A Pheromone Trails Model for MAX-MIN Ant System

Nikola Ivkovic[1], Marin Golub[2], Mirko Malekovic[1]

[1] Faculty of Organization and Informatics, University of Zagreb
{nikola.ivkovic, mirko.malekovic}@foi.hr
[2] Faculty of Electrical Engineering and Computing, University of Zagreb
marin.golub@fer.hr

Abstract. Pheromone trails are the main media for gathering collective knowledge about a problem, and have a central role in all ant colony optimization algorithms. Setting appropriate trail limits for the MAX-MIN ant system (MMAS) is important for good performance of the algorithm. We used rigorous analysis to develop expressions that model the influence of trail limits on MMAS behavior. Besides the general model, specific formulas for ATSP, TSP and QAP are presented. Assumptions on which our model is founded are experimentally validated. The paper gave general guidance for estimating the trail limits ratio established on exact analytical models. Experiments on tested problems showed a high level of agreement with predictions made by the presented model.

Keywords: pheromone trail, trail limit, MAX-MIN ant system, Ant colony optimization, Swarm intelligence

1 Introduction

Many practical computational problems have a too high a complexity to be solved with exact algorithms. One common approach to overcome high complexity is to use heuristic algorithms that cannot guarantee finding an optimal or sufficiently good solution, but in practice they often do.

Ant colony optimization (ACO) [1], [2] is a metaheuristic inspired by the foraging behavior of a colony of biological ants. Together with the ant colony system [3], the MAX-MIN ant system [4] is one of the most popular and successful [5] ACO algorithms. The MAX-MIN ant system has desirable characteristics like robustness, as it works with a population of solutions, natural parallelism and applicability to a wide range of problems [6], [7]. The disadvantage of MMAS, as with many nature inspired algorithms, is the usage of parameters that require appropriate tuning otherwise the algorithm may fail to find admissible solutions [8].

The main characteristic of the MMAS is the existence of trail limits, where all trails are maintained inside an interval bounded by some predefined lower and upper trail limits. The motivation for limiting pheromone trails is to avoid algorithm stagnation [4]. There are two possible approaches for setting the trails limits. The first is to set minimum and maximum limits after experimental measuring of the algorithm's performance. The second is to use an analytical expression to choose appropriate limits. Of course, analytically estimated limits can be fine-tuned with experimental measurements.

The paper is structured as follows. Section 2 gives a brief description of the MMAS and Section 3 gives a brief description of studied problems. In Section 4, a trail separation effect is described, modeled and experimentally evaluated. In Section 5, formulas for trail limits and probabilities are constructed, analyzed and compared. In Section 6, predictions of the presented model are experimentally evaluated. In Section 7 we summarize our findings and stress the importance of using exact formulas in trail limit estimation.

2 MAX-MIN Ant System Description

The MAX-MIN ant system is an extension of the ant system with improved performance for many optimization problems. It uses a colony of ants that construct a population of solutions, based on pheromone trails and heuristic values, both associated with building components. Heuristic values are used only with some optimization problems and for others it is always 1. The algorithm constructs a solution by adding solution components in the list of components that specify partial solutions, until an entire solution is constructed. The probability of selecting solution component $c(i)$ is given in (1). Index i denotes a solution construction step, $\tau_{c(i)}$ is the trail value associated with component $c(i)$, and $\eta_{c(i)}$ is the heuristic value associated with $c(i)$. In step $i,$ a component is selected from L_i, a set of components. Parameters α and β are used to maintain balance between trails and heuristic values.

$$p_{c(i)} = \frac{\tau_{c(i)}^{\alpha} \cdot \eta_{c(i)}^{\beta}}{\sum_{k \epsilon L_i} \tau_k^{\alpha} \cdot \eta_k^{\beta}} \tag{1}$$

After the population of solutions is constructed the best solution is found and trails are updated. The update process includes trail evaporation (2) for all trails, and trail reinforcement (3) for all components included in the iteration or global best solution.

$$\tau_{c(j)} = (1 - \rho) \cdot \tau_{c(j)} \tag{2}$$

$$\tau_{c(k)} = \tau_{c(k)} + \frac{1}{f(S^{best})} \tag{3}$$

Parameter ρ is the trail evaporation rate, and $f(S^{best})$ is the goodness of the iteration best or the global best solution. If a trail gets smaller than the minimum trail limit, the trail is set to the minimal value and if a trail gets bigger than the maximum trails limit, the trail is set to the maximal value. After the trails update, ants construct a new population of solutions and the process repeats. If algorithm stagnation is detected, the trails are reset to their initial values. If a fitness function $f(S^{opt})$ of the optimal solution can be estimated, then initial trails and maximum trail limit are set as:

$$\tau_0 = \tau_{max} = \frac{1}{\rho \cdot f(S^{opt})} . \qquad (4)$$

3 Optimization Problems

The Quadratic Assignment Problem (QAP) and the Travelling Salesman Problem (TSP) are well known optimization problems that arise in many practical applications. These are known to be NP-hard problems. Also, TSP and QAP cannot be approximated by polynomial approximation algorithms, unlike some other NP-hard problems, and are NPO-complete [9], [10]. Only brief problem descriptions are presented here. For a more comprehensive introduction, it is advisable to consult literature that is substantial for the selected problems.

For the TSP, there is a set of cities and all distances between cities are known. The problem is to find a tour with minimum total length. All cites must be visited exactly once and a traveler must end the tour by coming back to the starting city. Alternatively, cities are called nodes or vertices and direct links between two cities are called edges or arcs. If edges have directions, the problem is called asymmetrical. Otherwise, if all edges have the same distances in both directions, the problem is called symmetrical. Although TSP is a general term and includes asymmetrical and symmetrical variants, more often only symmetrical variants are studied and denoted as a TSP.

For the QAP, there is a set of facilities and an equally sized set of locations. Flow weights between facilities and distances between locations are known in advance. The problem is to allocate facilities to locations in a way such that the sum of the products of flow weights and distances is minimized.

All ATSP and TSP test problems used in empirical studies presented in this article are taken from *TSPLIB* library publicly accessible at *http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/* and *VLSI Data Set* accessible at *http://www.tsp.gatech.edu/vlsi/*. All QAP test problems used in the article are from *QAPLIB* library publicly accessible at *http://www.seas.upenn.edu/qaplib/*.

4 Trails Separation

Trails serve as a media for transferring collectively gathered knowledge about the problem into solution construction step. As the algorithm progresses, trails are changed to hopefully raise the probability for constructing an optimal or near optimal solution. The components that were reinforced in the previous steps are more likely to be chosen in the next solution construction. Because of this autocatalytic process, it is probable that the algorithm will converge toward one solution; hopefully an optimal or near optimal one. The trails of components that form this solution will probably be around maximal and all the others will be around minimal trail value.

The trail separation can be defined as a state in which the most trails are separated into two non-overlapping intervals I_{MIN} and I_{MAX}, separated by the interval I_{MID}. The interval I_{MIN} includes the lower and I_{MAX} includes the upper trail limit. If the set S_X denotes the set of components inside interval I_X, than the separation effect can be formally defined as a state that satisfies the inequalities (6) and (7) for some arbitrary small r, $0 \leq r < 1$.

$$S_{MID} = (S_{MIN} \cup S_{MAX})^C \tag{5}$$

$$|S_{MID}| \leq r \cdot |S_{MIN}| \tag{6}$$

$$|S_{MID}| \leq r \cdot |S_{MAX}| \tag{7}$$

4.1 Experimental Evaluations

To evaluate trail separation effect on the test problems, 100 runs of the algorithm were executed and distributions of trails were gathered. Table 1 lists selected problems that consist of 90 QAP, 18 ATSP and 59 TSP problems, giving 16700 runs in total. A problem name contains information about the problem size; but for ATSPs, this rule is sometimes violated, so the sizes for ATSPs are explicitly listed. The parameters were set to α=1, ρ=0.1 and the number of ants was set equal to a problem size n. The iteration best reinforcement strategy was used, except for bigger problems (more than 300 cities for ATSPs and more than 400 for TSPs) where the global best strategy was used. For ATSPs and TSPs β=4 and for QAPs β=0.

Table 1. List of selected problems used in experiments

<table>
<tr><th colspan="10">ATSP</th></tr>
<tr><th>Problem</th><th>Size</th><th>Problem</th><th>Size</th><th>Problem</th><th>Size</th><th>Problem</th><th>Size</th><th>Problem</th><th>Size</th></tr>
<tr><td>br17</td><td>17</td><td>p43</td><td>43</td><td>ft53</td><td>53</td><td>kro124p</td><td>100</td><td>rbg403</td><td>403</td></tr>
<tr><td>ftv33</td><td>34</td><td>ftv44</td><td>45</td><td>ftv55</td><td>56</td><td>ftv170</td><td>171</td><td>rbg443</td><td>443</td></tr>
<tr><td>ftv35</td><td>36</td><td>ftv47</td><td>48</td><td>ftv64</td><td>65</td><td>rbg323</td><td>323</td><td></td><td></td></tr>
<tr><td>ftv38</td><td>39</td><td>ry48p</td><td>48</td><td>ft70</td><td>70</td><td>rbg358</td><td>358</td><td></td><td></td></tr>
<tr><th colspan="10">TSP list</th></tr>
<tr><td colspan="10">gr17, gr21, gr24, fri26, gr48, hk48, eil51, berlin52, st70, eil76, pr76, rat99, kroA100, kroB100, kroC100, kroD100, kroE100, rd100, eil101, lin105, pr107, gr120, pr124, xqf131, pr136, pr144, ch150, kroA150, kroB150, pr152, u159, rat195, d198, kroA200, kroB200, tsp225, pr226, xqg237, gil262, pr264, pr299, lin318, linhp318, pma343, pka379, bcl380, pbk411, fl417, pbn423, pbm436, pr439, pcb442, d493, u574, pr654, xql662, u724, dkg813, pr1002</td></tr>
<tr><th colspan="10">QAP list</th></tr>
<tr><td colspan="10">tai10a, tai10b, had12, nug12, rou12, tai12a, had14, chr15a, esc16a, esc16b, esc16c, esc16e, esc16g, esc16h, esc16i, esc16j, had16, nug16a, nug16b, nug17, tai17a, had18, nug18, had20, lipa20a, lipa20b, nug20, rou20, tai20a, nug21, nug22, bur26a, nug27, nug28, kra30a, kra30b, lipa30a, lipa30b, nug30, tai30a, tai30b, tho30, esc32b, esc32c, esc32d, esc32h, kra32, tai35a, lipa40a, lipa40b, tai40a, tai40b, sko49, lipa50a, lipa50b, tai50a, tai50b, wil50, sko56, lipa60a, lipa60b, tai60a, tai60b, esc64a, sko64, tai64c, lipa70a, lipa70b, sko72, lipa80a, lipa80b, tai80a, tai80b, sko81, lipa90a, lipa90b, sko90, sko100a, sko100b, sko100c, sko100d, sko100e, sko100f, tai100a, tai100b, wil100, esc128, tai150b, tho150, tai256c</td></tr>
</table>

Possible trail values were divided into k intervals: $I_1 = I_{MAX} = [10^{-1/2}\tau_{\max}, \tau_{\max}]$, ... , $I_i = [10^{-i/2}\tau_{\max}, 10^{-(i-1)/2}\tau_{\max}\rangle$, ... , $I_{k-1} = [10^{-(k-1)/2}\tau_{\max}, 10^{-(k-2)/2}\tau_{\max}\rangle$, $I_k = I_{MIN} = [\tau_{\min}, 10^{-(k-1)/2}\tau_{\max}\rangle$. For ATSPs, there are n*(n-1) components from which n components are selected for one solution. This gives a predicted number of components $Count(I_{MAX}) = n$ for I_{MAX} interval and $Count(I_{MIN}) = n*(n-2)$ for I_{MIN}. For TSPs, the predicted number of components is $Count(I_{MAX}) = 2n$ and $Count(I_{MIN}) = n*(n-3)$, and for QAPs $Count(I_{MAX})=n$ and $Count(I_{MIN}) = n*(n-1)$.

Figure 1 shows a scatter plot of the experimentally measured $|S_{MAX}|$ marked with circles and $|S_{MIN}|$ marked with squares for ATSPs (a), TSPs (b) and QAPs (c) after 500 iterations. Respective predicted values are drawn with lines. In all cases it is noticeable that experimentally obtained data follows predicted values rather well. There are a few reasons why the measured values differ from the predicted values.

Firstly, the algorithm has not yet reached the trail separation phase. An example of this is the trail distribution for the TSP problem kroB100 shown on Fig. 1d. After 100 iterations, there is still a noticeable number of components with trails between I_{MIN} and I_{MAX} intervals. After 200 iterations, trails outside of I_{MIN} and I_{MAX} intervals are barely noticeable on the graph and values are very close to the predicted one. After 300 iterations, the situation is further improved by a barely noticeable increase of I_{MIN}. Greater sized problems, as expected, often need more time to converge to a solution and to achieve trail separation.

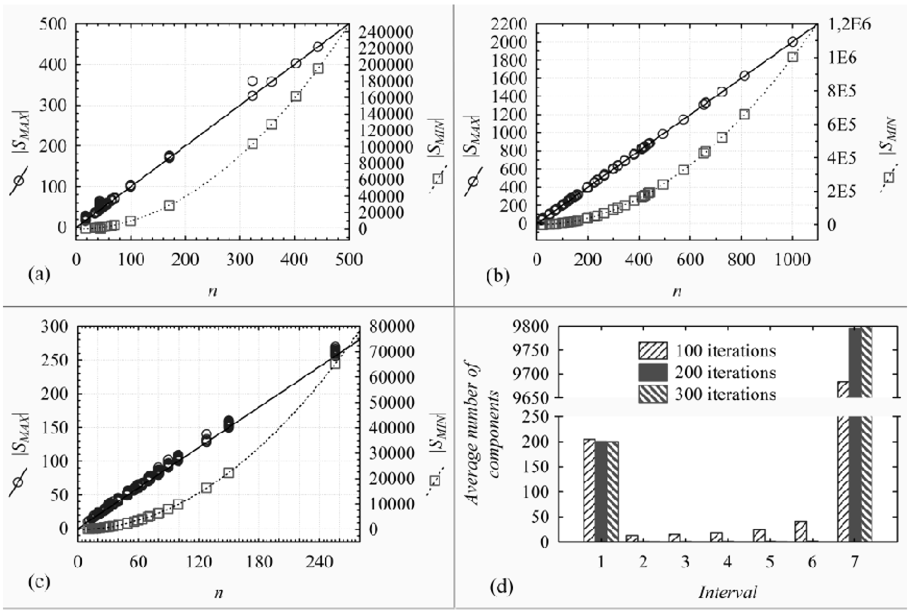

Fig. 1. First three graphs represent scatter plots for ATSP (a), TSP (b) and QAP (c) with related prediction curves. Bar graph (d) shows trails distribution for kroB100 instance of TSP problem.

Secondly, when the trail separation occurs at a certain iteration and the algorithm has constructed a near optimal solution, it is possible that in the next iteration, the algorithm would find a better solution that differs from the previous in a few components. These new components will gradually rise from I_{MIN} interval to I_{MAX} interval, and the components that are being replaced will go from I_{MAX} to I_{MIN} interval.

The third case is when two or more different solutions with some common components are constructed as iteration best solution in an alternating manner. Then, when the separation effect occurs, more components than predicted can be in the I_{MAX} interval. As the algorithm proceeds, one solution can manage to be reinforced more than others and take others out of the I_{MAX} interval. This can be observed on Fig. 1d where the average $|S_{MAX}|$ for kroB100 between 100 and 200 iterations falls closely to the predicted values.

Coefficients of the determination R^2, a measure of how well experimentally obtained values agree with predicted values, are calculated and listed in Table 2. The coefficient of determination for linear functions $Count(I_{MAX})$ is equal to the square of the correlation coefficient. Data in Table 2 shows that the predicted values agree with the experimentally obtained values very well, since for all cases R^2 is very close to 1 (exactly 1 would mean perfect matching).

Table 2. Coefficient of determination R^2

ATSP		TSP		QAP	
$\lvert S_{MAX}\rvert$	$\lvert S_{MIN}\rvert$	$\lvert S_{MAX}\rvert$	$\lvert S_{MIN}\rvert$	$\lvert I_{MAX}\rvert$	$\lvert I_{MIN}\rvert$
0.9993548	0.9999519	0.9990805	0.9996935	0.9993020	0.9999996

5 Trail Limits and Solution Construction Probabilities

For a fixed iteration of the MMAS algorithm it is possible to calculate the exact probability for one ant to construct predefined solution identified by an ordered list of components. Order in the list is not important for all optimization problems, but it is inherent for the MMAS solution construction process. The probability for constructing a solution as an ordered list is equal to the product of the probabilities of selecting individual components. To construct such expression it is necessary to know all the trails and heuristic values, and also the α and β parameters that influence the components selection.

5.1 Stützle – Hoos Expression for Trails Limits

Stützle and Hoos proposed in [4] an analytical expression (8) for calculating appropriate maximal and minimal trail values.

$$\tau_{min} = \tau_{max} \cdot \frac{1-\sqrt[n]{p_{best}}}{(avg-1)\cdot\sqrt[n]{p_{best}}} \; . \tag{8}$$

The average number of components that can be selected in construction steps is *avg*, the probability of constructing the best solution is p_{best}, and the number of components in the constructed solution is *n*. Heuristic values, as well as the *α* parameter, are neglected in expression (8). Mathews extended the Stützle-Hoos expression in [11] by setting the *α* parameter as an exponent on minimal and maximal trails. In [4] and [11] there is no differentiation between a solution described with a particular order of components, its probability is further on denoted as p_π, and a class of solutions, which are equivalent to a solution described with a particular order of components.

5.2 Exact Expressions for Trail Limits

In the presumption of complete trail separation and by neglecting heuristic values, if any are used, exact expressions that directly follow the process of constructing a solution can be obtained. All possible solution components that can be used for solution construction, in the solution construction step *i*, constitute the set L_i. Using the random-proportional rule (1), the probability of selecting the component with maximal trail value is:

$$p_{c(i)} = \frac{\tau_{\max}^{\alpha}}{\tau_{\max}^{\alpha}+\tau_{\min}^{\alpha}\cdot(|L_i|-1)} \tag{9}$$

$$\vartheta = \frac{\tau_{min}}{\tau_{max}} \tag{10}$$

$$p_{c(i)} = \frac{1}{1+\vartheta^{\alpha}\cdot(|L_i|-1)} \tag{11}$$

The probability for constructing the solution described with an ordered list of components constituting only of components associated with maximal trails is given in (12). The number of components contained in the solution is denoted as *n*.

$$p_\pi = \frac{1}{\prod_{i=1}^{n}(1+\vartheta^{\alpha}\cdot(|L_i|-1))} \tag{12}$$

5.2.1 Expressions for ATSP

Solution construction starts by randomly selecting the first node. The probability of selecting the node that is first in the predefined permutation of components is 1/*n*. Without losing generality, nodes can be labeled sequentially starting from 1 to *n*, so that edge (*i*, *i*+1) is the component of the predefined solution as shown in the Fig. 2. The case when the first node is selected is shown in Fig. 2a.

The next step is to select the second node, and by this, implicitly the first edge. For the first edge that can be selected, one component is associated with maximal trail value and $|L_1|$-1=*n*-2 are associated with minimum trail value. In general, if an ant stays on a predefined path and has already selected *i* nodes, than it can select from $|L_i|$-1=*n*-*i*-1 nodes as shown in Fig. 2b. Again, only one edge has the maximal value and all the others have minimal trail value.

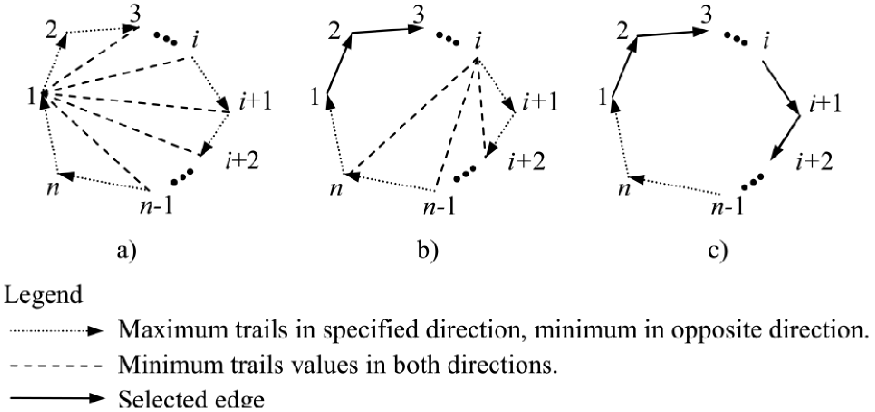
2
3
i
1
i+1
i+2
n
n-1
a)
b)
c)
Legend
Maximum trails in specified direction, minimum in opposite direction.
Minimum trails values in both directions.
Selected edge

Fig. 2. Solution construction phases for ATSP

At the end of solution construction, the ant is in the node labeled with *n*-1. As shown in Fig. 2c, there is no real choice for selection since $|L_{n-1}|$-1= 0 and only the component with maximum trail can be selected. This permutation of nodes identifies a solution, and although the edge (*n*, 1) is not explicitly selected, it is selected implicitly and the solution construction is over. Multiplying single probabilities for selecting nodes in a predefined order gives (13), the probability for constructing an ATSP solution with all the edges associated with the maximum trail value. One particular permutation represents *n* equivalent solutions that are trivially generated by rotating permutations. All cyclic permutations of the nodes have the same tour length. The probability for constructing any of *n* cyclic permutations that represent one equivalent solution is given by (14).

$$p_{\pi}^{ATSP}(n, \vartheta^{\alpha}) = \frac{1}{n} \cdot \left[\prod_{i=1}^{n-2}(1 + \vartheta^{\alpha} \cdot i)\right]^{-1} \qquad (13)$$

$$p^{ATSP}(n, \vartheta^{\alpha}) = n \cdot p_{\pi}^{ATSP}(n, \vartheta^{\alpha}) = \left[\prod_{i=1}^{n-2}(1 + \vartheta^{\alpha} \cdot i)\right]^{-1} \qquad (14)$$

5.2.2 Expressions for TSP

The solution construction for a TSP is very similar to that for an ATSP, but the probability of selecting the first component is different. Fig. 2 can be reused as visualization aid if arrows are neglected and all edges are bidirectional. When the first node is randomly selected, the same as with ATSP, there is a choice of selecting two edges that have maximal trails (edges: (1, 2) and (1, *n*)) and *n*-3 edges with minimal trails. After that, solution construction has defined constructing direction and probabilities of selecting following components are the same as with ATSP. The probability for constructing a particular solution coded directly by one permutation of nodes is given in (15). One permutation identifies *2n* equivalent TSP solutions. Along with choosing one of *n* nodes as starting node, we can choose 2 distinct directions. The expression for the class of equivalent solutions for TSP is given in (16).

$$p_\pi^{TSP}(n,\vartheta^\alpha) = \frac{1}{n} \cdot \frac{1}{2+\vartheta^\alpha \cdot (n-3)} \left(\prod_{i=1}^{n-3}(1+\vartheta^\alpha \cdot i)\right)^{-1} \quad (15)$$

$$p^{TSP}(n,\vartheta^\alpha) = 2n \cdot p_\pi^{TSP}(n,\vartheta^\alpha) = \frac{2 \cdot \left[\prod_{i=1}^{n-3}(1+\vartheta^\alpha \cdot i)\right]^{-1}}{2+\vartheta^\alpha \cdot (n-3)} \quad (16)$$

5.2.3 Expressions for QAP

To construct a QAP solution is to select couplings of elements from a facilities set and a locations set. In every construction step, an element is selected randomly with uniform distribution from the available elements from one set (set of available facilities or locations), and then its coupling element from the other set is selected using the random-proportional rule (1). The probability of selecting one particular element from the first set is $1/n$ in the first construction step, $1/(n\text{-}1)$ in the second construction step, and generally $1/(n\text{-}i\text{+}1)$ in the *i-th* construction step. In the first step $|L_1|=n$, and generally $|L_i|=n\text{-}i\text{+}1$ for the step i. The probability of constructing one particular component in a predefined permutation is given by (17). The solution is the set of couplings, not their permutations, so there are $n!$ equivalent solutions coded with one permutation. The expression for constructing a predefined solution that has all the trails with maximal value is given in (18).

$$p_\pi^{QAP}(n,\vartheta^\alpha) = \prod_{i=1}^{n}\left[\frac{1}{(n-i+1)} \cdot \frac{1}{(1+\vartheta^\alpha \cdot (n-i))}\right] \quad (17)$$

$$p^{QAP}(n,\vartheta^\alpha) = n! \cdot p_\pi^{QAP}(n,\vartheta^\alpha) = \prod_{i=1}^{n-1}\frac{1}{(1+\vartheta^\alpha \cdot i)} \quad (18)$$

5.3 Expressions Comparisons and Analysis

Expressions for ATSP, TSP and QAP are similar, so it is convenient to define a function (19) and rewrite expressions for ATSP (20), TSP (21) and QAP (22).

$$p^{LIN}(\mathrm{n},\vartheta^\alpha) = \frac{1}{\prod_{\mathrm{i}=1}^{\mathrm{n}-1}(1+\vartheta^\alpha \cdot \mathrm{i})} = \frac{\vartheta^{-\alpha\mathrm{n}}}{(\vartheta^{-\alpha})_\mathrm{n}} = \frac{\vartheta^{-\alpha\mathrm{n}} \cdot \Gamma(\vartheta^{-\alpha})}{\Gamma(\vartheta^{-\alpha}+\mathrm{n})} \quad (19)$$

$$p^{ATSP}(n,\vartheta^\alpha) = p^{LIN}(n-1,\vartheta^\alpha) \quad (20)$$

$$p^{STSP}(n,\vartheta^\alpha) = \frac{2}{(2+\vartheta^\alpha \cdot (n-3))} \cdot p^{LIN}(n-2,\vartheta^\alpha) \quad (21)$$

$$p^{QAP}(n,\vartheta^\alpha) = p^{LIN}(n,\vartheta^\alpha) \quad (22)$$

Figure 3a shows that the p^{LIN} function changes considerably with ϑ^α and n arguments, and only for a rather limiting region gives a probability that is not very close to either 0 or 1. If, instead of using $|L_i|$ in construction steps, average value is used as in Stützle-Hoos and the α parameter is taken into account, then the probability of constructing a solution is (23). An equivalent expression, in a different formulation, was previously developed by Matthews [11].

$$p^{AVG}(n,\vartheta^\alpha) = \left(1+\vartheta^\alpha \cdot (avg-1)\right)^{-n}. \quad (23)$$

5.3.1 Stützle – Hoos and Average Approximation Error

The error caused by using an average approximation instead of an exact formula is shown on Fig. 3b. Both axes use a logarithmic scale. The vertical axis represents the ratio of p^{AVG}/p^{LIN}. Although this graph was originally drawn for the QAP (note that $p^{QAP}=p^{LIN}$), the same graph gives the error ratios for the ATSP and the TSP because on a scale like this graph lines are the same.

The graph shows that except for very small ϑ^{α} values, introduced error is huge and this becomes more serious as n grows. In an extreme case, when $p^{AVG}\cdot avg^{n}<1$, ϑ^{α} becomes bigger than one, that is $\tau_{min} > \tau_{max}$. The Stützle-Hoos expression causes an error ratio with even higher magnitudes if $\alpha\neq1$, otherwise it is equal to those for p^{AVG}. The problems with the Stützle-Hoos expression are empirically detected by others. Wong and See noted in [12] that minimal trail limits are often set too low and that this affects the algorithms performance. These findings correspond with the error ratio shown in Fig. 3b, where it is noticeable that the Stützle-Hoos probability, as a special case of p^{AVG}, underestimates ϑ^{α} often by many orders of magnitude.

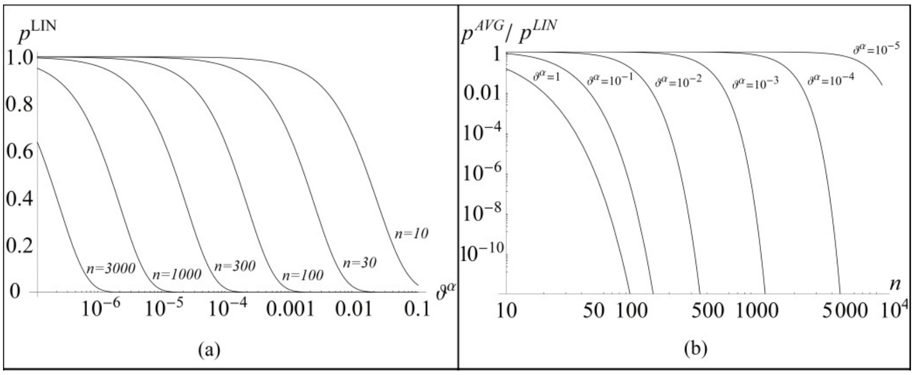

Fig. 3. Dependence of p^{LIN} function on n and ϑ^{α} arguments (a) and error ratio of average approximation (or Stützle – Hoos) and exact expressions (b).

6 Experimental model testing

To test the proposed model, we compare the performance of the algorithm for different ϑ ratios and different α parameters. The aim of these experiments is to see how well the proposed model explains differences in the performance of the algorithm. The results of the experiments are listed in the Table 3 as normalized mean solutions over 100 runs, for each parameter settings. For a problem instance and the α parameter (one row in the table), the minimal mean solution was selected and then all mean solutions were divided with this value. The parameters were set to α=1, β=4 (for the ATSP and the TSP), ρ=0.1, the number of ants was set equal to a problem size n, the iteration best strategy was used and the maximum number of iterations was set to 1000.

For minimal mean solutions, with normalized value 1, respective ϑ^{α} is provided in the brackets. The results show that optimal ϑ depends on problem size *n* and the *α* parameter as predicted by the model. In general, data shows that when *α* changes, so does the optimal ϑ, but the optimal ϑ^{α} stays approximately the same. The results in Table 3 also show that it is better to set ϑ^{α} too low than to set it too high (relatively close to 1), since generally the leftmost column has better mean solutions then the rightmost column. (This is of course not true for small problems that have optimal ϑ^{α} in rightmost columns. For small problems, too high ϑ^{α} values are not measured by these experiments.) These behaviors occur because too high ϑ^{α} prevents the algorithm from properly using trails to guide the algorithm towards the (near) optimal solution. When ϑ^{α} is set too low, at first the algorithm progresses normally, but after passing a certain minimum/maximum trail ratio, a search space is no longer explored. Instead, the algorithm constructs the same solutions all over again.

Table 3. Normalized mean solutions for MMAS

Problem	α	ϑ=1e-6	ϑ=1e-5	ϑ=1e-4	ϑ=1e-3	ϑ=1e-2	ϑ=1e-1
ftv35 (ATSP)	2/3	1.002462	1.001899	1.001543	**1(1.0e-2)**	1.000449	1.040856
	1	1.005004	1.004676	1.00383	1.002428	**1(1.0e-2)**	1.007097
	3/2	1.00962	1.009681	1.008015	1.007027	1.00303	**1(3.2e-2)**
ft70 (ATSP)	2/3	1.000522	1.000408	**1(2.2e-3)**	1.029509	1.108673	1.235636
	1	1.00392	1.003653	1.002417	**1(1.0e-3)**	1.009122	1.205311
	3/2	1.008973	1.008685	1.009779	1.007142	**1(1.0e-3)**	1.128595
kro124p (ATSP)	2/3	1.008777	1.004721	**1(2.2e-3)**	1.003099	1.06977	1.231649
	1	1.023612	1.022723	1.013809	1.004179	**1(1.0e-2)**	1.175048
	3/2	1.027704	1.028072	1.027028	1.021173	**1(1.0e-3)**	1.031121
eil51 (TSP)	2/3	1.002898	1.001613	1.000701	**1(1.0e-2)**	1.010144	1.065729
	1	1.005541	1.005308	1.004606	1.002128	**1(1.0e-2)**	1.014099
	3/2	1.008455	1.009225	1.006586	1.006446	1.001004	**1(3.2e-2)**
pr124 (TSP)	2/3	1.001093	1.000788	**1(2.2e-3)**	1.000431	1.012313	1.08072
	1	1.004805	1.004036	1.001035	**1(1.0e-3)**	1.000607	1.036979
	3/2	1.007934	1.008671	1.006843	1.003956	**1(1.0e-3)**	1.008198
ch150 (TSP)	2/3	1.005388	1.001836	**1(2.2e-3)**	1.013028	1.10239	1.205364
	1	1.014749	1.00848	1.00451	**1(1.0e-3)**	1.000149	1.140743
	3/2	1.025587	1.022809	1.022631	1.008084	**1(1.0e-3)**	1.034762
chr20 (QAP)	2/3	1.232387	1.114547	1.061805	**1(1.0e-2)**	1.012952	1.669481
	1	1.331188	1.310806	1.272797	1.102553	**1(1.0e-2)**	1.113854
	3/2	1.524881	1.447104	1.479266	1.411661	1.174907	**1(3.2e-2)**
lipa40a (QAP)	2/3	1.001597	1.000158	**1(2.2e-3)**	1.005087	1.011528	1.012395
	1	1.005012	1.003934	1.001738	**1(1.0e-3)**	1.005246	1.012295
	3/2	1.006576	1.006538	1.005973	1.003545	**1(1.0e-3)**	1.010981
sko100e (QAP)	2/3	**1(1.0e-4)**	1.003395	1.052245	1.085047	1.093219	1.094417
	1	1.022404	1.011157	**1(1.0e-4)**	1.018971	1.085858	1.094746
	3/2	1.028558	1.027869	1.021032	**1(3.2e-5)**	1.010537	1.085239

7 Conclusion

This paper explains the trail separation effect and shows with extensive experimental testing that in practice this effect occurs very closely as predicted with theoretical analysis. The paper stresses the importance of using exact models, instead of approximations, to avoid errors that can make calculated values completely unusable. Experimental results published by others detected that minimal trail limits are often set too low. This correlates with our findings that outside a limited domain, previous approximate expressions give significantly lower minimum trail limits. It is experimentally confirmed that choosing appropriate trail limits depends on the problem size and the α parameter as predicted by our model. If the trail limits ratio is set high (relatively close to 1) it will prevent the algorithm from learning and by this from successfully solving a problem, but setting it too low will cause algorithm stagnation and prevent it from further improving the solution. The presented model shows that appropriate trail limits, more precisely their ratio, vary considerably with a problem size and the α parameter.

References

1. Dorigo, M., Di Caro, G.: Ant colony optimization: A new meta-heuristic. In: Angeline, P.J. et al. (eds.), Proceedings of the Congress on Evolutionary Computation, vol. 2, pp. 1470–1477. IEEE Press (1999)
2. Dorigo, M., Di Caro, G., Gambardella, L. M.: Ant Algorithms for Discrete Optimization. Artificial Life, 5(2):137--172 (1999)
3. Dorigo, M., Gambardella, L.M.: Ant Colony System: A Cooperative Learning Approach to the Travelling Salesman Problem. IEEE Transactions on Evolutionary Computation, 1(1):53--66, 1997
4. Stützle, T., Hoos, H. H.: MAX–MIN Ant System. Future Generation Computer Systems, 16(9): 889 -- 914, 2000
5. Dorigo, M.: Ant colony optimization. Scholarpedia, 2(3):1461, revision #82084 (2007)
6. Dorigo, M., Stützle, T.: Ant Colony Optimization, p. 39, MIT Press (2004)
7. Dorigo, M., Birattari, M., Stützle, T.: Ant Colony Optimization-- Artificial Ants as a Computational Intelligence Technique. IEEE Computational Intelligence Magazine (2006)
8. Pellegrini, P., Birattari, M.: Implementation effort and performance: a comparison of custom and out-of-the-box metaheuristics on the vehicle routing problem with stochastic demand. In: Proceedings of the 2007 international conference on Engineering stochastic local search algorithms: designing, implementing and analyzing effective heuristics (2007)
9. Orponen, P., Mannila, H.: On approximation preserving reductions: Complete problems and robust measures. Technical Report C-1987-28, Department of Computer Science, University of Helsinki (1987)
10. Arkin, E. M., and Hassin, R.: Approximating the maximum quadratic assignment problem. In: SODA '00, Proceedings of the eleventh annual ACM-SIAM symposium on Discrete algorithms (2000)
11. Matthews, D. C.: Improved Lower Limits for Pheromone Trails in Ant Colony Optimization. In: Rudolph, G., Jansen, Th., Lucas, S.M., Poloni, C., Beume, N. (eds.) PPSN X. LNCS, vol. 5199, pp. 508--517. Springer, Heidelberg (2008)
12. Wong, K.Y. and See, P.C.: A new minimum pheromone threshold strategy (MPTS) for max-min ant system. Applied Soft Computing, 9(3): 882--888 (2009)