Extending the STRADA framework to design an AI for ORTS

Laurent Navarro[1] and Vincent Corruble[1]
[1]Laboratoire d'Informatique de Paris 6
Université Pierre et Marie Curie (Paris 6) – CNRS
4, Place Jussieu
75252 Paris Cedex 05
{Laurent.Navarro,Vincent.Corruble}@lip6.fr

Abstract. Strategy games constitute a significant challenge for game AI, as they involve a large number of states, agents and actions. This makes indeed the decision and learning algorithms difficult to design and implement. Many commercial strategy games use scripts in order to simulate intelligence, combined with knowledge which is in principle not accessible to human players, such as the position of the enemy base or the offensive power of its army. Nevertheless, recent research on adaptive techniques has shown promising results. The goal of this paper is to present the extension such a research methodology, named STRADA, so that it is made applicable to the real-time strategy platform ORTS. The adaptations necessary to make STRADA applicable to ORTS are detailed and involve the use of dynamic tactical points and specific training scenario for the learning AI. Two sets of experiments are conducted to evaluate the performances of the new method.

Keywords: Game AI, learning, real-time strategy games

1 Introduction

The quality of a commercial video game depends largely on its capacity to entertain human players. After having invested significant efforts to increase the graphic quality of their games, making them more realistic, game designers try to focus on improving the gameplay of their products. Nevertheless, the Artificial Intelligence (AI) available in games remains usually limited and predictable, often forcing the players to compete against other humans instead of synthetic entities [3,5].

Some of the most highly used AI techniques in video games, such as Finite State Machines (FSM) or Scripting languages which are powerful solutions, easy to implement, let programmers describe behaviors in a static and somewhat detailed manner. They can lead to realistic behaviors [8], but they are also plagued by complexity [4]. Moreover, their determinism makes them predictable by the human player after a certain amount of runs [5]. A promising evolution to go beyond this limitation is to look for adaptive techniques, where the knowledge necessary to the behavior is not produced by the programmer, but is learned automatically through experience (i.e. through playing). It has been proposed for example with Dynamic Scripting [1] which uses weighted rules to adapt scripts. Though some promising results have been obtained in research labs developing learning techniques for games [7, 9], they remain so far underused in commercial games [5].

In the following section, this paper briefly introduces the STRADA framework for an adaptive game AI on which this paper is based, and the ORTS platform which is used as an environment for our experimentation. It then tackles the challenge of adapting the STRADA framework to the ORTS platform for real-time strategy (RTS) games, in particular looking at the question of map analysis and tactical points. The resulting platform is then tested against the winner of the 2007 ORTS competition, and against a random AI. Encouraging results are further improved by proposing the notion of specific training scenarios where the learning AI is set in an environment that favors the acquisition of key game concepts.

2 Background

The goal of the STRADA framework [2] was to propose a generic model for the automatic generation of adaptive strategic behaviors in strategy games. It combines recent AI techniques, like reinforcement learning, with new ideas to handle the large complexity of modern games. Three main axes were explored: a decision-making system architecture based on a military hierarchy and a map analysis algorithm, whose goal were to reduce the complexity of the state and action spaces, and specific combined reinforcement signals which dispatch the information through the hierarchy and help the coordination between the different learning agents.

This approach has been applied to the turn-based game Battleground: Napoleon in Russia (Talonsoft). Experimental results showed that the STRADA approach reaches higher performances than those obtained by the original game AI, and is able to compete with a trained human player. A coherent and adapted military strategy was learned for the two scenarios studied. Only Battleground was used so far for the evaluation of STRADA, while the ambition behind it was to have a somewhat generic framework for strategy games adaptive AI. Trying to test and adapt the STRADA framework on a modern real-time strategy game is an important motivation behind the work presented here.

The ORTS platform is an Open Source project aiming at providing the scientific community with a shared framework for RTS AI testing [6]. It is based on a client–server architecture where all the central data, like the position of all the playing units, are handled by the server. This environment is nearly comparable to commercial real-time strategy games like Command & Conquer (Westwood Studios), but some important differences with *Battleground* (used in past experiments with STRADA) need to be highlighted, as they require specific adaptations explained further in the following section. The random generation of map in ORTS (for each new game) highlights the need for a new approach to the map analysis method proposed in STRADA. The presence of an economy in ORTS, symbolized by the management of resources in order to develop a base and an offensive army, where Battleground focused on the tactical aspect of the conflicts, requires that the new platform adapts the notion of hierarchy previously used in STRADA. Finally, considering that ORTS is a real-time game and that Battleground is turn-based only, the new model has to adapt the learning algorithms to this new complex testing environment.

3 Extension of the STRADA model

3.1 Hierarchical structure

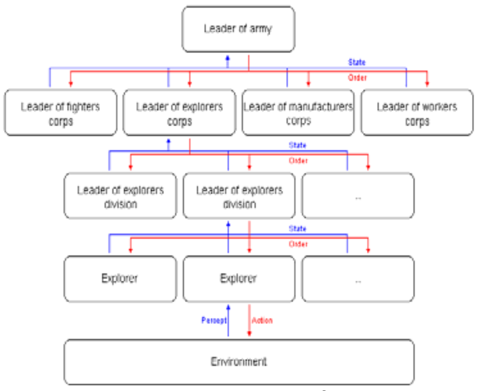

Fig. 1: Military hierarchy from STRADA adapted to ORTS corporations

The platform described here replicates the vertical dimension of the military hierarchy already used in STRADA, from army leader at the top to individual units at the bottom.
However, the presence of *fog of war* in ORTS requires creating special unit dedicated to exploration. Similarly, the simple economy in ORTS requires the implementation of workers for collecting resources and building production structures, and of manufacturers, for creating new units. All those corporations have specific orders and perceptions. Therefore, the new platform introduces a horizontal dimension to the original vertical hierarchy introduced by STRADA with four components (fighters, explorers, manufacturers, and workers) as shown in Figure 1**Erreur ! Source du renvoi introuvable.**.

3.2 Dynamic tactical points

Tactical points are structures created by the platform in real-time to abstract the knowledge acquired while playing. At each round, the engine locates and identifies special areas of interest (see Fig. 2 for an example), mainly defined by the presence of groups of buildings, and extract specific information which is stored to create the memory of the game. Those parameters, which are discretized and normalized, describe (1) the strength of friendly forces, of (2) enemy forces, (3) a risk factor, (4) a force ratio, (5) resource availability.

Thus, tactical points are a combination of those parameters, calculated and modified in real-time by the engine. In this study, the different values used for each setting allows the creation of 36 singular tactical points. Each of them is then combined with the different orders usable by the 3 operating level to create the action space. For the purpose of this study, only buildings can create a tactical point, even if surrounding units offensive and defensive power are represented in its description. Creating tactical points with only troops or landscape singularities would be feasible but adding parameters to their description will exponentially increase the number of allowed tactical points, as well as the size of the action space.

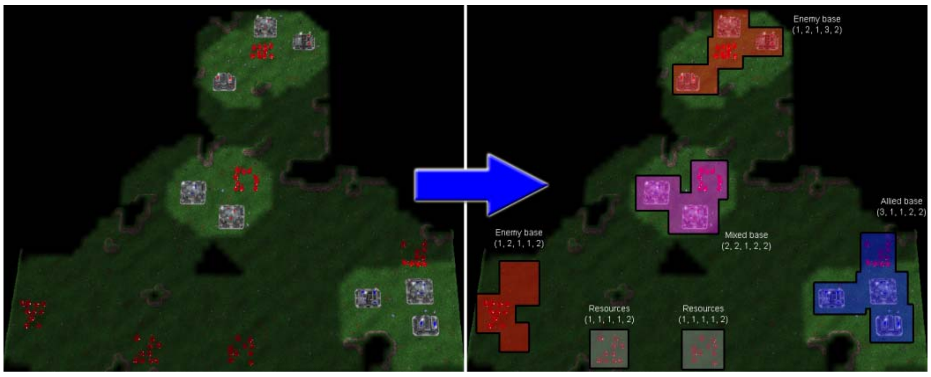

Fig. 2: Example of inference of 6 tactical points during a game.

3.3 Reinforcement signals

The structure of the reinforcement signals used in the new platform is a consequence of the military hierarchy described previously. Most of them are similar to the one illustrated in the STRADA model: the global reward, calculated from the score obtained by the agent mainly by collecting resources and killing opponents, the local reward, specified for each leader of the different corporations, and the order reward, representing how a leader follows the order given by his direct hierarchy.

However, to represent the horizontal axis added by our new framework, a specific local combined reward has been introduced, whose definition is shown below. Its goal is to symbolize the interaction between the different corporations. Finally, the complete reward, named combined reward, is a linear combination of the main rewards explained above:

$$R_{Local\ combined} = \alpha R_{Fighters} + \beta R_{Explorers} + \gamma R_{Manufacturers} + \delta R_{Workers}$$

$$R_{Combined} = \alpha R_{Global} + \beta R_{Order} + \gamma R_{Combined\ Local}$$

This final reward is used within a SARSA-λ learning algorithm [10]. The combined rewards associated to the state/action couples are memorized using neural networks. Finally, the action selection strategy is based on a Boltzmann-Gibbs probabilistic distribution.

4. Experiments and results

During this first experiment, the new platform has been trained on the third scenario of the ORTS AIIDE competition, during 5.000 steps. It is opposed both to the *RTSComp07* game AI, winner of the AIIDE07 challenge, and to a *test AI* using the same engine as the new platform but performing a *random* action at each decision cycle. Every 100 runs, an evaluation was performed during 20 games to measure different performance indicators such as the score, the offensive and defensive power of the army, the size of the explored map and the amount of farmed resources.

The score evolution detailed in Figure 3a (left) shows that the platform is able to increase its performance through learning. After 5.000 runs, its score is 78% higher than the one obtained by the *random AI* but 45% lower than the estimated score of *RTSComp07*. Moreover, the platform takes more than 1500 steps before it begins increasing its performance.

After learning, the AI is able to farm resources, explore the map and optimize its global score by creating a few offensive units. Nevertheless, it does not learn to colonize unused resources spots and cannot launch significant assaults against the opponent. These somewhat poor results can be explained by the large amount of stages needed to be completed before being able to develop a massive army. At the opposite, the *RTSComp07* AI rushes the opponent base early in the game.

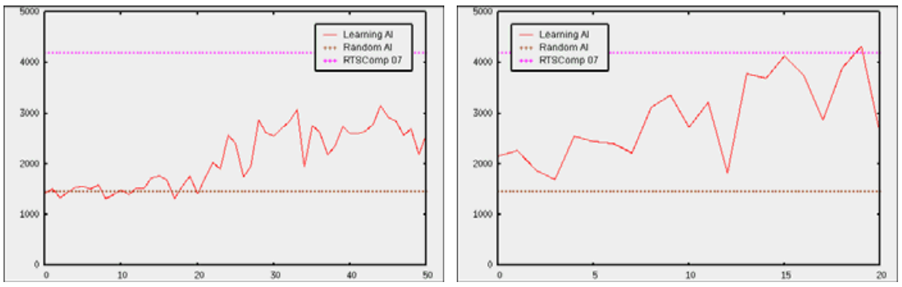

Fig. 3: Evolution of the score function with the number of evaluation steps (a) (left) without using specific training scenarios (b) (right) with specific training scenarios.

Following the half-satisfying results described above, an attempt was made to increase the performance of the platform by creating specific training scenarios, which are designed to let the agent acquire important skills without fearing an early attack. Three training scenarios have been designed, each one to learn a specific ability: (1) exploration with an empty map, (2) tactical coordination with an already created assault force and an identified enemy base to target, and (3) army development with a pre-built complete base. Finally, the skilled agent having learned through these three training scenarios has been tested again against *RTSComp07* during 2.000 runs with the same evaluation method as the one defined previously.

This time, the score evolution displayed in Fig. 3b (right) shows an important enhancement of the performance, which is after learning now 235% higher than the one obtained by the *random* AI, and only 16% lower than the estimated score of *RTSComp07*. With the use of the learning scenarios, the agent is now fully able to explore the map, farm resources, manage the production of units regarding the amount of resources collected and produce a massive army. Nevertheless, it is still not able to colonize unused resources spots and efficiently coordinate its army to attack the enemy bases or defend its own base. Let us note that the AI performance is highly dependent on the quality of the scenarios it has been trained on. Those should be further improved to focus on the defaults observed previously.

5 Conclusion

The results analyzed previously showed a great capacity of the new framework to learn a strategy and to increase its performance in a complex STR environment.

Exploration and basic economical behaviors have been successfully learned during both experimentations. The use of training scenarios has improved military tactics with the production of a massive infantry army and the emergence of artillery. Nevertheless, no decisive assault is launched on the enemy base and only very few victories against *RTSComp07* have been registered.

As shown previously, those points could be improved by slightly modifying the model and optimizing most of the low-level AI algorithms used in the engine. Moreover, the training scenarios used to increase the agent's initial knowledge appeared to be difficult to design and not completely satisfactory. Another idea would be to replace them by a military doctrine. It would represent what servicemen learned at school as a basis before learning "in battle".

Finally, the performances obtained by this new platform are very encouraging. Most of its aspects can be improved to allow it to produce a high-quality strategy and to be able to defeat most of the script-based AI with its adaptive capacities. Such progress would allow it to be a great opponent in the next ORTS competitions, and ultimately a more entertaining opponent to human players.

References

1. Spronck, P., Ponsen, M., Sprinkhuizen-Kuyper, I., and Postma, E. 2006. Adaptive game AI with dynamic scripting. *Mach. Learn.* 63, 3, 217-248.
2. Madeira, C. and V. Corruble. STRADA : une approche adaptative pour les jeux de stratégie modernes. *Revue d'Intelligence Artificielle*. Hermès, Lavoisier. Vol. 23 – N° 2-3/2009, pp. 293-326.
3. Forbus, K. D. and Laird, J. 2002. AI and the Entertainment Industry. *IEEE Intelligent Systems* 17, 4 (Jul. 2002), 15-16.
4. S. Russell, P. Norvig, 2002, Artificial Intelligence: A Modern Approach. Prentice Hall.
5. Nareyek, A. 2004. Artificial Intelligence in Computer Game - State of the Art and Future Directions. *ACM Queue* 10, 58-65.
6. M. Buro, 2004, Call for AI Research in RTS Games. Proceedings of the *Challenges in Game Aritificial Intelligence* workshop. AAAI 200, pp 139-141.
7. Marthi, B., Russell, S. J., Latham, D., and Guestrin, C. 2005. Concurrent hierarchical reinforcement learning. In *Proceedings of IJCAI-2005*, 779-785.
8. Khoo, A., Hunicke, R., Dunham, G., Trienens, N., and Van, M. 2002. FlexBot, Groo, Patton and Hamlet: research using computer games as a platform. In *Eighteenth National Conference on Artificial intelligence*, 1002-1002.
9. Tesauro, G. 2002. Programming backgammon using self-teaching neural nets. *Artif. Intell.* 134, 1-2, 181-199.
10. Sutton, R., and Barto, A. 1998. Reinforcement Learning, An Introduction. MIT Press, Cambridge, MA.