Recent Advances on Multi-Agent Patrolling

Alessandro Almeida[1], Geber Ramalho[1], Hugo Santana[1], Patrícia Tedesco[1], Talita Menezes[1] Vincent Corruble[2], and Yann Chevaleyre[2]

[1]Universidade Federal de Pernambuco
Centro de Informática
Cx. Postal 7851
50732-970,Recife – PE Brasil
Phone: +55 8132718430
E-mail: {all, glr, hps, pcart, trm}@cin.ufpe.br

[2]Université Paris 6
Laboratoire d'Informatique de Paris VI
Boîte 169 – 4 Place Jussieu75252
PARIS CEDEX 05
E-mail: {Vincent.Corruble, Yann.Chevaleyre}@lip6.fr

Abstract. Patrolling is a complex multi-agent task, which usually requires agents to coordinate their decision-making in order to achieve optimal performance of the group as a whole. In previous work, many patrolling strategies have been developed, based on different approaches: heuristic agents, negotiation mechanisms, reinforcement learning techniques, techniques based on graph-theory and others. In this paper, we complement these studies by comparing all the approaches developed so far for this domain, using patrolling problem instances that were not dealt with before (i.e. new map topologies). The final results constitute a valuable benchmark for this domain, as well as a survey of the strategies developed so far for this task.

Keywords: autonomous agents and multi-agent systems, coordination and patrolling

1. Introduction

Patrolling is the act of walking around an area in order to protect or supervise it [1]. Informally, a good patrolling strategy is one that minimizes the time lag between two visits to the same location. Patrolling can be useful for domains where distributed surveillance, inspection or control is required. For instance, patrolling agents can help administrators in the surveillance of failures or specific situations in an Intranet [2] or detect recently modified or new web pages to be indexed by search engines [3]. Patrolling can be performed by multiple agents, requiring coordinated behavior and decision-making in order to achieve optimal global performance.

Despite its high potential utility and scientific interest, only recently multi-agent patrolling has been rigorously studied. In the context of an international AI project, we initiated pioneering work on this issue, proposing different agent architectures and empirically evaluating them [4, 5, 6]. In this paper, we put together, summarize and compare the recent advances, pointing out the benefits and disadvantages of each one, as well as the situations where they are most appropriate. This kind of comparative study of different AI paradigms for multi-agent patrolling has not been done so far. Moreover, in order to provide a more precise and richer comparison of the current approaches, we introduce novel patrolling scenarios corresponding to some environment topologies, which are thought to be representative instances of the task.

Section 2 of this paper describes the patrolling task; section 3 presents various approaches to this task. The experimental results and comparative study are described in section 4. Finally, section 5 presents conclusions and discusses future research.

2. The Patrolling Task

In order to obtain a generic, yet precise, definition of the patrolling task, we represent the terrain being patrolled as a graph, where the nodes correspond to specific locations and the edges to possible paths, as illustrated in Fig. 3~~Fig. 3~~. The main advantage of adopting this abstract representation is that it can be easily mapped to many different domains, from terrains to computer networks. Given a graph, the patrolling task studied in the remainder of this paper consists in continuously visiting its nodes.

One of the contributions of our previous work [4] was the identification of *evaluation criteria* for patrolling, since no patrolling related works [7, 8] have proposed performance measures. Considering that a cycle is a simulation step, the *instantaneous node idleness* (or simply idleness) for a node n at a cycle t is the number of cycles elapsed since the last visit before t (i.e. the number of cycles the node n remained unvisited). The *instantaneous graph idleness* is the average instantaneous idleness over all nodes in a given cycle. Finally, the *average idleness* is the mean of the instantaneous graph idleness over a t-cycle simulation. In the same context, another interesting measure is the *worst idleness*: the highest value of the instantaneous node idleness encountered during the whole simulation. In order to better evaluate the contribution of each agent when the population size varies, the idleness can be *normalized* by multiplying the absolute value by the number of agents divided the number of nodes in the graph.

There are some variations in the patrolling task. For instance, the terrain may contain mobile obstacles or assign different priorities to some regions. In some cases, typically military simulations, agent communication is forbidden.

3. Survey on Current Approaches

In this section, we give an overview of the approaches taken by our research groups in the investigation of the patrolling task. The current approaches for multi-agent patrolling are based on three different techniques: (1) *Operations research algorithms*, in particular those used for the Traveling Salesman Problem (TSP); (2) Non-learning Multi-Agent Systems (MAS), involving classic techniques for agent coordination and design; and (3) Multi-Agent Learning which use Reinforcement Learning allows the agents to continuously adapt their patrolling strategies. In this work we will not compare our findings to the results obtained in robot patrolling [9], since the latter results were rather concerned with dealing with challenges inherent to the complexity of the real world, such as the necessity for robots to recharge.

3.1 Pioneer Agents

The pioneer work on the patrolling, as formulated above, has been done by Machado *et al.* [4]. In their article, they proposed several multi-agent architectures varying parameters such as agent type (reactive vs. cognitive), agent communication (allowed vs. forbidden), coordination scheme (central and explicit vs. emergent), agent perception (local vs. global), decision-making (random selection vs. goal-oriented selection). The choice of the next node to visit is basically influenced by two factors: (1) node idleness, which can shared (generated by all agents) vs. individual (corresponding a single agent visits); (2) field of vision, which can be local (agent's neighborhood) or global (entire graph). The experiments they conducted showed two interesting facts: first, agents moving randomly achieved very bad results. Second, agents with absolutely no communication ability which strategies consisted in moving towards the node with the highest idleness performed nearly as well as the most complex algorithm they implemented.

In the experiments (Section 4), we will the compare the most efficient architecture of each approach with the following two agents: (a) *Conscientious Reactive*, whose next node to visit is the one with the highest individual idleness from its neighborhood; (b) *Cognitive Coordinated,* whose next node to visit is the one with the highest shared idleness from the whole graph, according to the suggestions given by a central coordinator. This coordinator is responsible for avoiding that more than one agent chooses the same next node.

3.2 TSP-Based Single-cycle Approach

Consider a single agent patrolling over an area. The simplest strategy which comes to mind would be to find a cycle covering all the area, and then to make the agent travel around this cycle over and over. In our case, in which areas are represented by nodes in a graph, the cycle is defined as a closed-path starting and ending on the same node and covering all nodes possibly more than once. It was shown in [6, 10] that, for a single agent, the optimal strategy in terms of worst idleness is this cyclic-based one, where each cycle is calculated as the optimal solution for the Traveling Salesman Problem [6, 10]. The idea is then to reuse all the classic and solid results from the operations research and graph theory in determining the cycle.

One way to extend single-agent cyclic strategy to the multi-agent case is to arrange agents on the same closed-path such that, when they start moving through it, all in the same direction, they keep an approximately constant gap between them. Although this multi-agent extension is not optimal, it can be shown that a group of agents performing this strategy can achieve a performance, according to the worst idleness criterion, which is lower or equal to 3 x *opt* + 4 x $\max_{ij}\{c_{ij}\}$, where *opt* is the optimal value for the worst idleness and c_{ij} is the length of the edge connecting nodes i and j. Fig. 1~~Fig. 1~~ illustrates two agents patrolling according to this TSP single-cycle approach in two different graphs. It is expected that this strategy will perform better in topologies like (a) than in (b), since the worst idleness is influenced by the longest edge (c_{ij}).

We call *Single-Cycle* the agents following this TSP single-cycle strategy. They will be compared with other agents in Section 4.

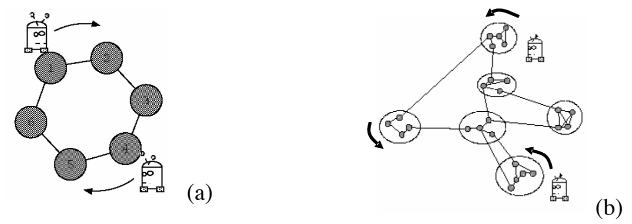

Fig. 1 – TSP Single-cycle patrolling strategy

3.3 Heuristic Agents

In the pioneer work [4], a cognitive agent reaches a node using the shortest path. However, the average idleness of the graph would be lower if the agent had taken longer paths, which passed through idler nodes instead. Suppose an agent is located in the node ***v*** (**Fig. 3**~~**Fig. 2**~~), and has chosen node ***w*** as objective. The shortest path is *v-a-w* (length of 100), which passes through node *a*, whose idleness is 10. However, the path *v-b-w*, which is just slightly longer (length of 101), passes through the idler node *b* (with idleness 99). This second path, which is much idler, should be chosen. That is the idea behind the *Pathfinder Agents* [5].

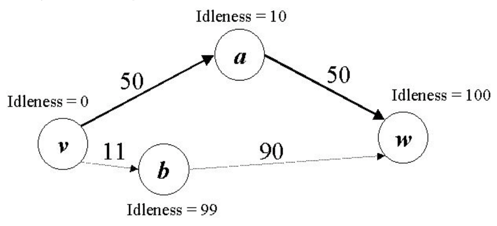

Fig. 3~~2~~ – Example of idleness vs. distance dilemma

Besides improving the path-finding technique, we have also enhanced the agent's decision making. In previous work [5], the next-node choice criteria could be either random or based on node idleness, where an agent chose the node with the highest idleness. However, if the node with the highest idleness was too far from the agent, it would be better to choose a nearer node with high idleness. In other words, the choice of the goal node should be guided by a utility function that takes into account the cost (distance) and the reward (idleness). Imagine an agent located at node ***v*.** In this case, the node with the highest idleness (***w*** with idleness 100) is relatively far from the agent (distance of 100). However, ***b*** (with idleness of 99) is much nearer (distance of 11). Node ***b*** should be chosen. This is the idea behind the *Heuristic Agents* [5]. These new variations of the decision-making process and the path-finding were applied to the previous agents [5].

In the experiments section we will compare the other agents with the *Heuristic Pathfinder Cognitive Coordinated*, which uses both advanced decision-making and path-finding, as well as the guidance of a central coordinator.

3.4 Negotiation Mechanisms for Patrolling

We removed the communication restrictions of the previous works [4], allowing the patrolling agents to exchange messages freely. These agents may need to interact to reach their objectives, either because they do not possess enough ability or resources to solve their problem, or because there are interdependences (both of resources and of results) between them [11]. In this approach, negotiations have been implemented as auctions, market-based [11]. They are simple interaction scenarios, which makes them a good choice for reaching agreements [12].

In this light, a typical patrolling scenario has each agent randomically receiving graph nodes to patrol. Each agent tries to get a set of nodes very near to each other (to minimize the gap between two visits, and increase the agent's/patroller's utility). When the agent detects a node that it cannot visit within a reasonable amount of time, it initiates an auction. The other agents (bidders) then check whether there is a node in their own set that can be traded by the offered one. If such a node is found, the corresponding bidder offers it as a bid. The auctioneer then chooses the best bid (i.e. the node that is nearest from the others in its set) and makes the deal with the bidder.

Following the incremental process suggested in [4], we developed new agents whose utility function only considers the distance between nodes. The main diference between them is the way the agents perform their auction. In our work, auctions are either one or two shot, private value and sealed-bid. We have investigated various market-based MAS architectures, the most eficient are detailed below:

1. Two-Shot-Bidder Agent (TSBA) – only bids for nodes that increase its utility function and it promotes a two-shot auction, and thus increases the chances of getting better offers.
2. Mediated Trade Bidder Agent (MTBA) – in this architecture, there is a broker agent, which informs the auctioneer which agents can be good bidders and which nodes they can trade.

We then enhanced these agents with the decision-making process for choosing the next node and the path finding algorithm presented in 3.3. This has resulted in the following architectures: *Heuristic Cooperative Bidder, Heuristic Mediated Trade Bidder* and *Heuristic Two-shots*. We have also developed the *Heuristic Pathfinder Cooperative Pathfinder Bidder, Heuristic Pathfinder Mediated Trade Bidder*, and *Heuristic Pathfinder Two-shots Bidder*. Section 4 shows the results obtained with these most efficient architectures.

3.5 Reinforcement Learning in Patrolling

Reinforcement learning (RL) is often characterized as the problem of learning how to map situations to actions, in order to maximize some numerical rewards [13]. In [14], we developed adaptive agents that learn to patrol using RL techniques. In order to enable these agents to adapt their patrolling strategies automatically via RL, we mapped the patrolling problem into the standard RL framework, namely the Markov Decision Process (MDP) formalism [13]. This consisted in defining a state and action space for each agent individually, and in developing proper models of instantaneous rewards which could lead to a satisfactory long term performance. Then, a standard RL algorithm, Q-Learning [13], was used to train individual agents.

Two different agent architectures were considered in this study: the Black-Box Learner Agent (BBLA), in which the agents do not communicate explicitly, and the Gray-Box Learner Agent (GBLA), in which the agents may communicate their intentions for future actions. The action space is the set of actions that let the agent navigate between adjacent nodes in the graph. For the BBLA, the state space is represented by: a) the node where the agent is located; b) the edge where it came from; c) the neighbor node which has the highest idleness; d) the neighbor node which has the lowest idleness. The GBLA has the same state representation, plus the information from the communication of intentions: e) the neighbor nodes which other agents intend to visit. These simple state definitions (comprising 10.000 and 200.000 possible states for BBLA and GBLA respectively for the maps studied) are feasible even with traditional Q-Learning implementations.

A function for the instantaneous reward, which showed to be particularly well suited for the average idleness criterion, was the idleness of the node visited by the agent. We showed that this reward model implements the concept of *selfish utility*, and we compared it with other utility models, such as the *team-game utility* and the *wonderful life utility* [14], that try to take into account indication of the global (systemic) performance. The selfish model, combined with the GBLA architecture, showed the best performance among the RL architectures so far.

In the experiments section we will compare *GBLA* with the other agents.

4. Experimental Results

This section presents the methodology used in order to make possible comparisons between the best MAS of each subsection of section 3. It shows the set-up of the experiments, the proposed scenarios, the obtained results and a discussion about them.

4.1 Experimental Set-up

All experiments were carried out using same simulator, previously developed by our research groups. The experiments were performed in six maps, which are shown in Fig 2. Black blocks in the maps represent obstacles. White blocks are the nodes to be visited. Possible paths in the graphs are shown in blue. The best architectures of each approach, indicated in the last paragraph of sections 3.1 to 3.5, were compared for each Map. Each MAS was simulated with populations of five and fifteen agents. For each pair (map, architecture) we ran 10 experiments. At each experiment, all the agents start at the same node, but this starting node varies among the 10 experiments.

Regarding the problem of choosing the appropriate number of cycles for a simulation, a reasonable approach is to visit each node k times. In the worst case, a node will stay WI iterations without being visited, where WI is the worst idleness of the simulation. Hence, each simulation run for 15000 cycles, 15 times the experimentally observed WI (1000).

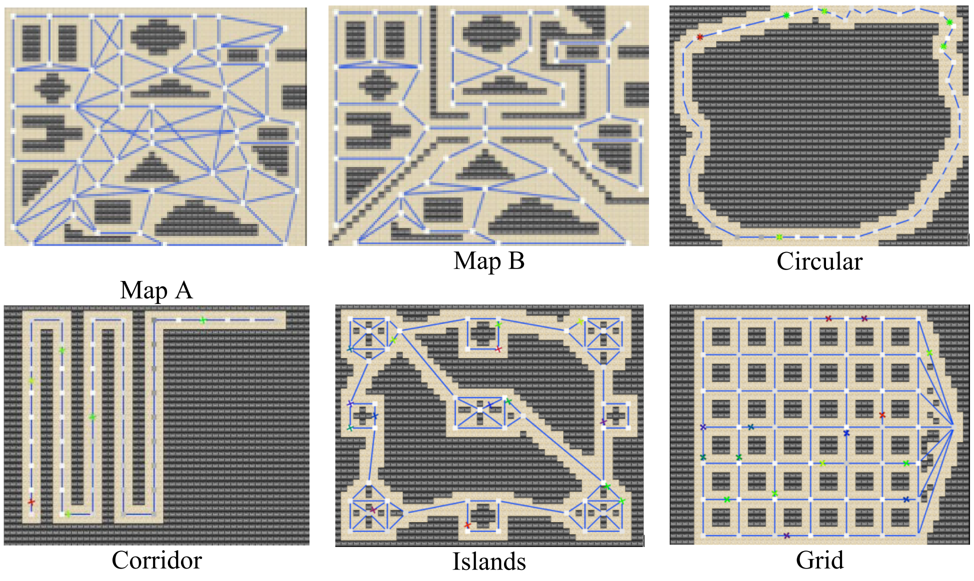

Fig. 3 - Maps with different corresponding graph topologies on our simulator

4.2 Results

Fig. 6~~Fig. 4~~ and Fig. 8~~Fig. 5~~ show the results for all maps and agent architectures, for populations of 5 and 15 agents, respectively. The y-axis indicates the average graph idleness: the smallest the idleness, the better is the performance of the architecture.

It is clear from the graphics that Single Cycle (SC) has the best performance for all cases, except in the islands map with 15 agents (where Heuristic Pathfinder Cognitive Coordinated (HPCC) wins) and in the corridor with 15 agents (where there is a draw). On the other extreme, the pioneer agents, Conscientious Reactive (CR) and Cognitive Coordinated (CC), have the worst performances. Regarding the remaining architectures, for 5-agent population and all graphs, HPCC and Gray-Box Learner Agent (GBLA) have the second best performance (followed by Heuristic Pathfinder Two-shots Bidder (HPTB). For 15 agents and all graphs, HPCC, HPTB and GBLA have an equivalent performance.

It is also interesting to analyze the influence of graph topologies and population size on architectures' performance. To understand these points, Fig. 10~~Fig. 6~~ shows the results using the normalized average idleness (Cf. Section 2), as well as its standard deviation. These results are the mean of the results of all maps.

With respect to graph topology influence, SC, GBLA and HPCC exhibit the best, and equivalent, performance variation (the standard deviation varies from 0.77 to 0.99). The negotiating agents' performance varies more according to the population size: HPTB has a standard deviation of 1.26 for 5, and Heuristic Pathfinder Mediated Trade Bidder (HPMB), 1.63 for 15 agents. CC and CG are still the worst architectures on this standpoint (the standard deviations go from 1.79 to 10.24). Finally, regarding population variation, SC and GBLA exhibit the most stable behavior, i.e., the same performance no matter the number of agents. The others vary their performance (for the best or the worse): adding agents to the population improves the normalized performance, but retracting agents, worsens it.

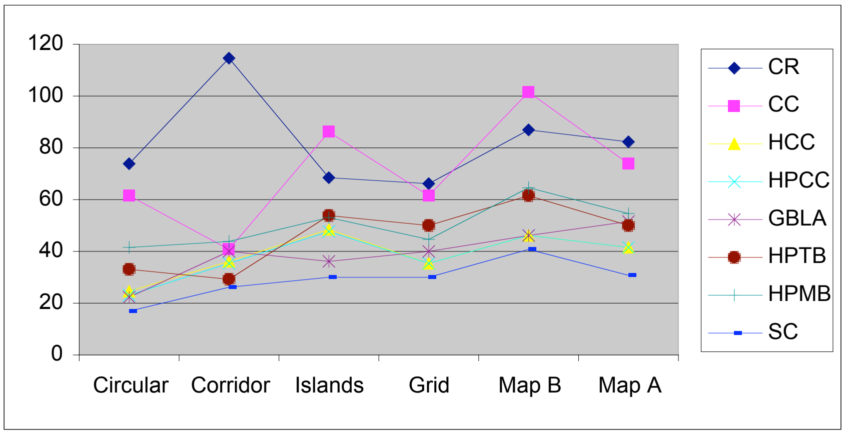

Fig. 64 - Results for a population of five agents (where CR = Conscientious Reactive, CC = Cognitive Coordinated, HPCC = Heuristic Cognitive Coordinated, GBLA = Gray-Box Learner Agent, HPTB = Heuristic Pathfinder Two-shots Bidder, HPMB = Heuristic Pathfinder Mediated Trade Bidder, SC = Single-Cycle)

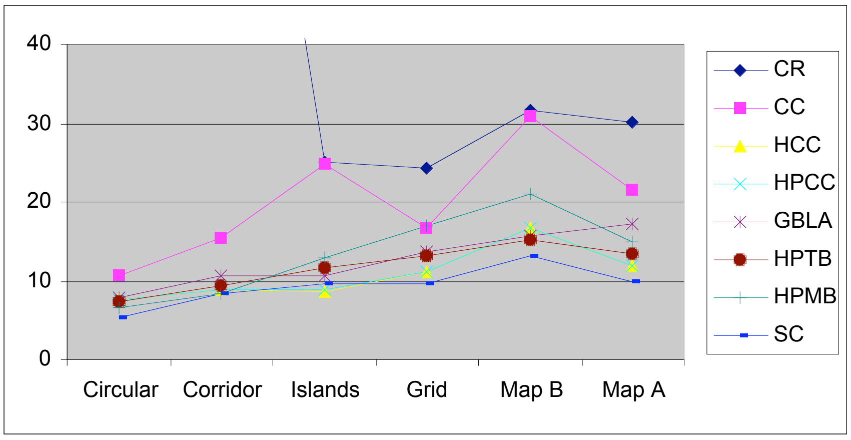

Fig. 8~~5~~ - Results for a population of fifteen agents. For sake of readability, we have hidden the first two results of CR, which were 73,71 and 107,01, respectively

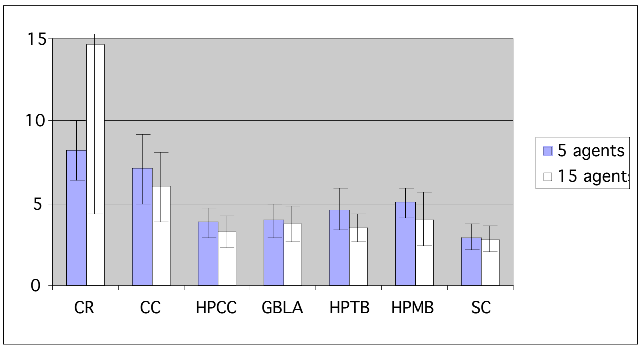

Fig. 10~~6~~ - Normalized average idleness with their standard deviations. These results are the mean of all maps

4.3 Discussion

The most striking finding is the outstanding performance of SC for the three discussed points of view (the only exception occurred in the island map, as theoretically expected, since this graph contains long edges). This excellent performance can be explained by its "non-spontaneous" coordination scheme (one agent following another in an optimum line and maintaining an equal distance between them), which is intuitively very effective. However, since this scheme is centralized, predefined and fixed, this kind of architecture will have problems in some situations such as: dynamic environment, as shown in the case of robots needing to recharge [9]; huge graphs (thousands of nodes), because of the TSP complexity; and patrolling task where different regions have different patrolling priorities.

Another interesting finding is the bad performance of the pioneer agents (CR and CC), indicating that without more sophisticated architectures it is not possible to tackle such a complex problem as patrolling.

Concerning the other agents, HPCC architecture exhibits the overall second best performance. It is more adaptive than SC, but it needs tuning for each situation (map and population), to balance the weight of distance and idleness in path finding and decision-making. Like SC, HPCC has also limitations in huge graphs because of pathfinding complexity. GBLA, which has the third overall best performance (even slightly loosing for HPTB with 15 agents), shows a great adaptation capacity, even when the population size varies. Moreover, it is the only architecture using local information, which is plausible scenario for some applications. The drawback of GBLA is its training cost, which can be prohibitive in huge graphs. The negotiating agents (HPTB and HPMB) do not show such a good performance, except in corridor and circular maps. However, they do not suffer from the problems mentioned above.

5. Conclusions and Further Work

Despite its potential usefulness and scientific interest, patrolling has begun to be addressed only fairly recently. In particular, our research groups had been working on this task, using different approaches [4, 5, 6, 10, 13], but had yet compared them in a richer benchmark. This article presents an overview of the recent advances in patrolling and it introduces an original comparative study among them, pointing out benefits and disadvantages of each one. Such comparison serves not only as a benchmark and guideline for new multi-agent patrolling approaches, but also for improving the current ones.

In the near future, we intend to deepen our comparative study, finding better explanations for the current architecture behaviors. We also plan to introduce new scenarios and MAS architectures, in particular more dynamic scenarios, huge graphs, and graphs containing regions with different priorities.

6. References

1 Abate, F.R. The Oxford Dictionary and Thesaurus: The Ultimate Language Reference for American Readers, Oxford Univ. Press 1996.

2 Andrade, R.d.C, Macedo, H.T., Ramalho, G.L., Ferraz, C.A.G: Distributed Mobile Autonomous Agents in Network Management, Proceedings of International Conference on Parallel and Distributed Processing Techniques and Applications (2001).

3 Cho, J. Garcia-Molina, H.: Synchronizing a database to Improve Freshness, In Proceedings of 2000 ACM International Conference on Management of Data (SIGMOD) (2000).

4 Machado, A., Ramalho, G., Zucker, J.D., Drogoul, A.: Multi-Agent Patrolling: an Empirical Analysis of Alternative Architectures, Multi-Agent Based Simulation (MABS'2002) (2002), 155-170.

5 Almeida, A. Patrulhamento Multi-Agente em Grafos com Pesos. MSc. dissertation. Universidade Federal de Pernambuco, 2003.

6 Chevaleyre, Y.: The patrolling problem. Tech. Rep. of Univ. Paris 9. (2003) available at http://l1.lamsade.dauphine.fr/~chevaley/patrol
7 Pottinger and C. Dave. in Game Developer, Vol. January, pp. 42-51 1999.
8 Pottinger and C. Dave. in Game Developer, Vol. February, pp. 48-58 1999.
9 Sempé, F. Auto-organisation d'une collectivité de robots: application à l'activité de patrouille en présence de perturbations. PhD Thesis. Université Paris VI. 2004.
10 Chevaleyre, Y. & Sempé, F. "A theoretical analysis of multi-agent patrolling strategies". Submitted to the International Conference on Autonomous Agents and Multi-Agents Systems (AAMAS) (2004).
11 Faratin, P., Sierra, C., Jennings, N. "Using Similarity Criteria to Make Issue Trade-Offs in Automated Negotiations". Artificial Intelligence Journal, 2, (2002), 205-237. available at: http://citeseer.nj.nec.com/faratin02using.html.
12 Faratin, P., Sierra, C., Jennings, N.: "Negotiation Decisions Functions for Autonomous Agents". International Journal of Robotics and Autonomous Systems, 24, (1998), 159-182. available at: http://citeseer.nj.nec.com/faratin98negotiation.html
13 Sutton, R., Barto, A. Reinforcement Learning: An Introduction. Cambridge, 1998
14 Santana, H., Ramalho, G., Corruble, V., Ratitch, B. “Multi-Agent Patrolling with Reinforcement Learning”. Submitted to the International Conference on Autonomous Agents and Multi-Agents Systems (AAMAS) (2004).