Re-Implementing Dynamic Scheduling and Division of Labor in Social Insects

Daniel Carmona Pedrajas

Universidad Politécnica de Madrid

Abstract. This paper shows the re-implementation of the dynamic scheduling and division of labor presented based on ant specialization so that it takes into account that an agent has limited capacity of resources and needs to recharge them after a number of completed tasks.

Keywords: Dynamic Scheduling · Task Allocation · Task Assignment.

1 Introduction

The base system that will be modified was presented in [1] by Campos et al. The authors present a new algorithm for the task assignment problem and compare it to the market based approach, presented in [2]. They call this approach the *ant-based* approach that mimics the social behavior of ants and their specialization is certain tasks. In this system, an agent k bids for a task using the following formula:

$$P_k = \frac{D_{t_i}^2}{D_{t_i}^2 + \alpha \cdot \theta_{k,t_i}^2 + \Delta T^{2 \cdot \beta}} \tag{1}$$

Where D_{t_i} is the demand of task t_i is given by $D_{t_i} = \sum_j w_j \cdot \delta(t_j - t_i)$, that is, the weighted sum of all the tasks of type i in the systems' queue of tasks (δ is the Dirac function). θ_{k,t_i}^2 is the threshold of agent k for the task t_i, ΔT is a time component dependent on the problem calculates an approximation of the time until task t_i is performed. Once all the agents have made a bid, the one with the highest bid will add the task to the agent's task queue. If there is a tie, the agent will be randomly selected. It is important to note that all the agents can have a maximum of n tasks in its queue, if all of them have their queue at maximum capacity, the task will be returned to the system's task queue. When an agent k is assigned a task of type i, two operations are performed:

- The agent's threshold for task i is decreased by ξ.
- The rest of the agents' threshold is increased by ϕ.

A task threshold is bounded by a minimum and maximum value, θ_{min} and θ_{max} respectively.

This ant-based system was applied to the truck painting problem, where a number of trucks need to be painted in different colours by painting booths

and the goal is to reduce the total paint flushes in the process. The market-based approach was demonstrated to have worse performance than the ant-based. However, this approach assumes that the paint in a painting booth is unlimited and that it doesn't need to refill its paint tank and this paper proposes a slight modification in the above formulas to address this problem.

In section 2 a new environment for the task assignment problem is defined, section 3 proposes the modifications for the algorithm, in section 4 we will define the parameters of the experiments performed and their results will be analyzed in section 5. Finally, section 6 presents some conclusions and future lines of work.

2 Problem restatement

This section specifies the environment where the ant-based approach modifications will be tested.

The new environment will simulate a farm where drones need to take care of the harvest. The field is a grid of $n \times n$ where each cell can be in three different states, which are ciclyc: **empty**, **planted** and **grown**. When a cell transitions to a state, it will publish a task that needs to be performed so that the cell transitions to the next state. An *empty* cell transitions to the *planted* state when a drone performs the task **to plant**, a *planted* cell transitions to a *grown* state when a drone performs the task **to water**, a *grown* cell transitions to an *empty* state when a drone performs the task **to harvest**.

Each drone has a maximum capacity (or space in the case of harvesting) of its resources and needs to refill (or discharge in the case of harvesting) each time they have exhausted their resources. It is assumed that a drone is able to perform any task and switch between any of them at any given time, but this has a time penalization. To perform a task the drone will need to move to the corresponding cell and to recharge it will need to move to the corresponding refill station.

As mentioned in section 1, ΔT depends on the problem, in this case:

$$\Delta T = n_q \cdot t_t + n_c * t_c * t_r + d \tag{2}$$

where n_q is the number of tasks in the queue, t_t is the time that it takes to perform a task, n_c is the number of task changes, t_c is the time it takes to change for a task, t_r, the time it takes to recharge (or discharge) the resource and d is the time it takes to move from the last cell in the queue to the cell that has published the auctioned task.

3 Ant-based approach modifications

In the new system presented above flushing the paint booth equals to changing a task but the recharge is a new concept that needs to be adressed. Two mechanisms have been designed to try to minimize the recharges:

– Modification of the **threshold update**: The motivation of this change is to keep a momentum on the specialization, when the drones are busy and their task queue are full, it is probable that a drone k that is not specialized in task i wins the auction and lowers the threshold corresponding to that task and thus raises the task threshold of other drones specialized in i. To keep the equilibrium, we will perform a moderate update if the drone is not specialized and a bigger one if it is:

$$\theta_{k,i} = \theta_{k,i} - \frac{\xi}{c} \quad (3)$$

(c * E)/max capacity

$$\theta_{j,i} = \theta_{k,j} + \frac{\phi}{c}, \quad \forall j \neq k \quad (4)$$

Where c is the minimum between the maximum capacity of the drones and the number of consecutive times that the drone k has performed the task i.

– Modification of the **bidding formula**: In the same way, a drone k should bid higher for a task i that has performed consecutively in a recent period of time and this is modeled as follows:

$$P_k = \frac{c \cdot D_{t_i}^2}{D_{t_i}^2 + \alpha \cdot \theta_{k,t_i}^2 + \Delta T^{2 \cdot \beta}} \quad (5)$$

Where c is the same as in the modification above.

4 Experimental setup

The system has been implemented in Unity. Three batches of experiments have been performed, one with the original ant-based approach, one with the threshold update modification and the last one with both modifications presented in section 3. For each batch, 15 experiments of 15 minutes each have been monitored each 5 seconds measuring total changes, total recharges, total tasks completed, each drone speciality and its minimum threshold. The grid is of side 8×8 and the parameters used are shown in Table 1. These parameters have been fixed taking

Table 1: Parameters of the experiments.

Drones	Max capacity	Speed	t_t	t_c	t_r	ξ	ϕ	θ_{min}	θ_{max}	α	β
6	10	5	2	5	3	10	2	5	100	400	2

as a starting point the values presented in [1] and after some experimentation it was found that this configuration is better for this problem. No optimization algorithm has been used to tune these parameters and could be interesting to define a better configuration in future works. However, as all the modifications will be tested with the same configuration, their impact can be measured.

5 Results

In the table 2 we can see the means and standard deviations of the tasks completed, recharges and changes done during the experiments. The original algorithm has better results than the two versions proposed in this paper for the task and recharges metrics. This difference though, is not very significant, the

Table 2: Comparison between the three batches of experiments

	Original		**Threshold**		**Threshold + Bidding**	
	mean	std	mean	std	mean	std
Total tasks	384.40	13.36	384.13	13.13	382.20	11.00
Total recharges	251.93	5.53	253.13	8.83	254.33	6.86
Total changes	176.53	5.98	175.86	5.23	178.33	6.58

ANOVA tests shows that there is no significant differences between the versions of the algorithm as we can see in the table 3 where the p-value is greater than 0.05 which rejects the hypothesis that states there is a mean that is different to the others. We can conclude that the modifications proposed don't have the

Table 3: One way ANOVA test for the metrics for the different methods

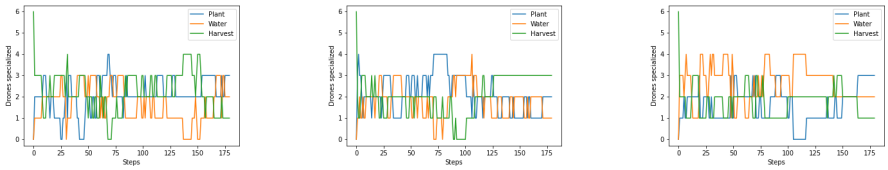
	Tasks	**Recharges**	**Changes**
F-value	0.1373	0.6871	0.4161
p-value	0.8720	0.6622	0.5085

impact desired, the recharges and task changes do not vary with respect with the original algorithm. However, we can see that the modifications impact the flexibility of the drones, the variation on the number of drones specialized in a certain task tends to be more stable with the *Threshold+Bidding* modification (see Fig.1c), followed by the *Threshold* modification (see Fig.1b) while the original algorithm (see Fig.1a) shows more flexibility. This is just an intuition that needs further investigation.

6 Conclusions

As we have seen in section 5, both of the modifications proposed have no impact on the outcome of the simulation. Nonetheless, the fact that they don't improve the performance in this problem doesn't mean that this won't work in other type of problem.

The reason for which there is no difference, and again this is just an intuition, is because in this environment, the new tasks that are created are directly dependant on the tasks completed due to the cyclic nature of the cell transition



(a) Original algorithm (b) Threshold (c) Threshold + Bidding

Fig. 1: Specialized drones by task over time using the 3 variants

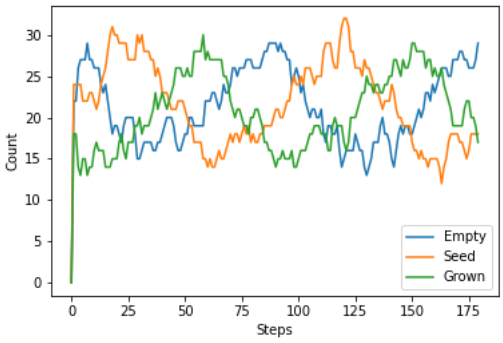


Fig. 2: Cell types over time

system as we can see in Fig.2. A future line of work could start digging in this direction, proposing a new task allocation algorithm specialized in cyclic events. As a final note, you can find the experiment results in this repository as well as an .exe for Windows with the *threshold+bidding* variant of the algorithm implemented.

References

1. Campos, M., Bonabeau, E., Théraulaz, G., Denebourg, JL.: Dynamic Scheduling and Division of Labor in Social Insects, Adaptive Behavior **8**(2), 83=-96 (2000)
2. Morley, R.: Painting Trucks at General Motors: The Effectiveness of a Complexity-Based Approach. In: Embracing Complexity: Exploring the Application of Complex Adaptive Systems to Business, pp. 53–58. MA: The Ernst & Young Center for Business Innovation, Cambridge (1996)