An Ant Colony Optimization Approach to Dynamic TSP

Michael Guntsch[1], **Martin Middendorf**[2], **Hartmut Schmeck**[3]
Institute AIFB
University of Karlsruhe
D-76128 Karlsruhe, Germany
{[1]mgu,[2]mmi,[3]hsch}@aifb.uni-karlsruhe.de

Abstract

An Ant Colony Optimization (ACO) approach for a dynamic Traveling Salesperson Problem (TSP) is studied in this paper. In the dynamic version of the TSP cities can be deleted or inserted over time. Specifically, we consider replacing a certain number of cities with new ones at different frequencies. The aim of the ACO algorithm is to provide a good solution quality averaged over time, i.e. the average taken of the best solution in each iteration is optimized. Several strategies for pheromone modification in reaction to changes of the problem instance are investigated. The strategies differ in their degree of locality with respect to the position of the inserted/deleted cities and whether they keep a modified elitist ant or not.

1 Introduction

Evolutionary methods are in general capable of reacting to dynamic changes of an optimization problem. Different ways to trim Evolutionary Algorithms (EAs) for dynamic problems has have been proposed over the last years (see (Branke, 1999) for a short overview). A key aspect is whether information connected to solutions found for older stages of a problem can be used to quickly find a good solution for the problem after a change occurred.

In this paper we explore strategies to apply Ant Colony Optimization (ACO) for solving dynamic optimization problems (see (Bonabeau, 1998), (Dorigo and Di Caro, 1999) for an overview of ACO). The particular test problem we study is a dynamic Traveling Salesperson Problem (TSP) where instances change at certain intervals through the deletion and insertion of cities. This problem is general enough to be interesting as a benchmark problem and allows modifying the degree of change easily. As a practical application, consider the case of a factory with a fluctuating set of active machines. In case of a failure in the production system it is necessary to start an inspection tour immediately to check all previously active machines. This makes it beneficial to constantly know a short tour for the set of active machines.

We use modified and extended strategies from a previous study done by two of the authors where the reaction of the ant algorithms to a single change of the problem instance was investigated (Guntsch and Middendorf, 2001). The only other dynamic optimization problems to which ACO has been applied are routing problems in communication networks where the traffic in the network continually changes (e.g. (Di Caro and Dorigo, 1998), (Schoonderwoerd et al., 1996)). The ants were used to continually measure the travel time between pairs of nodes and this information is used to update the routing tables (which contain the pheromone information). This allows the system to adapt to new traffic situations but it does not provide any means for reacting explicitly to single changes.

Dynamic optimization problems are most interesting when changes of the problems instances occur frequently and each change is not too large so that it is likely that the new optimal solutions will be in some sense related to the old ones. In this case a simple restart of the algorithm which discards all old information after a change has occurred might not be a good strategy. Instead, maintenance of some previously determined knowledge in the form of pheromone information should be beneficial. To this end a tradeoff must be found between the opposing goals of preserving pheromone information and resetting enough to allow the ants to explore new relevant areas of the search space in later iterations. Based on this general idea, we propose and test three different strategies and combi-

nations thereof to make ant algorithms more suitable for the optimization in dynamic environments. Moreover, we propose an elitist strategy for use in dynamic environments. A standard elitist strategy for ant algorithms is that an elitist ant updates the pheromone values in every generation according to the best solution found so far. But after a change of the problem instance the best solution found so far will usually no longer represent a valid solution to the new instance. Instead of simply forgetting the old best solution, we study an alternative approach where the old best solution is adapted so that it becomes a reasonably good solution for the new instance.

The basic structure of the ant algorithm is presented in Section 2. In Section 3, the strategies for modifying the pheromone information are described. The test problems and the used parameter values are provided in Section 4, with the results discussed in Section 5. The paper concludes with a summary in Section 6.

2 The Ant Algorithm

In this section we describe only the general approach of our algorithm for the TSP. The strategies added for applying it to the dynamic TSP are presented later in Section 3. Ant algorithms have been applied for the (static) TSP problem by several authors (Bullnheimer et al., 1999), (Dorigo, 1992), (Dorigo et al. 1996), (Dorigo and Gambardella, 1995), (Dorigo and Gambardella, 1997), (Stützle and Hoos, 1997). Our algorithm for the TSP follows (Dorigo et al., 1996).

In every iteration each of m ants constructs one tour through all the given n cities. Starting at a random city an ant builds up a solution iteratively by always selecting the next city based on heuristic information as well as pheromone information. Pheromone information serves as a form of memory by indicating which choices were good in the past. The heuristic information, denoted by η_{ij}, and the pheromone information, denoted by τ_{ij}, are indicators of how good it seems to move from city i to city j. The heuristic value is $\eta_{ij} = 1/d_{ij}$ where d_{ij} is the distance between city i and city j.

With probability q_0, where $0 \leq q_0 < 1$ is a parameter of the algorithm, an ant at city i chooses the next city j from the set S of cities that have not been visited so far which maximizes $[\tau_{ij}]^\alpha [\eta_{ij}]^\beta$, where α and β are constants that determine the relative influence of the heuristic values and the pheromone values on the decision of the ant. With probability $1 - q_0$ the next city is chosen according to the probability distribution over S determined by

$$p_{ij} = \frac{[\tau_{ij}]^\alpha [\eta_{ij}]^\beta}{\sum_{h \in S} [\tau_{ih}]^\alpha [\eta_{ih}]^\beta}$$

Before doing a global pheromone update some of the old pheromone is evaporated on all edges according to $\tau_{ij} \mapsto (1 - \rho) \cdot \tau_{ij}$ where parameter ρ determines the evaporation rate. For pheromone update the ant that found the best solution in that generation updates pheromone along its solution, i.e. for every city i some amount of pheromone is added to element (i, j) of the pheromone matrix when j is the successor of i in the tour. Observe that pheromone is added to exactly two edges incident to a node i. The amount of pheromone added is $\rho/4$, that is $\tau_{ij} \mapsto \tau_{ij} + \frac{1}{4}\rho$. We also apply an elitist strategy where one elitist ant updates pheromone along the best solution that has been found so far.

For initialization we set $\tau_{ij} = 1/(n-1)$ for every edge (i, j). Observe, that for every city i the sum of the pheromone values on all incident edges is one, which is not changed by the pheromone update.

3 Reacting to a Change

The ant algorithm that was described in the last section can not handle the dynamic TSP. When a change of the problem instance occurred it is necessary to initialize the pheromone information for the new cities. Moreover, it might also be important to modify the pheromone information for the old cities that were not deleted. We describe three strategies and combinations thereof for resetting part of the pheromone information in reaction to a change of the problem instance. Resetting information is achieved by equalizing the pheromone values to some degree, which effectively reduces the influence of experience on the decisions an ant makes to build a solution.

Strategies for the modification of pheromone information have been proposed before to counteract stagnation of ant algorithms. In (Gambardella et al., 1999) it was proposed to reinitialize the whole pheromone matrix while (Stützle and Hoos, 1997) suggested to increase the pheromone values proportionately to their difference to the maximum pheromone value. Similar to these approaches we use a global pheromone modification strategy which reinitializes all the pheromone values by the same degree. However, this method, which we call Restart-Strategy, is limited because it does not take into account where the change of the problem instance actually occurred. Usually, the most extensive resetting of pheromone values should be performed in the close vicinity of the inserted/deleted

cities. A more locally oriented update strategy is the "η-Strategy" which uses heuristic based information, distances between cities in this case, to decide to what degree equalization is done on the pheromone values on all edges incident to a city j. The "τ-Strategy" uses pheromone based information, i.e. the pheromone values on the edges, to define another concept of "distance" between cities. Equalization of pheromone values is then again performed to a higher degree on the edges of "closer" cities.

All three strategies work by distributing a reset-value $\gamma_i \in [0:1]$ to each city i which determines the amount of reinitialization the pheromone values on edges incident to i according to the equation

$$\tau_{ij} \mapsto (1-\gamma_i)\tau_{ij} + \gamma_i \frac{1}{n-1} \qquad (1)$$

In case of a problem with symmetric η-values like Euclidean TSP, the average of the reset-values $(\gamma_i+\gamma_j)/2$ is used instead of γ_i in equation 1 for modifying the pheromone value on the edge connecting cities i and j. An inserted city i always receives an unmodifiable reset-value of $\gamma_i = 1$, resulting in all incident edges to i having the initial pheromone value of $1/(n-1)$. We will now describe in more detail how the different strategies assign the values γ_i.

3.1 Basic Strategies

The Restart-Strategy assigns each city i the strategy-specific parameter $\lambda_R \in [0,1]$ as its reset-value, i.e. $\gamma_i = \lambda_R$.

In the η-Strategy, each city i is given a value γ_i proportionate to its distance from the nearest inserted/deleted city j. This distance d^η_{ij} is derived from η_{ij} in such a way that a high η_{ij} implies a high d^η_{ij} and that scaling the heuristic η-values has no effect:

$$d^\eta_{ij} = 1 - \frac{\eta_{avg}}{\lambda_E \cdot \eta_{ij}}$$

with $\eta_{avg} = \frac{1}{n(n-1)} \sum_{i=1}^{n} \sum_{k \neq i} \eta_{ik}$ and the strategy-specific parameter $\lambda_E \in [0,\infty)$ scaling the width of the distance-cone. A city i then receives $\gamma_i = \max\{0, d^\eta_{ij}\}$ (see the example in Figure 1).

The τ-Strategy uses a distance measure based on pheromone information to calculate the reset-values. The pheromone-distance d^τ_{ik} between two cities i and k is basically defined as the maximum over all paths P_{ik} from i to k of the product of pheromone-values on the edges in P_{ik}. To prevent any incompatibility due to the size of absolute values, the pheromone-values on the edges are scaled by the maximum possible pheromone value on an edge τ_{max} [1]. Formally,

$$d^\tau_{ik} = \max_{P_{ik}} \prod_{(x,y)\in P_{ik}} \frac{\tau_{xy}}{\tau_{max}}.$$

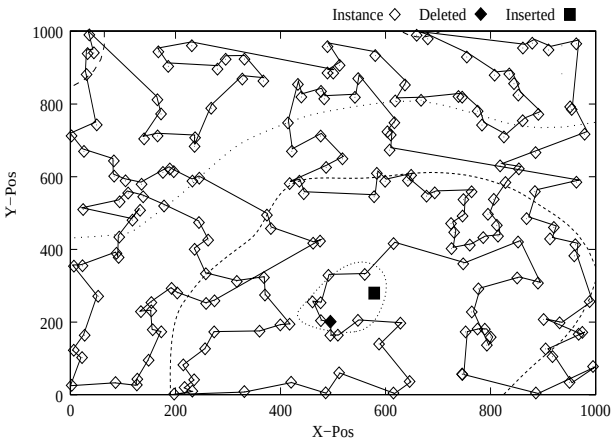


Figure 1: η-Strategy: TSP test instance with best found solution; contures show different level of reset-values - highest reset occurs near inserted/deleted city.

For the case of insertion, we set the pheromone value of the edges from the inserted city to the two closest cities, i.e. those with the highest value for η_{ij}, to τ_{max} during the application of this strategy, since the new city does not yet have any utilisable pheromone information. With J being the set of all cities that are inserted or deleted during the same change, only the maximum value $\max_{j\in J} d^\tau_{ij}$ is recored for each city i. When multiplied with a strategy-specific parameter $\lambda_T \in [0,\infty)$, with the result limited to 1 for application of equation 1, this gives the reset-value for city i: $\gamma_i = min\{1, \lambda_T \cdot d^\tau_{ij}\}$.

3.2 Combined Strategies

A combination of the global Restart-Strategy with one of the two more locally acting η- or τ-Strategies could be advantageous in a situation where strong local resetting near the inserted/deleted cities is necessary to incorporate a change while a lower global resetting is needed to maintain the flexibility for the algorithm to change the best tour found more strongly if beneficial. This combination can be realized by having each of the two strategies involved distribute reset-values according to their respective scheme and then

[1] τ_{max} is 0.5 for symmetric and 1.0 for asymmetric TSP.

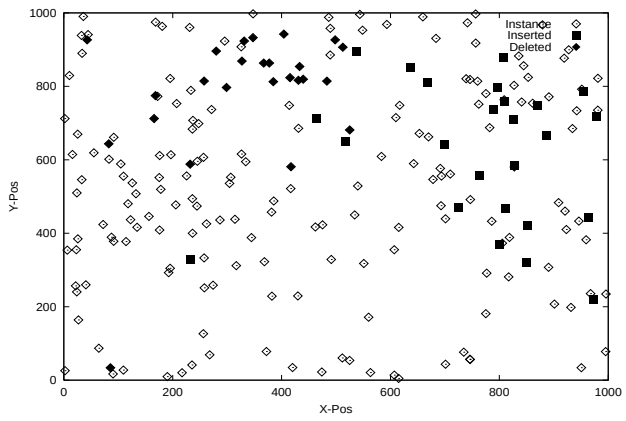


Figure 2: TSP test instance with $k = 25$ deleted and inserted cities determined with $p = 2$.

choosing for each city i the maximum of the two reset-values determined by the two strategies. Formally, if the first strategy distributes $\gamma_{i,1}$ and the second $\gamma_{i,2}$, then $\gamma_i = \max\{\gamma_{i,1}, \gamma_{i,2}\}$.

3.3 Keeping the Elitist Ant

Whenever a change to the instance that the algorithm is running on occurs, the elitist ant, which enforces the best solution found so far, no longer represents a valid solution. Consequently, it must be dropped and a new elitist ant is determined after the first iteration of ants has worked on the changed instance. The possible loss of information from dropping the old best solution can be alleviated somewhat by modifying the former best tour so that it once again yields a valid and presumably good solution to the changed instance. We use two greedy heuristics for performing this modification: i) all cities that were deleted from the instance are also deleted from the old best tour, effectively connecting their respective predecessors and successors, ii) the cities that were added are inserted individually into the tour at the place where they cause the minimum increase in length. The tour derived from this process is the new tour of the elitist ant. We call this method KeepElitist. Clearly, this modification can be combined with the strategies explained above.

4 Test Setup

For our tests we chose subproblems of the Euclidean TSP instance `rd400` from the (TSP-Library, 2001). Specifically, 200 random cities were taken away from the 400 making up the problem instance to form a spare pool of cities before the start of the algorithm, leaving the instance with 200 cities. During the run of the algorithm the actual problem instance was changed every t iterations by exchanging k cities between the actual instance and the spare pool, i.e. k cities were deleted from the actual instance and the same number of cities from the spare pool were inserted. When deciding which cities to delete, the first city j was chosen at random and all other cities i according to a probability distribution defined by η_{ij}^p, with p being a parameter that determines the relative influence of the distances between i and j. The cities that were inserted were chosen analogously from the spare pool. An example is shown in Figure 2.

We tested all combinations of parameter values $k \in \{1, 5, 25\}$, $t \in \{50, 200, 750\}$, and $p \in \{0.0, 2.0\}$. Note that for $k = 1$, the parameter p has no effect as only one city is removed/inserted. For each configuration (k, t, p), 10 test runs of 8999 iterations were done (in iteration 9000, the next change would occur for all tested t), each starting with a different random subset of 200 cities. All results that were used as a basis for comparison are averages over these 10 runs. Only the results during iterations 3000-8999 were used to measure the performance of the applied strategies, since the behaviour of the ant algorithm during the first iterations is not representative for the latter stages.

The parameter values for the ant algorithm used in the tests were $m = 10$ ants, $\alpha = 1$, $\beta = 5$, $q_0 = 0.9$, and $\rho = 0.05$. The heuristic weight of $\beta = 5$ has been used by several authors (e.g. (Bullnheimer at al. 1999), (Stützle and Hoos, 1997)) for TSP.

We tested the parameters $\lambda_R \in \{0.25, 0.5, 0.75, 1.0\}$ for the Restart-Strategy, $\lambda_E \in \{0.5, 1.0, 2.0, 5.0\}$ for the η-Strategy, and $\lambda_T \in \{0.5, 1.0, 1.5, 2.0\}$ for the τ-Strategy. A parameter value of 0.0, which is equivalent for all strategies and corresponds to not applying the strategy at all, was also tested. Furthermore, we combined the Restart-Strategy with $\lambda_R \in \{0.25, 0.5\}$ with the η- and τ-Strategy using their respective parameter-values above to determine if such a combination can yield better results than the "pure" strategies by itself. Finally, all of the above settings were tested with and without keeping a modified elite ant as described in Section 3.3 after the exchange of cities.

Besides the best solutions found by the ant algorithm we also recorded the normalized entropy $E \in [0, 1]$ of the pheromone matrix in every iteration, which is defined as

$$E = \frac{1}{n \log n} \sum_{i=1}^{n} \sum_{j=1}^{n} -\tau_{ij} \log(\tau_{ij})$$

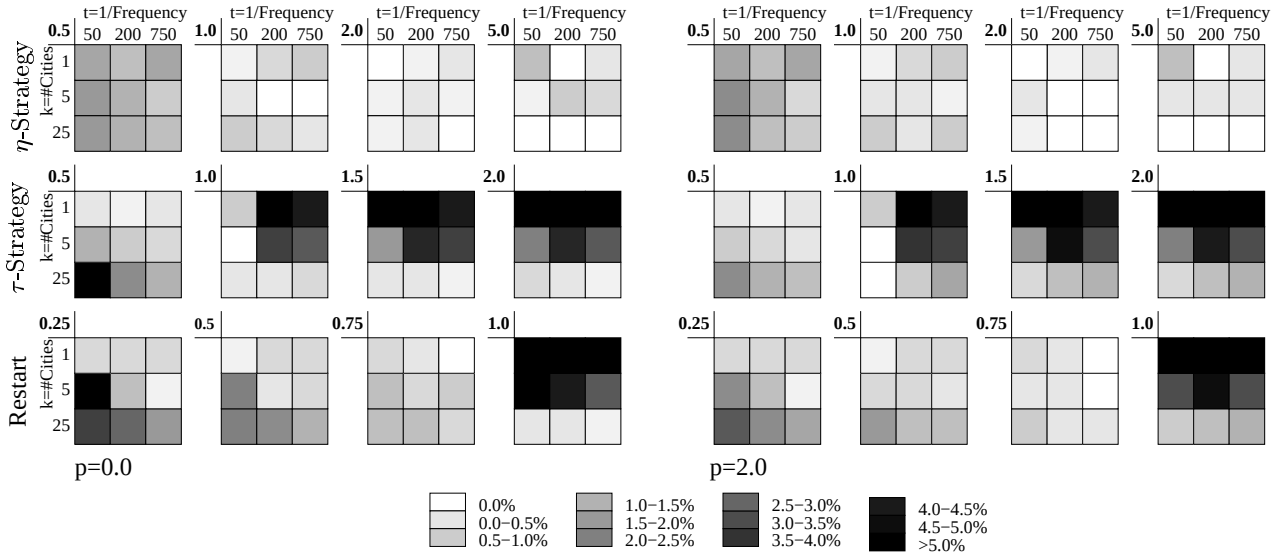


Figure 3: Relative performance of Restart-, η- and τ-strategy for $p = 0$ (left) and $p = 2$ (right) and different values of k and t: loss in quality of the best found solution averaged over iterations 3000-8999 compared to the best performing variant.

Normalized entropy has been used previously in (Guntsch and Middendorf, 2001) to help understand the current state of the ant algorithm for a given iteration.

5 Empirical Evaluation

A comparison of the Restart-, η- and τ-Strategies for the respective parameter values described in Section 4 is shown Figure 3. Judging from "average darkness", the best overall strategy is the the η-Strategy with a parameter $\lambda_E = 2.0$, especially for a high degree of proximity for the cities inserted and removed. The τ-Strategy with $\lambda_T = 1.0$ provides good to very good solutions when changes occur quickly, i.e. for $t = 50$. The Restart-Strategy, when given enough time and not confronted with changes that are too severe, is also able to achieve good solutions for $\lambda_R = 0.75$. A complete restart, i.e. using the Restart-Strategy with $\lambda_R = 1.0$, is only comparable to the other strategies for the cases where many cities are exchanged, even beating some of the other strategies when they do not reset enough information. This would likely increase if even more cities were transferred as the changed problems would become almost independent of one another.

As for the influence of the proximity-value p, is seems that the difference in the solution-quality achieved by the individual strategies becomes less for $p = 2$ compared to $p = 0$. For the local strategies, a stronger proximity of the exchanged cities is beneficial because a cluster of cities being inserted or deleted will cause a distribution of reset-values that is not as much dependent on the number of cities comprising the cluster as on their position in the graph or their degree of connectivity in the pheromone matrix. Therefore, although the transfer of cities might be large, the local confinement of this change makes it easier to incorporate for the local strategies. The Restart-Strategy, however, also benefits from a higher degree of proximity. This is probably again due to the "bad" pheromone information being more centralized than for the case of equal distribution, and therefore easier to deal with for the ant algorithm.

Figure 4 shows a more detailed view of the optimization behavior for the individual strategies and its dependency on their respective λ-parameters (λ_E, λ_T, λ_R) for the case of $(k, t, p) = (1, 50, 0)$, i.e. frequent occurring small changes. In particular, the effect on increasing the λ-parameter-values can be seen. The η-Strategy only slowly becomes worse in terms of solution quality, despite resetting a lot of pheromone for high values of λ_E, as is indicated by the entropy-curves in Figure 4. In contrast, the τ-Strategy shows a significant loss of performance for $\lambda_T > 1$, even though the entropy curve indicates that the increase in reset information is only moderate. For the Restart-Strategy, we see U-shaped parameter-dependency curves for resulting solution quality, which shows that not resetting enough as well as resetting too much information has

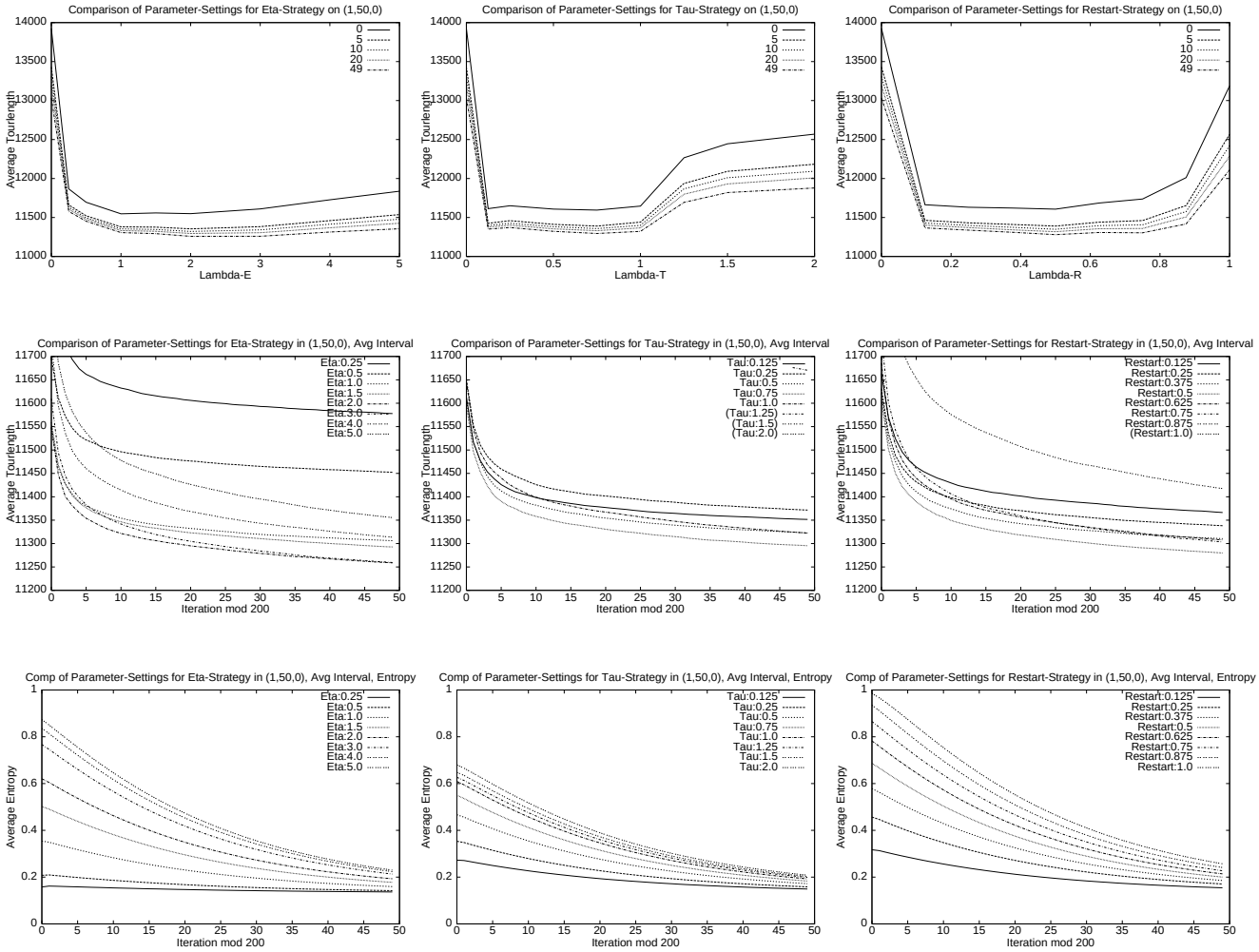
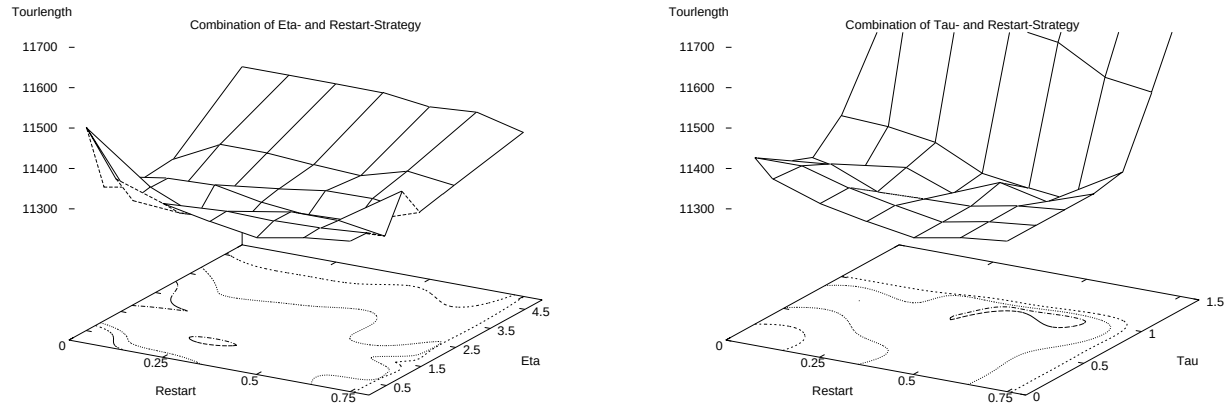


Figure 4: Average quality of best found solution with respect to number of iteration after a change forη-Strategy (left), τ-Strategy (center), and Restart-Strategy (right) for different parameter values $\lambda_E, \lambda_T, \lambda_R$ and the configuration $(k, t, p) = (1, 50, 0)$. The upper row shows curves of average solution quality for certain iterations after a change, the middle row the averaged behavior for a certain parameter over a t-Interval, and the lower row shows the corresponding entropy values to the middle row.

a negative effect on the derived solution. As with the local strategies, the biggest performance gain can be observed when going from doing nothing, i.e. using a parameter-value of 0.0, to doing even just a little, i.e. setting $\lambda_E = 0.25$, $\lambda_T = 0.125$, and $\lambda_R = 0.125$. The curves for the Restart-Strategy also show that the difference from resetting almost all pheromone information to actually resetting all of it is enormous in terms of solution quality when using this strategy.

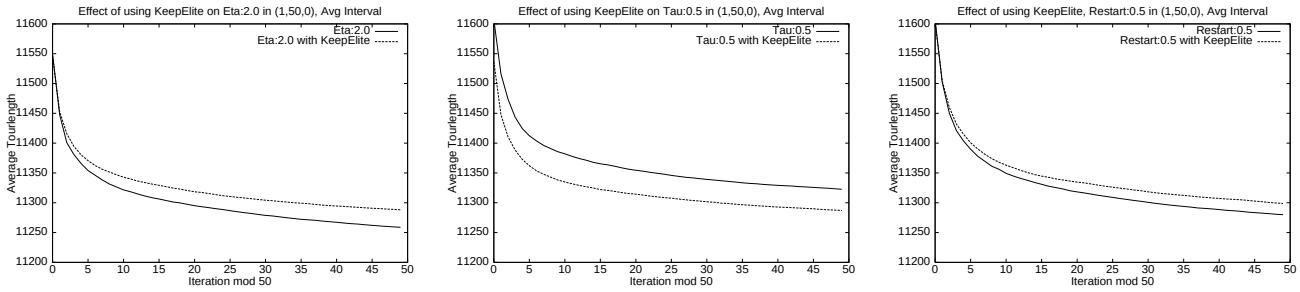
As mentioned in Section 4, we also analyzed the performance of combinations of the local η- and τ-Strategies with the Restart-Strategy. For some cases, this combination provided better solutions than any of the strategies could achieve by itself. An example of this is the configuration $(k, t, p) = (1, 50, 0)$ shown in Figure 5, for which we performed additional parameter-tests to make a more precise analysis. The contour lines for the combination of the η- and Restart-Strategy show that there are two areas in which good performance was achieved, one of them a true combination with $\lambda_E = 1$ and $\lambda_R = 0.25$, and the other one, which is better in terms of solution quality as well as larger, with $\lambda_E \in \{2, 3\}$ and $\lambda_R = 0$. This suggests that the η-Strategy does not benefit much, if at all, from being combined with Restart. For the τ-Strategy on the other hand we see a promising area located around a combination of medium λ_T and λ_R values, specif-



(a) η-Restart Combination

(b) τ-Restart Combination

Figure 5: Combinations of the local η- and τ-Strategies with the Restart-Strategy for the problem-configuration $(k, t, p) = (1, 50, 0)$.



(a) η-Strategy

(b) τ-Strategy

(c) Restart-Strategy

Figure 6: Combination of the individual strategies with KeepElitist for the configuration $(k, t, p) = (1, 50, 0)$.

ically for $\lambda_T = 1.0$ and $\lambda_R \in \{0.5, 0.675\}$, and also for $\lambda_T = 0.75$ and $\lambda_R = 0.375$. Thus the combination of the τ- and Restart-Strategy performs better than either strategy by itself, and also better than the η-Strategy in this case, justifying its application.

Finally, we combined the KeepElitist method with the individual strategies as well as the combinations of the η- and τ-Strategies with the Restart-Strategy. Figure 6 shows how this modified the average behavior of the pure strategies with their respectively best λ-parameter on configuration $(k, t, p) = (1, 50, 0)$. As can be observed, for the η- and Restart-Strategy the combination on average entailed a worse solution, while for the τ-Strategy the effect on average was an improvement. Overall, the heuristic of keeping a modified elite ant was beneficial only when the number of cities k that was inserted and deleted was not too large and when the time for adapting to the problem t was small. If too many cities were exchanged, then the heuristic would no longer provide a good solution, and the ants would find a better solution in the first iteration after the change. This case is not dangerous, since keeping the modified elitist ant would be the same as not keeping it; only the "new" elitist ant would update the pheromone matrix. The second case in which keeping an elitist ant does not entail better solutions is when the interval t between changes is long enough to permit the algorithm to adapt very well to the new instance, and the guidance provided by an early good solution leads toward stagnation in the end. This case is potentially dangerous, as the elitist ant survives the first generation(s) and influences the pheromone ma-

trix, thereby restricting the search space to a region that is perhaps not very promising.

6 Conclusion

In this paper, we studied strategies for helping ant algorithms deal with a highly dynamic TSP. We modified three strategies proposed for the case of a single change of the problem instance. Using combinations and a heuristic for keeping a modified elitist ant, we were able to find better solutions for various problem classes than the pure strategies by themselves. We have also distinguished what type of problem classes seem to favor which strategy for dealing with changes, and what type of parameter to use for the different strategies in such a case.

Overall, we have shown empirically that the local strategies perform best when problem changes occur frequently so that the algorithm does not have enough time to reset "blindly" and reoptimize the entire instance. Future work could clarify where exactly the boundary lies between sensible local resetting and global resetting. Also, the state of convergence that the ant algorithm has achieved could codetermine the ideal strength of resetting in reaction to a change. Lastly, it might be that the strategies for resetting pheromone could successfully be applied in a static environment when stagnation of the search process is imminent for the entire instance or parts of it.

References

B. Bullnheimer, R.F. Hartl, and C. Strauss (1999). A New Rank Based Version of the Ant System - A Computational Study. *Central European Journal of Operations Research* **7**: 25-38.

E. Bonabeau, M. Dorigo, and G. Theraulaz (1999). *Swarm Intelligence: From Natural to Artificial Systems*, Oxford University Press, New York.

J. Branke (1999). Evolutionary approaches to dynamic optimization problems - a suvey. In A. Wu (ed.) GECCO Workshop on Evolutionary Algorithms for Dynamic Optimization Problems, 134-137. San Mateo, CA: Morgan Kaufmann.

G. Di Caro and M. Dorigo (1998). AntNet: Distributed Stigmergetic Control for Communications Networks. *Journal of Artificial Intelligence Research* **9**: 317-365.

M. Dorigo (1992). Optimization, Learning and Natural Algorithms (in Italian). PhD Thesis, Dipartimento di Elettronica, Politecnico di Milano, Italy, pp.140.

M. Dorigo and G. Di Caro (1999). The ant colony optimization meta-heuristic. In D. Corne, M. Dorigo, F. Glover (eds.), *New Ideas in Optimization*, 11-32, McGraw-Hill.

M. Dorigo and L.M. Gambardella (1995). Ant-Q: A Reinforcement Learning approach to the traveling salesman problem. In *Proceedings of ML-95, Twelfth International Conference on Machine Learning*, 252-260. San Mateo, CA: Morgan Kaufmann.

M. Dorigo and L.M. Gambardella (1997). Ant colony system: A cooperative learning approach to the travelling salesman problem," *IEEE Transactions on Evolutionary Computation* **1**: 53-66.

M. Dorigo, V. Maniezzo, and A. Colorni (1996). The Ant System: Optimization by a Colony of Cooperating Agents. *IEEE Trans. Systems, Man, and Cybernetics – Part B* **26**: 29-41.

L.M. Gambardella, E.D. Taillard, and M. Dorigo (1999). Ant Colonies for the Quadratic Assignment Problem. *Journal of the Operational Research Society* **50**: 167-176.

M. Guntsch and M. Middendorf (2001). Pheromone Modification Strategies for Ant Algorithms applied to Dynamic TSP. To appear in Proceedings of EvoWorkshops 2001 - First European Workshop on Evolutionary Computation in Combinatorial Optimization (EvoCop2001), Lake Como, Italy. Springer LNCS Series.

D. Merkle, M. Middendorf, and H. Schmeck (2000). Ant Colony Optimization for Resource-Constrained Project Scheduling. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2000)*, 893-900. San Mateo, CA: Morgan Kaufmann.

R. Schoonderwoerd, O. Holland, J. Bruten, and L. Rothkrantz (1996). Ant-based Load Balancing in Telecommunications Networks. *Adaptive Behavior* **5**: 168-207.

T. Stützle and H. Hoos (1997). Improvements on the ant system: Introducing MAX(MIN) ant system. In G. D. Smith et al. (eds.), *Proc. of the International Conference on Artificial Neutral Networks and Genetic Algorithms*, 245-249. Springer-Verlag.

T. Stützle and H. Hoos (1999). *MAX-MIN* Ant System. *Future Generation Computer Systems* **16**: 889-914.

TSP-Library (2001). http://www.iwr.uni-heidelberg.de/iwr/comopt/software/TSPLIB95/.