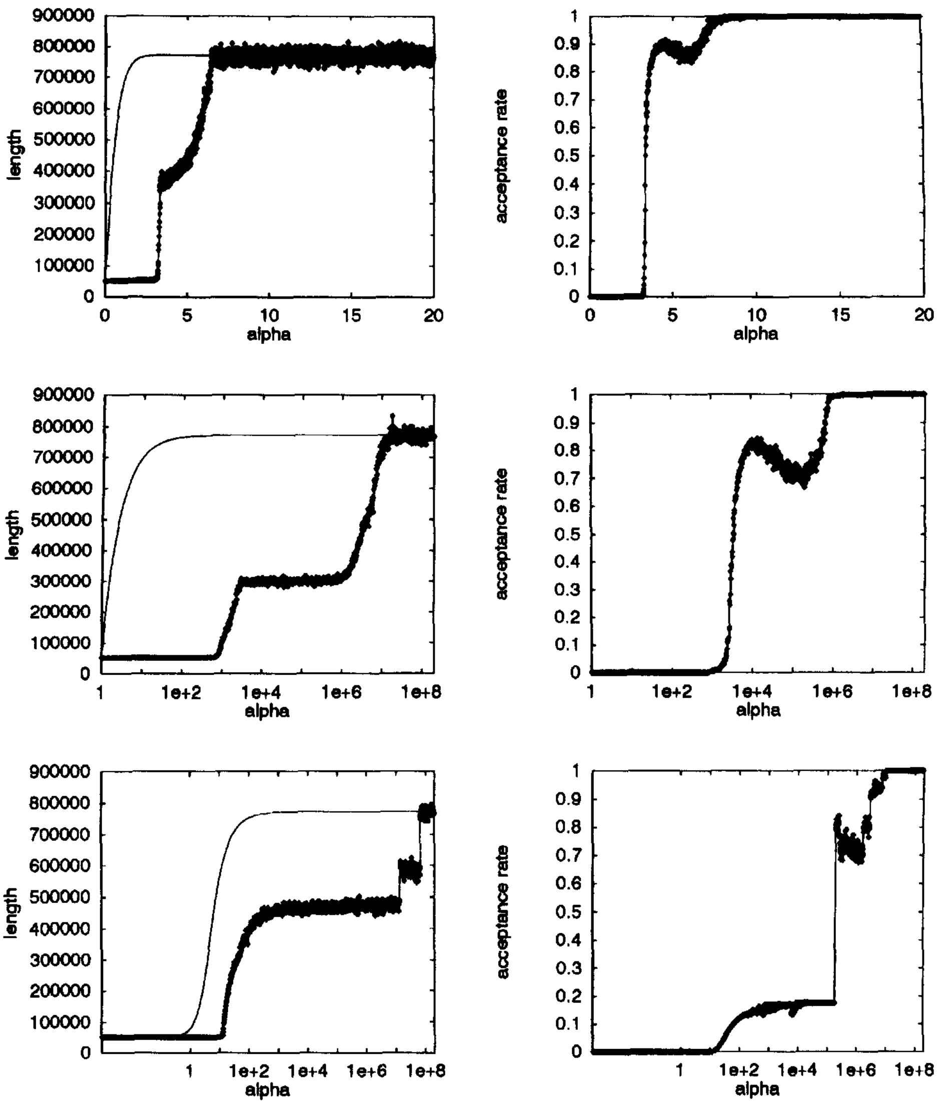


Fig. 19. Exponential (top), hyperbolic (middle row) and sigmoidal (bottom) smoothing combined with Great Deluge Algorithm, length (left column) and acceptance rate (right column), respectively, for the PCB442 instance.

freezing into a suboptimal state does only take little additional time in comparison to the combination of SSS with Greedy. Again we are quite sure to reach a local optimum on a smoothed landscape within one step.

If we compare the results for GDA combined with power-law – i.e. original Jun Gu-like – smoothing (Fig. 18) with the results shown in Fig. 10 we see that the graphs are quite similar except for a third range of α: the length according to the original metric and the acceptance rate do not break down at once. Till $\alpha = 16$ one has a plain

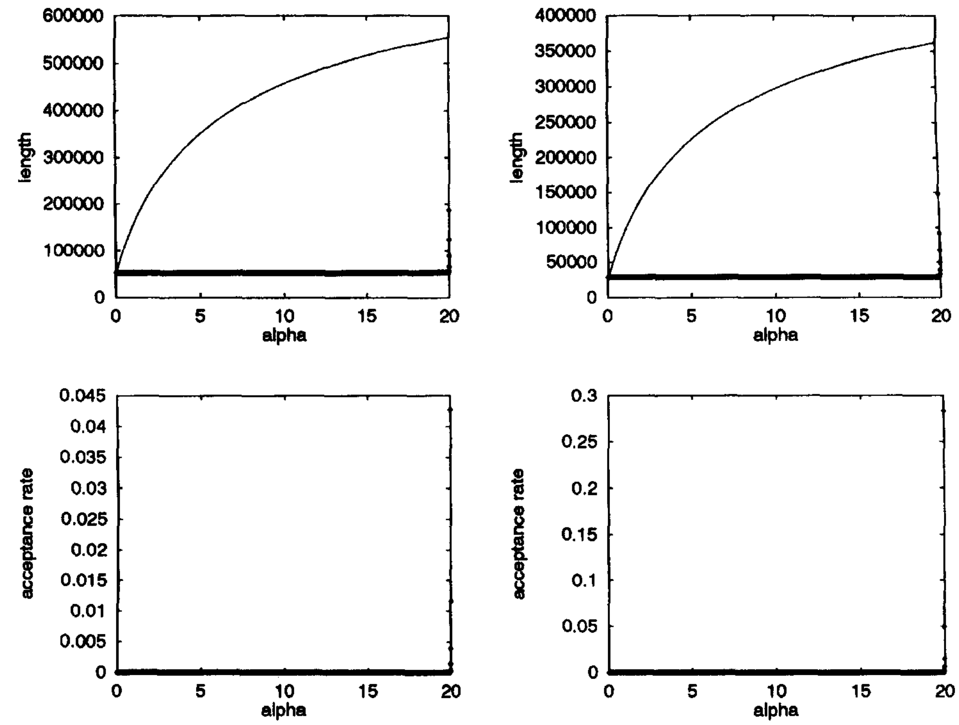


Fig. 20. Logarithmic smoothing combined with Great Deluge Algorithm, length (top) and acceptance rate (bottom), PCB442 (left column) and ATT532 (right column); the initial $\alpha = 20$, the decrement $\Delta\alpha = 0.01$. In this figure we see that the behavior is similar to using the Greedy only, the smoothed metric just assimilates the original metric. However, there are many rearrangements so that we get better results than with a simple Greedy.

plateau in the PCB442-system. Then the length decreases with some fluctuations more and more slowly till $\alpha = 7.8$, the acceptance rate even increases for a certain range. Afterwards, there is a steep decrease of the length according to the original metric, similar to Fig. 10. For the ATT532-problem it is comparable, here we also have a third new middle range. The length development according to the smoothed metric, however, stays nearly exactly the same as using Greedy.

Also for the combined search-space smoothing–GDA concept the exponential smoothing has a behavior similar to the power law smoothing (compare Figs. 18 and 19, we find again a new third range, smaller than the one at power law smoothing). The same observation is valid for hyperbolic smoothing, too, the behavior is similar to power-law smoothing, but with the new middle range being much more extended, the length according to the original metric stays roughly constant for a wide range. In all three cases the acceptance rate shows this more or less extended new third range, where it decreases first and then increases again. Obviously, in all three cases the system stays on the plateau avoiding new rising “energy mountains”. First their number grows, but then it seems that the subset in which the system stays, becomes larger and larger, in

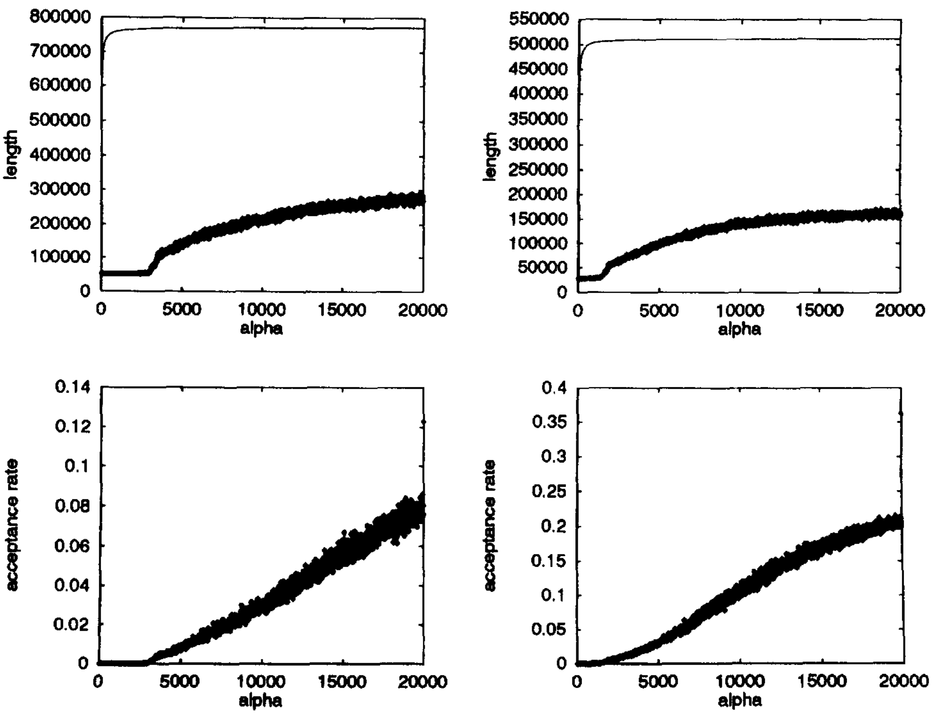


Fig. 21. Logarithmic smoothing combined with Great Deluge Algorithm, length (top) and acceptance rate (bottom), PCB442 (left column) and ATT532 (right column); the initial $\alpha = 2 \times 10^4$, the decrement $\Delta\alpha = 10.0$.

a way that the length according to the original metric stays constant. Finally, there is a sudden breakthrough in the system, and it proceeds down from the plateau along new search space paths.

The graphs for sigmoidal smoothing show a similar behavior as in case of the combined Greedy-smoothing procedure, shown in Fig. 11. The acceptance rate decreases later than in the previous case and it increases in a small range similar to the other graphics of GDA.

Comparing the results of the Tables 3–6 we see that GDA can be combined with search-space Smoothing, they do not contradict each other. With the calculation time we had, however, we cannot show that the combination of SSS with Greedy might be better or worse than the combination of SSS with GDA.

5.2.2. Computational results: Logarithmic smoothing

Again, like in the case of a combined Greedy-logarithmic smoothing we want to show a series of results for the combination of GDA with a logarithmic smoothing ("GDA-log$S$") formula, starting with Fig. 20. As we can see, the "non-Greedy-behavior", i.e. the decrease of the length according to the original distance matrix and of the acceptance rate, begins much earlier (earlier in terms of lower values of α)

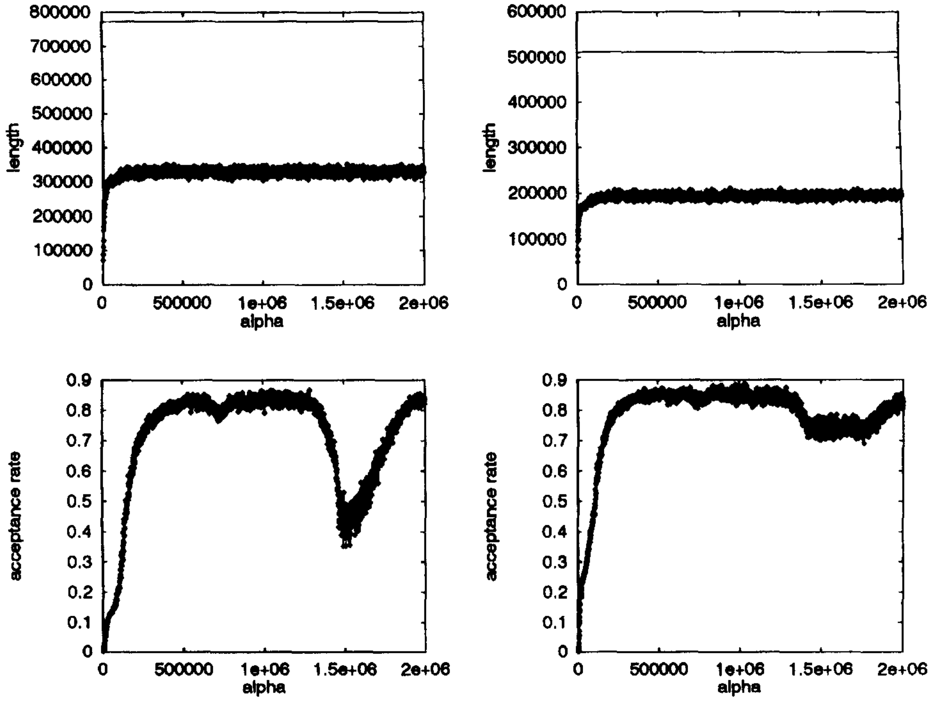


Fig. 22. Logarithmic smoothing combined with Great Deluge Algorithm, length (top) and acceptance rate (bottom), PCB442 (left column) and ATT532 (right column); the initial $\alpha = 2 \times 10^6$, the decrement $\Delta\alpha = 10^3$.

than if we use the Greedy only. (See Fig. 21: there it happens two orders of magnitude of α later, as can be seen in Fig. 13.) A further difference is that the decrease of both parameters is non-quadratic here. A new behavior of GDA-log$S$ can be seen for α-values a two additional orders of magnitude larger (Figs. 22 and 23). Here we have a behavior which appears to be completely new: while the lengths according to the smoothed and according to the original landscape roughly stay constant there are many changes in the acceptance rate. We explain this with the hypothesis that the system is on constant height plateaux, respectively; the plateaux only differ in their absolute height. A further increase of α by a factor 100 reveals the development of the lengths, the same behavior as in Fig. 15. The acceptance rate, however, shows a different behavior: there are no oscillations anymore, every move is roughly equally accepted. Note that in the right part of the acceptance rate graphs the acceptance rate is not exactly 1 but slightly smaller than 1. In addition, the acceptance rate behavior is also non-monotonous. Finally, we have a look at GDA-log$S$ at very large α (Fig. 24). After leaving the range of random walk in which all distances are equal to $\bar{d}$ the system is on a new, for a wide range fairly constant level, whereas the acceptance stays slightly below 1. Obviously, here we return again to our picture of a Monument Valley.

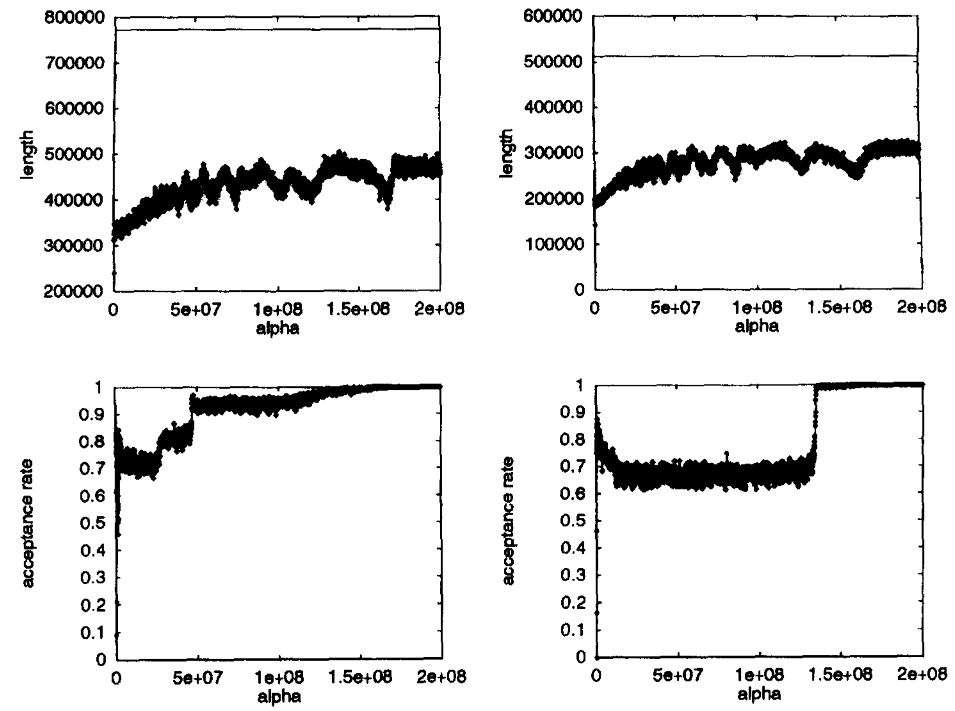


Fig. 23. Logarithmic smoothing combined with Great Deluge Algorithm, length (top) and acceptance rate (bottom), PCB442 (left column) and ATT532 (right column); the initial $\alpha = 2 \times 10^8$, the decrement $\Delta\alpha = 10^5$.

6. Conclusion

To conclude the presented results we want to state that search-space smoothing – based on an earlier idea in [7] and related work – in combination with local search algorithms provides a new and successful approach. It avoids the problem of getting trapped in local suboptimal minima which is common to nearly all local search concepts besides Simulated Annealing, which provides – via the temperature concept – a similar control handle to escape these traps; the advantage over the latter one is that a certain amount of computational effort usually provides a better result than in the case of SA. This statement is nevertheless only based on empirical data up to now and not on an exact theory as it exists – at least partially – for SA.

The most promising results – promising in terms of coming close to the global optimum – are found for smoothing recipes which allow a wide transition region between the ordered and the disordered phase: there one should especially mention the hyperbolic and logarithmic approach. In addition, we want to formulate a provocative statement: we believe that this concept is in a certain sense even superior to Simulated Annealing since it allows a much more deterministic (and in connection with that a simpler) prescription for the manipulation of the control parameter. Nevertheless, we provided only a very rudimentary conceptual sketch and made only limited simulations

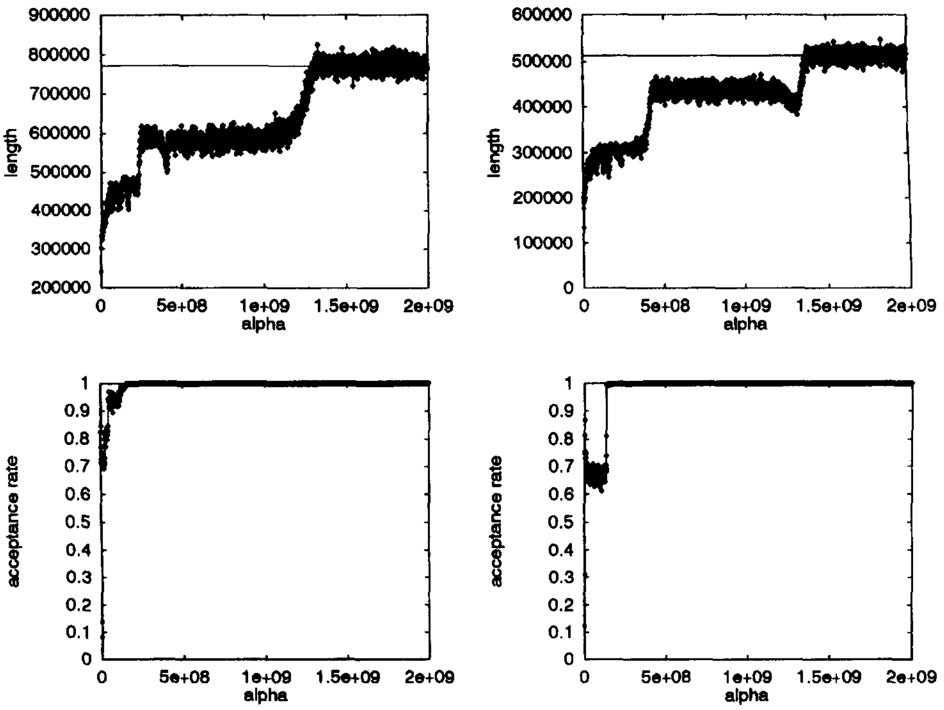


Fig. 24. Logarithmic smoothing combined with Great Deluge Algorithm, length (top) and acceptance rate (bottom), PCB442 (left column) and ATT532 (right column); the initial $\alpha = 2\times10^9$, the decrement $\Delta\alpha = 10^6$.

on a small number of TSP instances; therefore, it is still necessary to extend the concept as well as the experiments to get a more profound knowledge on this promising approach.

There are also a few interesting possibilities to parallelize this optimization algorithm, e.g. we have begun transferring the rule from Ensemble Based Simulated Annealing [16] in order to decrease the smoothing parameter α instead of the temperature T. Further work will be done in this field.

Acknowledgements

We acknowledge important hints and discussions with Gunter Dueck, Tobias Scheuer, Gerhard Schrimpf, Hermann Stamm-Wilbrandt and Klaus Volk from the IBM Scientific Center Heidelberg, and we are very grateful to D. Stauffer for an enlightning discussion on several aspects of our work. Furthermore, we wish to thank the KFA Jülich for a generous grant of computer time and kindly acknowledge the help and assistance of Jutta Docter, Katja Fälski, Renate Knecht and Ulrich Detert.

We also want to thank Uwe Krey, Silvia Hauptmann, Markus Böhmer and Matthias Weiser for useful discussions.

Part of the work (Uni Zürich) has been financed by the Swiss National Fund. Werner Fettes wishes to thank the DFG for financial support.

References

[1] J. Schneider, Parallelisierung physikalischer Optimierungsverfahren, Diploma Thesis, 1995.
[2] M. Dankesreiter, Anwendung physikalischer Methoden auf Probleme der Logistik, Diploma Thesis, 1996.
[3] J. Schneider, Ch. Froschhammer, I. Morgenstern, Th. Husslein, J.M. Singer, Searching for backbones – an efficient parallel algorithm for the traveling salesman problem, Comput. Phys. Commun. 96(2/3) (1996) 173–188.
[4] S. Kirkpatrick, C.D. Gelatt Jr., M.P. Vecchi, Optimization by simulated annealing, Science 220 (1983) 671–680.
[5] G. Dueck, T. Scheuer, Threshold accepting: a general purpose optimization algorithm appearing superior to simulated annealing, J. Comput. Phys. 90 (1990) 161–175.
[6] J.C. Schön, M. Jansen, Angew. Chemie 108 (1996) 1358–1377.
[7] Jun Gu, Efficient local search space smoothing: a case study of the traveling salesman problem (TSP), IEEE Trans. Systems Man Cybernet. 24 (5) (1994).
[8] M. Grötschel, O. Holland, Solution of large-scale symmetric travelling salesman problems, Math. Programming 51 (1991) 141–202.
[9] M.W. Padberg, G. Rinaldi, Optimization of a 532-city symmetric travelling salesman problem by branch and cut, Oper. Res. Lett. 6 (1987) 1–7.
[10] G. Reinelt, TSPLIB - Version 1.2, Report No. 330, University of Augsburg, Germany, 1991.
[11] P.F. Stadler, W. Schnabl, The landscape of the traveling salesman problem, preprint 91-047, Preprintreihe Sonderforschungsbereich 343 Diskrete Strukturen in der Mathematik at the University of Bielefeld, Germany, 1991.
[12] S. Kirkpatrick, G. Toulouse, Configuration space analysis of the travelling salesman problem, J. Physique 46 (1985) 1277–1292.
[13] K. Binder, D.W. Heermann, Monte Carlo Simulation in Statistical Physics, Springer, Berlin, 1992.
[14] G. Dueck, New Optimization Heuristics: The Great Deluge Algorithm and the Record-to-Record Travel, J. Comput. Phys. 104 (1993) 86–92.
[15] I. Morgenstern, D. Würtz, Z. Phys. B: Condens Matter 67 (1987) 397.
[16] R. Tafelmayer, K.H. Hoffmann, Adaptive Schedules for Ensemble Based Threshold Accepting, TU Chemnitz–Zwickau, 1994.