SimPatrol: Establishing a Testbed for Multi-agent Patrolling

Daniel H. Moreira Luiz J. da Silva Filho Patrícia R.Tedesco Geber L. Ramalho

Universidade Federal de Pernambuco (UFPE), Centro de Informática (CIn)
Programa de Pós-graduação em Ciência da Computação
Avenida Professor Moraes Rego, 1235. 50732970 – Recife, PE – Brazil

Abstract

Multi-agent patrolling is a recurrent task in many game styles. It has been studied but it is not yet easy or accurate to perform comparisons of the proposed solutions. Having this in mind, this poster describes multi-agent patrolling as a benchmark for AI, and it presents a testbed to parameterize such benchmark. Concretely, this testbed is represented by *SimPatrol*, a simulator of multi-agent systems constructed to support the analysis of the performance of the various proposed patrolling strategies.

Keywords: multi-agent patrolling, benchmark, testbed, *SimPatrol*

Authors' contact:
{dhm, ljsf2, pcart, glr}@cin.ufpe.br

1. Introduction

As said by Almeida et al. [2004], multi-agent patrolling has been rigorously studied since 2002. Informally, patrolling is the act of walking around an area in order to protect or supervise it, having the goal to minimize the time lag between two visits to a same location.

One could ask the reasons to study such subject. Focusing on games, multi-agent patrolling is useful in game AI mechanisms related to tasks like detecting mobile (enemy) characters, discovering new items, detecting new enemy buildings and protecting cities and resources. So, it is present in strategy games, role playing games, combat games, first person shooters, arcade games, simulators and sport games.

Considering the recurrence of multi-agent patrolling in various game styles, a developer having to apply it in different situations (from games with fog of war – where parts of the patrolled territory are not visible in the beginning – to real-time strategy games with very dynamic maps, for example) would face problems to compare distinct strategies, as well as reuse old ones, due to the lack of well-defined problem-related parameters and supporting tools.

Considering the concept of benchmarks presented by Drogoul et al. [2007], multi-agent patrolling can be enunciated as one of them, leading to the identification of the necessity of a testbed (i.e. an interface to specify parameters of the benchmark [Hanks et al. 1993]), in order to facilitate the comparison among the existing patrolling strategies [Machado 2002; Almeida 2003; Santana 2004; Chevaleyre 2005; Menezes 2006] and the eventual new ones.

Having this all in mind, this short paper aims to introduce *SimPatrol*, a software simulator resulting from efforts to establish a testbed for the patrolling task. So, it is organized as follows: next section formally defines multi-agent patrolling, while section 3 describes it as a benchmark. After that, section 4 brings some critics to the previous works on the topic, being followed by section 5 that presents *SimPatrol*. Finally, conclusions are discussed.

2. Multi-agent Patrolling

Computationally, multi-agent patrolling is stated as a multi-agent system of which environment is represented by a graph and of which agents act as tokens moving on it, visiting the nodes through the edges. So, the objective is to minimize the time lag between two visits for each node.

Almeida [2003] defined the metrics used to evaluate the proposed patrolling strategies. The most fundamental one (used to obtain all the others) is the concept of *instantaneous idleness of a node n* (i_n), that – for a specific node – measures the time elapsed since the last visit until the current moment. Directly based on this metric, there is the *mean instantaneous idleness of the graph* (i), i.e. the mean of the instantaneous idleness of all the nodes at the current moment, and also there is the *maximum instantaneous idleness of the graph* (**max** i), that is just the highest instantaneous idleness at the current moment, once more considering all the nodes.

Additionally, collecting the mean instantaneous idlenesses of the graph at a specific time rate permits the calculus of the *mean idleness of the graph* (I), which is the mean value of all such collected values until now. Similarly, one can measure the *maximum idleness of the graph* (**max** I), that simply stated is the highest instantaneous idleness ever found in the graph.

Historically, perceiving that the patrolling task is a situation that requires coordinated behavior and decision making among the patrollers, Machado [2002] started the study of the subject as a multi-agent system. Since then, a series of works were realized. While some of them tested empirically various distinct

techniques (like reactive behavior [Almeida 2003], machine learning [Santana 2004] and negotiation mechanisms [Menezes 2006]), Chevaleyre [2005] presented a theoretical analysis of the topic. Add to that, two works were published in order to compare the various proposed patrolling techniques [Almeida et al. 2004; Chevaleyre 2004]. In both cases, the proposed strategies were compared based on the performance of the patroller agents when submitted to each one of the six maps described by Santana [2004] (actually, an effort to represent the topologies he found to be the most possible to face in real problems).

Despite the quantity of already proposed solutions, multi-agent patrolling was not analyzed as a benchmark for AI yet. If done so, its popularity would increase (more AI researchers would enjoy the topic) and as a natural consequence, more new techniques and related tools would be developed. Certainly, among the benefited ones would be the game programmers that recurrently have to implement, test and improve patrolling strategies when attending their developing duties.

This way, a natural question is: can multi-agent patrolling be described as a benchmark?

3. Can Multi-agent Patrolling be a Benchmark for AI?

As noticed by Drogoul et al. [2007], Computer Science inherited from the industrialization process the need to evaluate the performance of distinct logic systems (algorithms, architectures, etc.) applied to a same problem, in order to identify which ones are the best, and specifically in what situations. Informally, such problem can be called benchmark.

Focusing on the AI discipline, benchmark is every problem sufficiently generic in order to be solved by plenty of distinct techniques, sufficiently specific to let such distinct techniques be compared, and sufficiently representative of a class of real applied problems [Drogoul et al. 2007]. Examples of AI benchmarks involve the *Towers of Hanoi*, the *Traveler Salesman Problem*, the *N-queens Problem*, the *Blocksworld*, etc.

Considering the concept of benchmarks given by the AI discipline, one natural question is if multi-agent patrolling is one of them. So, the following questions must be answered:

- Is multi-agent patrolling sufficiently generic?
- Is multi-agent patrolling sufficiently specific?
- Is multi-agent patrolling sufficiently representative?

The answer for the first question is proven positive by the number of existing distinct techniques developed to solve the patrolling problem, despite the relative newness of the theme [Machado 2002; Almeida 2003; Santana 2004; Chevaleyre 2005; Menezes 2006]. Actually, more efforts have been engaged by the AI research group of UFPE in order to obtain new methods of patrolling, using ant colony optimization and graph partition among the patroller agents, for example.

Taking the second question into consideration, the answer is again positive. Multi-agent patrolling is specific enough in order to let the distinct proposed techniques be compared; otherwise the comparing works of Almeida et al. [2004] and Chevaleyre [2004] could not be done. Due to the well defined metrics first presented in the work of Almeida [2003] (see section 2) and to the six maps proposed by Santana [2004] as an effort to represent the most usual topologies of territories to be patrolled, the results obtained from the various techniques can be measured in a deterministic way, letting the establishment of a total order among the collected values.

Finally, having in mind the third question, multi-agent patrolling can be considered sufficiently representative, given the diversity of possible applications. Besides the field of games, Almeida et al. [2004] noticed that it is useful for every domain where distributed surveillance, inspection or control is required.

Answered these questions, one relevant detail is how good multi-agent patrolling is, as a benchmark. Drogoul et al. [2007] defines a good benchmark as the one that turns the representation and the understanding of new methods easier, letting the researcher focus on the solution rather than on the representation of the problem. The patrolling task reaches it into two ways:

1. The territories to be patrolled are represented by generic graphs. So, when a virtual patroller agent visits the nodes of a graph, it can be the night watchman visiting the rooms of a museum, or maybe a bot inspecting the links of an intranet, or perhaps an adventurer hunting ghosts in a haunted house.

2. The collected metrics are all based on the time lag between two visits in a same node (see section 2). So, it does not matter how the agents decided the order and the exact time to visit the nodes; to impact on the collection of the metrics, all they have to do is to visit the nodes.

Said this all, some criticism related with the previous works of multi-agent patrolling are made in the next section.

4. Criticism to the Previous Works of Multi-agent Patrolling

One source of criticism to the previous works of multi-agent patrolling is related to the features that were not taken into consideration by researchers yet, due to the

newness of the topic. Aspects like graphs of which nodes have distinct priorities of visiting (like it can occur in the patrolling of a bank, where the room of the main safe must be visited more frequently than the others), or of which nodes and edges can become unavailable in simulation time were not considered in the first proposed patrolling strategies [Machado 2002; Almeida 2003; Santana 2004; Chevaleyre 2005; Menezes 2006]. In the same way, open societies of agents (i.e. societies of which agents can be born or die in simulation time) and the possibility to associate energetic costs to the actions and perceptions of the patrollers were not explored.

Actually, one of the most important features not yet analyzed is related to the counting of time. Taking into consideration the comparisons done among the various proposed patrolling strategies [Almeida et al. 2004; Chevaleyre 2004], all of them were experimented in simulators that counted the time of simulation based on the agents' perceive-think-act cycle. So, when researchers simulated their strategies in 1000 cycles (for example), they were actually letting their agents to perceive, think and act 1000 times. Similarly, the metrics of idleness were calculated based on such numbers, what means that in the worst case, the maximum possible idleness for the graph being patrolled was 1000. Due to the diversity of applied techniques and reasoning mechanisms, it was perceived that such method of time simulation represents a problem to the comparing studies, as it does not take into consideration the real time spent by the agents to make their decisions of their next actions. So, if a strategy really chooses the best actions but, as a consequence, spend too much time deciding, it will not be penalized in any form.

Another source of criticism is the lack of an environment to let researchers easily implement their own techniques and compare it to the previous ones. If deciding to use one of the simulators applied to the previous works, the developers should study carefully the internal mechanisms of such programs, due to the high coupling of modules. Someone wanting to add new behaviors to the patroller agents, or maybe to implement a new graphical interface to show the simulation, should first know the internal models of graphs, agents, actions and perceptions, and also understand how the eventual controlling classes manipulate them. Add to that, they should also codify their routines in the *C/C++* programming language.

Having in mind such situations and considering the description of the multi-agent patrolling as a benchmark for AI, the necessity to establish a related testbed was noticed. As said by Hanks et al. [1993], testbeds are environments that provide an interface to specify parameters of a benchmark problem, as well as instrumentation to measure performance. So, such a testbed for the patrolling task would lead to a well documented software system of which content shall already bring implemented all the mechanisms to collect the existent patrolling metrics (see section 2), some interesting maps, like the six instances proposed by Santana [2004] (as well as a manner to easily load new ones), the previous developed patrolling strategies and a way to let researchers develop new strategies preferably in a free-programming language way (as it occurs with the simulator of RoboCup [Chen et al. 2002]).

As an effort to reach these requisites, *SimPatrol* was created.

5. *SimPatrol*: a Simulator for Multi-agent Patrolling

SimPatrol is a simulator of multi-agent systems constructed for the patrolling task. Currently, its engineering is in the construction phase, being codified in the Java platform with open source. Someone wanting to check its development can do it at any time in a SVN-manner by the following address: http://simpatrol.googlecode.com/svn/trunk/.

Its initial version has the basic requisites of territory simulation, where a new graph to be patrolled can be initially loaded, since it is in a XML adequate form. Besides this, there are also the action and perception simulations provided to the patroller agents, where fundamental mechanisms of movement through the nodes and edges of the graph, communication and vision of the neighborhood and near agents are already implemented.

Technically inspired by the simulator of RoboCup [Chen et al. 2002], *SimPatrol* operates using a client-server architecture. Succinctly, the graph and other configurations are loaded in the server side, while the patroller agents must connect to ports on the server in order to perceive and act on the generated environment (working as clients). Due to this feature, such agents can be coded in any language, as soon as they implement the defined communication protocols. So, the internal models are updated based on the actions intended by the agents, while the simulator periodically provides them their appropriated perceptions. Additionally, some auxiliary ports are reserved to output the patrolling metrics, and others are reserved for logging the main events of the simulation, in a way that it can be visualized online, or properly stored to be played later.

As an effort to extend the patrolling research, the simulator also permits the creation of dynamic territories, i.e. graphs of which edges and nodes are associated to time probability distributions that rule the appearing and disappearing of such elements. Add to that, it is possible to give distinct priorities of visiting to each node, as well as create open societies of agents on the simulation, i.e. in some user defined cases, new patrollers can be added or die in simulation time. With these features, developers can test their strategies in

very dynamic environments, adding a new step to the patrolling issue.

Additionally, having in mind the robotic aspect of the patrolling task, *SimPatrol* provides mechanisms to set a stamina feature to the agents, what means that they can spend an amount of energy when perceiving or acting on the environment. In such cases, the territory can have some supply points where agents can recharge their stamina.

Finally, the simulator brings the option to count the time of simulation in a real time way, as an alternative to deal with the time simulation problem discussed in the previous section. So, with the adoption of *SimPatrol*, besides the possibility to count the time of patrolling in the traditional manner (based in the agents' perceive-think-act cycle), researchers can also analyze the performance of their techniques in real time, having an idea of the time spent by their agents to think and decide their next action.

After this all, the next step to consolidate the proposed testbed is to implement the previous patrolling techniques [Machado 2002; Almeida 2003; Santana 2004; Chevaleyre 2005; Menezes 2006] in *SimPatrol*.

6. Conclusion

Multi-agent patrolling has recently gained attention from researchers due to its high scientific interest and potential for practical applications. As a contribution to such field, this poster presented multi-agent patrolling as a benchmark for AI and identified the necessity of developing new tools and environments to consolidate it.

In order to help the promotion of such consolidation, it was noticed the need for the establishment of a testbed that could let researchers primarily focus on the solution, rather than on the representation of the problem, as well as define parameters that could let them easily apply the proposed techniques in many varied situations (with dynamic environments, open societies of agents, energetic-expensive actions and perceptions, for example). So, as a very first step to obtain such testbed, *SimPatrol* was proposed as a software simulator of multi-agent systems that aims to support the solution of the patrolling problem and is deeply inspired by the simulator of RoboCup. Currently, *SimPatrol* is in the construction phase.

References

ALMEIDA, A., 2003. *Patrulhamento Multiagente em Grafos com Pesos*. MSc dissertation, Universidade Federal de Pernambuco.

ALMEIDA, A., RAMALHO, G., SANTANA, H., TEDESCO, P., MENEZES, T., CORRUBLE, V. AND CHEVALEYRE, Y., 2004. Recent Advances on Multi-agent Patrolling. *In: Proceedings of the XVII Brazilian Symposium on Artificial Intelligence, São Luís*. Berlin: Springer Verlag, 474-483.

CHEN, M., FOROUGHI, E., HEITNZ, F., HUANG, Z., KAPETANAKIS, S., KOSTIADIS, K., KUMMENEJE, J., NODA, I., OBST, O., RILEY, P., STEFFENS, T., WANG, Y., YIN, X., 2002. *Robocup Soccer Server: Users Manual* [online]. Available from: http://sserver.sourceforge.net/docs/manual.pdf [Accessed 17 August 2007].

CHEVALEYRE, Y., 2004. Theoretical Analysis of the Multi-agent Patrolling Problem. *In: Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology, Beijing*.

CHEVALEYRE, Y., 2005. Le Probleme Multi-agents de la Patrouille. *In: Annales du LAMSADE n.4*. Post-doctoral report.

DROGOUL, A., LANDAU, S., MUÑOZ, A., 2007. RoboCup: un benchmark pour l'IAD? *Unpublished paper*.

HANKS, S., POLLACK, M., COHEN, P., 1993. *Benchmarks, Testbeds, Controlled Experimentation, and the Design of Agent Architectures*. Technical Report, Washington University.

MACHADO, A., 2002. *Patrulhamento Multiagente: uma Análise Empírica e Sistemática*. MSc dissertation, Universidade Federal de Pernambuco.

MENEZES, T., 2006. *Negociação em Sistemas Multiagente para Patrulhamento*. MSc dissertation, Universidade Federal de Pernambuco.

SANTANA, H., 2004. *Patrulha multiagente com aprendizagem por reforço*. MSc dissertation, Universidade Federal de Pernambuco.