Title
Higher order pheromone models in ant colony optimisation

Author
Erin Montgomery

Bibliographic citation
Montgomery, Erin (2006). Higher order pheromone models in ant colony optimisation. University Of Tasmania. Conference contribution.
https://figshare.utas.edu.au/articles/conference_contribution/Higher_order_pheromone_models_in_ant_colony_

Is published in: 10.1007/11839088_42

Higher Order Pheromone Models in Ant Colony Optimisation

James Montgomery

Faculty of Information & Communication Technologies,
Swinburne University of Technology, Melbourne, Australia
`jmontgomery@ict.swin.edu.au`

Abstract. Ant colony optimisation is a constructive metaheuristic that successively builds solutions from problem-specific components. A parameterised model known as pheromone—an analogue of the trail pheromones used by real ants—is used to learn which components should be combined to produce good solutions. In the majority of the algorithm's applications a single parameter from the model is used to influence the selection of a single component to add to a solution. Such a model can be described as first order. Higher order models describe relationships between several components in a solution, and may arise either by contriving a model that describes subsets of components from a first order model or because the characteristics of solutions modelled naturally relate subsets of components. This paper introduces a simple framework to describe the application of higher order models as a tool to understanding common features of existing applications. The framework also serves as an introduction to those new to the use of such models. The utility of higher order models is discussed with reference to empirical results in the literature.

Keywords: Ant colony optimisation, pheromone model, model-based search.

1 Introduction

Ant colony optimisation (ACO) is a constructive metaheuristic that belongs to the model-based search (MBS) class of optimisation algorithms [15]. In an MBS algorithm, successive solutions are built using a parameterised probabilistic model, the parameters of which are revised over time using the solutions produced by the algorithm in order to direct its search towards promising areas of the solution space. The model used in ACO is referred to as a *pheromone model* in reference to the trail pheromones laid down by real ants to mark paths from their nest to a food source. In essence, the model describes relationships between components in the solution, such as one component succeeding another or whether a component is in a solution or not. The problem being solved thus partially dictates what can be modelled. Although the initial application of ACO to the well-known travelling salesman problem (TSP) used a model that very closely resembles the environment in which real ants move—there is a clear similarity between a Hamiltonian cycle in an edge-weighted graph and alternative routes between nest and food—as the range of problems to which it is applied has grown so too has the range of models [9].

During a single iteration of a typical ACO algorithm, each artificial ant constructs a solution by successively adding problem-specific solution components. The relative utility of alternative components is given by the parameters of the pheromone model. When the utility of adding a candidate component to a partial solution is described by a single parameter, the model is said to be first order [2, 3, 9]. A higher order model is one in which the utility of adding a candidate component to a partial solution is described by several parameters, requiring that the information be aggregated before judging that component's utility.

This paper introduces a simple framework for higher order pheromone models that serves both as a tool to understand existing applications of such models and as a guide for their future application. The utility of such models is also discussed with reference to their empirical performance. These topics are organised as follows. Section 2 formalises the discussion of pheromone models, introducing some necessary notation, before Section 3 describes the framework. The empirical performance of higher order models is compared to that of first order alternatives in Section 4. Section 5 provides some concluding remarks.

2 Pheromone Models

An ACO algorithm consists of a number of iterations of solution construction, within which each (artificial) ant builds its solution by successively selecting a solution component to add to its sequence. Solution components are typically selected probabilistically, biased by the parameters of the pheromone model, which provide an estimate of the utility of adding a solution component to an ant's partial solution. It should be noted that the term *solution component* is somewhat overloaded in the ACO literature, at times being used to refer to components of the model rather than the components from which solutions are built (sometimes referred to as *natural solution components* [2]). As both kinds of "component" are discussed in this paper, the term *solution characteristic* [9] is introduced to describe components of the model. A pheromone model thus consists of a set of solution characteristics, and is denoted by C. To each solution there corresponds a set of solution characteristics that is a proper subset of that defined by the model.

In essence, the model describes relationships between components or components and the solution. For instance, the model used with the TSP describes a relationship between exactly two components, the candidate being considered and the previous component added. The model commonly used with knapsack problems represents a relationship between a candidate and the solution as a whole, indicating the utility of including a component at all. The relationships described by a pheromone model may also make reference to aspects of solutions other than the components from which they are built. For instance, some models used in the literature represent the absolute position of components in a solution, while others represent the assignment of items from one set of entities to another (with the solution components being drawn from only one of the two sets). When a model represents a relationship between a candidate solution component and at most one other component in the partial solution, as in these examples, that model is said to be first order. In other words, the utility of adding a candidate component is described by a single pheromone value (a non-zero, real-valued number denoted by τ). When the relationships modelled relate to multiple solution components, the pheromone model is higher order.

Higher order pheromone models implicitly define two sets of solution characteristics: one set relates to decisions to include individual solution components based on the current state of a partial solution, while the other set relates to higher order decisions, i.e., whether a single solution should exhibit two or more particular solution characteristics at the same time. Given a first order model C, an n^{th}-order model may be a set of n-tuples of solution characteristics, denoted by C^n. Higher order models may be contrived by modelling combinations of n solution characteristics from some first order model or, when the solution characteristics modelled relate many parts of the solution to each other, will form naturally as a consequence of having to combine information from each relationship. In the latter case there is typically no related first order model.

3 Using Higher Order Models

The two main issues that arise when using a higher order model are how the higher order information is used, and the trade-off between the computational overhead associated with the larger pheromone model versus the benefits of using the extra information it provides. The former is discussed in this section, while the latter is discussed in Section 4 below.

Although higher order pheromone models have been used in a number of ACO algorithms, there is no single approach to their use. Nevertheless, there are common features of each of the approaches currently described in the literature that allow a general framework to be proposed.

When using a first order pheromone model, each constructive step is a competition between individual solution characteristics (and hence between the solution components they implicitly represent). When using a higher order model, the pheromone associated with adding a particular solution component is an aggregate of a number of pheromone values. Given a first order solution characteristic $c \in C$, denote the set of all other single solution components or characteristics to which c is related, and which consequently should be used to inform the decision to include c in the current partial solution, by C_c. In general, for an n^{th} order pheromone model it is important to know to which tuples of $(n-1)$ solution components or characteristics a first order characteristic c is related, denoted C_c^{n-1}. Given an appropriate definition for C_c^{n-1}, a suitable aggregation function must also be defined, as well as an alternative when $C_c^{n-1} = \varnothing$.

Assuming that a solution characteristic c corresponds to a single constructive step (e.g., the addition of a single solution component or a single assignment), and denoting the pheromone associated with adding c to the partial solution s^p using an n^{th} order pheromone model by $\tau(s^p, c, n)$, a generic function for $\tau(s^p, c, n)$ where $n \geq 2$ is given by

$$\tau(s^p, c, n) = \begin{cases} f(s^p, c, \tau_n) & \text{if } \exists\, \tau_n \text{ and } |C_c^{n-1}| > 0 \\ \tau(s^p, c, n-1) & \text{otherwise} \end{cases} \tag{1}$$

where $\tau_n : C \times C^{n-1} \to \mathbb{R}^+$ is a function from collections of n solution characteristics to pheromone values, and $f(s, c, \tau_n)$ is an aggregation function over the pheromone values associated between c and the elements of C_c^{n-1}, which is discussed in more detail below. Note that the equation is recursive; if C_c^{n-1} is empty or τ_n does not exist then a lower order pheromone model is sought. To ensure that the recursion defined by Equation 1 is well-founded, τ_1 must be defined, either to be a constant value or a separate first order pheromone model. In pheromone models where the elements of C_c^{n-1} are taken from s^p, early in solution construction s^p contains few solution characteristics and it is likely that $C_c^{n-1} = \varnothing$, and hence a lower order pheromone model such as τ_1 must be used until the n^{th} order model τ_n can be used. Conceivably, for $n > 1$, if an n^{th} order model is used, $n - 1$ other pheromone models may also be employed to deal with the first $n - 1$ steps of solution construction. In practice, most higher order pheromone models are only second order, so at most two pheromone models may be required.

Instances of higher order models are specified by providing definitions for the three components of this general framework, C_c, f and τ_1. The definition of C_c is highly problem specific and closely tied to the way solutions are constructed.

A number of options are available for the aggregation function f, four of which are to take the minimum, maximum, mean or sum of the different pheromone values involved. These four alternative definitions of f are denoted by $min(\tau_n)$, $max(\tau_n)$, $mean(\tau_n)$ and $sum(\tau_n)$ respectively.

The definition of C_c^{n-1} also determines which pheromone values from τ_n are updated by a solution s. For instance, using a second order pheromone model that represents the learned utility of having pairs of solution characteristics $(c_i, c_j) \in C^2$ copresent in a solution, pheromone is updated for all pairs (c_i, c_j) such that $c_i, c_j \in s, c_i \neq c_j$. Alternatively, when using a second order model that represents the utility of placing a solution component before certain other solution components, the value of C_c^{n-1} *when the solution was constructed* must be used to identify which pheromone values to update.

3.1 Defining C_c, f and τ_1 for a problem

The definition of C_c is problem specific and typically apparent from the higher order solution characteristics being modelled. For instance, if a second order model is used to learn whether pairs of components should be part of the same solution, then intuitively C_c should contain those components already in the partial solution. Alternatively, given a different second order pheromone model that models pairs of components that should not be part of the same solution

Table 1. Sample of customisations of Equation 1. τ_{max} is an upper bound on pheromone values imposed in the $\mathcal{MAX} - \mathcal{MIN}$ Ant System algorithm [14]

C_c	f	τ_1	Example(s)
$\in \mathfrak{s}^p$	*sum*	unknown	[10, 13]
		equivalent to 1	[7]
		1^{st} order model	[1]
	mean	1	[5, 11]
		1	[6]
		1	[8]
		1^{st} order model	[8]
$\notin \mathfrak{s}^p$	*min*	τ_{max}	[12]
		∞	[3]

(or where there is a relationship based on the relative order of the pairs of components as in [3]), and faced with a first order decision about whether to include a candidate characteristic, intuitively C_c should contain only those components that have not yet been added to the partial solution.

The definitions of f and τ_1 can be somewhat separate from the solution characteristics modelled and so may appear to be arbitrary choices. Nevertheless, a number of observations may be made concerning existing applications of higher order models. Table 1 categorises the usage of higher order pheromones found in the literature, based on whether the elements of C_c come from the partial solution or its complement, the aggregation function used and definition of τ_1. Full details of the retrospective application of the framework to the works cited are given in [8].

With regards to the aggregation function f, all the examples in Table 1 use *min*, *mean* or *sum*, while none uses *max*. The use of the *min* function can be characterised as a cautious approach—any single low pheromone value can in effect veto the first order decision being considered. Conversely, *max* allows any single high pheromone value to make the decision more likely. The functions *sum* and *mean* allow each higher order solution characteristic's pheromone value to influence the first order decision, with the choice of whether to use *sum* or *mean* dependent on the number of higher order solution characteristics available for each candidate first order characteristic (or component). In the examples cited, *sum* is used in all cases where $|C_c| = |C_{c'}| \; \forall \; c \neq c'$ for a fixed partial solution size, while *mean* is used in those cases where this is not the case (or where the magnitude of observed pheromone values must be kept constant).

Notably, *min* is used only in those cases where the elements of C_c are not present in the partial solution. Blum and Sampels [3] describe an ACO algorithm for shop scheduling problems in which each solution characteristic indicates the relative order of operations that must be processed on the same machine. In this application, the rationale for using *min* is that if any pheromone value is low then there must exist at least one related operation that should be scheduled before the one being considered. The *min* function is also used in an ACO algorithm for a university timetabling problem developed by Socha, Knowles and Sampels [12], where higher order pheromone values are used to learn which events should *not* be placed in the same timeslot. Conceivably, taking the minimum value between the current event and those already assigned a timeslot might produce similar results to considering unscheduled events. However, this approach may allow an event to be placed in a timeslot that suits another unscheduled event better and which may increase solution cost if that other event were later placed in the same timeslot. Consequently, taking the minimum value may avert such undesirable actions. Thus, in both examples, using *min* in relation to those solution components or characteristics that have yet to be added to a partial solution appears to avoid making decisions that may force the algorithm to make an inferior decision later in solution construction. In contrast, the use of *sum* and *mean* with pheromone values associated with solution components or characteristics already in a partial solution appears to promote the selection of a solution component that is well suited to the existing partial solution.

In those examples where τ_1 is clearly defined and the elements of C_c are taken from the partial solution, the first solution characteristic is chosen either randomly or using a first order pheromone model when used in conjunction with *sum*, while it is assigned a constant value when used with *mean*. Where the elements C_c do not come from the partial solution, τ_1 is set to either a high value (τ_{max}) or a candidate component is chosen as if it had a high value (such as ∞).

4 Utility of Higher Order Pheromones

When implemented, higher order pheromone models require greater computational resources than their first order counterparts. While storage overhead is typically not problematic—most higher order models are second order, representing a squaring in size—higher order models necessarily take longer to process as multiple pheromone values must be considered for each solution characteristic. This increased computational overhead must be weighed against any potential improvements to the quality of solutions produced by the algorithm, as the following examples show.

A comparative study of first and second order pheromone models for the k-cardinality tree problem found that, given the same amount of execution time, the latter produces fewer solutions and thus the algorithm makes less progress towards good solutions [1]. It was concluded that the first order model is consequently a better choice for this problem.

Roli, Blum and Dorigo [10] compared the performance of an ACO algorithm for constraint satisfaction using three alternative pheromone models, including a first order pheromone model that represents which assignments should be made and a second order pheromone modelling which pairs of assignments should be made. Both models performed similarly well. However, again due to the increased computational overhead for the second order model, the first order model is promoted as the best-suited to that problem.

Montgomery [8] compared first and second order pheromone models for the knapsack problem, also finding that the two gave equivalent performance in terms of solution quality, with the second order model increasing the required computation time for an equivalent number of solutions produced.

Montgomery [8] also compared a first and two second order pheromone models for a car sequencing problem in which different car models must be assigned positions in a production sequence such that the separation penalty between cars of the same model is minimised (i.e., it is desirable keep cars of the same model apart). The penalty varies between models. The first order pheromone model represents the assignment of a car model to a sequence position. One of the higher order models represents pairs of sequence positions assigned the same car model, similar to the model used by Costa and Hertz [5] for the graph colouring problem in which nodes in a partially connected graph are partitioned into colour groups. The other higher order model represents pairs of sequence positions assigned the same car model plus which model is assigned. The study found that the first order model and the second order model that includes the actual car model assigned both outperformed the model inspired by that used by Costa and Hertz. This finding is commensurate with suggestions by Montgomery, Randall and Hendtlass [9] regarding appropriate pheromone models. However, the first order model performed best overall.

These four studies would appear to suggest that higher order models have little or no utility. However, there are some combinatorial problems, such as the maximum-clique [7] and graph colouring [5] problems, for which first order models may not be appropriate—both of these problems involve the assignment of components to groups. Indeed, potential problems have been identified with first order models for the graph colouring problem [9]. Furthermore, the best performing model for shop scheduling problems is the second order model developed by Blum and Sampels [3, 4], a key feature of which is that it is not derived from a first order model. Taken together, these results suggest that if a simpler pheromone model is "appropriate" for a given problem, it is unnecessary to use a higher order (typically, a second order) model. Montgomery, Randall and Hendtlass [9] put forward a number of qualities of a model that make it "appropriate" for a particular problem, the chief one being that it represents characteristics of solutions that directly impact on solution cost. Using this guiding criterion, the use of higher order pheromone models is implicated in a number of problem domains.

5 Conclusions

The majority of pheromone models used in ACO algorithms are first order, with a single parameter of the model representing the learned utility of adding a single solution component to a partial solution. A number of higher order models have also been developed which give more detailed information about the utility of adding a single component. This paper has introduced a simple framework for describing higher order pheromone models, which serves as a tool to understand common features of existing applications and may assist in the future development of new higher order models for other problems.

A review of studies that compare first and second order models suggests that higher order models will often give equivalent performance to first order counterparts, but at the expense of greater computation times. However, there are some problems where the use of higher order models appears necessary. Therefore, if a problem appears to require the use of a higher order model then a first order model should also be developed and its performance examined.

References

[1] C. Blum. Ant colony optimization for the edge-weighted k-cardinality tree problem. In W. Langdon, editor, *GECCO 2002*, pages 27–34, New York, USA, 2002. Morgan Kaufmann Publishers.

[2] C. Blum. *Theoretical and practical aspects of ant colony optimization*, volume 282 of *Dissertations in Artificial Intelligence*. IOS Press, Nieuwe Hemweg, The Netherlands, 2004.

[3] C. Blum and M. Sampels. Ant colony optimization for FOP shop scheduling: A case study on different pheromone representations. In *CEC 2002*, pages 1558–1563, 2002.

[4] C. Blum and M. Sampels. An ant colony optimization algorithm for shop scheduling problems. *J. Math. Model. Algorithms*, 3(3):285–308, 2004.

[5] D. Costa and A. Hertz. Ants can colour graphs. *J. Oper. Res. Soc.*, 48:295–305, 1997.

[6] F. Ducatelle and J. Levine. Ant colony optimisation and local search for bin packing and cutting stock problems. *J. Oper. Res. Soc.*, 55(7):705–716, 2004.

[7] S. Fenet and C. Solnon. Searching for maximum cliques with ant colony optimization. In G. Raidl et al., editors, *EvoWorkshops 2003*, volume 2611 of *LNCS*, pages 236–245, Essex, UK, 2003. Springer-Verlag.

[8] E. J. Montgomery. *Solution biases and pheromone representation selection in ant colony optimisation*. PhD thesis, Bond University, 2005.

[9] J. Montgomery, M. Randall, and T. Hendtlass. Automated selection of appropriate pheromone representations in ant colony optimisation. *Artificial Life*, 11(3):269–291, 2005.

[10] A. Roli, C. Blum, and M. Dorigo. ACO for maximal constraint satisfaction problems. In *MIC'2001*, pages 187–192, Porto, Portugal, 2001.

[11] L. Schoofs and B. Naudts. Solving CSPs with ant colonies. In *Abstract Proceedings of ANTS2000*, Brussels, Belgium, 2000.

[12] K. Socha, J. Knowles, and M. Sampels. A $\mathcal{MAX}-\mathcal{MIN}$ ant system for the university course timetabling problem. In M. Dorigo et al., editors, *ANTS 2002*, volume 2463 of *LNCS*, pages 1–13, Brussels, Belgium, 2002. Springer-Verlag.

[13] C. Solnon. Ants can solve constraint satisfaction problems. *IEEE Trans. Evol. Comput.*, 6(4), 2002.

[14] T. Stützle and H. Hoos. $\mathcal{MAX}-\mathcal{MIN}$ ant system. *Future Gen. Comp. Sys.*, 16:889–914, 2000.

[15] M. Zlochin and M. Dorigo. Model-based search for combinatorial optimization: A comparative study. In *PPSN2002*, pages 651–662, 2002.