# The Influence of Memory in a Threshold Model for Distributed Task Assignment

Harry Goldingay and Jort van Mourik,
Information Engineering
Aston University
Aston Triangle, Birmingham B4 7ET, UK

## Abstract

*A nature inspired decentralised multi-agent algorithm is proposed to solve a problem of distributed task selection in which cities produce and store batches of different mail types. Agents must collect and process the mail batches, without a priori knowledge of the available mail at the cities or inter-agent communication. In order to process a different mail type than the previous one, agents must undergo a change-over during which it remains inactive.*

*We propose a threshold based algorithm in order to maximise the overall efficiency (the average amount of mail collected). We show that memory, i.e. the possibility for agents to develop preferences for certain cities, not only leads to emergent cooperation between agents, but also to a significant increase in efficiency (above the theoretical upper limit for any memoryless algorithm), and we systematically investigate the influence of the various model parameters. Finally, we demonstrate the flexibility of the algorithm to changes in circumstances, and its excellent scalability.*

## 1. Introduction

Distributed systems are an active and important field with applications ranging from distributed heterogeneous computing systems [9] to mobile sensor networks [11]. In such systems, resource and task assignment are important in order to provide performance efficiently. Many current solutions to these problems focus on a centralised approach. However, there is growing feeling that the size of some distributed systems is growing to the point where decentralised approaches may become a necessity [6]. Note that a decentralised solution to a problem in principle can not be better than the best centralised solution. At minimum a central controller could issue instructions causing agents to act as they would have under the autonomous rules of the best decentralised solution. In fact, access to global information and the ability to coordinate agents should allow better performance than any collection of individuals. In practice, however, limitations on resources such as computational power and/or communication costs [10] mean that centralised solutions are not efficient in practice. This is particularly true for large systems as the calculation time of an optimal allocation of tasks becomes a major limitation [6]. Large systems also decrease the effectiveness of global inter-agent communication. Shehory et al. [15] point out that if $n$ agents are communicating with each other, this involves a total of O($n^2$) communications, potentially "overwhelming" the communication network.

Rana et al. [14] suggest that many practical applications will require large numbers of agents, and because of this the poor scalability of centralised systems rules them out as solutions to these problems. The challenge, therefore, is to design decentralised solutions (involving only simple local interactions) with a performance close to the best centralised solution.

The parallels between multi-agent systems and social insects are well established and have inspired many algorithms [2]. When taken at colony level, social insects can be seen as examples of a self-organising multi-agent system and, within these systems, behaviour has been observed which fits well with the desirable properties described above. Of particular interest is the fact that, within a colony, the fraction of individuals engaged in particular tasks changes in response to demand, which Beshers et al. [1] describe as "one of the most prominent features of social insect colony behaviour". Additionally, Grassé [8] explained that this behaviour emerges due to individuals acting to modify the local environment and these modifications causing a change in local behaviour. This mechanism, known as stimergy, involves no centralised control and no global communication, such that any task allocation algorithms based on this principle should avoid problems of scalability.

In this paper, we study a task allocation problem from a decentralised perspective in which agents must travel to distributed task sites and perform tasks under certain constraints. As previous work on this problem indicated that the efficiency of any solution is greatly limited when agents

cannot build up preferences towards task sites [7], we introduce a system of memory to allow agents to specialise in particular localities. We then test the performance of the algorithm, its dependence on model parameters, and its scalability through a series of simulations.

The rest of the paper is organised as follows. In section 2, we introduce the problem and identify the ways in which performance can be lost. We then introduce the base unit of agent decision making, i.e. the threshold model, and define how it applies to our problem. In section 3, we define conditions which a good solution to the problem should fulfil and introduce a system of agent memory designed to meet these conditions. Next in section 4, we present simulation results of the system, and compare them to both theoretical and practical results of memoryless algorithms. Finally in section 5, we summarise and draw conclusions from our results and give an outlook to future research.

## 2. The Model

We propose a solution to a generic distributed task allocation problem, the mail processing problem, introduced by Bonabeau et al. [3] and developed into its current form by Price et al. [12, 13]. In this problem, there is a set of $N_c$ cities, each of which is capable of producing and storing one batch each of $N_m$ mail types. The cities are served by a set of $N_a$ agents, each of which has an associated mail processing centre. Agents must travel to a city, choose a batch of mail and take this batch to their processing centre, before repeating this process. There are, however, differences between the mail types and each agent $a$, has a mail-specialisation $\sigma_a$, indicative of the mail type its processing centre can efficiently process. Processing mail of this type takes the centre a fixed time $t_p$, while the centre must undergo a *changeover* in order to process mail of type $m \neq \sigma_a$, taking a total time $t_c > t_p$. After the changeover, the centre is specialised to deal efficiently with mail type $m$.

Each centre also has a mail queue in order to buffer the immediate effects of changeovers. This queue is capable of holding up to $L_q$ batches of mail and, while there is space in the queue, the agent continues to collect mail (i.e. remains *active*). When a processing centre finishes processing a batch of mail, it will immediately start processing the next batch in its queue (i.e. the batch which was collected the longest time ago), thus freeing a space in the queue. As centres must process mail in the order in which it was collected, $\sigma_a$ denotes the mail type last taken by agent $a$ and will be the specialisation of the centre when it comes to process the next collected piece of mail.

In order to simulate this system, we discretise time into steps of the amount of time it takes an agent to visit a city and return with mail. This allows us to define our measure of an algorithms performance, i.e. the *efficiency*, as the average amount of mail processed per agent per time step. During each time step the following happens:

1. Each city which is missing a batch of mail of any type produces a new batch of mail of this type.
2. Each active agent chooses and visits a city.
3. Each city randomly determines the order in which its visiting agents are allowed to act.
4. Each agent examines the available mail at the city in a random order, selecting or rejecting each batch individually until either one is selected or all are rejected.
5. Each agent returns to its processing centres and deposits any mail it has collected in the queue.
6. Each processing centre either processes the next piece of mail in its queue, or continues its changeover.

Note that the definition of this problem as "mail processing" is completely arbitrary. Cities are merely localised task-sites, and the mail types are just different types of tasks. The processing centres and their behaviour can be seen as a generic way to introduce a cost into switching task type. In fact Campos et al. [5] study an almost identical problem but describe it as one of truck painting.

The problem of efficiency maximisation can also be seen as minimisation of loss of efficiency, and in [7] we have identified the causes for agents to fail take up mail. For an agent with specialisation $\sigma_a$, the efficiency loss sources are:

($\ell.1$) The agent is inactive due to a full queue.

($\ell.2$) The visited city has mail of type $\sigma_a$, but the agent rejects all mail.

($\ell.3$) The visited city has some mail, but none of type $\sigma_a$, and the agent rejects all mail.

($\ell.4$) The visited city has no available mail at the time of the agent's action.

### 2.1. Variable Threshold Model

While a fair proportion of the loss sources (in particular $\ell.4$) is determined by the agent's city choice, an agent still needs an efficient method of decision making concerning the take up of mail once it has arrived at a city. In particular, a method that strikes a good balance between $\ell.1$ (caused by repeated change overs) and $\ell.3$ (rejection of non-specialised mail types), is necessary. It has been shown that the a model known as the variable threshold model is a good mechanism for controlling this tradeoff [12, 13, 7]. The original threshold model is a social-insect inspired method of task

allocation developed by Bonabeau et al.[4] in order to explain the flexibility of of social insect colonies. It postulates that tasks can be split into distinct tasks and that each individual within a colony has a threshold, $\theta$ for each type of task which determines its likelihood of engaging in that task. Each instance of a task has a stimulus $s$, which indicates the priority of the task for completion. Upon encountering a task with stimulus $s$, the individual compares it with its corresponding threshold. It will engage in the task with high probability if $s >> \theta$, low probability if $s << \theta$, and probability close to $0.5$ if $s \approx \theta$.

The threshold model was then developed into the variable threshold model by Theraulaz et al. [16], in which individuals undertaking a task increase their probability of undertaking similar tasks by decreasing their threshold for this type of task and by increasing their thresholds for all other task types. This allows for the population to have a distribution of specialisations that follows demand.

In order to apply this model to our problem, we define each mail type as being a different type of task. A batch of mail is seen as an instance of a task and the stimulus is taken to be the amount of time it has been waiting at a city. When a batch of mail is created it is assigned a waiting time of 1 and this time increases by 1 every time step. Each city $c$, therefore, has a set of waiting times $\mathbf{w}_c = (w_{c,1}, ...w_{c,N_m})$ where $w_{c,m}$ is the waiting time of the $m^{th}$ mail type.

Each agent $a$ is given a set of thresholds $\boldsymbol{\theta}_a = (\theta_{a,1}, ...\theta_{a,N_m})$, where $\theta_{a,m}$ is the agent's threshold for taking mail type $m$. Upon encountering a task of type $m$ with stimulus $w$, the probability of an agent $a$ accepting is determined by its threshold $\theta_{a,m}$ and its *threshold function* $\Theta(\theta_{a,m}, w)$. Here, we have opted for the exponential threshold function [7]:

$$\Theta(\theta, w) = \begin{cases} 0 & \text{if } w = 0, \\ \frac{w^\lambda}{w^\lambda + \theta^\lambda} & \text{otherwise,} \end{cases} \tag{1}$$

where $\lambda$ is some appropriately chosen positive exponent.

Both the efficiency and the the flexibility (the capacity of the system to adapt to new situations) are determined by the mechanism in which the agents adapt their thresholds. Initially an agent takes its thresholds uniformly in the interval $[\theta_{min}, \theta_{max}]$. Upon taking mail, the agent applies an *update rule* to each of its thresholds. Here, we use the switch-over (SO) update rule [7] which has been shown to give near optimal efficiency in the case of random city choices. Upon accepting mail of type $m$ an agent $a$ updates its thresholds as follows:

$$\theta_{a,n} = \begin{cases} \theta_{min} & \text{if } m = n, \\ \theta_{max} & \text{otherwise.} \end{cases} \tag{2}$$

## 3. Memory

While the threshold model with well chosen of rules and parameters can provide a good solution to this problem, it is always limited by the likelihood of poor city choices. An agent choosing a city at random is no more likely to visit a city which has no other agents visiting it than it is to visit a city at which has already had all its mail taken, thus leaving the first city unserved and the agent without mail. The upper limit this puts on the efficiency can be calculated [7] analytically in the limit $N_a \to \infty$, and is given by:

$$\sum_{k=1}^{N_m} \chi_k(t) \left( 1 - P_{R_{a/c}}(k) - \frac{R_{a/c} - k}{R_{a/c}} \sum_{j=k+1}^{\infty} P_{R_{a/c}}(j) \right), \tag{3}$$

where $\chi_k(t)$ is the proportion of cities with exactly $k$ available pieces of mail at the beginning of time step $t$, and where $P_{R_{a/c}}$ is the Poisson distribution with parameter $R_{a/c}$ (the ratio of agents to cities). Since in the current setting cities always replace any taken mail, we have that $\chi_k(t) = \delta_{k,N_m}$, where $\delta$ is the Kronecker delta.

For fixed resources (number of agents) and environment (number of cities and mail types) the only way to improve on this limit is to change the profile of agents visiting cities, and hence the way in which agents choose cities. When designing an efficient method for agents to visit cities, it is useful to establish some conditions on how agents should be allocated to cities in ideal circumstances:

($\boldsymbol{c}$.1) No city should be visited by fewer agents than it has mail available.

($\boldsymbol{c}$.2) For each mail type $m$ that a city has available, exactly one of its visiting agents should have specialisation $\sigma_a = m$.

Globally, $\boldsymbol{c}$.1 minimises $\boldsymbol{\ell}.4$ while $\boldsymbol{c}$.2 reduces the tradeoff between $\boldsymbol{\ell}.1$ & $\boldsymbol{\ell}.3$. However, while a lack of appropriately specialised agents can be rectified by the threshold model, $\boldsymbol{c}$.1 is impossible to fulfil with if $R_{a/m} < 1$, where $R_{a/m} = \frac{N_a}{N_c \times N_m}$ is the ratio of agents to mail. Hence, we must add a third condition to ensure that no cities go unserved for long periods of time.

($\boldsymbol{c}$.3) If $\boldsymbol{c}$.1 & $\boldsymbol{c}$.2 cannot be fulfilled over a single time step, they should be fulfilled uniformly at all cities over a longer period.

In an attempt to fulfil these conditions while maintaining the decentralised nature of the algorithm (no knowledge of the state of cities before agents visit them), and the relative simplicity of its component agents, we propose a *Stimulus Based (SB)* system of agent memory in which taking mail from a city increases an agent's chance of revisiting the city in the future.

Each agent $a$ is assigned a memory $\vec{\mathcal{M}}_a$, consisting of a set of $\mu_a$ paired variables $\vec{\mathcal{M}}_a \equiv \{(\mathcal{C}_{a,\ell}, \mathcal{W}_{a,\ell}),\ \ell = 1, .., \mu_a\}$, where $\mathcal{C}_{a,\ell}$ is a city which the agent has taken mail from in the past, and where $\mathcal{W}_{a,\ell}$ is a weight assigned to this memory.

An agent $a$ either bases its selection of city on its memory $\vec{\mathcal{M}}_a$ with probability $\rho_a$, or chooses one randomly with probability $1 - \rho_a$. The parameter $\rho_a \in [0, 1]$ allows us to move continuously from a memoryless scenario ($\rho_a = 0$), to one completely dominated by the memory ($\rho_a = 1$). The total probability that agent $a$ visits city $c$ is given by

$$V(c|\vec{\mathcal{M}}_a) = \rho_a\ M(c|\vec{\mathcal{M}}_a) + (1 - \rho_a)\frac{1}{N_c} \qquad (4)$$

where $M(c|\vec{\mathcal{M}}_a)$ is the probability that city $c$ is visited given that memory $\vec{\mathcal{M}}_a$ is used. Initially, the probability for choosing a specific city from the memory was taken to be its normalised weight $\frac{\mathcal{W}_{a,\ell}}{\sum_{j=1}^{\mu_a} \mathcal{W}_{a,j}}$. However, we encountered the problem that self-reinforcement tends to lead to one weight becoming so much larger than all the others that it completely dominates, thus making stable multiple city-specialisation virtually impossible. In general, this may lead to some cities being well served while others are neglected in violation of condition $\boldsymbol{c}.3$ (for similar reasons values of $\rho_a < 1$ are needed). This effect can be avoided by the introduction of a maximum usable weight $\mathcal{L}_a$, and by replacing the $\mathcal{W}_{a,\ell}$ with $\min(\mathcal{W}_{a,\ell}, \mathcal{L}_a)$, such that

$$M(c|\vec{\mathcal{M}}_a) = \sum_{\ell=1}^{\mu_a} \frac{\min(\mathcal{W}_{a,\ell}, \mathcal{L}_a)}{\sum_{j=1}^{\mu_a} \min(\mathcal{W}_{a,j}, \mathcal{L}_a)}\ \delta_{c,\mathcal{C}_{a,\ell}}\ , \qquad (5)$$

when $\sum_{j=1}^{\mu_a} \mathcal{W}_{a,j} > 0$, and $M(c|\vec{\mathcal{M}}_a) = \frac{1}{N_c}$ otherwise. This allows for uniform probabilities in a small subset of cities (needed for $\boldsymbol{c}.3$), while allowing the weights themselves to become large (giving the agents' city choices stability). Note that although this stability is an advantage in the current scenario, it would leave the agent vulnerable if a breakdown were to occur at a city for which it has built up a large weight.

In a similar spirit to the threshold model, we propose a memory weight update that is stimulus based. In SB memory, cities do not occur more than once in an agent's memory, such that an agent can remember up to $\mu_a$ cities. Each city in the agent's memory is assigned an individual weight which increases when an agent takes mail from that city proportionally to the stimulus (waiting time) of the taken mail, and decreases when it does not. We loosely define a city to be *well served* if it has a set of agents which return to it repeatedly and ensure that all types of mail are taken from it with regularity. Specialisation of any new agent in a city which is already well served adds nothing to $\boldsymbol{c}.1$, $\boldsymbol{c}.2$, and should be avoided. As agents have no direct knowledge of other agents' memories and specialisations, they must infer it from the only available information at a city, namely the waiting times. Mail at well served cities will tend to have low waiting times compared to poorly served cities, such that it makes sense to make the increase in weights proportional to the waiting time of taken mail. Upon taking mail with waiting time $w$ from city $c$ the agent's memory is updated as follows:

1. If $\mathcal{C}_{a,\ell} = c$ (city $c$ is already in the agent's memory), then its weight is increased by $w$:

$$(\mathcal{C}_{a,\ell}, \mathcal{W}_{a,\ell}) \rightarrow (\mathcal{C}_{a,\ell}, \mathcal{W}_{a,\ell} + w) \qquad (6)$$

2. Otherwise if $w$ is at least as big as the least weight $\mathcal{W}_{a,\ell}$ then city $c$ replaces this lowest weighted element.

$$(\mathcal{C}_{a,\ell}, \mathcal{W}_{a,\ell}) \rightarrow (c, w) \qquad (7)$$

   Note that in case of multiple equal minimum weights, only the city which was last visited the longest time ago is replaced.

3. All unmodified, non-zero weights decay.

$$(\mathcal{C}_{a,\ell}, \mathcal{W}_{a,\ell}) \rightarrow (\mathcal{C}_{a,\ell}, \mathcal{W}_{a,\ell} - 1) \qquad (8)$$

Note that if no mail is taken, all non-zero weights decay. We define an agent which has a weight of at least $\mathcal{L}_a$ in a city is *city-specialised* in that city. Note that an agent $a$ can in principle be city-specialised in up to $\mu_a$ cities. Now, we can formalise the definition of a well served city as a city which has an agent $a$ specialised in it with mail specialisation $\sigma_a = m$ for every mail type $m$, such that $\boldsymbol{c}.1$, $\boldsymbol{c}.2$ are fulfilled locally. If it persists, a well served city can be seen as an example of emergent cooperation between agents. Each agent minimises the waiting time of its given mail type which decreases the chance of changeovers for other city-specialised agents. In return its own chance of a changeover is decreased by the low waiting times of all other mail types. This situation resolves the conflict between $\boldsymbol{\ell}.1$ and $\boldsymbol{\ell}.3$ and could, in principle, lead to perfect efficiency.

Assuming that $N_m$, $R_{a/m}$ and $\mu_a$ are fixed and finite, both the memory requirements to implement this algorithm, and the number of operations per time step, scale *linearly* with the system size $N_a$. An agents behaviour (including memory) is only affected by the stimulus detected at cities and this is independent of $N_a$, hence, each agent performs $\mathrm{O}(1)$ operations. Cities must perform $N_m$ operations to increase their waiting times and in the worst case (all agents visiting a single city) must perform $N_a - 1$ operations to randomly order the agents.

## 4. Results

While the full optimisation of the model's parameters for particular circumstances is beyond the scope of this paper, we do study the influence of some key parameters on the system's qualitative behaviour. Parameters that are not explicitly varied, are set to some default values, which we take to be the same as in [7] for comparison with memoryless algorithms. Hence, we take $N_m = 2$ which is the most interesting case, as the distribution of agents to cities becomes more uniform with increasing $N_m$ such that the upper limit on the efficiency (3) tends to 1. Furthermore, we take $R_{a/m} = 1$, as this is both the minimum ratio at which all cities could in principle be served perfectly, and the maximum ratio at which all agents could take mail every iteration (no wasted resources). We also set $N_a = 5 \times 10^4$ and have shown in [7] that this is sufficient to neglect finite size effects. For the threshold function (1), we take $\lambda = 2$ and set $\theta_{min} = 0$ in order to minimise $\boldsymbol{\ell}.2$, and $\theta_{max} = 50$ sufficiently high to avoid most repeated changeovers. Finally, for the memory parameters, we take $\mu_a = 10$, $\mathcal{L}_a = 10$ and $\rho = 0.95$.

Figure 1 shows the performance of the algorithm over the course of a single run. We see that efficiency quickly tends to a high value while $\boldsymbol{\ell}.1$-$\boldsymbol{\ell}.4$ all take small values. We see the reason behind this in the specialisation behaviour, with cities being well served by either singly or double specialised agents, fulfilling $\boldsymbol{c}.3$. Subsequently, some of the remaining unspecialised agents gain specialisation in a city being served by a doubly specialised agent, which then loses its second specialisation, This fulfils $\boldsymbol{c}.1$ and $\boldsymbol{c}.2$ and allows further increases in efficiency.

Figure 2 shows the influence of $\rho$ on the efficiency. The efficiency is a monotonically increasing function of $\rho$, with the most marked increase taking place for $\rho > 0.5$. This increase can be explained by the specialisation behaviour, with city-specialisation starting to become prevalent at this point, as agents return often enough to cities for their average weights to increase. Although specialisation in up to 10 cities is in principle possible, we observe that most agents specialise in a single city. Double city-specialisations also occur for intermediately high values of $\rho$, but for $\rho \approx 1$ the chances of a specialised agent to visit another city become so small that very few doubly specialised agents emerge. We note, furthermore, that although the overall efficiency is maximised for $\rho = 1$, this is clearly not optimal for the fraction of well served cities or the maximum waiting times. With increasing $\rho$ a smaller and smaller fraction of agents visit cities at random, such that not well served cities are less and less likely to be visited. The sharp increase after $\rho > 0.95$ is due to the fact that city-specialised agents not only never choose cities at random, but also stop specialising in multiple cities.

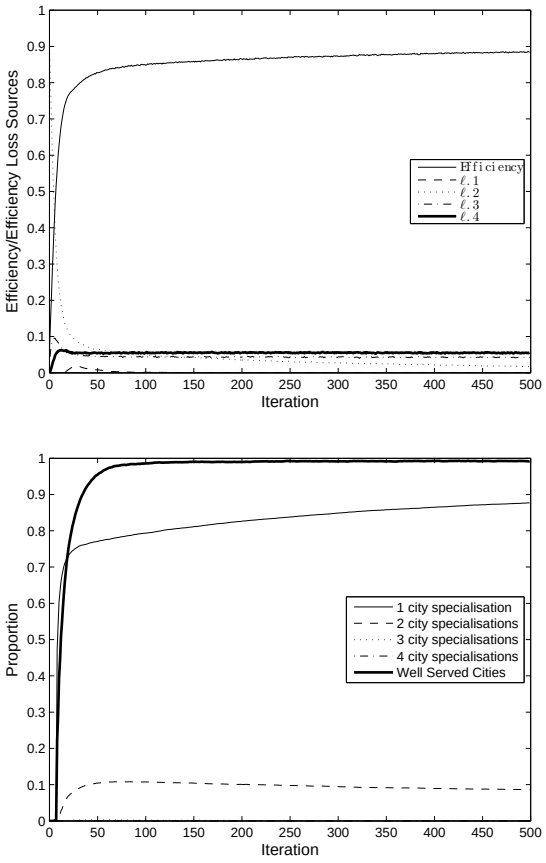


**Figure 1. Efficiency and loss sources (top), and specialisation behaviour (bottom) for $R_{a/m} = 1$ and $\rho = 0.95$ over a single run of $500$ iterations. After a sharp initial increase, there is a slow increase in the efficiency to its asymptotic value which is mainly due to a corresponding decrease in $\ell.2$. Other losses become approximately static from the $50^{th}$ iteration onwards. Most agents are either singly or doubly specialised by this point which is followed by a slow loss of doubly specialised agents. The proportion of well served cities quickly tends to $1$.**

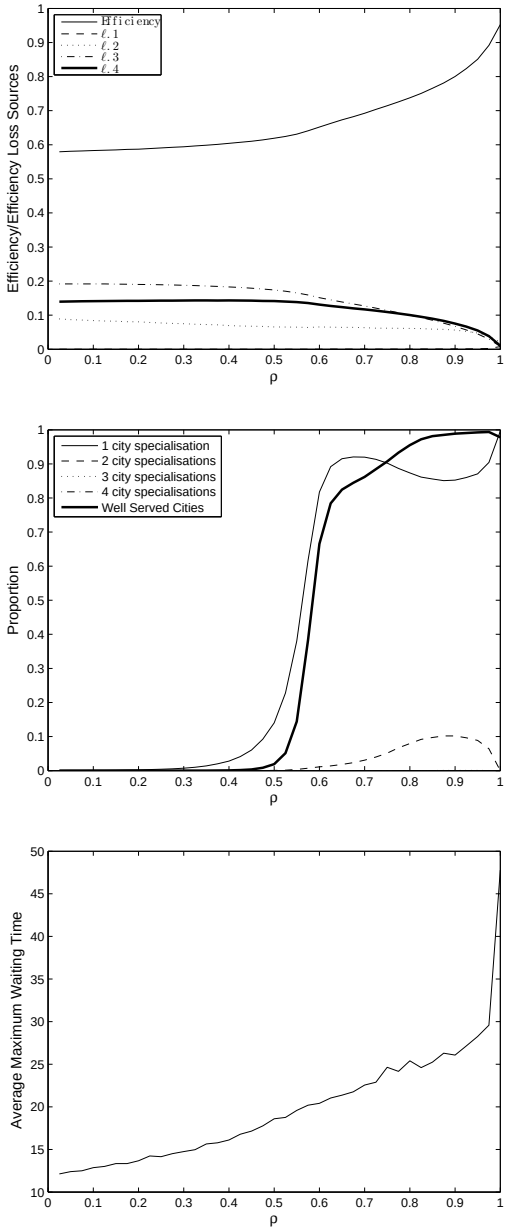


**Figure 2. Efficiency and loss sources (top), specialisation behaviour (middle), and maximum waiting time (bottom) as a function of $\rho$ for $R_{a/m} = 1$ averaged from iteration $401$ to $500$. Efficiency increases with $\rho$ with the rate increasing after $\rho = 0.5$. $\ell.1$ and $\ell.3$ decrease correspondingly with $\ell.2$ static and $\ell.4$ negligible. Singly specialised agents appear near $\rho = 0.5$, followed shortly by well served cities. Some of agents change from singly to doubly specialised agents at higher values of $\rho$ before decreasing again. Maximum waiting times increase steadily with a sharp increase at $\rho = 1$.**

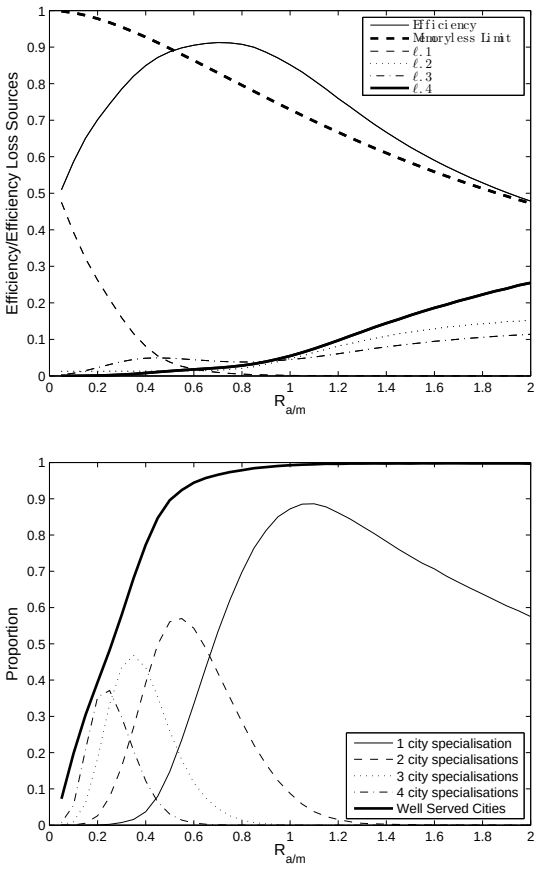


**Figure 3. Efficiency and loss sources (top), and specialisation behaviour (bottom) as a function of $R_{a/m}$ for $\rho = 0.95$ (which is optimal for $R_{a/m} = 1$ only) averaged from iteration $401$ to $500$. The efficiency initially increases sharply and surpasses the memoryless limit before decreasing and asymptotically approaching the limit from above. $\ell.1$ decreases with $R_{a/m}$ while $\ell.2$-$\ell.4$ increase with the rate of increase growing after efficiency reaches its maximum. The modal number of specialisations decrease as $R_{a/m}$ grows with peaks occurring at approximately the inverse of the number of specialisations. As expected, the fraction of well served cities initially shows a sharp increase and asymptotically tends to $1$.**

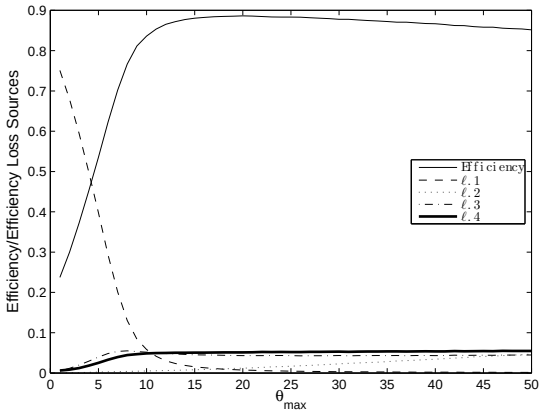


**Figure 4. Efficiency and loss sources as a function of $\theta_{max}$ for $R_{a/m} = 1$ and $\rho = 0.95$ averaged averaged from iteration $401$ to $500$. The efficiency increases sharply initially, before peaking and declining slowly. $\ell.1$ decreases with $\theta_{max}$ while $\ell.3$ and $\ell.4$ increase following a similar trend to efficiency but at lower values, while $\ell.2$ exhibits a slow monotonic increase.**

In figure 3 we compare efficiency with the upper bound on efficiency of any algorithm using random city choices, given by equation 3. At low values of $R_{a/m}$ high $\boldsymbol{\ell}.1$ leads to low efficiency compared to the limit as high average waiting times overwhelm the selectivity of the threshold function, leading to multiple changeovers. Note that this is a consequence of our choice of $\theta_{max}$, higher values would lead to increased efficiency at low $R_{a/m}$ (due to lower $\boldsymbol{\ell}.1$) and decreased efficiency at high $R_{a/m}$ (due to higher $\boldsymbol{\ell}.2$, $\boldsymbol{\ell}.3$).

As $R_{a/m}$ increases, $\boldsymbol{\ell}.1$ decreases without much of an increase in $\boldsymbol{\ell}.2$-$\boldsymbol{\ell}.4$ as city specialisation becomes useful. Note that the average number of city specialisations becomes approximately $R_{a/m}^{-1}$ meaning that the agents are acting to serve approximately all the mail. This leads to most cities being well served well before $R_{a/m} = 1$, fulfilling $\boldsymbol{c}.3$ and allowing efficiency to surpass the memoryless limit. The proportion of singly specialised agents continues to increase, fulfilling $\boldsymbol{c}.1$ and $\boldsymbol{c}.2$, but this does not lead to increased efficiency as it is inherently limited by the fact that there are less batches of mail than there are agents. Efficiency decreases as $\boldsymbol{\ell}.2$-$\boldsymbol{\ell}.4$ increase, with these increases caused by both the lack of mail and the low average waiting times of the remaining batches. This means that less agents specialise in cities and efficiency approaches the memoryless limit, which also tends to the true upper limit $R_{a/m}^{-1}$ for high values.

| | Efficiency | Final Efficiency |
|---|---|---|
| Best Memoryless | 0.626 | 0.633 |
| Memoryless Limit | N/A | 0.729 |
| With Memory | 0.953 | 0.986 |

**Table 1. Comparison of the algorithm's average efficiency with memory ($\rho = 1$) and using the best memoryless algorithm with the theoretical limit for a memoryless algorithm. Starting from uniform initial conditions and with no city-specialised agents, the efficiency is averaged over $500$ iterations and the final efficiency is averaged over a subsequent $100$ iterations. We see that memory provides a large boost in efficiency, and an even larger boost in the final efficiency.**

Figure 4 shows the influence of $\theta_{max}$ on the efficiency. We observe that (too) low values for $\theta_{max}$ cause a high probability of changeovers (high $\boldsymbol{\ell}.1$). The peak in efficiency is reached at approximately $\theta_{max} = 20$, after which it decreases due to increases in $\boldsymbol{\ell}.2$. These increases are due to higher average initial thresholds which lower the probability of initial mail uptake and hence specialisation. The behaviour is markedly different from that found in a memoryless system, for which much lower values of $\theta_{max}$ are optimal [7]. The increase in the optimal value of $\theta_{max}$ is due to the increased need to avoid changeovers. At a well served city an agent that undergoes a changeover must not only cope with the penalty in processing time, but must now also compete for mail with another agent with whom it was previously cooperating.

Table 1 compares the theoretical upper limit of memoryless efficiency, the best actual efficiency for a fully optimised memoryless algorithm [7], and the best efficiency obtained in this paper using SB memory with partially optimised parameters. Note that we use efficiency as the measure of performance and so take $\rho = 1$ even though this dramatically increases maximum waiting times and is not optimal for other measures such as the fraction of well served cities. The values in the left column include the initial phase in which the agents have to (self-) organise their behaviour, and as such are also indicative for the adaptability and the speed at which this organisation takes place. The values in the right column give the all out performance once the self-organisation is more or less completed. Note that the final efficiency of the only partially optimised SB memory system is at 98.6%, which is indeed very close to perfect efficiency.

## 5. Conclusions and Future Work

In this paper, we have introduced the SB model of agent memory as a solution to a problem of distributed task selection. The performance of this model has been investigated under the variation of key parameters and is close to that of the best centralised solution, while retaining the necessary conditions for good scalability such as a (very) limited information flow, localised decision making and relatively simple agents. In particular, the elimination of random city choices allows the system to exceed the theoretical upper limit on memoryless efficiency by $35.3\%$ after convergence and to obtain near perfect efficiency. This is partially due to emergent cooperation between the agents, which resolves the conflict between the need for agent flexibility, and the constraints of the model.

The performance, however, may still be limited by both the (not fully optimised) choice of agent parameters, and by the rules for decision making (SO update rule and exponential threshold function). Hence, the use of a genetic algorithm to find optimal parameters and of genetic programming to find improved rules, should lead to even better results. In addition, it would be interesting to study the performance of SB memory in a dynamical environment (with non-constant probabilities of mail production) as has been done for memoryless algorithms [7], and to compare it with other mechanisms of memory.

## References


[1] S. N. Beshers and J. H. Fewell. Models of division of labor in social insects. *Annual Review of Entomology*, 46:413–440, 2001.

[2] E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm Intelligence: From Natural to Artificial Systems*. Oxford University Press, 1999.

[3] E. Bonabeau, A. Sobkowski, G. Theraulaz, and J.-L. Deneubourg. Adaptive task allocation inspired by a model of division of labor in social insects. In *Biocomputing and Emergent Computation*, pages 36–45, 1997.

[4] E. Bonabeau, G. Theraulaz, and J.-L. Deneubourg. Fixed response thresholds and the regulation of division of labour in insect societies. *Bull. Math. Biol.*, 60:753–807, 1998.

[5] M. Campos, E. Bonabeau, G. Thraulaz, and J.-L. Deneubourg. Dynamic scheduling and division of labor in social insects. *Adaptive Behavior*, 8(2):83–92, 2001.

[6] Y. Chevaleyre, P. E. Dunne, U. Endriss, J. Lang, M. Lemaître, N. Maudet, J. Padget, S. Phelps, J. A. Rodríguez-Aguilar, and P. Sousa. Issues in multiagent resource allocation.

[7] H. Goldingay and J. van Mourik. Genetics and competing strategies in a threshold model for mail processing. Technical Report NCRG/2008/003, Aston University, 2008.

[8] P. P. Grassé. La reconstruction du nid et les interactions inter-individuelles chez les bellicositermes natalenis et cubitermes sp. la théorie de la stigmergie: essai d'interprétation des termites constructeurs. *Insectes Sociaux*, 6:41–83, 1959.

[9] B. Hong and V. K. Prasanna. Distributed adaptive task allocation in heterogeneous computing environments to maximize throughput. In *IPDPS*, 2004.

[10] P. Janacik, T. Haimfarth, and F. Rammig. Emergent topology control based on division of labour in ants. In *Proceedings of the IEEE 20th International Conference on Advanced Information Networking and Applications (AINA 2006)*, 2006.

[11] K. H. Low, W. K. Leow, and M. H. Ang, Jr. Task allocation via self-organizing swarm coalitions in distributed mobile sensor network. In *Proc. 19th National Conference on Artificial Intelligence (AAAI-04)*, pages 28–33, 2004.

[12] R. Price. Evaluaton of adaptive nature inspired task allocation against alternate decentralised multiagent strategies, 2004.

[13] R. Price and P. Tiňo. *Parallel Problem Solving from Nature VIII*, chapter Evaluaton of Adaptive Nature Inspired Task Allocation Against Alternate Decentralised Multiagent Strategies, pages 982–990. Springer Berlin / Heidelberg, 2004.

[14] O. F. Rana and K. Stout. What is scalability in multi-agent systems? In *AGENTS '00: Proceedings of the fourth international conference on Autonomous agents*, pages 56–63, 2000.

[15] O. Shehory, S. Kraus, and O. Yadgar. Emergent cooperative goal-satisfaction in large scale automated-agent systems. *Artificial Intelligence*, 110(1):1–55, 1999.

[16] G. Theraulaz, E. Bonabeau, and J.-L. Deneubourg. Response threshold reinforcement and division of labour in insect societies. In *Proc. Roy. Soc. London B 265*, pages 327–332, 1998.