11 Further Techniques Changing the Energy Landscape of a TSP

11.1 The Convex–Concave Approach to Search Space Smoothing

Based on the search space smoothing (SSS) approach by Gu and Huang [76], discussed in the previous chapter 10, Coy et al. introduced a related approach [42] based on considerations of the shape of the smoothing functions introduced in [76] and [186] and discussed in the last chapter 10. All these smoothing functions $f(d(i,j))$ except the hyperbolic one can be split at the mean normalized distance $\bar{d}$ into a convex and a concave part: a real function $f(x)$ is said to be convex in some interval $[a;b]$ if

$$f(t \times x_1 + (1-t) \times x_2) \leq t \times f(x_1) + (1-t) \times f(x_2)\,, \tag{11.1}$$

with $0 \leq t \leq 1$ and $x_1, x_2 \in [a;b]$. $f(x)$ is said to be concave in some interval if $-f(x)$ is convex there.

Figure 10.3 shows that the power law smoothing formula (10.6) is convex for $d(i,j) > \bar{d}$ and concave for $d(i,j) < \bar{d}$. On the other hand the sigmoidal smoothing formula, e. g., is concave for $d(i,j) > \bar{d}$ and convex for $d(i,j) < \bar{d}$. Thus, the question can be asked whether the convex or the concave part of the smoothing function is the important one for the smoothing process and, based on that, whether a completely convex or a completely concave smoothing function would lead to even better results [42].

A straightforward approach to investigating this idea is to use the approach of Gu and Huang: introduce a smoothness control parameter α, work with normalized distances $d(i,j) = D(i,j)/D_{\max}$, i. e., $0 \leq d(i,j) \leq 1$, and define the convex smoothing formula

$$d_\alpha(i,j) = d(i,j)^\alpha \tag{11.2}$$

and the concave smoothing formula

$$d_\alpha(i,j) = d(i,j)^{1/\alpha} \tag{11.3}$$

with $\alpha \geq 1$ [42]. Note that these formulas are nearly identical with the power law smoothing formula (10.6) by Gu and Huang, but the concept of using the mean distance $\bar{d}$ is omitted here in order to get an overall convex or concave

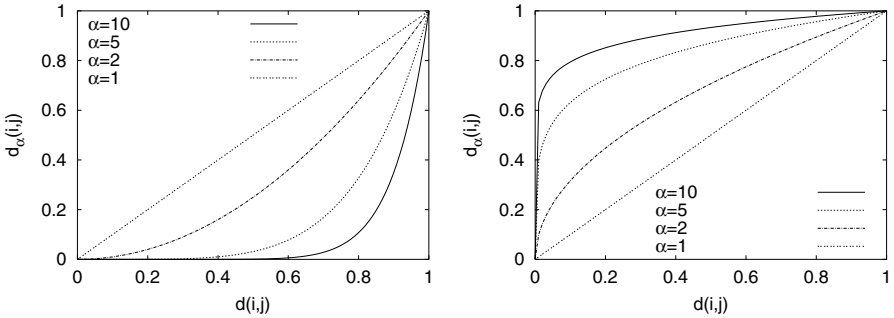

Fig. 11.1. Distances smoothed with the convex smoothing formula (11.2) (*left*) and with the concave smoothing formula (11.3) (*right*): the *curves* show the values for the smoothed distances $d_\alpha(i,j)$ as functions of the original normalized distances $d(i,j)$ for various values of the smoothness-control parameter α. One gets $d_\alpha(i,j) = d(i,j)$ for $\alpha = 1$

function (Fig. 11.1). Thus, all nonzero distances converge to 1 for $\alpha \to \infty$ when using the concave formula. Analogously, all smoothed distances except those for $d(i,j) = 1$ converge against 0 for $\alpha \to \infty$ when using the convex formula. Thus, we get flat landscapes at large α and the original landscape at $\alpha = 1$ for both formulas.

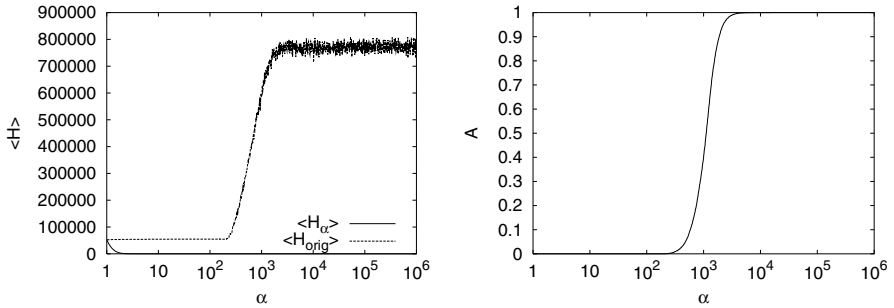

Fig. 11.2. Computational results for the mean energy $\langle \mathcal{H} \rangle$ (*left*) and the total acceptance rate A (*right*) of the PCB442 instance using the convex formula (11.2) for SSS. *Left*: both the mean energy $\langle \mathcal{H}_\alpha \rangle$ according to the smoothed distance values $D_{\max} \times d_\alpha(i,j)$ and the mean energy $\langle \mathcal{H}_{\text{orig}} \rangle$ calculated with the original distances $D(i,j)$ are shown

Again we first study the developments of the mean energy and of the acceptance rate with decreasing smoothness control parameter α (Fig. 11.2) for the convex smoothing formula (11.2). We find that $\langle \mathcal{H}_\alpha \rangle$ is hardly visible in the left graphic as it remains zero until small values of α are reached. On the other hand, $\langle \mathcal{H}_{\text{orig}} \rangle$ shows the behavior we are used to. It decreases from the energy range of the random configurations in some range of the control parameter α to the regime of the ordered configurations. In the same

range, the total acceptance rate A drops from slightly below 1 to values slightly larger than 0. Note that when using this convex smoothing formula, the greedy approach can never completely coincide with the random walk (RW), as the longest edge is always of length 1, whereas the other lengths are decreased to 0. Thus, the longest edge will never be accepted by the greedy algorithm. We find that at α-values smaller than 5×10^4, the system leaves this quasi-RW mode. In the production runs, we thus decrease α exponentially from 5×10^4 to 1 with a factor of 0.99. Furthermore, we add one greedy step at $\alpha = 1$ in the original landscape. At small α, the curve for $\langle \mathcal{H}_\alpha \rangle$ increases toward the curve for $\langle \mathcal{H}_{\text{orig}} \rangle$, while the curve for $\langle \mathcal{H}_{\text{orig}} \rangle$ shows that sometimes improvements can be found even at small α.

Although the convex smoothing formula (11.2) is related to the power law smoothing formula (10.6), we find that here the system orders itself on a logarithmic α-scale (α must be decreased over several orders of magnitude), whereas it ordered itself on a linear scale for the power law smoothing formula. Thus, we want to compare the results achieved with the convex smoothing formula with those for another smoothing formula that also orders the system over several orders of magnitude of the control parameter. As the hyperbolic smoothing formula (10.10) also orders the system in a logarithmic way and as it is analytically a special case of a convex function, we will compare the results for the convex smoothing formula with those for the hyperbolic formula here. Figure 11.3 shows the results for both smoothing formulas.

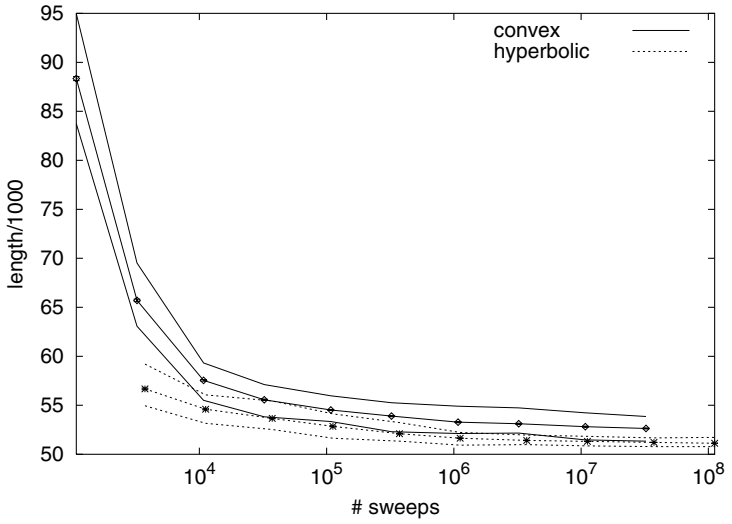

Fig. 11.3. Quality of the results achieved with SSS using the convex formula (11.2) (*straight lines*) and the hyperbolic formula (10.10) (*dotted lines*) for the PCB442 instance vs. overall number of sweeps: the three *lines* show minimum lengths, mean lengths (*error bars* are mostly the size of the *symbols*), and maximum lengths

Note that $1, 3, 10, 30, \ldots$, or 30,000 sweeps were performed per α-step in both cases, but that 1078 α-steps were performed using the convex function and 3736 α-steps when working with the hyperbolic function. We find that the hyperbolic smoothing function leads to significantly better results than the convex smoothing function.

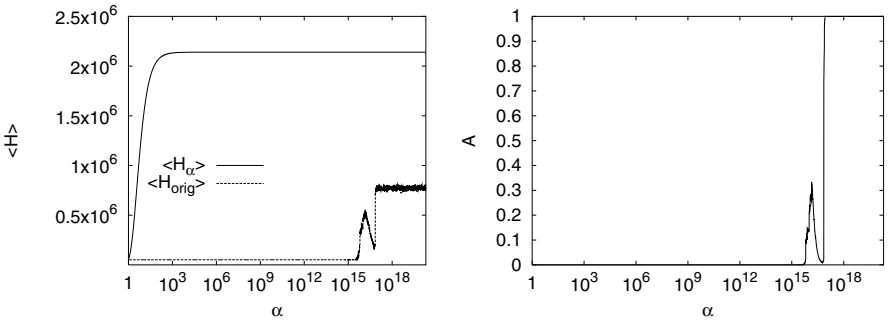

Fig. 11.4. Computational results for the mean energy $\langle \mathcal{H} \rangle$ (*left*) and the total acceptance rate A (*right*) of the PCB442 instance using the concave formula (11.3) for SSS. *Left*: both the mean energy $\langle \mathcal{H}_\alpha \rangle$ according to the smoothed distance values $D_{\max} \times d_\alpha(i, j)$ and the mean energy $\langle \mathcal{H}_{\text{orig}} \rangle$ calculated with the original distances $D(i, j)$ are shown

Next we consider the results for the concave smoothing function (11.3). Figure 11.4 shows the development of the mean energy and of the total acceptance rate with decreasing α. Again $\langle \mathcal{H}_\alpha \rangle$ is rather constant over a wide α-range at a value of $D_{\max} \times N \approx 4841.487 \times 442 \approx 2.14 \times 10^6$, as all smoothed normalized distances are equal to 1 for very large α and deviate from 1 only slightly with decreasing α. These starting deviations lead to the breakdown of both $\langle \mathcal{H}_{\text{orig}} \rangle$ and A at $\alpha \approx 7 \times 10^{16}$. Then the curves for these observables gradually increase again before a second decrease occurs. At small α, the curve for $\langle \mathcal{H}_\alpha \rangle$ decreases strongly and tends toward the curve for $\langle \mathcal{H}_{\text{orig}} \rangle$. The last improvements for the system can be found at $\alpha \approx 2$. Thus, this smoothing formula also orders the system over several orders of magnitude of the smoothness control parameter, such that we use again an exponential cooling schedule.

In the production runs, we decreased α from 9×10^{16} exponentially to 1 with a factor of 0.99 and added a greedy step in the original landscape at the end, so that we had overall 3886 α-steps here. Figure 11.5 shows the results for the concave smoothing formula (11.3) compared to those for hyperbolic smoothing. We find that the results for concave smoothing are on average significantly better than those for hyperbolic smoothing, so that the concave smoothing formula may be considered to be the best smoothing formula found so far. As the results do not spread as widely as for hyperbolic smoothing,

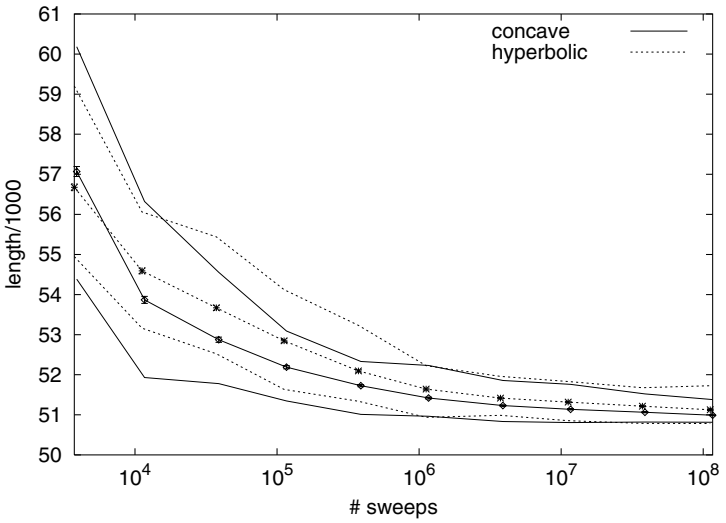

Fig. 11.5. Quality of the results achieved with SSS using the concave formula (11.3) (*straight lines*) and the hyperbolic formula (10.10) (*dotted lines*) for the PCB442 instance vs. overall number of sweeps: the three *lines* show minimum lengths, mean lengths (*error bars* are the size of the *symbols*), and maximum lengths

we did not obtain a result here with optimum length using it in contrast to the hyperbolic and the logarithmic smoothing formulas.

Finally, one might ask whether one could combine the convex smoothing formula (11.2) with the concave smoothing formula (11.3). This can be be done, e. g., by first performing some sweeps using the concave smoothing formula and then performing the same number of sweeps using the convex smoothing formula. Figure 11.6 shows the results for this approach. For each α-step, we perform the measurements separately for the concave and for the convex part. Thus, the data points jump between the curves for convex and concave smoothing shown in Figs. 11.2 and 11.4. For large α, the system is in the RW mode for the convex smoothing formula but already in the greedy mode for the concave smoothing formula. Thus, the system performs a RW if the convex formula is applied; then it is quenched down in some local valley after switching to the concave formula. With the next switch to the convex formula, the system leaves the local minimum again and walks freely through the energy landscape.

As Fig. 11.7 shows, switching between the convex and the concave formula does not lead to a further improvement in the results. Here we decrease α from 5×10^4 to 1 exponentially with a factor of 0.99. At each α-step with $\alpha > 1$, we perform two steps, one in the convex and one in the concave mode. At $\alpha = 1$, we perform a greedy step in the original landscape. Thus, we multiply here the number of sweeps per α-step with $2 \times 1078 - 1 = 2155$,

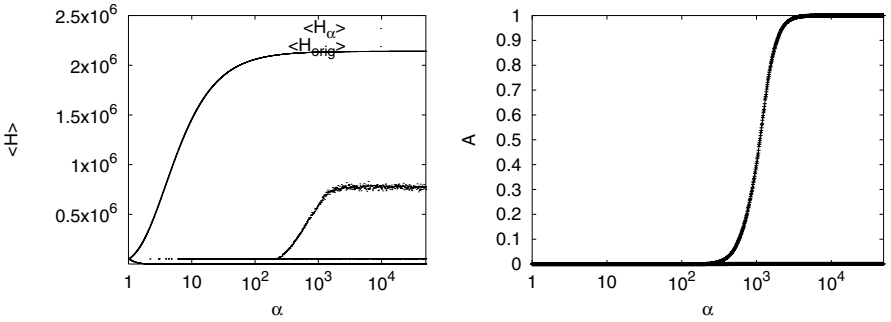

Fig. 11.6. Computational results for the mean energy $\langle \mathcal{H} \rangle$ (*left*) and the total acceptance rate A (*right*) of the PCB442 instance using both the convex formula (11.2) and the concave formula (11.3) for SSS. *Left*: both the mean energy $\langle \mathcal{H}_\alpha \rangle$ according to the smoothed distance values $D_{\max} \times d_\alpha(i,j)$ and the mean energy $\langle \mathcal{H}_{\text{orig}} \rangle$ calculated with the original distances $D(i,j)$ are shown. At each value of α, we first perform the sweeps in the concave mode, then print the values for $\langle \mathcal{H}_\alpha \rangle$, $\langle \mathcal{H}_{\text{orig}} \rangle$, and A, and then perform the same number of sweeps in the convex mode and print the results from that. Thus, the data *points* cannot be connected: since they jump between two *curves*, the *lines* would simply fill the *area* between both *curves*

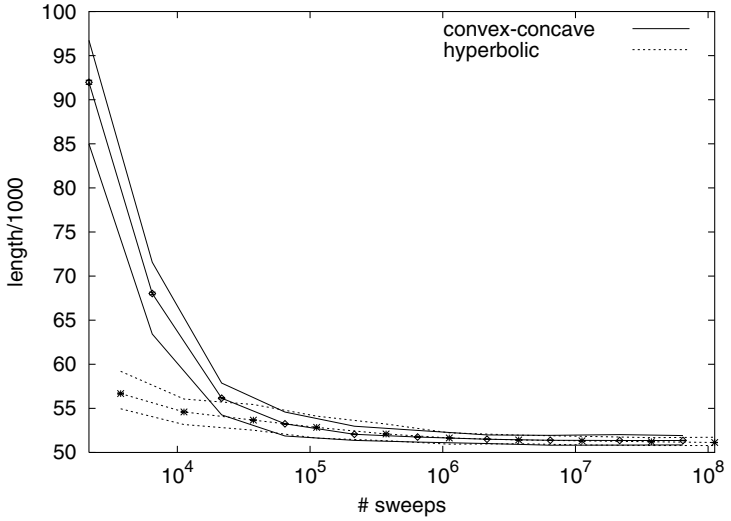

Fig. 11.7. Quality of the results achieved with SSS switching between the concave formula (11.3) and the convex formula (11.2) (*straight lines*) and using the hyperbolic formula (10.10) (*dotted lines*) for the PCB442 instance vs. overall number of sweeps: the three *lines* show minimum lengths, mean lengths (*error bars* are mostly the size of the *symbols*), and maximum lengths

the number of steps, in order to get the overall number of sweeps. We find that the results for this combined technique are again rather good but not as good as when using the concave formula alone.

Coy et al. changed the original power law smoothing formula by Gu and Huang to make it completely convex or concave. This can, of course, also be done with the other smoothing formulas introduced in the last chapter. For example, the sigmoidal smoothing formula (10.8) can be altered to

$$d_\alpha(i,j) = \frac{\tanh(\alpha d(i,j))}{\alpha}\,, \tag{11.4}$$

thus resembling an overall concave formula in the interval $[0;1]$. Figure 11.8 shows results for the mean energy and for the acceptance rate. The picture for the mean energy looks similar to that for the convex formula shown in Fig. 11.4: at large α, the mean value calculated with the smoothed distance matrix vanishes, while the mean value calculated with the original distance matrix shows that the system is in a RW mode. $\langle \mathcal{H}_\alpha \rangle$ continuously increases over the whole α-range. The mean value calculated with the original distance matrix exhibits a breakdown, after which both curves start to coincide. Much more interesting is the sight of the acceptance rate: for large α, the total acceptance rate jumps irregularly between total acceptance of all moves and a value of $0.428\ldots$. Both values start to decrease at the same α-values.

Figure 11.9 shows the results achieved with the sigmoidal–concave smoothing formula in comparison with results achieved using the original sigmoidal smoothing formula. We decreased α from 2000 to 0.01 by a factor of 0.99 when working with the sigmoidal–concave formula and added a greedy step in the original landscape at the end, thus having only 1216 α-steps. Just as with the power law smoothing formula, we find here also that this convex–

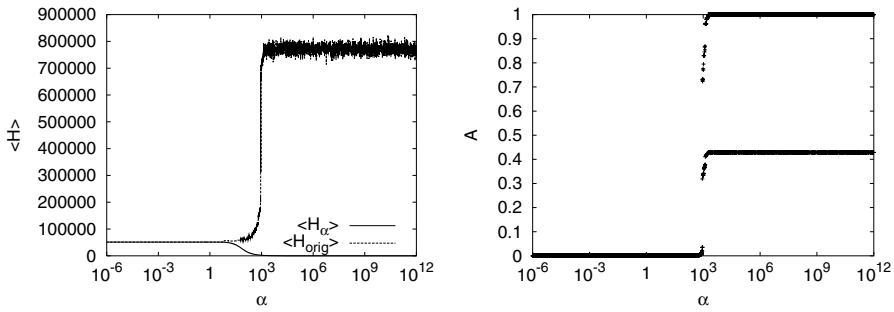

Fig. 11.8. Computational results for the mean energy $\langle \mathcal{H} \rangle$ (*left*) and the total acceptance rate A (*right*) of the PCB442 instance using the sigmoidal–concave formula (11.4) for SSS. *Left*: both the mean energy $\langle \mathcal{H}_\alpha \rangle$ according to the smoothed distance values $D_{\max} \times d_\alpha(i,j)$ and the mean energy $\langle \mathcal{H}_{\text{orig}} \rangle$ calculated with the original distances $D(i,j)$ are shown. The acceptance rate shown on *right* jumps between two values for large α

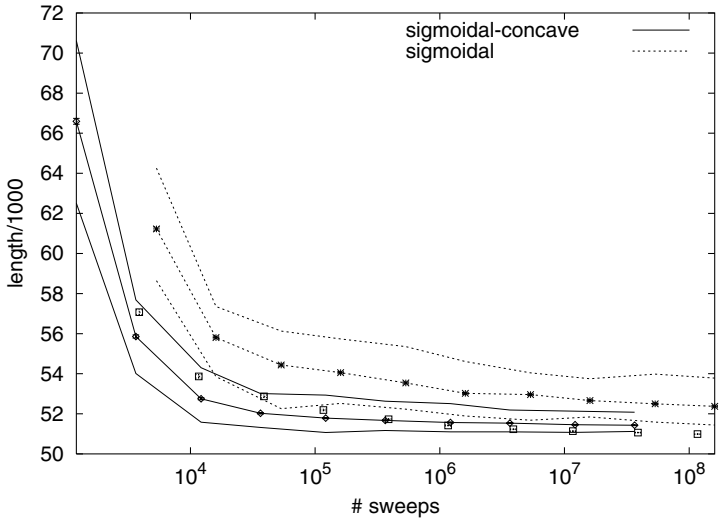

Fig. 11.9. Quality of the results achieved with SSS using the sigmoidal–concave formula (11.4) (*straight lines*) or the original sigmoidal smoothing formula (10.8) (*dotted lines*) for the PCB442 instance vs. overall number of sweeps: the three *lines* show minimum lengths, mean lengths (*error bars* are the size of the *symbols*), and maximum lengths

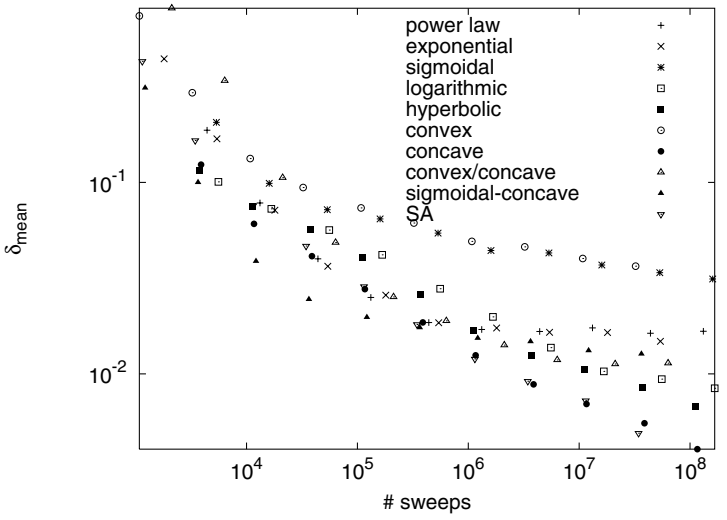

Fig. 11.10. Comparison of the qualities of results achieved for the PCB442 instance with various smoothing formulas introduced in the previous and in this chapter. δ_{mean}: mean relative deviation to the optimum as defined in formula (8.1). For comparison, the results for SA are also shown

concave approach by Coy et al. leads to an improvement over the original deviation approach of Gu and Huang.

Summarizing, Fig. 11.10 shows the mean deviation of the results achieved for the PCB442 instance for all the smoothing formulas introduced in this and in the last chapter: we clearly find that the concave smoothing formula leads to the best results, followed by hyperbolic, logarithmic, and convex/concave smoothing. The worst results are achieved with the convex and the sigmoidal smoothing formulas. The results for simulated annealing (SA), which are shown for comparison and redrawn from Fig. 7.9, show that SSS can lead to results similar to those for SA if an appropriate smoothing function is used.

11.2 Noising the System

Besides these and related techniques for smoothing the energy landscape, additional algorithms for changing the energy landscape have been developed. Based on the finding that similar problem instances usually exhibit similar ground states, the noising technique, which is also called the permutation technique, was developed. The basic idea is to change the original problem instance that is to be solved slightly again and again into other instances that look quite similar, get into a good local minimum of these, and then switch to the next altered problem instance. One hopes thus to cross barriers in the energy landscape of the original problem instance even when using the greedy algorithm for optimization.

For the traveling salesman problem (TSP), an instance can be altered by changing the entries in the distance matrix. This can be done in two ways: either one addresses the distances directly and changes them in a random way or one changes the locations of the nodes randomly, thus changing the distances. We prefer to change the locations of the nodes in order to stay with geometric Euclidean TSP instances. Furthermore, we found in our tests that it is advantageous to use some variable control parameter R that governs the size of the displacement of the nodes and that is decreased during the optimization run. Thus, the outline of our approach is as follows:

1. Create a random initial configuration σ for the proposed TSP instance.
2. Choose an appropriate starting value R of how much a node can be displaced at a maximum.
3. Derive from the original TSP instance a new instance: each node of the original TSP instance is displaced randomly within a squere of radius R around its original location.
4. Perform a greedy optimization run, starting at σ and ending at a configuration τ.
5. Set $\sigma := \tau$ and decrease R slightly.
6. If R has not yet reached some final small value, return to step 3.

7. Finally, perform a greedy optimization run starting at σ on the original problem instance (i. e., at $R = 0$) and print out the resulting configuration.

According to our tests on various TSP instances, it is advantageous to choose the initial and final value of R depending on the minimum distance $D_{\min} = \min\{D(i,j)|D(i,j) > 0\}$ occurring in the original problem instance. We found empirically that it is a good approach to decrease R from $10^3 \times D_{\min}$ to $10^{-2} \times D_{\min}$ exponentially before finally setting $R = 0$.

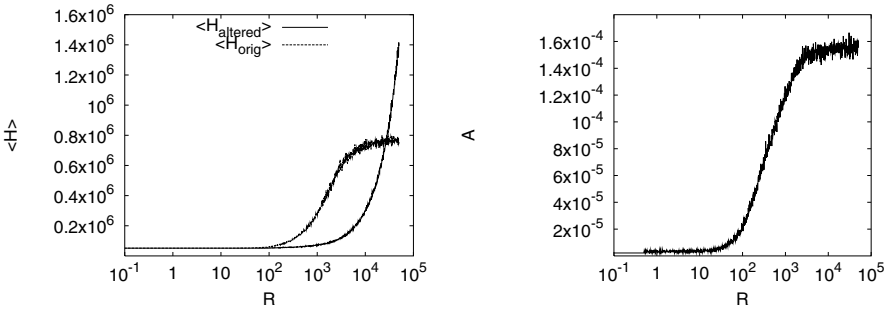

Fig. 11.11. Computational results for the mean energy $\langle\mathcal{H}\rangle$ (*left*) and the total acceptance rate A (*right*) of the PCB442 instance using the noising approach. *Left*: both the mean energy $\langle\mathcal{H}_{\text{altered}}\rangle$ of the altered instances and the mean energy $\langle\mathcal{H}_{\text{orig}}\rangle$ calculated with the original distances $D(i,j)$ are shown

The minimum distance in the PCB442 instance is 50, so that we decrease R from 5×10^4 to 0.5 exponentially with a factor of 0.99. At the end, we add a greedy step for the original instance, corresponding to $\alpha = 0$, such that we have overall 1147 steps. Figure 11.11 shows results for the mean energy and the total acceptance rate of the moves for the PCB442 instance. We have again two energies of interest: first, the energy of the altered instances that are optimized within an R-step is of interest. The curve for this $\langle\mathcal{H}_{\text{altered}}\rangle$ decreases linearly at large R. This is due to the fact that the individual nodes are also displaced far outside the boundaries of the curcuit board at these large R. In fact, the random displacements dominate the locations of the nodes at these large values. With decreasing R, the area where the nodes are located becomes smaller, such that the lengths found for these altered TSP instances decrease linearly with R. In addition, we calculate what length the sequence of the configuration would have if this sequence were the configuration of the original problem. We find that $\langle\mathcal{H}_{\text{orig}}\rangle$ takes values like those for random instances at first; thus, the system is in a kind of RW mode at these large R, when considered from the point of view of the original problem, although the acceptance rate is very small, of course, as the greedy criterion is applied. With decreasing R, $\langle\mathcal{H}_{\text{orig}}\rangle$ and the total

acceptance rate A decrease sigmoidally. In this range, the noising approach tackles a series of TSP instances that become increasingly identical with the original instance. Finally, the deviations from the original instance are so small that basically the original instance is tackled all the time, so that both $\langle \mathcal{H}_{\text{orig}} \rangle$ and $\langle \mathcal{H}_{\text{altered}} \rangle$ freeze and finally coincide at $R = 0$.

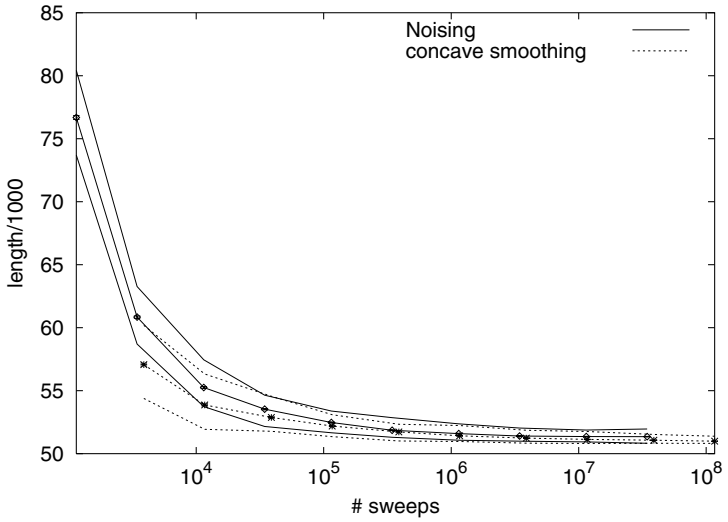

Fig. 11.12. Quality of the results achieved with the noising approach (*straight lines*) and with SSS using the concave formula (11.3) (*dotted lines*) for the PCB442 instance vs. overall number of sweeps: the three *lines* show minimum lengths, mean lengths (*error bars* are mostly the size of the *symbols*), and maximum lengths

Next we want to consider the quality of the results provided by this noising approach. In Fig. 11.12, these results are compared to the results of the concave smoothing formula (11.3), which provides the best results among all smoothing formulas for SSS introduced here. We find that the results achieved with our noising approach are only slightly worse than those achieved with SSS using the convex smoothing formula.

11.3 Weight Annealing

The weight annealing (WA) technique is closely related to the noising approach. Here weights are assigned to the individual parts of the problem in order to change the sizes of their corresponding addends in the cost function and thus their importance in the optimization run. In the application to the TSP, the individual parts are the nodes. We first rewrite the Hamiltonian of

the TSP as follows:

$$\begin{aligned}\mathcal{H}(\sigma) &= \sum_{i=1}^{N} D(\sigma(i), \sigma(i+1)) \\ &= \frac{1}{2} \sum_{i=1}^{N} \left(D(\sigma(i), \sigma(i+1)) + D(\sigma(i), \sigma(i-1)) \right) \\ &= \sum_{i=1}^{N} J_\sigma(\sigma(i)) \,,\end{aligned} \tag{11.5}$$

with $J_\sigma(\sigma(i)) = (D(\sigma(i), \sigma(i+1)) + D(\sigma(i), \sigma(i-1)))/2$, $\sigma(0) \equiv \sigma(N)$ and $\sigma(N+1) \equiv \sigma(1)$.

Thus, each node i contributes the value

$$J_\sigma(i) = (D(i, \sigma(\sigma^{-1}(i)+1)) + D(i, \sigma(\sigma^{-1}(i)-1)))/2 \tag{11.6}$$

to the Hamiltonian $\mathcal{H}(\sigma) = \sum_i J_\sigma(i)$. If we want to strengthen the importance of a certain node i, we will thus generally enlarge its addend to the Hamiltonian by some weight factor $W(i)$. Thus, we get a weighted Hamiltonian

$$\mathcal{H}_W(\sigma) = \sum_{i=1}^{N} W(i) \times J_\sigma(i) \,. \tag{11.7}$$

Note that here the sum runs over the nodes and not over the tour positions as above.

This approach opens a whole slew of new possibilities for optimization algorithms, especially if a control parameter like the temperature T in SA is used: when working with SA, the individual parts of the system could be treated at different local temperatures. Local bouncing strategies could be developed. These and other techniques could look during or after the optimization run at how well the individual parts of the problem are solved and then readjust the individual weights in order to give more emphasis to obviously harder to solve parts of the problem.

We want to restrict ourselves to the combination of this WA approach with the greedy algorithm. As with the noising approach, it is our aim to overcome barriers in the energy landscape by changing the TSP instance with these weights. As one can easily see by rewriting the Hamiltonian to

$$\begin{aligned}\mathcal{H}_W(\sigma) &= \sum_{i=1}^{N} \frac{1}{2} \left(W(\sigma(i)) + W(\sigma(i+1)) \right) \times D(\sigma(i), \sigma(i+1)) \\ &= \sum_{i=1}^{N} D_W(\sigma(i), \sigma(i+1)) \,,\end{aligned} \tag{11.8}$$

the distances of the TSP instance are "weighted" by incorporating the weights of the nodes, such that this approach is analogous to the noising approach.

We want to start out by choosing the weights of the nodes randomly. Based on the results of the noising approach in the last section, it is advantageous to decrease the amount of noise gradually, such that one starts out with instances that are rather random and ends at the original instance. Thus, the outline of this approach is as follows:

1. Initialize the system and create some random configuration σ.
2. Choose some large initial value for the parameter α by which the degree to which the weights can deviate from 1 is governed.
3. Choose the weights $W(i)$ randomly but in dependence on the parameter α.
4. Calculate the weighted distances $D_W(i,j)$.
5. Perform a greedy step on the Hamiltonian $\mathcal{H}_W$, starting from configuration σ and ending at configuration τ.
6. Set $\sigma := \tau$.
7. Decrease parameter α.
8. If α has not yet reached its final value, return to step 3 of this approach.
9. Finally, set all weights $W(i) = 1$ and perform a greedy step on the original instance, starting at σ.

Now we must develop some function with which we calculate these weights $W(i)$. Of course, this function must depend on α, but it also must incorporate some random element. We will try the following two "weighting functions" here:

- In the first approach, which is a power law approach, we choose random numbers $r(i)$ that are uniformly distributed in the interval $[-1;1]$ and set

$$W(i) = \alpha^{r(i)}\,. \tag{11.9}$$

  Thus, the weights have values in the interval $[1/\alpha;\alpha]$ and are uniformly distributed within this interval when plotting this interval using a logarithmic scale.
  We start out with a large initial α (using a value of 10^5 for the PCB442 instance) and decrease α exponentially toward 1.
- In the second linear approach, we increase α linearly from 0 to 1 in steps of, e. g., 10^{-3} and choose random weight values $W(i)$ that are uniformly distributed in the interval $[\alpha; 2-\alpha]$.

Figure 11.13 shows the development of the mean energy and of the total acceptance rate for both weighting approaches. The curve for the reweighted Hamiltonian fluctuates around and decreases with the curve for the original Hamiltonian for the power law weighting technique. We find that mostly $\langle\mathcal{H}_W\rangle \leq \langle\mathcal{H}\rangle$ for small α. The acceptance rate shows a sigmoidal decrease as in the noising approach. The linear weighting approach leads to other results: here α increases linearly from 0 to 1, such that we must read the graphic from left to right here. $\langle\mathcal{H}_W\rangle$ increases with increasing α, while $\langle\mathcal{H}\rangle$ decreases. Both curves coincide for $\alpha \to 1$.

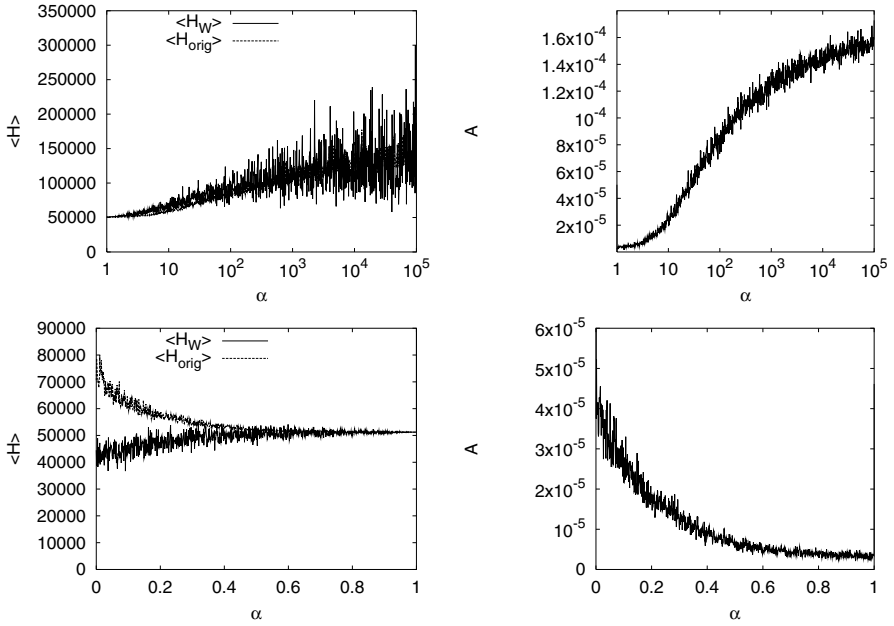

Fig. 11.13. Computational results for the mean energy $\langle \mathcal{H} \rangle$ (*left*) and the total acceptance rate A (*right*) of the PCB442 instance using the WA approach. *Left*: both the mean energy $\langle \mathcal{H}_W \rangle$ of instances with the reweighted distances and the mean energy $\langle \mathcal{H}_{\text{orig}} \rangle$ calculated with the original distances $D(i, j)$ are shown. *Top*: results for the power law weighting technique; *bottom*: results for the linear weighting technique

Finally, we want to compare the quality of the results achieved with these weighting approaches. We used 1147 α-steps when working with the power law weighting approach and 1001 α-steps using the linear weighting approach. Figure 11.14 shows the quality of the results of these approaches. We find that the linear approach leads to significantly better results than the power law approach for short computing times and remains still better for long computing times.

Of course, these two approaches are only a start in the struggle to apply this WA technique to the TSP, as they change the weights of the nodes at random. One might think of more elaborate techniques according to the philosophy of this approach: if a part of the system is badly solved, then the corresponding weight of this part will be enlarged. Thus, the question arises as to how to measure how well or how poorly a part is solved. We tried to introduce a measure with a rather simple approach: as introduced in Sect. 1.8, the misfit parameter measures how large the frustration in a TSP instance is. One can also use this parameter here: a TSP instance is solved locally optimally at some node if this node is connected to its two nearest neighbors. Let $n_1(i)$ and $n_2(i)$ be the nearest and the next nearest neighbors

of node i. Then one must calculate

$$D(\sigma(i), \sigma(i+1)) + D(\sigma(i), \sigma(i-1)) - D(\sigma(i), n_1(\sigma(i))) - D(\sigma(i), n_2(\sigma(i))) \tag{11.10}$$

in order to measure how large the deviation from local optimality is for node $\sigma(i)$. This measure could then be introduced in finding new weight values $W(\sigma(i))$. We tried several approaches but found none better than our linear approach with the randomly chosen weights above. We think that this is due to the fact that usually some local part of the system should not be solved optimally in order to get to the optimum of the overall system. This is just the point when dealing with complex problems, that the overall global optimality is not given by the sum of the local optimalities.

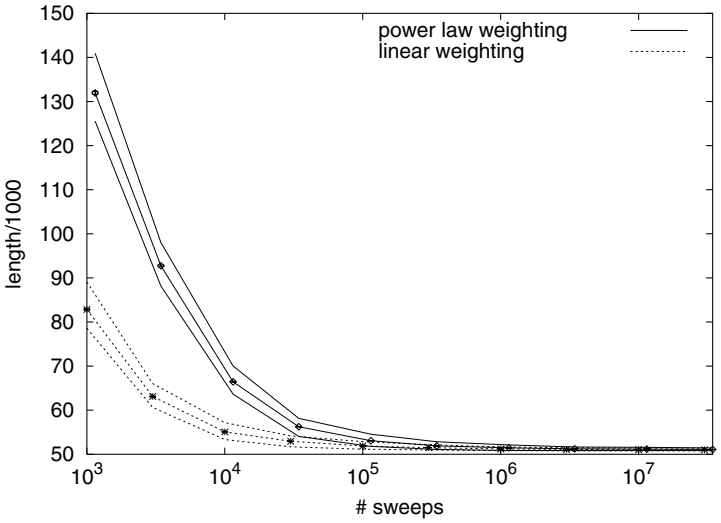

Fig. 11.14. Quality of the results achieved with WA using the power law approach (*straight lines*) and the linear approach (*dotted lines*) for the PCB442 instance vs. overall number of sweeps: the three *lines* show minimum lengths, mean lengths (*error bars* are mostly the size of the *symbols*), and maximum lengths

11.4 Final Remarks on Application of Changing Techniques

In our implementations of SSS, noising, and WA, we generally worked with a control parameter, usually called α, introduced by Gu and Huang [76]. This control parameter was gradually decreased over many steps. One sometimes also finds implementations in the literature in which the schedule for

α is decreased to only a few steps but for which still moderately or even very good results are achieved. The exact values the smoothness control parameter takes in these applications usually depend not only on, e. g., the smoothing function but also on the problem instance. Thus, in these approaches, first, these important smoothing values must be found either by trial and error or by some assumptions or within the kind of optimization runs we performed here. Of course, if only these important smoothness values are used, the calculation time in the production runs can then be decreased to a large extent. However, we prefer to present results here for straightforward implementations for which it is easier to work with a schedule with many α-values as the important ones are usually unknown a priori and are sometimes hard to find.

Secondly, we would like to comment on the basic heuristics to be used in combination with these approaches that change the energy landscape: it is not necessary to work strictly with the greedy algorithm, which does not accept any deterioration. But one cannot start a complete run with, e. g., SA at each α-step, as the local valley will be left at large temperatures, such that the guidance effect of SSS and of the other techniques is lost. As shown in [186], the greedy algorithm can be replaced by the great deluge algorithm (GDA): in that work an optimization run was performed using the GDA at each α-step. The initial value for the water level was chosen as the energy of the current configuration such that the system was forced to stay in the local valley. The GDA is thus very well suited to be combined with these approaches. But the problem with this combination is that the GDA converges very slowly, such that there is no great advantage from using the GDA instead of the greedy and if one does not want to spend too much additional calculation time. However, threshold accepting (TA) may also be used in combination with, e. g., SSS: in the threshold search space smoothing (TSSS) approach, a small constant threshold is introduced and the TA criterion is applied instead of the greedy acceptance criterion. With decreasing α, the energy differences occurring in the smoothed energy landscape become larger, such that more and more moves are rejected. Thus, the system performs both a SSS and a TA transition. However, when α approaches its final value, the system might not freeze completely. Thus, one must memorize the best solution found in this late stage of the optimization algorithm and print this solution as the result of the optimization run.

The third comment we want to make is that, although there are differences in the philosophies and the behaviors of the algorithms between SA and its relatives on the one side and SSS and its relatives on the other side, the algorithms have much in common: both types of algorithms construct a Markov chain of configurations that hopefully ends at a (quasi) optimum configuration. Both types inherit some control parameter that is changed during the optimization such that the system is transferred gradually from a quasi-RW mode into a quasigreedy mode. Now we will move on to completely different approaches. We will, however, find similar concepts in these approaches, such as the usage of a variable control parameter.