Theoretical Study of Ant-based Algorithms for Multi-Agent Patrolling

Arnaud Glad and **Olivier Simonin** and **Olivier Buffet** and **François Charpillet** [1]

Abstract. This paper addresses the multi-agent patrolling problem, which consists for a set of autonomous agents to visit all the places of an unknown environment as regularly as possible. The proposed approach is based on the ant paradigm. Each agent can only mark and move according to its local perception of the environment. We study EVAW, a pheromone-based variant of the EVAP [3] and VAW [12]. The main novelty of the paper is the proof of some emergent spatial properties of the proposed algorithm. In particular we show that obtained cycles are necessarily of same length, which ensures an efficient spatial distribution of the agents. We also report some experimental results and discuss open questions concerning the proposed algorithm.

1 INTRODUCTION

Deploying autonomous agents or robots in unknown or dynamic environments is a challenging problem for a growing number of tasks (e.g. military surveillance, rescue after natural disasters, etc.). In this paper we address an important task: the patrolling of an unknown environment. It consists in several agents that are in charge of the surveillance of a limited area. We suppose that this area is not known in advance and the number of agents can change dynamically. So we are looking for a patrolling approach that provides adaptability and robustness.

To address such a challenge we study a bio-inspired algorithm that mimics ant mechanisms. Ants provide decentralized algorithms relying on very simple individual behaviors [6]. A particularity of ants is their ability to use the environment as a shared memory by dropping and sensing pheromones, defining temporary information (due to the evaporation process). Such a paradigm has been used to define several pheromone-based algorithms and meta-heuristics to deal with spatial or more generally distributed problems [5, 2, 4, 9, 10].

The patrolling problem can be defined, for a group of agents, as the problem of visiting a set of places while minimizing the time between two consecutive visits. This time is called idleness. For about ten years, several models have been proposed to deal with patrolling. Most of these approaches propose to search for a policy offline by ant-walk and consider a priori known environments represented as graphs [8, 1, 7]. On the contrary, few models have been proposed to deal with unknown and dynamic environments and online computation. We can mention Wagner et al. [13, 11] who proposed ant-based algorithms for the covering problem. In these papers they explored the capabilities of self-organized systems, in which each agent can only read and write integers on the edges of a graph. In this paper we study such systems when the environment is a grid. So we present the EVAP algorithm, introduced in [3], that just uses the pheromone evaporation process, and we compare it to a variant of the VAW algorithm [12]. Those algorithms exhibit interesting properties. After an exploration phase, agents self-organize into stable partial cycles of equal length that completely cover the environment. As a consequence, cells are visited at a very regular frequency. As this property is desirable in the patrolling problem, our main objective is to demonstrate this property formally.

The paper is organized as follows. In Section 2 we introduce the multi-agent patrolling problem. Then Section 3 presents the EVAP and VAW ant-based algorithms allowing to deal with covering and patrolling problems, and we show that they have similar behaviors. In Section 4 we study emergent spatial properties of EVAW, a combination of these two algorithms, by focusing on the emergence of optimal cycles. Before concluding, Section 5 discusses some open questions about the proposed approach.

2 THE PATROLLING PROBLEM

2.1 Definition

Patrolling consists in deploying several agents in order to visit at regular time intervals some defined places of an area. It aims at gathering reliable information, seeking objects and watching over places in order to defend them against any intrusion, etc. An efficient patrol in an environment requires that the delay between two consecutive visits of a given place is minimal. Related work on multi-agent patrolling generally considers that the environment is known, two-dimensional and that it can be reduced to a graph $G(V, E)$ (V the nodes to be visited, E the arcs defining the valid paths between nodes).

2.2 Covering vs. Patrolling

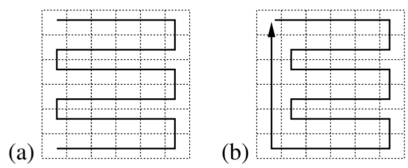

Figure 1. Optimal covering is not necessary optimal patrolling

Covering aims, for one or multiple agents, at visiting each place of the environment once within the shortest possible time. Then patrolling can be intuitively considered as the process of repeatedly covering an environment. But a simple example can show that repeating an optimal solution to cover the environment is not necessary

[1] INRIA/Nancy University, Loria Lab., MAIA project, Nancy, France
email: firstname.lastname@loria.fr

optimal for patrolling. Indeed, in the case of Figure 1 we have two optimal covers but only the second one is an optimal patrol since the last visited cell is adjacent to the first one. Covering approaches may not be relevant in the scope of the patrolling problem.

In next sections we address the patrolling problem by using simple agents that cannot communicate directly.

3 ANT-INSPIRED ALGORITHMS

3.1 Presentation of the Algorithms

3.1.1 The EVAP Algorithm

The EVAP algorithm has been introduced in [3]. This algorithm solves the multi-agent patrolling problem even when the environment is unknown. It is based on a digital pheromone model in which pheromones are represented as numbers whose value decreases over time (simulating the evaporation process of biological pheromones).

Agents evolve in a 2D grid. They can perceive and move to the four adjacent cells representing their neighborhood (noted $N(x)$, x being the current cell). Algorithm 1 describes the individual behavior of each agent. When an agent visits a cell, it drops a quantity Q_{max} of pheromone, then moves according to the negative gradient of pheromone. As the environment evaporates pheromones, with rate ρ (see Algorithm 2), the remaining quantity in a cell x (noted $q(x)$) represents the time elapsed since its last visit. So, an agent's local behavior is defined by moving to the cell of its neighborhood which has not been visited for the longest time.

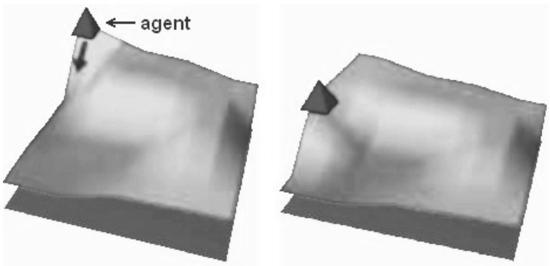

Figure 2. 3D illustration of the EVAP algorithm (with one agent)

Algorithm 1 EVAP Agent (situated on cell x)

A) Find a cell y in $N(x)$ such that $q(y) = min_{w \in N(x)} q(w)$
in case of multiple choices make a random choice
B) Move to cell y
C) Set $q(y) \leftarrow Q_{max}$ (drop the Max quantity of pheromone)

Algorithm 2 EVAP Environment

For every cell x of the environment
If $q(x) \neq 0$ then $q(x) \leftarrow \rho.q(x)$
($\rho \in]0, 1[$)

3.1.2 The Vertex-Ant-Walk (VAW) Algorithm

In this section, we present an earlier version of the VAW algorithm (noted WAV$_0$ in the rest of the paper) introduced by Wagner and co-authors in an appendix of [12]. The local behavior of the agents is the same as the EVAP algorithm (gradient descent), but the dropped information is the date $s(x)$ of the visit instead of laying a quantity of pheromone. So, in the VAW$_0$ algorithm, agents must have synchronised time counters (same frequency) and start at the same time with counter $t = 0$.

Algorithm 3 Vertex-ant-walk$_0$ (ant situated in cell x)

A) Find a cell y in $N(x)$ such that $s(y) = min_{w \in N(x)} s(w)$
in case of multiple choices make a random choice
B) Set $s(x) \leftarrow t$
C) Move to cell y
D) $t = t + 1$

3.2 Comparison of the EVAP and VAW$_0$ Algorithms

Lets compare both algorithms. One can see that the next cell selected by an agent is the same in both algorithms (step A). Indeed, agents follow the numerical gradient, choosing in the surrounding neighborhood the cell with the minimum value. So agents necessarily choose the one which has not been visited for the longest time.

Concerning the numerical fields q and s built by the algorithms, they both allow to express the elapsed time $\delta t(x)$ since the last visit of a x cell:

$$\begin{aligned} \delta t(x) &= \log(q(x)/Q_{max})/\log(\rho) \quad \text{in EVAP,} \\ \delta t(x) &= t - s(x) \quad \text{in VAW}_0. \end{aligned}$$

It is then possible to express $q(x)$ as a function of $s(x)$ and reciprocally. There is clearly a bijection between the EVAP evaporation function and the VAW$_0$ time function. So, we can freely swap the time computation functions of these two algorithms.

However, it is important to note that, in the multi-agent case, EVAP and VAW$_0$ are not strictly equivalent as steps B and C are not performed in the same order. EVAP agents move and drop pheromones whereas VAW$_0$ agents drop pheromones and move to the next cell. As a consequence, two EVAP agents may only meet on the same cell in very particular topologies. On the contrary, VAW$_0$ agents may find themselves on the same cell more often and then follow each other until some random choice has to be made. This subtle difference leads to a more efficient exploration with EVAP.

We prefer EVAP because it favors exploration, yet VAW$_0$'s time computation function is easier to manipulate. As a result, we propose — and will study — the EVAW algorithm (Exploring VAW) which uses EVAP's order of operations with VAW$_0$'s maths formulae (see Algorithm 4). Note that EVAP and EVAW exhibit identical behaviors for the same initial conditions and the same random seed.

Algorithm 4 EVAW Agent (situated on cell x)

A) Find a cell y in $N(x)$ such that $s(y) = min_{w \in N(x)} s(w)$
in case of multiple choices make a random choice
B) Move to cell y
C) Set $s(y) \leftarrow t$
D) $t = t + 1$

3.3 Known Properties

In [12], Wagner et al. proved that k VAW$_0$ agents cover the environment in bounded time t_k. This proof can be extended to show that the algorithm performs the patrolling task (each cell will be visited at most every t_k time steps). These results are also valid for the EVAW algorithm. As Wagner et al. we have also experimentaly observed that the agents self-organize, so that each of them reaches a stable cycle. A cycle ζ is a finite sequence of adjacent cells that the agent repeatedly covers, some cells possibly appearing several times in the sequence. We are interested in formally studying those cycles. Before considering the multi-agent case in next section, we start by giving a result in the single agent case.

In [11], Wagner et al. present a VAW variant (which we call VAW$_1$) in which ants smell traces made up of a pair (μ, τ) in which μ is the number of visits to the cell so far and τ the last time the cell was visited. Considering a single agent, they proved that, when an Hamiltonian cycle[2] has been reached, the ant repeats it forever. Using the proof schema, we now show the same result for the EVAW algorithm.

We note $s_t(x)$ the value of cell x at time t.

Proof: Assume that ζ is an Hamiltonian cycle denoted by $\zeta(t) = (x_t, x_{t+1}, \ldots, x_{t+n})$ the sequence of $n+1$ consecutive vertices in the tour, starting at x_t. The next tour starts at time $t+n+1$ and only depends on the gradient values along the vertices. So, to prove that the cycle is stable, we have to prove that, for vertices u, v, if it holds that $s_t(u) > s_t(v)$ then $s_{t+n}(u) > s_{t+n}(v)$. This is true as, for all u, $s_{t+n}(u) = s_t(u)+n$.

So if a single Hamiltonian cycle is obtained it remains stable forever. In the next section we study the stability of cycles (Hamiltonian or not) when several agents interact in the same environment.

4 STUDY OF THE MULTI-AGENT CASE

4.1 Introduction

In the multi-agent setting, cycles only interact in pairs so that we will focus on the two-agent case. We suppose for now that both agents (agt_1 and agt_2) remain on their own cycles (ζ_1 and ζ_2, of respective lengths l_1 and l_2). These cycles are neighbors by at least two adjacent cells. We note (c_1, c_2) a couple of adjacent cells such that $c_1 \in \zeta_1$ and $c_2 \in \zeta_2$ (see Fig. 3).

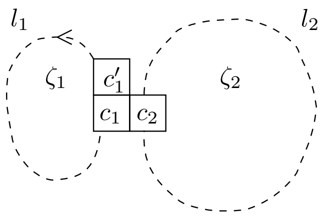

Figure 3. Two cycles of different lengths connecting in cells (c_1, c_2)

We will now show that the obtained cycles can not be stable if they have different lengths, then study the stability of equal length cycles.

4.2 Instability of Cycles of Different Lengths

We suppose $l_1 < l_2$. Each time agt_1 visits c_1, it continues its cycle on cell c'_1 (see Fig. 3). We make the assumption that c'_1 appears only once in the cycle (which is in particular verified in Hamiltonian cycles). As a result, at time t, when agt_1 is in c_1, we have:

$$s_t(c'_1) = s_t(c_1) - l_1 + 1 = t - l_1 + 1. \tag{1}$$

Lemma Under these conditions, two distinct cycles patrolled each by one EVAW agent will not be maintained if they have different lengths.

Proof

If agt_2 breaks its cycle first, the problem is solved. Let us therefore consider that this is not the case and observe what happens for agt_1.

Agent agt_1 goes to cycle ζ_2 (on cell c_2) if and only if it is in cell c_1 at time t and

$$s_t(c'_1) \quad \geq \quad s_t(c_2). \tag{2}$$

This inequality relies on the EVAW agent behavior that ensures it always moves to its minimal neighbor cell. We therefore have to show that inequality (2) will be true in a finite time.

The property that both agents visit c_1 and c_2 alternatively infinitely often would be written:

$$t_2 \leq t_1 \leq t_2 + l_2 \leq t_1 + l_1 \leq \cdots \leq t_2 + k \cdot l_2 \leq t_1 + k \cdot l_1,$$

where t_2 and t_1 are two reference visit dates t_2 and t_1 (agt_2 visiting c_2 just before agt_1 visits c_1). This inequality obviously holds only if $l_1 = l_2$.

Thus, there exist two dates t_1 and t_2 of the visit of agt_1 in c_1 ($s_{t_1}(c_1) = t_1$) and agt_2 in c_2 ($s_{t_2}(c_2) = t_2$) such that

$$t_2 \leq t_1 < t_1 + l_1 < t_2 + l_2.$$

We can then write (using Equation 1):

$$\begin{aligned}
s_{t_1}(c'_1) &= t_1 - l_1 + 1, \\
s_{t_1+l_1}(c'_1) &= (t_1 + l_1) - l_1 + 1 = t_1 + 1, \\
s_{t_1}(c_2) &= s_{t_2}(c_2) = t_2 \quad \text{(because } t_1 < t_2 + l_2\text{) and} \\
s_{t_1+l_1}(c_2) &= s_{t_2}(c_2) = t_2 \quad \text{(because } t_1 + l_1 < t_2 + l_2\text{).}
\end{aligned}$$

Then, at $t_1 + l_1$, we have (using Eq. 2):

$$\begin{aligned}
s_{t_1+l_1}(c'_1) &= t_1 + 1 \\
&> t_2 \\
&= s_{t_1+l_1}(c_2).
\end{aligned}$$

So, agt_1 changes to cycle ζ_2. □

Note that, as we take into account only cell c_2, the previous result does not depend on the direction of agt_2's walk. Another remark concerns the stability of n cycles created by n agents. The stability of the system can only be obtained if cycles have the same length.

4.3 Stability of Equal Length Cycles

From now on we consider that $l_1 = l_2$. Will cycles ζ_1 and ζ_2 be maintained ? We show that some patterns are fixed points and others are not.

Lets start with an illustrated example. Figure 4 presents an environment in which two cycles have emerged, and that will persist, i.e. a fixed point was attained. Such a solution illustrates the emergence of an optimal patrolling with two agents. Fig. 4-b shows step 7 and Fig. 4-c shows step 15 (i.e. after one more turn). One can see that

[2] A cycle is Hamiltonian when each cell is visited exactly once.

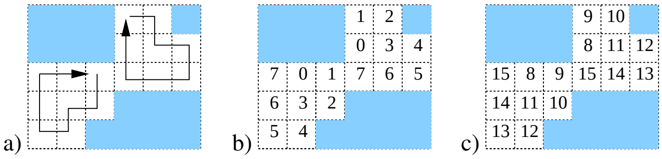

Figure 4. A fixed point composed of two cycles of equal length

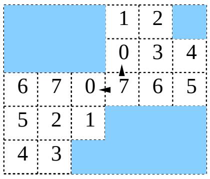

Figure 5. Two cycles of equal length that cannot be maintained

the difference of values between adjacent cells from one cycle to the next remains the same.

We show below that, under defined conditions, when agents converge to distinct cycles of equal length, the cycles will be stable.

Remark When both cycles have the same length, an agent has a choice between two options (see Fig. 5) if and only if it sees not only the tail of its own cycle, but also the tail of the other agent's cycle.

We will try to find out in which situations such a choice is possible by first studying a special case where both cycles are contiguous on half of their length, as depicted on Figures 6-a and 7-a.

In this setting, we will distinguish two cases depending whether both agents run along their boundary in opposite or similar directions.

Agents Going in Opposite Directions — Because the length of the boundary is half the length of their cycles, agt_1 and agt_2 meet each other at some point along this boundary. Then, they can either always end up on a couple of neighbouring cells (c_1, c_2) —so that each remains on its own cycle (see Fig. 6-b)— or they always "miss" each other —so that they both see each other's tail and have the choice to switch cycles or not (see Fig. 6-c)—. As a consequence, the agents have one chance out of two to have stable cycles.

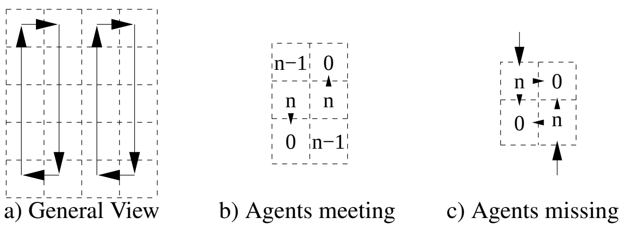

Figure 6. Agents going in opposite directions along their boundary

Agents Going in Similar Directions — Both agents "follow each other". In most cases the distance between agt_1 and agt_2 is different from 1, so that they never see each other's tail (Fig. 7-b) and remain stable. Otherwise, one agent (say agt_1) is in front of the other (agt_2) and may switch to agt_2's cycle which has to find another path to follow (Fig. 7-c).

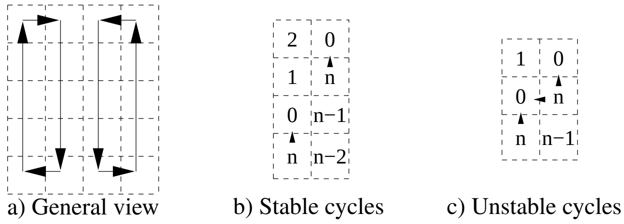

Figure 7. Agents going in similar directions along their boundary

Non Continuous Boundary — The same reasoning can be extended to more complex settings where the boundary is not made of a single segment as in previous examples. Fig. 8-a shows two agents which have reached stable cycles whose boundary is made of five segments.

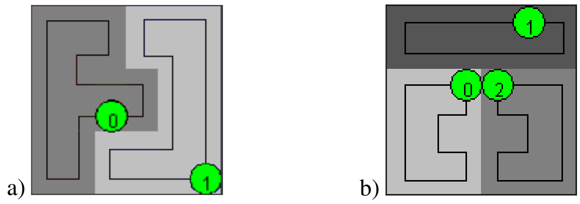

Figure 8. Solutions with a) complex boundaries and b) more than two agents

Beyond Two Agents — The same reasoning can also be extended to more than two agents by considering boundaries in pairs as illustrated in Fig. 8-b.

4.4 Shared Cycles

Up to now, cycles were distincts, meaning that each cell belonged to a single cycle. However, EVAW agents can also reach cycles where some cells are visited by different agents.

Common Cycle — We distinguish a first case where several agents cover a common cycle. Figure 9-a illustrates such a situation. Trivially, both agents describe a cycle with the same length as the other.

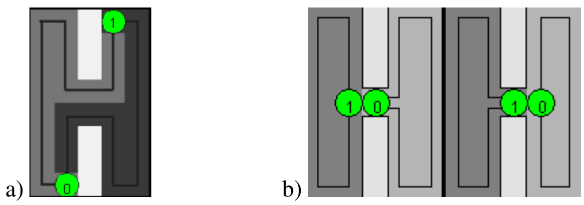

Figure 9. Solutions with a) a common cycle and b) two overlapping cycles

Overlapping Cycles — A second case is that of agents whose cycles share only a subset of their cells. Experimentaly, this case seems to appear more frequently than common cycles. Fig. 9-b gives an example of cycles overlapping on the central cell of the environment.

5 DISCUSSION

We have demonstrated that the obtained cycles can only stabilize if they have the same length. As a consequence the EVAP algorithm ensures a balanced spatial distribution of agents in the environment. Indeed, the average and worst-case idlenesses are minimized, which is a desired property in the context of patrolling.

Wagner et al. [11] asked the question whether VAW_1 — when used with a single agent and in an environment allowing Hamiltonian cycles — can converge to a non-Hamiltonian cycle. Our experiments with EVAW raise the same question as we never found a counter-example. It is interesting to note that, in a multi-agent setting, EVAW may reach suboptimal solutions, when the environment is Hamiltonian (i.e. when it can be covered by a set of non overlapping Hamiltonian sub-cycles). Yet the length of these resulting cycles is always close to the Hamiltonian one. We also observed the formation of optimal or close-to-optimal cycles in non-Hamiltonian environments. In this last case, some agents follow a path that crosses itself in order to extend it and ensure that all cycles have the same length.

Although we have proved that EVAW achieves the patrolling task (agents repeatedly visiting all cells), a theoretical proof that cycles are necessarily obtained is still missing. Furthermore, we plan in future work to study the mechanism leading systematically to an organization in cycles, even if the time to converge to a stable solution is huge. The objective is to possibly improve the algorithm so as to find better solutions or find good solutions faster.

Concerning a real implementation of EVAP and VAW_0, both require that some computational entities be synchronized:

- the "smart cells" in the case of EVAP and
- the mobile robots for VAW_0.

If computations take place if different entities in each algorithm, both rely on digital pheromones —possibly based on sensor networks or future dust sensors— as a shared memory. Patrolling algorithms and pervasive technologies will have to jointly evolve so as to provide a real-world solution to the patrolling problem. Real-world settings will also add constraints such as limited resources, robots avoidance and human-robot interaction. These algorithms should also be considered for offline use: they are known to compare with the state of the art algorithms for finding Hamiltonian cycles in a graph [11].

It has also been shown experimentally in [3] that the number of agents asymptotically increases the performance up to a limit value. Moreover, such an algorithm is robust to perturbations:

- dynamic changes in the graph as studied in [13],
- asynchronicity between the cells or the robots' clocks,
- noisy observations and uncertain actions.

However a number of theoretical questions remain open:

- Will EVAW always self-organize in a set of cycles ?
- Could we compute a complexity bound for cycle formation ?
- If EVAW does not converge to a set of cycles, is the patrolling still guaranteed ?
- Could we bound the average/maximum idleness ?

6 CONCLUSION

In this paper we investigated emergent behaviors occurring in ant-based algorithms defined for the multi-agent patrolling problem. Such theoretical results are still rare in the reactive MAS community. We have presented and compared two similar algorithms: EVAP [3] and VAW_0 [12]. Then we have introduced EVAW for practical reasons, using it both for theoretical and experimental studies. The main novelty of the paper is the theoretical study of the stability of cycles generated by the algorithm. Whereas Wagner et al. only considered Hamiltonian cycles in a mono-agent setting, we proved that, in the multi-agent case, only cycles of same lengths can persist as limit cycles. Then we identified patterns that ensure that several cycles with same length will remain stable forever. We also presented and discussed different spatial self-organisations.

In future work, we plan to generalize our results and continue the theoretical study of the emergent behaviors of EVAW. In particular, we want to go deeper in the analysis of the mechanisms underlying cycles formation. We plan also to work on experimental and theoretical bounds of algorithm complexity. Concerning applications, we are currently experimenting this algorithm with simulated drones involved in military base surveillance (SMAART DGA project).

REFERENCES

[1] A. L. Almeida, P. M. Castro, T. R. Menezes, and G. L. Ramalho, 'Combining idleness and distance to design heuristic agents for the patrolling task', in *II Brazilian Workshop in Games and Digital Entertainment*, pp. 33–40, (2003).

[2] R. Beckers, O.E. Holland, and J.-L. Deneubourg, 'From local actions to global tasks: stigmergy and collective robotics', in *Artificial Life IV: Proc. of the 4th Int. Workshop on the synthesis and the simulation of living systems, MIT Press*, (1994).

[3] H. Chu, A. Glad, O. Simonin, F. Sempe, A. Drogoul, and F. Charpillet, 'Swarm approaches for the patrolling problem, information propagation vs. pheromone evaporation', in *ICTAI'07 IEEE International Conference on Tools with Artificial Intelligence*, pp. 442–449, (2007).

[4] A. Colorni, M. Dorigo, and V. Maniezzo, 'Distributed optimization by ant colonies', in *in proceedings of ECAL91, European Conference on Artificial Life*, pp. 134–142, Paris, (1991). Elsevier.

[5] A. Drogoul and J. Ferber, 'From tom thumb to the dockers: Some experiments with foraging robots', in *2nd Int. Conf. On Simulation of Adaptative Behaviors*, pp. 451–459, Honolulu, (1992).

[6] T. H. Labella, M. Dorigo, and J-L. Deneubourg, 'Division of labor in a group of robots inspired by ant's foraging behavior', *ACM Transactions on Autonomous and Adaptive Systems*, **1**, 4–25, (2006).

[7] F. Lauri and F. Charpillet, 'Ant colony optimization applied to the multi-agent patrolling problem', in *IEEE Swarm Intelligence Symposium*, (2006).

[8] A. Machado, G. Ramalho, J-D. Zucker, and A. Drogoul, 'Multi-agent patrolling: an empirical analysis of alternative architectures', in *Third International Workshop on Multi-Agent Based Simulation*, pp. 155–170, (2002).

[9] J. A. Sauter, R. Matthews, H. V. D. Parunak, and S. Brueckner, 'Evolving adaptive pheromone path planning mechanisms', in *Proc. of AAMAS'02*, pp. 434–440, (2002).

[10] J. A. Sauter, R. Matthews, H. V. D. Parunak, and S. Brueckner, 'Performance of digital pheromones for swarming vehicle control', in *Proc. of AAMAS'05*, pp. 903–910, (2005).

[11] I. Wagner and A. Bruckstein, 'Hamiltonian(t) - an ant-inspired heuristic for recognizing hamiltonian graphs', in *Ant-Algorithms Session in CEC'99 International Joint Conference on Neural Networks*, (1999).

[12] I. Wagner, M. Lindenbaum, and A. Bruckstein, 'Distributed covering by ant-robots using evaporating traces', *IEEE Transactions on Robotics and Automation*, **15**, 918–933, (1999).

[13] I. Wagner, M. Lindenbaum, and A. Bruckstein, 'Ants agents networks trees and subgraphs', *Future Generation Computer Systems Journal*, **16(8)**, 915–926, (2000).