# $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System and Local Search for Combinatorial Optimization Problems

THOMAS STÜTZLE AND HOLGER HOOS

TU Darmstadt, FB Informatik FG Intellektik
Alexanderstr. 10 64283 Darmstadt
Email(s): {stuetzle,hoos}@informatik.th-darmstadt.de

**Abstract**

*In this paper we present an extension of $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System applying it to Traveling Salesman Problems and Quadratic Assignment Problems. The extension involves the use of a modified choice rule and a hybrid scheme allowing ants to improve their solution by local search. The computational results show that this algorithm can be used to efficiently find near optimal solutions to hard combinatorial optimization problems and is one of the best methods for the solution of structured quadratic assignment problems.*

## 1 Introduction

Ant Colony Optimization (ACO) is a population based, cooperative search metaphor inspired by the foraging behavior of real ants. One of the basic ideas of ACO is to use the equivalent of the *pheromone trail* used by real ants as a medium for cooperation and communication among a colony of artificial ants. The seminal work on ACO is Ant System [8, 10] that was first proposed for solving the Traveling Salesman Problem (TSP). In Ant System, the ants are simple agents that are used to construct solutions, guided by the pheromone trail and heuristic information based on intercity distances. Since the work on Ant System, several extensions of the basic algorithm have been proposed, among those are Ant-Q [14], Ant Colony System [15], $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System [32, 34] and the rank-based version of Ant System [5]. All these extensions are in some sense "greedier" than Ant System, but differ significantly in main aspects of the search control.

$\mathcal{MAX}$–$\mathcal{MIN}$ Ant System [34, 32] yields a significantly better solution quality than Ant System for TSPs if only tour constructions are allowed. Additionally, the performance of ACO approaches can be considerably enhanced by adding a local search phase [32, 33, 9, 27], in which some ants are allowed to improve their solutions by a local search procedure. Thus, we obtain a hybrid algorithm consisting of a construction mechanism and a subsequent local search phase.

Ant System, and therefore also $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System ($\mathcal{MM}$AS), can be seen as an adaptive sampling algorithm — adaptive in the sense that it considers the experience gathered in earlier iterations of the algorithm. In case local search procedures are used, the information of earlier iterations is collected indirectly by the pheromone trails associated with *features* of local optimal solutions. For TSPs these features are arcs which are part of local optimal solutions. In case of Quadratic Assignment Problems (QAP) we use the location of specific items as a solution features. These features are combined in subsequent iterations by a global probabilistic construction mechanism biased by the pheromone trails, directing the sampling of new solutions towards promising regions of the search space. Important issues associated with adaptive sampling algorithms are which solutions (here ants representing the solutions) should be allowed to perform a local search (see [33] for a first discussion of this issue for the application of $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System to the TSP) and how powerful the local search method should be [19, 37].

In this paper we present an extension of $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System using a modified selection rule for the construction of solutions, the *pseudo-random-proportional rule*. This selection rule was first proposed for Ant-Q [14] and is also used in Ant Colony System [15], the successor of Ant-Q. The main effect of this selection rule is to yield a faster detection of high quality solutions by the algorithm. We apply the modified $\mathcal{MM}$AS to the Traveling Salesman Problem [24, 31], and the Quadratic Assignment Problem [11, 23] that models many practical problems arising in Location Science. $\mathcal{MM}$AS has been applied to TSP before in [34, 33], but the results presented here are significantly better, the application of $\mathcal{MM}$AS to the QAP is new. Both problems are $\mathcal{NP}$-hard [17], but whereas very large Euclidean TSPs can be solved to optimality, general instances of QAP with $n > 25$ are considered intractable. Apart from the practical interest of these problems, they also serve as a standard testbed for combinatorial optimization methods that try to find near-optimal solutions to $\mathcal{NP}$-hard problems. $\mathcal{MM}$AS is a *general purpose algorithm* that can be applied with minor modifications to combinatorial optimization problems yielding high quality solutions competitive with the best algorithms for both, TSPs and QAPs. The computational results presented on symmetric and asymmetric TSP instances from TSPLIB [30] *and* QAP instances of QAPLIB [6] support this claim.

The paper is organized as follows: To make the paper self contained, we briefly introduce $\mathcal{MM}$AS and its application to TSPs in Section 2. Subsequently, in Section 3, we describe how $\mathcal{MM}$AS is applied in combination with local search procedures to TSPs. In Section 4, we briefly introduce the QAP and present the modifications for $\mathcal{MM}$AS that have to be made for its application to the QAP. The computational results along with a discussion of specific issues concerning the selection of ants for local search are presented in Section 5, and we conclude after having discussed recent related work in Section 6.

## 2 $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System for TSPs

### 2.1 The Traveling Salesman Problem

In the Traveling Salesman Problem a given set of $n$ cities has to be traversed so that every city is visited exactly once and the tour ends in the initial city. The optimization goal is to find a shortest possible tour. More formally, a TSP can be represented by a complete weighted directed graph $G = (\mathcal{V}, \mathcal{A}, d)$ with $n$ nodes, $\mathcal{V}$ being the set of nodes, $\mathcal{A}$ being the set of arcs and $d : \mathcal{A} \mapsto \mathbb{N}$ a weight function associating a positive, integer weight $d_{ij}$ with every arc $(i, j)$ that corresponds to the distance between cities $i$ and $j$. The problem then is to find a shortest Hamiltonian cycle in the graph. For symmetric TSPs, the distances between nodes are independent of the direction of traversing the cities, i.e. $d_{ij} = d_{ji}$ for every pair of nodes. In the more general asymmetric TSP (ATSP) at least for one pair of nodes we have $d_{ij} \neq d_{ji}$. The TSP is a $\mathcal{NP}$-hard optimization problem [17] and is extensively studied in literature [24, 31].

### 2.2 $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System

$\mathcal{MAX}$–$\mathcal{MIN}$ Ant System shares some important features like the tour construction mechanism with Ant System, see also [32, 34] for a more detailed discussion. Ant System is a population based approach using a number $m$ ants. To solve TSPs, in Ant System and $\mathcal{MM}$AS a positive real value $\tau_{ij}$ is associated with each arc $(i, j)$, imitating the pheromone trails. Additionally the ants may also use heuristic information that in case of the TSP is some function of the distance between cities. Initially each ant is set on a randomly chosen city. Then in each step an ant selects from its current city $i$ one of the cities it still has not visited. In Ant System the ants make this selection according to the following probability distribution:

$$p_{ij} = \begin{cases} \frac{\tau_{ij}^{\alpha} \cdot \eta_{ij}^{\beta}}{\sum_{k \ not \ visited} \tau_{ik}^{\alpha} \cdot \eta_{ik}^{\beta}} & if \text{ city } j \text{ is not yet visited} \\ 0 & otherwise \end{cases} \tag{1.1}$$

where $\eta_{ij}$ is a local heuristic function which in Ant System and in $\mathcal{MM}$AS is defined for TSPs as $\eta_{ij} = 1/d_{ij}$. $\eta_{ij}$ is high if the distance between nodes $i$ and $j$ is small, thus shorter arcs are preferred by the heuristic function. The probability distribution for the selection of the next city is biased by parameters $\alpha$ and $\beta$ which determine the relative influence of the trail strength and the heuristic information. To keep track of the cities already visited, every ant maintains a list, that is used to store its partial tour is stored. After all ants have constructed a complete tour and have calculated the tour length $L_k$, the trails are updated. In $\mathcal{MM}$AS only the ant corresponding to the best tour found in an iteration is allowed to update the trails whereas in Ant System all ants lay down some pheromone. The chosen ant lays down a constant quantity $Q = 1$ of pheromone. Thus, the trail intensities are updated according to the formula

$$\tau_{ij}^{new} = \rho \cdot \tau_{ij}^{old} + \Delta\tau_{ij} \tag{1.2}$$

where $\rho$ with $0 < \rho < 1$ is the persistence of the trail, thus $1-\rho$ models the evaporation. The lower $\rho$, the faster information gathered in previous iterations is forgotten. The amount $\Delta\tau_{ij}$ is equal to $1/L_{\text{best}}$, $L_{\text{best}}$ being the tour length of the trail updating ant, if arc $(i, j)$ is used by the updating ant on its tour, otherwise zero. Thus, frequently used arcs receive a higher amount of pheromone and will be selected more often in future cycles of the algorithm. This can be interpreted as a learning of *good* arcs. The two basic steps *tour construction* according to (1.1) and *trail update* according to (1.2) are then simply repeated for a given number of iterations. We refer to the complete cycle of tour construction and trail update as one iteration, similar to the notion of generation in genetic algorithms. The most important part in Ant System is the treatment of the trail intensities. If some trails have very high intensities, the ants are very likely to choose the next arc among those. In practice, the long term effect of the trail intensities is to successively reduce the size of the effective search space by concentrating the search on a relatively small number of arcs.

The main differences between $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System and Ant System are that in $\mathcal{MM}$AS only one ant[1] is allowed to provide a feedback mechanism by updating

[1] Choosing only one ant for trail update was proposed also for Ant-Q and ACS. Other possibilities like choosing several of the best ants as proposed in [5] may also be used.

the trails and that the trails are limited to an interval between some maximum and minimum possible values $\tau_{\max}$ and $\tau_{\min}$. A minor difference is that in $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System the trails are initialized to their maximum value $\tau_{\max}$. The introduction of minimum and maximum trail limits is motivated by the following observation: Especially for larger TSP instances with $n \geq 50$, Ant System gives a rather poor solution quality, which can be enhanced by introducing some kind of elitism like allowing only the best ant to update the trails. Yet, then it may occur that the differences between the trail intensities on arcs get so high that the same tours are constructed again and again, i.e., premature stagnation occurs. This may lead even to the fact that many, possibly better tours cannot be constructed anymore. To alleviate this problem, in $\mathcal{MM}$AS explicit maximum and minimum trail strengths $\tau_{\min}$ and $\tau_{\max}$ on the arcs are used to counteract stagnation of the search. Using limits on the trails the difference among the trails on the arcs is limited and exploration of new, possibly better tours is increased. In [32, 34] we showed that this mechanism is effective and allows $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System to find significantly better tours than Ant System and an improved tour quality for longer runs compared to ACS.

### 2.3 Modifications to $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System

Compared to $\mathcal{MM}$AS as presented in Section 2 we considered the modification of the selection rule for the next city and adopted the *pseudo-random-proportional rule* used in Ant Colony System [15, 9]. Here with some probability $p_0$ the best possible choice is made, i.e, the ant chooses a city for which $\tau_{ij}^{\alpha} \cdot \eta_{ij}^{\beta}$ is maximal. Otherwise with probability $1 - p_0$ the next city is selected according to Equation 1.1. The parameter $p_0$ determines the degree of exploration an ant performs while constructing a tour. If $p_0 = 1$ the ant will always choose the best possible action and not do any exploration, for $p_0 = 0$ this selection rule corresponds to the selection rule used in Ant System. The main advantage of using the pseudo-random-proportional rule is that good solutions can be found considerably faster.

When using the pseudo-random-proportional rule in $\mathcal{MM}$AS, for higher values for $p_0$ tighter maximum and minimum trail strengths have to be chosen. We will give an intuitive explanation of this: For high values of $p_0$ almost always the best possible choice is made; in the cases where the choice is based on Equation 1.1 and the intervals for the trails are rather wide, it may happen that also in the probabilistic selection according to Equation 1.1 nearly always a city with highest $\tau_{ij}^{\alpha} \cdot \eta_{ij}^{\beta}$ will be chosen. Hence, nearly no exploration takes place. To limit the influence of the trails the trail limits have to be tighter, providing a better balance between exploitation and exploration. The probabilistic selection according to Equation 1.1 still is biased by the trails, giving a preference for arcs with higher trails. To determine the tightness of the trail limits, we add a parameter *max-factor* such that $\tau_{\max} = \tau_{\min} \cdot$ *max-factor*.

### 2.4 Adding local search to $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System

Constructive algorithms usually result in a poor solution quality compared to local search algorithms, see [31] for an extensive investigation of this issue for TSPs. Yet, it has been noted that repeating local searches from randomly generated initial solutions still lets a considerable gap to the optimal solution for most problems. Thus, to find solutions of very high quality some hybrid method is needed. An often used idea is to combine a probabilistic, adaptive construction heuristic with local search heuristics [4, 12], where the construction heuristic provides good starting points for the local search. $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System is such an adaptive construction heuristic. It can be used to identify promising regions of the search space using the pheromone trails as an adaptive memory of previously found solutions and to construct new promising starting points for the local search. Thus, we have such a hybrid algorithm, in which ants are used to construct initial solutions for a subsequent local search, see Figure 2.1 for an algorithmic frame. Another advantage of using $\mathcal{MM}$AS as a construction heuristic is that by generating good initial solutions, the subsequent local search needs much less iterations to reach a local optimum. Thus, for a given time limit more local searches can be done than by starting from randomly generated solutions.

```
procedure MAX–MIN Ant System with local search
(1) Initialize the pheromone trails and the parameters
(2) while ( termination condition not met) do
  for k := 1 to m do
    construct a solution for ant k
  end for;
  Select ants to improve their solution by local search
  Apply local search to selected ants
  Update the pheromone trail, ∀τ_ij τ_max ≥ τ_ij ≥ τ_min
(3) return best solution found
end MAX–MIN Ant System;
```

Figure 2.1: General algorithmic frame for $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System with local search procedures.

## 3 $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System applied to TSPs

While the principles of applying $\mathcal{MM}$AS to TSPs have been described before, in this section we present some important implementation aspects. To speed up the tour construction, we use candidate sets [31] of size 20 that contain for a given city $i$ a list of the 20 nearest neighbors. If possible, an ant chooses the next city among the members of the candidate set, otherwise it chooses the closest city. As a simple local search procedure for symmetric TSPs we implemented the so called `2-opt` algorithm and a version in which also moves corresponding to node insertions are made, denoted as `2-h-opt` [2, 31]. For ATSP the `2-opt` heuristic is not directly applicable as in a 2-opt move one part of the tour is inverted. So we use a particular 3-opt move that allows the re-insertion of arcs without reversing the direction of any partial tour, called `reduced 3-opt`. As we are not solving very large ATSP instances we use a straightforward implementation which is much slower than more optimized implementations. We use `reduced 3-opt` as a first-improvement heuristic, whereas `2-opt` and `2-h-opt` are run as best-improvement heuristics, i.e. select the best possible move in each iteration.

For larger TSPs a straightforward implementation of the local search procedures is computationally too expensive. As $\mathcal{MM}$AS, especially for symmetric TSPs, is run on larger problems we restricted the number of 2-opt moves to the 40 nearest neighbors

[21] and search for an improving move via a fixed radius near neighbor search, as described in [2]. Another improvement concerns the update of the trail matrix. In Ant System and $\mathcal{MM}$AS the trail matrix has $n^2$ entries, thus, the update according to Equation 1.2 would require time $O(n^2)$. To fasten the trail update, we only modify the trails leading from a node to members of its candidate set.

## 4 $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System for the Quadratic Assignment Problem

In the quadratic assignment problem (QAP) $n$ units have to be assigned to $n$ locations. The aim is to find an assignment that, knowing that a flow of value $a_{ij}$ has to go from unit $i$ to unit $j$ and that $b_{rs}$ is the distance between locations $r$ and $s$, minimizes the sum $S$ of the products flow×distance:

$$S = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} b_{\phi(i)\phi(j)}$$

where $A = (a_{ij})$ and $B = (b_{rs})$ are two $n \times n$ matrices, $\Phi(n)$ is the set of all permutations (assignments) of $\{1, \ldots, n\}$, and $\phi(i)$ gives the location of unit $i$ in the current solution $\phi \in \Phi(n)$.

When applying $\mathcal{MM}$AS to QAPs, the trails refer to specific locations for units, i.e. $\tau_{ij}$ gives the desirability of assigning unit $i$ to location $j$. For the construction of a solution essentially two steps have to be considered. First, we have to choose a unit that has to be assigned in the second step to one of the free[2] locations. In $\mathcal{MM}$AS the two steps are made as follows: We choose randomly some free unit that is placed on some free location according to the pseudo-random-proportional rule. If a location is chosen probabilistically this is done according to Equation 1.1 without using any heuristic information.

As a simple local search heuristic for QAP we implemented a greedy 2-opt heuristic whose neighborhood consists of all solutions that can be obtained by exchanging two units. In every step it explores the complete neighborhood $N(\phi)$ and chooses the move that gives the highest decrease of the objective function value, hence in every iteration $O(n^2)$ moves are explored. Note, that for the QAP the evaluation of the moves can use the information available from the preceding iteration. The first iteration of the local search has a computational cost of $O(n^3)$, whereas the following ones have a computational cost of $O(n^2)$, see [35] for details. As another possibility we consider applying short runs of Robust Tabu Search as proposed in [36].[3] The reason for doing so is that for most problem instances contained in QAPLIB, the best known solutions have been found by Tabu Search approaches. Thus, by adding Tabu Search a higher solution quality than by adding a simple greedy 2-opt procedure should be expected.

A first application of Ant System to the QAP is presented in [27]. Independently from our approach another new Ant Colony Optimization approach has been proposed for QAPs [16]. In Section 5.3 we will compare their results with ours showing that both approaches result in a comparable solution quality. Yet, the approach of [16]

[2] A unit is free if it is not yet assigned to a location and a location is free if no unit is assigned to it in the partial solution.

[3] A hybridization of genetic algorithms with short Tabu Searches was earlier proposed in [13].

differs strongly from the initial version of Ant Colony System and is fine tuned for the application to QAPs. On the other side, our approach is a straightforward application of $\mathcal{MM}$AS as also used for TSPs.

# 5 Experiments and Computational Results

## 5.1 Parameter Settings for $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System

For $\mathcal{MM}$AS various parameters have to be set, but good parameter settings can be found that give excellent results for both, the TSP and also the QAP. The parameter values for $\alpha$ is chosen $\alpha = 1.0$ throughout all experiments. The parameter values for $\rho$ are set as $\rho = 0.95$ in case of TSPs and $\rho = 0.8$ in case of the application to QAPs. Note, that the higher $\rho$, the slower is the learning of the solution features. The number of ants is chosen as 10 for the TSPs and 5 for the QAP. In the case of the application to TSPs we discuss the role of the heuristic information and show that with additional local search the heuristic information may be neglected, i.e., we may set $\beta = 0.0$ without significant solution quality degradation for most problem instances.

After each iteration the trails are modified according to Equation 1.2. Hence, also for the trail interval reasonable values around the expected amount of trail laid down by the best ant has to be found. To find a suitable trail interval as determined by $\tau_{\min}$ and $\tau_{\max}$ we calculate an estimate *estimate-opt* of the optimal solution by constructing a random solution and a subsequent run of a local search procedure and then chose $\tau_{\min} = 2/\textit{estimate-opt}$.[4] The upper trail limit is determined by the parameter *max-factor*, see Section 2.3. Throughout all experiments we set *max-factor* $= 5$. As we set a fixed interval for the trail limits, we modify the parameter value for $p_0$ according to the problem instance. For both TSP and QAP we chose $p_0$ in such a way that roughly 15 cities or units are chosen according to Equation 1.1, i.e., $p_0 = (n - 15)/n$. In some sense this is similar to the slight modifications made in iterated Lin-Kernighan [20] to the current solution: In iterated Lin-Kernighan the current solution is modified by a random 4-opt move that can not be reversed directly by the Lin Kernighan local search procedure. In $\mathcal{MM}$AS most of the time an ant makes the best possible choice and only sometimes a randomized choice leads to slightly modified solution with respect to the one that would be obtained by setting $p_0 = 1$.

For the selection of the ant to update the trails we depart from the initial approach to choose always the iteration best ant. It has been shown that, in combination with local search, for $\mathcal{MM}$AS it is advantageous to choose sometimes the best found solution for updating the trails [32]. On the other side, if only the best solution is chosen, the algorithm may get stuck early in suboptimal solutions. Thus, we decided for alternatingly choosing the iteration best and the overall best ant for the trail update.

## 5.2 $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System for Traveling Salesman Problems

### 5.2.a Adapting the number of ants that perform local search

When combining $\mathcal{MM}$AS with local search, one specific issue is which and how many ants should be allowed to perform local search. In [33] it was shown that there do not exist definite answers to this questions when using only the choice rule of

[4] Another possibility would be to use the best found solution to determine $\tau_{\min}$ and $\tau_{\max}$ dynamically during the run of the algorithm.

Equation 1.1. It was conjectured that the number of ants that is allowed to perform a local search should be increased during the run of the algorithm. Here we pursue such an approach. We start with a fixed number, here 10, ants and first allow only one ant, the ant that has constructed the best tour, to perform a local search. The number of ants that are allowed to improve their tours by local search is then successively increased by one after every `incr-ls` iterations. In Table 5.1, we compare different settings of `incr-ls` for a rather short run time on benchmark problems in Table 5.1. If `incr-ls` = 0, all ants are allowed to perform local search. As can be seen from Table 5.1, adapting the number of ants in the way outlined above actually improves the solution quality obtainable within a fixed time limit. The best results are obtained for values for `incr-ls` from 25 to 100. This result is a consequence of the fact that in early iterations the local search starts from rather bad initial solutions and take a longer time to find the local minimum. Additionally the number of iterations performed to learn good trail values is too limited. On the other side, the best ant may often have an initial solution that is very close the the previous best one and is more likely to again yield the previously found best solution. Thus, it is reasonable to increase the number of ants that perform a local search during the run of the algorithm.

For the experimental results presented in Section 5.2.b we chose `incr-ls` = 25 as this value seemed to yield reasonable performance for short runs.

Table 5.1: $\mathcal{MM}$AS for short run times, dynamic scheme of choosing ants for a local search, see text for details. Averages over 10 runs of $\mathcal{MM}$AS. We give the instance name, the number that appears is the problem size, and the time limit in seconds allowed on a Sun Sparc 5 Workstation. The results are indicated as percent deviation from the known optimal solution, best results are indicated in boldface.

| `incr-ls` | `d198` 10 sec | `lin318` 20 sec | `pcb442` 30 sec | `att532` 60 sec | `rat783` 180 sec |
|---|---|---|---|---|---|
| 0 | 0.48 | 1.99 | 2.24 | 2.13 | 2.11 |
| 1 | 0.43 | 1.73 | 1.86 | 1.89 | 2.06 |
| 3 | 0.43 | 1.46 | 1.71 | 1.67 | 1.85 |
| 5 | 0.45 | 1.24 | 1.54 | 1.77 | 1.51 |
| 10 | **0.31** | 1.02 | 1.36 | 1.26 | 1.60 |
| 25 | 0.34 | **0.64** | 1.22 | **1.12** | 1.51 |
| 50 | 0.40 | 0.78 | **1.06** | 1.16 | 1.40 |
| 100 | 0.45 | 0.76 | 1.21 | **1.12** | **1.27** |
| $\infty$ | 0.42 | 0.99 | 1.16 | 1.28 | 1.68 |

### 5.2.b Results for Longer Runs with Local Search

In this section we present results obtained with $\mathcal{MM}$AS and `2-h-opt` as a local search procedure in case of symmetric TSPs and `reduced 3-opt` in case of ATSPs. The solution quality of `2-h-opt` usually is slightly better than the one obtained with `2-opt`. We compare the results of $\mathcal{MM}$AS with those obtained by our own implementation of Ant Colony System (ACS) using the same local search procedure. For ACS we used the parameter settings given in [9].[5] The experiments were run using the measurement

[5] In [9] a `3-opt` local search algorithm is used, giving better results than `reduced 2-h-opt`. In the meanwhile we also implemented a fast 3-opt procedure using don't look bits [2]. First

protocol proposed for the First Competition for Evolutionary Computation.[6] In this protocol a fixed number of steps, each with a computational complexity of $O(n^2)$, is allowed, where $n$ is the problem size. Every tour construction and every improving step found by the local search procedure is counted as one step, in case of `reduced 3-opt` one complete neighborhood scan is counted as $n$ steps. For all algorithms we allowed $n \cdot 1000$ steps on symmetric TSPs and $2 \cdot n \cdot 2500$ steps on ATSPs. Thus, the comparison among the algorithms is fair, especially, as for all the same local search procedure was used. Additionally we give the computation times measured on a Ultra Sparc 2 Workstation. To investigate whether the heuristic information is still necessary if local search procedures are used, we run $\mathcal{MM}$AS once with ($\beta = 1.0$) and once without heuristic information. Both approaches are compared to an *iterated local search* approach. We apply the random 4-opt move used in the iterated Lin-Kernighan algorithm [20, 21] to modify the local optimal solution and use `2-h-opt` as the local search procedure. For the iterated local search approach we use the same local search as for $\mathcal{MM}$AS and allowed the same computation time as needed by $\mathcal{MM}$AS.

Table 5.2: Computational results for benchmark instances from TSPLIB comprising symmetric TSPs (upper half) and ATSPs (lower half). The known optimal solution is indicated below the instance name, the number in the identifier being the problem size, except **`kro124.atsp`** having 100 cities. We give the average percentage deviation over 25 runs from the optimal solution and the maximal computation times in seconds, best average results are indicated in boldface.

| Problem instance | iterated local search | ACS | $\mathcal{MM}$AS no heuristic | $\mathcal{MM}$AS + heuristic |
|---|---|---|---|---|
| `d198.tsp` 15780 | (0.0690%) 169 | (0.0196%) 170 | (0.0139%) 169 | **(0.00697%)** 162 |
| `lin318.tsp` 42029 | (0.211%) 443 | (0.198%) 506 | (0.124%) 443 | **(0.107%)** 429 |
| `pcb442.tsp` 50778 | (0.438%) 601 | (0.354%) 896 | (0.389%) 601 | **(0.353%)** 589 |
| `att532.tsp` 27686 | (0.316%) 1266 | (0.249%) 1477 | **(0.214%)** 1250 | (0.240%) 1266 |
| `rat783.tsp` 8806 | (0.660%) 2203 | (0.498%) 3088 | (0.386%) 2203 | **(0.280%)** 2165 |
| `ry48p.atsp` 14422 | (0.326%) 20 | (0.235%) 20 | **(0.0%)** 20 | **(0.0%)** 20 |
| `ft70.atsp` 38673 | (0.050%) 61 | **(0.0114%)** 60 | (0.0137%) 61 | (0.0212%) 61 |
| `kro124.atsp` 36230 | (0.822%) 190 | (0.756%) 187 | **(0.0%)** 190 | (0.0127%) 190 |
| `ftv170.atsp` 2755 | (0.396%) 966 | (0.428%) 982 | **(0.119%)** 966 | (0.189%) 966 |

The computational results show that both, $\mathcal{MM}$AS and ACS are competitive with the iterated local search approach yielding for nearly all instances a higher solution quality. Compared to ACS, $\mathcal{MM}$AS yields better average results for most problems. Furthermore, the computation times for $\mathcal{MM}$AS are, especially for the

results show a considerable quality improvement. For latest results, visit our WWW-page at http://aida.intellektik.informatik.tu-darmstadt.de/ tom/tsp.html.

[6] See also http://homepages.ulb.ac.be/~gseront/ICEO/2ndICEO.html.

larger instances, lower than for ACS. The reason for this difference is that ACS generates more solutions that are often already local optimal and can not be improved by the local search. Hence, ACS performs more iterations and constructs more tours than $\mathcal{MM}$AS, leading to higher run times. On the other side, solutions constructed by $\mathcal{MM}$AS are more diverse and offer more diversity to find better local minima by the local search.

The results show that for symmetric TSP instances the use of heuristic information gives, apart from instance `att532` slightly better results. On the contrary, for ATSPs the heuristic information degrades slightly the solution quality. It seems that the heuristic information in case of symmetric TSPs is of more use, whereas for ATSPs it may sometimes be misleading. We also verified that for ACS, neglecting the heuristic information results only in a small decline in solution quality. This is a noticeable result, as it was shown that for Ant System and ACS without local search, the heuristic information is very important [10, 9]. In case local search is allowed, the heuristic information is replaced by the information, which arcs are contained in the local optimal solutions, making the heuristic information of less use.

### 5.3 Quadratic Assignment Problems

In this section we present results obtained on some problem instances of QAPLIB [6]. As noted in [35], the problem instances may be classified in basically two types: Purely randomly generated problems without structure, for many of these instances the entries of the matrices $A$ and $B$ are uniformly distributed, and more structured real world problems. Additionally, in [35] randomly generated problem instances are proposed that exhibit a structure similar to the one observed in real world problems. It was also shown that the performance of heuristic methods depends strongly on the problem type. For real world problems in [35] it was shown that a Genetic Hybrid method (GH) [13] gave best results, whereas for randomly generated problems with uniform matrix entries robust Tabu Search [36] and Reactive Tabu Search [1] give the highest solution quality for the same computation times.

For the application of $\mathcal{MM}$AS to the QAP we especially investigate the issue which local search procedure should be added to $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System. We consider two possibilities: One is the application of a simple 2-opt greedy local search, while the second consists in running short Tabu Searches of length $4n$ as in GH [13]. We allow for a total of 250 applications of Tabu Search. For the greedy 2-opt local search a total of 1000 local search applications were allowed, resulting in computation times comparable (or less) to the 250 applications of Tabu Search. Here all ants perform a local search in every iteration. We compare our results to those obtained in [16] with HAS-QAP and their re-implementation of GH. Additionally, we present results of our own implementation of Robust Tabu Search, run for $1000 \cdot n$ iterations. For GH also 250 Tabu Searches of length $4n$ were allowed and in HAS-QAP 1000 applications of a first-improvement local search with a limit of 2 complete scans of the neighborhood are done. Note, that in case of the QAP the computational burden of the local search is so high, that the overhead caused by $\mathcal{MM}$AS is negligible. Hence, as [16] report that GH and HAS-QAP have roughly comparable run times, their results are comparable to ours also with respect to computation times.

The computational results presented were obtained with two slight modifications

Table 5.3: $\mathcal{MM}$AS compared to Genetic Hybrids, HAS-QAP and our implementation of Robust Tabu Search for unstructured (upper half) and structured (lower half) QAP instances. The number in the problem identifier indicates the dimension of the problem. Results for genetic hybrids and HAS-QAP are taken from [16]. The average solution quality over 10 runs for each algorithm is presented as percentage deviation from the best known solution values for the instances, see text for more details.

| Problem instance | Robust Tabu Search | GH + 250 TS | HAS-QAP | $\mathcal{MM}$AS + 250 TS | $\mathcal{MM}$AS + 2-opt |
|---|---|---|---|---|---|
| `sko72` | 0.146 | 0.143 | 0.277 | **0.103** | 0.301 |
| `sko81` | 0.135 | 0.136 | 0.144 | **0.077** | 0.153 |
| `sko90` | **0.128** | 0.196 | 0.231 | **0.128** | 0.366 |
| `tai50a` | 1.125 | **1.049** | 2.800 | 1.147 | 2.213 |
| `tai60a` | 1.203 | 1.159 | 3.070 | **0.980** | 2.211 |
| `tai80a` | 0.899 | 0.796 | 2.689 | **0.741** | 1.808 |
| `tai100a` | 0.894 | — | — | 0.684 | 1.66 |
| `bur26a-h` | 0.002 | 0.004 | **0.0** | **0.0** | **0.0** |
| `kra30a` | 0.268 | **0.134** | 0.629 | 0.135 | 0.821 |
| `kra30b` | 0.023 | 0.054 | 0.0711 | **0.015** | 0.121 |
| `tai35b` | 0.065 | 0.107 | 0.026 | **0.024** | 0.047 |
| `tai40b` | 0.531 | 0.2109 | **0.0** | 0.201 | **0.0** |
| `tai50b` | 0.345 | 0.2142 | 0.1916 | 0.125 | **0.001** |
| `tai60b` | 0.471 | 0.291 | 0.048 | 0.043 | **0.014** |
| `tai80b` | 1.031 | 0.827 | **0.667** | 0.725 | 0.741 |
| `tai100b` | 0.512 | — | — | 0.336 | **0.334** |

for $\mathcal{MM}$AS. We use the *smoothing of trails* approach [34]. This approach increases the trails proportionally to the difference between $\tau_{ij}$ and $\tau_{\max}$ if stagnation of the algorithm is detected. We noted that the best results were obtained with a re-initialization of the trails to $\tau_{\max}$. To achieve a stronger search diversification, after a new initialization of the trails we choose only the iteration best ant for trail update for a given number of iterations, here 4 iterations. After that we continue with alternatingly choosing the iteration best ant and the global best solution for the trail update.

The computational results show that both ant based approaches, $\mathcal{MM}$AS + 2-opt and HAS-QAP, have a comparable behavior.[7] $\mathcal{MM}$AS appears to be better on larger instances of `taixxa`, this may be due to the local search procedure chosen. Comparing the average solution quality of $\mathcal{MM}$AS + 2-opt and $\mathcal{MM}$AS + 250 Tabu Searches we observe that using Tabu Search as a local search operator pays of in case of unstructured instances, see upper part of Table 5.3, whereas for most of the structured QAP instances the simple 2-opt algorithm gives better results than the more sophisticated Tabu Search. Hence, for these instances it seems to be better to apply more often a rather simple local search procedure in order to identify promising regions of the search space. Comparing $\mathcal{MM}$AS + Tabu Search with the GH, the combination of Tabu Search with $\mathcal{MM}$AS gives better solution quality on most instances. Also note, that on nearly all instances the combination of $\mathcal{MM}$AS with TS gives better average results than Robust Tabu Search, indicating the usefulness of $\mathcal{MM}$AS as an diversification mechanism for Tabu Search. In resume, $\mathcal{MM}$AS seems to be one of the

[7] Note, that in [16] it was shown that Reactive Tabu Search [1] performs best for instances `taixxa`.

best approaches for the solution of structured real world QAPs.

## 6 Related Work

Many algorithms have been proposed and applied to both, the TSP and also the QAP. We can not discuss all of these approaches but just want to indicate some rather recent work on these two problems. For the TSP the most efficient approaches are the iterated Lin-Kernighan algorithm [20, 21] and the Genetic Local Search approach of [29]. Yet, both algorithms use the powerful Lin-Kernighan heuristic — it is regarded as the uncontested champion for TSPs — as the local search operator. It has been noted [37] that for the same time limit a much better solution quality can be obtained using the Lin-Kernighan local search heuristic than by applying, for example, 2-opt. We are convinced that these results also carry over to $\mathcal{MM}$AS and that the solution quality still can be improved by far. Ant System has been introduced using TSP as an example application. Recently, ACS another ant based approach to improve over Ant System was presented [15, 9]. With ACS very good results for symmetric TSP and especially for ATSPs with an extension of ACS by a 3-opt procedure are obtained.[8] For TSP another promising approach is the extension to ASPARAGOS presented in [18]. Like our approach she uses a very simple local search procedure for a spatially structured genetic algorithm. For ATSPs impressive results are reported solving all problems used here to optimality in all runs with computation times of only few seconds.

For QAP apart from HAS-QAP [16] two other very promising approaches have been proposed. In [29] a genetic local search algorithm for QAPs is presented, extending earlier work of the authors on TSPs. Their approach is in the same spirit as our application of $\mathcal{MM}$AS to QAPs as they adapt their successful genetic algorithm for the TSP to the QAP using only minor modifications. A new feature they present is a very fast local search procedures inspired by *don't look bits* [2]. Especially for larger problems they obtain very good results with comparably low computation times. Another new approach is based on Scatter Search and uses Tabu Searches as the local search operators [7]. They obtain very high quality solutions but apparently at the cost of rather high run times.

## 7 Conclusion

In this paper we have presented an enhancement of $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System using the pseudo random proportional rule first proposed for Ant-Q and ACS. We discussed a simple approach of how to adapt the number of ants that are allowed to perform a local search dynamically to increase the solution quality obtained for short runs of the algorithm. Furthermore, we showed that the heuristic information may be neglected when additional local search is performed giving only a minor degradation of solution quality. Compared to ACS, better solution quality could be obtained using $\mathcal{MM}$AS, while both approaches behave better than the iterated local search algorithm used for comparison. To show that $\mathcal{MM}$AS is a general purpose algorithm that can be easily extended to other problems we presented its application to the QAP. For the QAP, $\mathcal{MM}$AS is among the best approaches for the solution of structured QAP instances.

[8] More interesting work an Ant Colony Optimization may be found at the Ant Colony Home page http://iridia.ulb.ac.be/dorigo/ACO/ACO.html.

For the largest QAPLIB instance `tai256c` we could even find a new best solution with an objective function value of 44759294 in some longer runs of our algorithm [22]. For the other structured problems very good results were obtained, finding the best known solutions in nearly all runs for problem instances up to $n = 60$. Especially, for the QAP the local search method that is combined with $\mathcal{MM}$AS in a hybrid algorithm has a strong impact on the performance. For unstructured problem instances the use of short Tabu Search runs, instead of simple 2-opt, has shown to be particularly effective to obtain competitive results with other procedures, whereas for structured instances with 2-opt a better performance could be achieved. On the other side, for TSPs general knowledge seems to be that it is always best to choose Lin-Kernighan for the local search, if available [21]. Yet, if one cannot use this sophisticated local search procedure which is much more difficult to implement than e.g. `2-opt` or `3-opt`, $\mathcal{MM}$AS in its presented form is a viable alternative for the solution of TSPs. Generally, $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System seems to be a very promising tool providing a general adaptive framework for the solution of hard combinatorial optimization problems.

In future we will add a more powerful local search procedure like the Lin-Kernighan algorithm to $\mathcal{MM}$AS to improve the solution quality on TSPs. Also for attacking larger QAP efficiently, choosing faster local search procedures is an important issue. Another important issue is to identify the problem classes which can be efficiently solved using ACO approaches. For many problems it has been shown that their search space provides a structure in the way that locally optimal solutions are clustered in a small part of the search space [4]. It is on problems exhibiting such a kind of behavior, that current Ant Colony Optimization approaches work specifically well. Thus, we conjecture that current ACO approaches will proof particularly competitive on structured problem domains.

## Bibliography


[1] R. Battiti and G. Tecchiolli. The Reactive Tabu Search. *ORSA J. Comput.*, 6(2):126–140, 1994.

[2] J.J. Bentley. Fast Algorithms for Geometric Traveling Salesman Problems. *ORSA J. Comput.*, 4(4):387–411, 1992.

[3] H. Bersini, M. Dorigo, S. Langerman, G. Seront, and L. Gambardella. Results of the First International Contest on Evolutionary Optimisation. Technical Report IRIDIA/96-18, IRIDIA, Université Libre de Bruxelles, 1996.

[4] K.D. Boese, A.B. Kahng, and S. Muddu. A New Adaptive Multi-Start Technique for Combinatorial Global Optimization. *Oper. Res. Letters*, 16:101–113, 1994.

[5] B. Bullnheimer, R. Hartl, and C. Strauss. A New Rank Based Version of the Ant System: A Computational Study. POM Working Paper 3/97, University of Viena, 1997.

[6] R. E. Burkard, S. Karisch, and F. Rendl. QAPLIB - A Quadratic Assignment Problem Library. *European J. Oper. Res.*, 55:115–119, 1991. Updated Version: http://fmatbhp1.tu-graz.ac.at/

[7] V.-D. Cung, T. Mautor, P. Michelon, and A. Tavares. A Scatter Search based Approach for the Quadratic Assignment Problem. In *Proc. IEEE Int. Conf. Evol. Comp. (ICEC'97)*, pages 165–170, 1997.

[8] M. Dorigo. *Optimization, Learning, and Natural Algorithms.* PhD thesis, Politecnico di Milano, 1992.

[9] M. Dorigo and L.M. Gambardella. Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem. *IEEE Trans. Evolut. Comp.*, 1:53–66, 1997. 1(1), 1997.

[10] M. Dorigo, V. Maniezzo, and A. Colorni. The Ant System: Optimization by a Colony of Cooperating Agents. *IEEE Trans. Systems, Man, and Cybernetics – Part B*, 26(1):29–41, 1996.

[11] A.N. Elshafei. Hospital Layout as a Quadratic Assignment Problem. *Operations Research Quarterly*, 28:167–179, 1977.

[12] T.A. Feo and M.G.C. Resende. Greedy Randomized Adaptive Search Procedures. *J. Global Optim.*, 6:109–133, 1995.

[13] C. Fleurent and J.A. Ferland. Genetic Hybrids for the Quadratic Assignment Problem. In Panos M. Pardalos and Henry Wolkowicz, editors, *Quadratic Assignment and Related Problems*, pages 173–189. American Mathematical Society, 1994.

[14] L.M. Gambardella and M. Dorigo. Ant-Q: A Reinforcement Learning Approach to the Traveling Salesman Problem. In *Proc. 12th Int. Conf. Mach. Learning*, pages 252–260, 1995.

[15] L.M. Gambardella and M. Dorigo. Solving Symmetric and Asymmetric TSPs by Ant Colonies. In *Proc. IEEE Int. Conf. Evolut. Comp. (ICEC'96).* IEEE Press, 1996.

[16] L. Gambardella, E.D. Taillard and M. Dorigo. Ant Colonies for the QAP. Technical Report IDSIA-4-97, 1997.

[17] M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* Freeman, San Francisco, CA, 1979.

[18] M. Gorges-Schleuter. Asparagos96 and the Travelling Salesman Problem. In *Proc. IEEE Int. Conf. Evolut. Comp. (ICEC'97)*, pages 171–174, 1997.

[19] W.E. Hart. *Adaptive Global Optimization with Local Search.* PhD thesis, University of California, San Diego, 1994.

[20] David S. Johnson. Local Optimization and the Travelling Salesman Problem. In *Proc 17th Coll. Automata, Lang. and Prog.*, volume 443 of *LNCS*, pages 446–461. Springer, 1990.

[21] D.S. Johnson and L.A. McGeoch. The Travelling Salesman Problem: A Case Study in Local Optimization. to appear in E.H.L. Aarts and J.K. Lenstra, editors, Local Search in Combinatorial Optimization. John-Wiley and Sons, Ltd, 1997.

[22] Stefan Karisch, personal communication.

[23] T.C. Koopmans and M.J. Beckman. Assignment Problems and the Location of Economic Activities. *Econometrica*, 25:53–76, 1957.

[24] E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, and D.B. Shmoys. *The Travelling Salesman Problem.* John Wiley & Sons, 1985.

[25] Y. Li, P.M. Pardalos, and M.G.C. Resende. A Greedy Randomized Adaptive Search Procedure for the Quadratic Assignment Problem. In P.M. Pardalos and H. Wolkowicz, editors, *Quadratic assignment and related problems*, pages 237–261. American Mathematical Society, 1994.

[26] S. Lin and B.W. Kernighan. An Effective Heuristic Algorithm for the Travelling Salesman Problem. *Oper. Res.*, 21:498–516, 1973.

[27] V. Maniezzo, M. Dorigo, and A. Colorni. The Ant System Applied to the Quadratic Assignment Problem. Technical Report IRIDIA/94-28, Université Libre de Bruxelles, Belgium, 1994.

[28] P. Merz and B. Freisleben. A Genetic Local Search Approach to the Quadratic Assignment Problem. *Proc. Seventh Int. Conf. Genetic Algorithms (ICGA'97)*, pages 465–472, 1997.

[29] P. Merz and B. Freisleben. Genetic Local Search for the TSP: New Results. In *Proc. IEEE Int. Conf. Evolut. Comp. (ICEC'97)*, pages 159–164, 1997.

[30] G. Reinelt. TSPLIB — A Traveling Salesman Problem Library. *ORSA J. Comp.*, 3:376–384, 1991.

[31] G. Reinelt. *The Traveling Salesman: Computational Solutions for TSP Applications*, volume 840 of *LNCS*. Springer, 1994.

[32] T. Stützle and H. Hoos. Improving the Ant-System: A Detailed Report on the $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System. Technical Report AIDA–96–12, FG Intellektik, TH Darmstadt, August 1996.

[33] T. Stützle and H. Hoos. The $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System and Local Search for the Traveling Salesman Problem. In *Proc. IEEE Int. Conf. Evolut. Comp. (ICEC'97)*, pages 309–314, 1997.

[34] T. Stützle and H. Hoos. Improvements on the Ant System: Introducing $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System. In *Proc. Int. Conf. Artificial Neural Networks and Genetic Algorithms*, 1997. Springer Verlag, Wien.

[35] Éric D. Taillard. Comparison of Iterative Searches for the Quadratic Assignment Problem. *Location Science*, 3:87–105, 1995.

[36] Éric D. Taillard. Robust Taboo Search for the Quadratic Assignment Problem. *Parallel Computing*, 17:443–455, 1991.

[37] N.L.J. Ulder, E.H.L. Aarts, H-J. Bandelt, P.J.M. van Laarhoven, and E. Pesch. Genetic Local Search Algorithms for the Travelling Salesman Problem. In *Proc. 1st Int. Workshop Parallel Problem Solving from Nature (PPSN)*, number 496 in LNCS, pages 109–116. Springer, 1991.