MULTI-AGENT STRATEGY GAME OVER ANTS

Vojtěch Šimetka
Master Degree Programme (1), FIT BUT
E-mail: xsimet00@stud.fit.vutbr.cz

Supervised by: Jiří Král
E-mail: ikral@fit.vutbr.cz

Abstract: This paper describes challenges in design and development of a multi-agent real-time strategy game. The resulting game implements, apart from features and game control which can be found in nowadays RTS games, three different levels of artificial intelligence in which each unit is an agent. Moreover, there is a unique cooperation mode, where units can be controlled by user and AI at the same time. A number of experiments was performed in order to evaluate both game design and capabilities of artificial intelligence.

Keywords: Multi-agent systems, RTS, Jason, strategy game, BDI agent, reactive systems, reasoning, A*, ray-casting.

1 INTRODUCTION

Games are an integral part of our life. From early childhood we play games in order to develop various abilities that may be useful some-when in the future. Computer games are not an exception. In fact, it has been proven that some computer games may be more beneficial for modern human in their everyday life than classic games like hide-and-seek [2]. Computer games can be divided into a number of genres. We will be particularly interested in so called real time strategy games (RTS).

A real time strategy is a computer game where you take control of armies, direct their development and fight battles. The game is played in real time so there are no turns and therefore players with lower reaction times have an advantage. Game objectives usually are elimination of all enemy buildings and/or units. From the genre we can already determine some requirements for properties of the artificial intelligence. The AI has to consider a large number of inputs, process them and act accordingly in a timely fashion. In most RTS games the AI acts as a human - it manages all units centrally and delegates actions they should take. We have decided to try another, admittedly more computer resource demanding, approach.

Every unit will reason independently and will be responsible for its actions. Such behaviour can be well described as an BDI agent [1, 4, 5]. All units will have a limited knowledge of the game environment so their decisions will not be fully informed. While this makes decision making simpler, it is more difficult to coordinate units and exchange their knowledge of the environment. Our goal is to develop a real time strategy game with a focus on the multi-agent AI.

2 DRAFT OF ARTIFICIAL INTELLIGENCE

We have implemented three distinct levels of artificial intelligence. The basic level of AI does not communicate and therefore every unit works for its best benefit. The second AI, called cooperative, is partially informed about perceptions of other friendly units and can commit its beliefs to achieve greater benefits across community of friendly units. The last level of AI, advanced, goes even further. It does not just blindly perform the best possible actions that are available now but it plans ahead by prioritizing collection of one resource type and allocating them to individual units.

AI	**Resource collection**	**Exploration**	**Combat**
Basic	No cooperation	No cooperation	No cooperation
Cooperative	Decentralized	Decentralized	Decentralized
Advanced	Centralized	Decentralized	Centralized + Decentralized

Table 1: Cooperation overview for each level of AI.

For purely user controlled units the execution of actions is handled completely within Java code and is not really worth mentioning. However, we have implemented so called assistance mode where units are controlled jointly by both player and the computer utilizing any of the previously mentioned AIs. In this mode actions from the player are prioritized and only if such actions are not issued, the unit control is up to the AI. The constant interruption of the AI reasoning cycle by the user causes quite a lot of problems. Especially advanced AI with its resource allocation suffers quite heavily. Nevertheless, there are no special modification to these AIs in the agent source code, which is quite convenient for further game customization.

3 EXPERIMENT EVALUATION

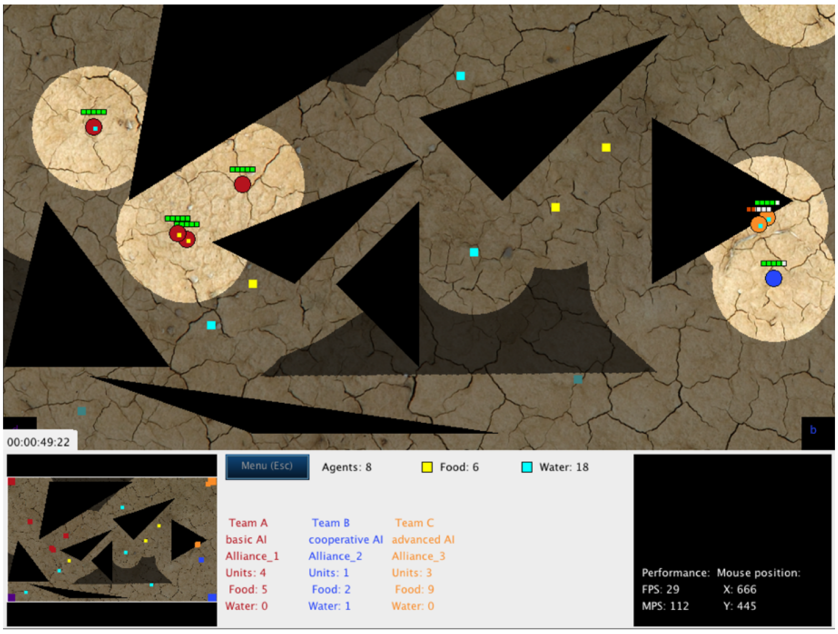
In order to determine effectiveness of each distinct level of AI we have set a number of experiments focusing on key aspects relevant to the success in the game. In the resource collection experiment we tried to analyse AIs' ability to collect resources and propagate theirs population. The advanced AI was, for small number of units, approximately 30% faster in collecting resources and expanding its unit population than the basic AI. The cooperative AI performed at about 12% better in comparison to the basic AI. However, with increasing number of units, the performance difference becomes less striking. Such phenomenon can be explained by overhead and information jam caused by communication between agents. The advanced AI is also much better at performing so called effective actions per minute (EAPM). Effective action is an action which changes game environment. Examples of such actions are: collecting resources, attacking enemy unit or choosing a destination where to go. In other words, the EAPM tells us how many times the unit had to altered its behaviour and usually lower value means better planning. Surprisingly, the worst AI in terms of EAPM is the cooperative which could be explained by inefficient resources collection delegation.

When it comes to combat, the win-loss ratio for the cooperative AI versus the basic AI is 59.38%, for the advanced AI versus the basic AI 53.3% and for the advanced AI versus the cooperative AI 50.07%. In general, the advanced AI competes much worse than the cooperative AI despite better resource collection and management. The EAPM grows linearly with increasing number of units. Once again, the EAPM of the cooperative AI is 25% worse than for the basic AI which is 5-10% worse than the EAPM of the advanced AI.

4 CONCLUSION

Our goal was a creation of a fully functional real-time strategy game featuring three different levels of the artificial intelligence and a single player mode. The resulting game implements most of nowadays standard RTS game features and provides rich game-play experience. We have implemented a unique mode which allows players to play the game with a help of the AI so that they can fully focus on interesting aspects of the game without any worries of falling behind. The post-game analysis helps the player in better understanding of the game mechanics along with detailed overview of the just finished game.

One of the key parts of this project was an evaluation of the game performance. We have discovered



Agents: 8 🟨 Food: 6 🟦 Water: 18

Team A	Team B	Team C
basic AI	cooperative AI	advanced AI
Alliance_1	Alliance_2	Alliance_3
Units: 4	Units: 1	Units: 3
Food: 5	Food: 2	Food: 9
Water: 0	Water: 1	Water: 0

Performance:	Mouse position:
FPS: 29	X: 666
MPS: 112	Y: 445

Figure 1: Screenshot from the game.

how big the impact of the graphic user interface is. In current implementation the GUI takes around 70% of total game resources. Implemented AIs were also tested quite heavily. We have not verified our hypothesis that the advanced AI, combining decentralized and centralized decision making, would be superior. However, all implemented AIs have their strengths and weaknesses and when competing one to another the win-loss ratio is quite balanced.

To sum up, we have proved that it is possible to create playable RTS game in Java and Jason [3] using multi-agent approach. The game is released as an open source program to allow others to participate in the game development and learn agent programming.

ACKNOWLEDGEMENT

This paper was created in partnership with the project IT for inovations IT4I.

REFERENCES

[1] R. H. Bordini and J. F. Hübner and M. Wooldridge: Programming Multi-Agent Systems Using Jason, John Wiley & Sons Ltd., 2007, ISBN 978-0-470-02900-8

[2] WWW pages, Why SEO Is Like An RTS Game (and why you should care) |SEOmoz [online] [cit. 2013-05-10], http://www.seomoz.org/blog/why-seo-is-like-an-rts-game

[3] WWW pages. Jason | a Java-based interpreter for an extended version of AgentSpeak. http://jason.sourceforge.net/, [cit. 2013-01-20].

[4] Anand S. Rao and Michael P. Georgeff. Bdi agents: From theory to practice. In IN PROCEEDINGS OF THE FIRST INTERNATIONAL CONFERENCE ON MULTI-AGENT SYSTEMS (ICMAS-95), pages 312–319, 1995.

[5] S. Russel and P. Norvig. Artificial Intelligence, a Modern Approach. Pearson Education Inc., 2003. ISBN 0-13-080302-2.