

Multiagent Gathering in Real-Time Strategy Games

José Carlos Moura Júnior Geber L. Ramalho

Centro de Informática – Universidade Federal de Pernambuco

Abstract

One common task in Real-Time Strategy Games (RTS) is resource gathering (for instance, gold mining or gas extraction). This task is usually performed by some Non Player Characters (NPCs), which are implemented as intelligent agents. In order to maximize resource gathering by a team of agents, coordination is required. In this paper, we present an experimental work concerning the study of various architectures of multi-agent systems devoted to resource gathering. The pros and cons of each architecture is discussed according to some evaluation criteria we have proposed.

Keywords: resource gathering, foraging, multi-agent, RTS games.

Authors' contact:

{jcmj, glr}@cin.ufpe.br

1. Introduction

RTS games are a very popular game genre, and have an interesting domain to be studied by the Artificial Intelligence area, due to the challenges that the game has to offer. Many AI techniques can be used to solve the problems as combat, team coordination and patrolling. An interesting challenge for IA found on RTS Games is the resource gathering, a multi-agent task that needs to be coordinated.

The problem that will be analyzed here is part of a competition that happens since 2006 in AIIDE 2006 (Artificial Intelligence and Interactive Digital Entertainment). In this competition the participants should develop a group of agents that collects a bunch of resources, spread in the environment, and take them back to the Control Center, like the gathering that happens in a RTS game.

Multi Agent gathering is an area with many studies and techniques, but in RTS context still is an open subject. Despite the current studies on multi-agent gathering, none of them can be seen as a definite solution to the problem. In other words, multi-agent gathering is a problem that needs to be more explored and studied.

This paper presents a performance evaluation, and analysis, of several multi-agent architectures to this important task for RTS games.

2. Gathering

Gathering, or foraging, is the act of searching for food and provisions, and collect them in an area. The foraging have some characteristics that helps to define it:

- Mono Agent X Multi Agent: In mono agent foraging a single agent is responsible for collect the resources, while in the multi-agent a group of agents is responsible to do the gathering.
- Central Place X Various Place: It refers to the amount of spots where the collected objects should be deposited. In central place, there's only one sink, where all the collected objects should be put. In the second case there's two or more places where the collected objects should be deposited.
- Recurrent X One Shot: One Shot means that the agent needs to visit each resource only one time, in recurrent foraging, the agent needs to visit the same resource several times to collect the entire resource.

2.1 Gathering in RTS

In a RTS game the first problem the player will have is the foraging or resource gathering. The player needs to collect resources so he can create buildings, evolve new technologies, train units, etc. Teams with a more efficient resource gathering will have a greater amount of resources, so they will have more advantage in the battle.

Resource gathering is the basis of the economy in a RTS society, so some combat tactics are based on foraging to collect the most of resources as fast as possible to evolve; or attacks the enemy foraging points that will make his economy weaker.

Raiding for instance is a combat tactic that his main objective is to make constant attacks to the foraging enemy points, because the forager units have low combat power. It usually doesn't wins the game, but make it more fragile.

The Fast-Heroic tactic, is based on the fast technologic evolution, because if a society is more evolved, so your units will be. To achieve the resources needed to the evolution, it doesn't create combat units, it only creates forage units so they can collect the resources as fast as it's possible, allowing that way the evolution.

Foraging is a complex problem that needs to use the maximum capacity of each agent, making then to carry as many resources as they can, so they can collect it as soon as possible.

The RTS Games foraging, and that will be analyzed, can be described as Multi Agent Central Place Recurrent Foraging, and this environment is fully observable, deterministic, sequential, dynamic and

continuous. Multi Agent because a group of agents will collect the resources, and there's a need to create a coordination among them. Central Place, because there's a Control Center and all the resources gathered should be taken there. Recurrent, the agent needs to visit each mine several times to collect the resource. The environment is fully observable, each agent have complete information about the environment. The world next state is complete consequence of the actions in the present moment so the environment is deterministic.

The resource gathering can be disassembled in 2 main problems, "Where should I go?" (Resource allocation) and "How do I get there?" (Path-Finding), and later in various others problems studied by the Artificial Intelligence.

The resource allocation is about, what resource each agent should be responsible for collect. And a series of points should be taken in consideration, for instance:

- Distance to the resource: How far is the resource to the Control Center, the most interesting are those that are closer to the Control Center, so the agent could collect, go back to control center and then go to the resource again in a few time.
- The resource availability: It's necessary that the resource is available, there should be a way to get to the resource and no other agent should be collecting the resource.

Path-finding is the most popular, potentially frustrating, problem that can be found in the Artificial Intelligence in games. It's desirable that the path found be as less expensive as possible, in the time to search it and cost to execute it.

On point that can interfere in the resource allocation decision and in the path-finding decision is the coordination among the agents that will collect the resources; it needs some synchronization, so they won't get in the way of each other. It needs to keep in mind that each decision should be taken with the objective to get the best result to the team and not the best for each agent individually.

Another multi-agent gathering problem is the communication; it should be done in an efficient way so each agent can inform the others from his team his decisions, because his decision can interfere in the other agent decision. For instance, if an agent decides to go to a mine the others agents should be informed so any other agent would go there, avoiding this way a unnecessary travel to the mine.

3. State of the art

In this section it will be analyzed the different studies in multi-agent gathering, and how they fit or not in the specific RTS gathering:

Many strategies were described in different articles about foraging, most of them doesn't fit to the exactly

problem, from AIIDE 2006, [OSTEGAARD], [SHELL], [SIMONIN], [HAYAT], [WANG], [STEELS] and [SUGAWARA].

4. Methodology

The objective of this work, is to analyzing different approaches to the problems mentioned in the previous section.

The first parameter that should be considered is the *basic type of the agent*: reactive or cognitive. Reactive agents take his decisions based only in its current world perception; it cannot execute plans (scheduled actions). Cognitive agents can consider in its decision making events that did not happen yet, and can create and execute plans.

The second parameter is the *Path-Finding technique*. In this work two options will be analyzed: the first one is the A* (A-Star) Algorithm, the most used path-finding algorithm, because it guarantees to find the best path between two points if it exists [BOURG]. The other path-finding option is the LRA* (Local Repair A*), a modification of A* for multi-agent environments. [SILVER]. In LRA*, the agent creates its path ignoring all the other agents, except his current neighbors, the agent moves along its path until find some other agent, when it founds another agent blocking its path, it calculates his path from current point.

The next point to consider is the *coordination strategy*. Firstly we will use a central coordinator, a sort of oracle, which controls the gathering agents. The other coordination strategy we will study is decentralized, where the main decisions concerning where and when to gather and when are taken by each agent.

Another parameter concerns communication. The agents can, or can't communicate among them, their plans, their perceptions, and any other information that may be important to help the others agents' decisions. It will be used a blackboard to make the communication between agents.

The last element is the utility function used to allocate each agent to a mine (resource location). Many functions are possible. In this work the function utilized is the Euclidian distance from the mine to the Control Center, where the agents are supposed to lay down the gathered resource.

We have investigated two resource allocation centralized strategies (Auction and Elitist Social Welfare), and two decentralized ones (with and without communication), as described below:

- Auction Allocation: In this case, the central coordinator elects, based on the utility function, to which mine each agent will be allocated. The agent that has the best bid (best score in utility function) for a mine, is allocated to it.

- **Elitist Social Welfare:** This allocation score is equal to the best individual score from the agents. To get the best combination, each agent is allocated to the resource that has its best score, according to its utility function. If more than one agent have a preference for the same resource, the resource is allocated to the agent with best score.
- **Individual without communication:** Each agent, based on its utility function elects the best mining spot, and goes there. It has problems in the case another agent chooses the same mining spot.
- **Individual with communication:** Similar to the approach mentioned before, except that after each choice, the agent marks on a *BlackBoard*, how long time that mine will be occupied by it. If an agent picks an already occupied mine then it chooses the next option.

Implementation

All the simulations will happen in the same scenario (map), and will have the same game configuration. The evaluation criteria will be the amount of resource collected in the Control Center after the time has finished.

The reactive agent is based in [SIMONIN], the agents are going to create a potential field from the control center and spreading across all the map. During the creation of the potential field when an agent finds a mine it collects, and then go throw the gradient going to the lower values until it gets to the Control Center when it is closer enough to the Control Center it delivers the resource. While the agent is going to the Control Center he leaves “bread-crumbs” in the path he made it, so another agent that found those marks are going to follow it to the point with highest potential field value until it finds the resource.

Each cognitive agent has access to a class that is responsible to make his allocation to a mine (IAAllocation), a class (GridManager) that grants it access to the map grid with mine positions and also responsible for path-finding. Those agents have a internal state machine with the states described below:

- **IDLE:** It's the initial state from this agent. In this state the agents is looking for a mining place, so he asks the IAAllocation Class for one. When it gets his mining spot It asks for GridManager for a path to the spot, then it changes its internal state to GOING TO MINE.
- **GOING TO MINE:** In this state the agent follow the path that it has be given, until it finishes then sets its internal state to MINING.
- **MINING:** In this state the agent is going to mine the resource until the mine is empty or it gets full of resource. Then its state is set to GOING TO DELIVER.

- **GOING TO DELIVER:** In this state the agent is going to use the same path it got in IDLE state to get back to the Control Center, when it gets near the Control Center It delivers the resource that is carrying. Then its state is set to IDLE.

Above there's a diagram illustrating the cognitive agents state machine, and it's class diagram.

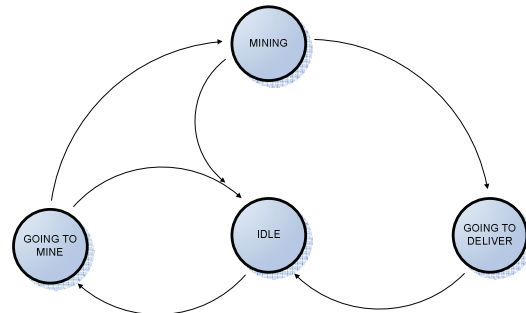


Figure 1 - Cognitive Agent State Machine

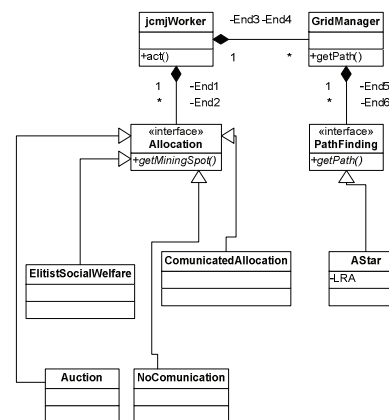


Figure 2 - Cognitive Agent Class Diagram

Scenario

The map (Figure 3) shows the scenario where the agents were simulated, during 800 cycles with 100 milliseconds each. The brown dots represents the resources, the gray dots are obstacles, the small blue dots are the workers and the big one is the Control Center.

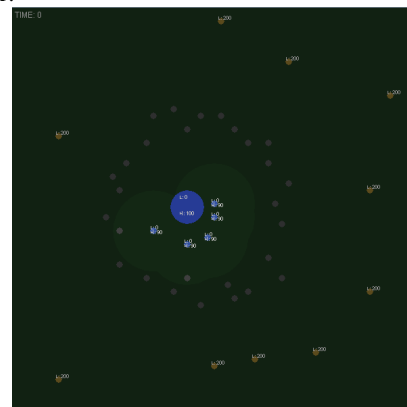


Figure 3 - Map

Results and Analysis

	Type	Coordination	Communication	Path-Finding	Allocation Strategy	Results
Worker0	Reactive		Yes	N/A	N/A	120
Worker1	Cognitive	Central	Yes	A-Star	Elitist	440
Worker2	Cognitive	Central	Yes	A-Star	Auction	120
Worker3	Cognitive	Individual	Yes	A-Star	Decentralized Communicated	340
Worker4	Cognitive	Individual	No	A-Star	Decentralized Without Communication	310
Worker5	Cognitive	Central	Yes	LRA	Elitist	550
Worker6	Cognitive	Central	Yes	LRA	Auction	490
Worker7	Cognitive	Individual	Yes	LRA	Decentralized Communicated	450
Worker8	Cognitive	Individual	No	LRA	Decentralized Without Communication	430

Table 1 - Summarizes the agents created with the combination of the different multi-agent architectures for resource gathering, and the results.

General Analysis

The reactive agents spent many time creating the gradient, but when an agent finds the mine it collects as much as it gets, and managed to get back to the control center, and back again to the resource very efficiently. As it was expected the agents with the LRA Path finding got better results then its same version with the simple A-Star path-finding.

In decentralized allocation the agent has a performance increase when they could communicate, also the agents visits more different mines if compared with the agent without communication, where most of them usually choose the same resource to collect, decreasing the team performance.

In centralized allocation, the auction allocation got a worse performance due to the fact the closest mines got very crowded, and the agents had to keep recalculating their routes, when it was possible (LRA*). The Elitist Allocation got a better result because the agents got more spread, avoiding crowding a region.

Comparing the two types of coordination, the centralized got the best results, even when compared with the decentralized with communication.

The results shows that the best approach, among the analyzed ones, is the cognitive agent with centralized allocation based on Elitist Social Welfare and a path-finding that allows recalculating the route when it collides with another agent.

3. Conclusion

Despite the clear application of multi-agent gathering in games, particularly in RTS, no systematic work has been reported in the literature. In this paper, we have presented in this paper an original analysis of an ensemble of possible architectures or approaches for multi-agent resource gathering. In future works we would like to add more parameters to the architectures like different utility functions besides the distance used in this work.

Also we would like to expand the possibilities covered by the already existing parameters, as different approaches of collective path-finding and another communication strategy, besides the blackboard and flags used in this work, as directed messages. Another possibility is the creation of one allocation strategy with negotiation between the agents.

References

- OSTEGAARD E., SUKHATME G., MATARIC M., Emergent Bucker Brigading – A simple mechanism for improving performance in multi-robot constrained space foraging tasks.
- SIMONIN O, Construction of Numerical Potential Fields with Reactive Agents.
- SHELL D, MATARIC M, On foraging strategies for large-scale multi-robot system. *In: Proceedings of the 2006 IEEE/RSJ, Intl. Conference on Intelligent Robots and systems. Beijing, China October 2006*
- HAYAT S, NIAZI M, Multi Agent Foraging- Taking a step further Q-Learning with Search. *In: Proceedings of the 2005 IEEE, International Conference on Emerging Technologies. Islamabad 2005.*
- WANG S, TAN M, Relation between task-based diversity and efficiency in multi-robot foraging. *In: Control, Automation, Robotics and Vision, 2002. ICARCV 2002. 7th International Conference.*
- STEELS, L. (1990). Cooperation between distributed agents through self organization. *In Demazeau, Y. and Müller, J.-P., editors, Decentralized AI — Proceedings of the First European Workshop on Modelling Autonomous Agents in Multi-Agent Worlds (MAAMAW- 89), pages 175–196. Elsevier Science Publishers B.V.: Amsterdam, The Netherlands.*
- SUGAWARA K, WATANABE T, Swarming Robots – Foraging Behavior of Simple Multi-Robot System. *In: Proceedings of the 2002 IEEE/RSJ, Intl. Conference on Intelligent Robots and systems. EPFL, Lausanne, Switzerland October 2002.*
- BOURG, D. AND SEEMANN, G.,. AI For Game Developers
- SILVER, D.,. *Cooperative Path Finding*