Edward Cruz

due 11/16/15

CS 4960

An Overview of "A Review of Real-Time Strategy Game AI"

The goal of this paper is to examine and review AI techniques used for video games, specifically real-time strategy video games (RTS). In that it'll cover the main areas of current academic research which are tactical and strategic decision-making, plan recognition, learning and the research gone into those sections and their importance. This paper will also look at the use of game AI in academia, and the various perspectives, approaches and research gone into them. The paper will also cover academic research into depth, the risk and challenges it faces and what can be done about them. It will also cover areas in AI for RTS games that need more research and investigation.

What is AI?

The quick and easy definition of Artificial Intelligence is that AI is branch of computer science concerned with making computers think or behave like humans or in a rational manner (the two are not always interchangeable, unfortunately). There are several specializations in the field of AI such as games playing AI (programming computers to compete in games against human opponents), natural language AI (computers that are made to understand natural human languages, neural networks (simulated intelligence that tries to create the types of physical connections that occurs in animal brains), robotics (computers that see, hear and react to their environment) and various others.

What is an RTS game?

The term Real Time Strategy was coined sometime in the early 1980's since Strategy games were becoming more popular and had more representation in the market, so it's only natural that people started pointing distinctions in various games in the genre. Real Time Strategy/RTS refers to a time-based video game that center around using available in-game resources to build units and defeat opponents. Knowing your location and your opponents immediate area and map, and where and how to acquire said resources (hopefully before your opponents) are vital elements to the genre, sometimes used in non-strictly strategy game genres like Fighting games, FPS/Arena FPS, Mobas, etc. etc. According to the author RTS are "simplified military simulations" (they're not entirely wrong, it is a subgenre within a subgenre but most RTS already share that feature) and ,"Players indirectly control many units and structures by issuing orders from an overhead perspective in "real-time" in order to gather resources, build an infrastructure and an army, and destroy the opposing player's forces." The real-time aspect in strategy games players refers to the rule of the genre in which players don't take turns, but instead play as many actions as they are physically able to make (in competitive settings some players can get an amazing 7 inputs per second), while the game runs at a steady and continuous flow (hopefully without latency).

Why is AI research and development important for video games?

Games are a great place for exploring the capabilities of (AI) within a constrained environment and a fixed set of rules, problem-solving techniques can and are often developed and evaluated before being applied to more complex real-world problems. AI has been used by a variety of games making competition which has sped the development of many heuristic-based search techniques (Heuristics reduce the number of alternatives from an gigantic number to a polynomial number, the term heuristic is used for any advice that is often effective at estimating the cost of an optimal path between a pair of states in a single-agent path-finding problem but not guaranteed to work in every case). According to Laird and van Lent, “video games as a potential area for iterative advancement in increasingly sophisticated scenarios, eventually leading to the development of human-level AI”. Video games are also a useful alternative to robotics for AI research because they can provide an increasingly complex and realistic environment for simulation unlike robotics that have cost and more strenuous needs. RTS games should be specifically looked at because they provide a big malleable tool for exploring various complex challenges that are central to game AI and other problems. There is long-term high level planning and decision-making in RTS games called macro management which is essential strategic decision making. That planning includes: balancing the development of the system’s economy, infrastructure, upgrades with the production of military units so they have enough power and units to attack and defend whenever needed and resources and upgrades to succeed later on in the match, deciding which units and structures to make and which technologies to advance throughout the game in order to have access to the right composition of units at the right times.

StarCraft

The author concentrates on StarCraft over other RTS available at the time this paper was written for a variety of reasons. One reason why it was chosen was due to its growing popularity for use in RTS game AI research, driven by the *Brood War Application Programming Interface* (BWAPI4) and the AIIDE5 and CIG6 StarCraft AI Competitions. BWAPI provides an interface to interact through programming code with StarCraft, allowing external code to retrieve the game state and execute actions as if they were a player in a match. The competitions (AIIDE5 and

CIG6) put StarCraft AI (CPU's or bots) against each other in full games of StarCraft to determine the best AI and improvements each year. Those competitions also had simple challenges based on various parts of the game like controlling a given army to defeat an opponent with an equal army. Because the StarCraft community already exist and is as big as it is, forming a community around the game as a research platform is easier and it enables people to work together and share and work of each other's work without having to repeat the needed groundwork before an AI system could be implemented. And because of StarCraft's lasting popularity and competitive scene there is a lot of high-quality footage and replays available for data mining and with the variety of players there is different skilled people to test the bots/AI against.

Techniques for Developing AI for StarCraft

There are various techniques used for creating AI in StarCraft, many come with varying results against different players of varying skills, and against different AI. Tactical and micromanagement decisions employs many techniques, it involves controlling a specific units or groups of various types of units over a short period of time, making use of a techniques from the AI making strategic decisions. These decisions follow simple goals like trying to maximize the amount of enemy resources, units and weapons that can be removed from the map in the quickest way possible without consuming too many resources yourself or putting too many of your units in danger. Finite state machines are commonly used to make said decision decisions in the industry, but even though the decisions taken are small-scale ones relatively speaking, a lot of factors need to be considered in making the best decisions at the given time, especially when using a variety of units each with different attributes.

One of the techniques used in Tactical Decision-Making is Reinforcement Learning. Reinforcement learning is an area of machine learning is an area in machine learning where the agent learns by trial and error, it concerns itself with what options it ought to take in order to get maximize a cumulative reward. Reinforced learning can find increasingly better solution than previously known ones through multiple iterations (trial and error). It's simple to apply to a new domain and just requires a description of the situation, possible actions it can take and end goal, unfortunately in a domain for an RTS, which is complex due to the amount of factors and possibilities one can take for tactical decision making, Reinforced learning need a clever state abstraction mechanism to learn in effectively. That's why this technique isn't used for large-scale strategic decision making in RTS games; it's very difficult due to the nature of the genre. However there have been attempts in StarCraft and other RTS games that also utilize artificial neural networks to know the consequences and expected results for attacking or fleeing with a particular unit during certain circumstances. The AI did manage to beat the inbuilt AI under certain game modes (3 on 3 battles).

Another technique used for Tactical Decision-Making is Game-Tree Search. Although search based techniques are not able to deal with the long term decision making of a typical RTS game like StarCraft, it has been applied to smaller scale versions of RTS combat which is worthy of note. A Game-Tree Search for a game is when a search starts at the initial position and has all possible moves from each position it can take, and RTS game has far too many options available, and something like GTS is more suited for checkers, tick tack toe or other simple games. The simplified RTS in the review does contain important aspects of an RTS but it's narrower on what it does focus on, like concentrating on multiple units' group movement and not worrying about long-term building or collecting or resources or other tedious micromanagement for controlling various units. Using simulations to compare expected outcomes from using various strategies that can be used by the AI and it's various opponents (everything they can do, we can too for simplicities sake) and then selecting the "Nash-Optimal strategy" (A simplified account of Nash-Optimal strategy is when 2 people playing a game are at a point where each is making the best decision possible while taking into account their opponents decision as long as their decisions don't change). It might seem pointless to purposeful avoid and skip such important aspects of the game but doing so allows us to skip to more eventful movements and examine really quickly all the way to the end-game state.

The Monte Carlo Planning technique has been used for simplified RTS like Game Tree search, and just like GTS, it's severely limited but worth mentioning. Using this method involves using randomly generated plans to find out which plan usually leads to more successful outcomes against opponents. Although it is beneficial for RTS research because it can deal with randomness and uncertainty, it has not been applied to StarCraft because of its lack of effective simulator and the complexity of the domain. The Bayesian model has also been used for tactical decision making AI to decide which direction units in battle should move to. Every sensory input is part of a probability equation which can be solved given enough data about the distributions of the inputs about the direction moved so that the best direction can be used. The Bayesian model by Synnaeve and Bessi`ere is paired with hierarchical finite state machine which lets the AI choose different sets of behavior for different situations like if enemies are scouting, engaging, etc. etc. Overall a lot of these techniques fail to account a variety of things important to the RTS genre like the macro and micromanagement aspects but can still produce meaningful results for actually beating players and enemy AI.

Strategic Decision Making

Planning systems have been created by researchers in order to make a system that can make intelligent actions at a strategic level in an RTS game, planning systems are capable of many things like being able to determine sequences of actions that can be taken in a specific situation to get to the end goal or win state. The lack of information or inability to account everything is a challenge in developing AI, but with planning systems, researchers hope to have AI be playing at a human level, while also reducing the development effort needed when compared with the standards in industry. An example of a strategic decision-making system was created for a game called, Kohan II: Kings of War, the system assigned resources and resource construction, research and upkeep capacities to goals, such as trying to maximize the priority of the total goals that could be meet. Priorities were set by a large number of hand-tuned values, they could be swapped for a different set to give the AI different “personalities”, each priority value was modified based on factors of the current situation and a goal commitment value to prevent flip-flopping after a goal had been set, and finally a random value to reduce predictability. The AI was then a lot easier to update for changes in the game’s design though the development process and most of all the system created a fun challenging opponent. A few of these techniques are:

Case-Based Planning

Case-Based Planning (CBP), which is a technique that finds similar past situations from which to draw potential solutions for the current situation. CBP systems are likely to show poor reactivity at the strategic level and having excessive amounts of reactivity at the action level, sometimes no reacting to high-level changes in the situation until a low-level action fails, or discarding an entire plan because a single action failed.

Aha, Molineaux, and Ponsen where one of the first few to create a CBP system for an RTS, they created a system that extended the “dynamic scripting” to select tactics and strategy based on the current situation. Their system was able to play against a non-static opponent instead of having more training every time the opponent changed. By abstracting states into a state lattice of possible orders in which buildings are constructed in a game combined with a small set of features, and abstracting actions into a set of tactics generated for each state they reduced the complexity of the state and action spaces. Their system was able to improve its estimate of the performance of every tactic in any given situation over many games, and eventually learn to consistently beat all of the tested opponent scripts.

Hierarchal Planning is the next technique mentioned by the author for Strategic Decision Making. It works by breaking up a problem hierarchically, although planning systems are able to deal with different parts of the situations separately at different levels of abstraction therefore reducing the complexity of a problem, there is the problem where you could create a new issue in coordination between the different levels. So a hierarchical plan maps well to the hierarchy of goals and sub-goals that are usually in an RTS, anywhere from the highest level goal like winning the game, to the lowest level goal that is related directly to an in-game action.

Behavior Trees

The next technique mentioned is behavior trees, which are hierarchies of decisions and action nodes, commonly used in the game industry to define behaviors in a game for agents. They’re very accessible and easy to understand for a beginner or no programming skills, they can be made and edited using visual tools. Behavior trees are very hierarchical and they cover a wide range of behavior so it's a good thing the structure of encourage the reuse because a tree defining specific behavior can be attached to another tree in a multitude of positions or it can be customized incrementally by adding nodes. Behavior trees were used my Palma to enable direct control of a case-based planner's behavior. Through a planner machine learning could be used to create complex and robust behavior while allowing the designers of the game to change particular parts of the behavior by using a behavior tree instead of an action or whole plan. This allows custom behavior for specific scenarios to be defined or to fix incorrectly learned behavior and modify it as needed.

Goal-Driven Autonomy

Another technique mentioned in the review for Strategic Decision is Goal-Driven Autonomy which is a model in were "an agent reasons about its goals, identifies when they need to be updated, and changes or adds to them as needed for subsequent planning and execution". By actively reasoning about and reacting to why a goal is doing well or not the Goal-Driven Autonomy (GDA) addresses high and low level problems experience by Case Base Planning. A GDA system for StarCraft using behavior language which is able to create plans with expectations about the outcome from said plans was described by Weber, Mataes and Jhala. If there were an unexpected situation or event that occurs, the system can record it as a discrepancy and generate an explanation for why it occurred. It would then form new goals to revise the plan which allows the system to react appropriately to unexpected event. I can also reason simultaneously about multiple goals at differing granularities. Initially it could learn goals, expectations or strategies so knowledge had to be input and updated manually, but later improvements allowed these to be learned from demonstration. This system was later used in

(AIIDE) StarCraft AI competition entry EISBotand was also evaluated by playing against human players on a competitive StarCraft ladder called International CyberCup (ICCup)12 where it was determined to be better than 48% of players.

State Space Planning

State Space planning is another technique used for Strategic Decision. In artificial intelligence, state space planning is a process used in designing programs to search for data or solutions to problems, the state space is a collective term for all the data that will be searched, the goal being to find a procedure or the best possible procedure for achieving the goal through a finite number of possible procedures. Automated planning and scheduling are branches of classic AI research from where heuristic state space planning techniques have been adapted for planning in RTS game AI. In those problems, an agent is given a start and goal state, and a set of actions which have preconditions and effects. Agent then must find actions that can achieve the goal from a starting state. There have been similar techniques applied to other genres successful such as GOAP(Goal Oriented Action Planning) which allows agents to automatically select the best actions for their current situation in order to satisfy a set of goals, hopefully resulting in more varied, complex, and interesting behavior but also keeping code more reusable and maintainable lets agents select the most best course of actions automatically for their current situation in order to satisfy a set of goals, ideally resulting in more varied, complex, and interesting behavior, while keeping code more reusable and maintainable, although there are problems in which the RTS version does fix. In the RTS domain, unlike GOAP, it is able to focus on a single goal like finding a plan to build a needed amount of units and building and resource gather in the quickest amount of time possible. Because the RTS domain is simplified by abstracting things like resource collection to an income per worker, or assuming the placement of buildings and movement of units always takes a constant amount of time, or just completely ignoring opponents. Ignoring opponents can be an okay thing to do in the beginning of a game because there is generally little to no opponent interaction like fights for resources. By doing so means the planner doesn't need to deal with uncertainty or external influences on the state. However both methods would still require expert knowledge to provide a goal state for them to pursue.

Evolutionary algorithms (EA) are also mentioned in the review, an EA is a subset of evolutionary computation and it uses mechanisms inspired by actual biological evolution, like reproduction, selection, mutation, and recombination. EA searches for an effective solution to a problem by looking and judging different possible solutions and combining components that are "high-fitness" potential solution to obtain better solutions. This technique isn't used frequent in RTS AI, it has been applied well in StarCraft 2. The technique is used to optimize partially defined build orders, given a set of resources and units to be gathered and ordered to do things in a finite amount of time in the match, the software searches for the best way to reach those goals by getting the fastest or least resources draining way.

Cognitive structures are also used to develop AI; this structure's goal is to get greater understanding of how humans reason so that we could create more human-like AI. Cognitive Structure AI is used to model a reasoning mechanism on how humans are thought to operate. To do this we need to be able summarize various results of cognitive psych in a comprehensive model. The results do need to be in a formalized for so that they could be the basis of a computer program. These models could then be used to refine a comprehensive theory of cognition even further. One of the most popular Cognitive structures is SOAR (a Cognitive structure made by John Laird, Allen Newell, and Paul Rosenbloom at Carnegie Mellon University).

Spatial Reasoning

Spatial Reasoning is a technique that has research that's very important in RTS games. They have a very suitable domain for RTS games in a controlled environment. AI agents in RTS games have to be able to reason about the actions of large number of objects that all have a variety of properties, all of the moving as the game progresses, some of them controlled by an opponent in a dynamic environment. For the most part, people are able to think quickly and act logically despite the complexity of the problem, and even predicting strategies on little information or intuition.

There are many areas of research for AI, and lot of researchers have decided to narrow their focus to just determining an opponent's strategy, which as you can guess is still really difficult and complex due to the nature of the game because of limited information about the opponent's actions, knowing the rules, attributes of your units, the terrain, etc. etc. So focusing on the plan recognition and learning aspect to strategic decision AI can cut out a lot of the meat but still give something meaningful to work with. Plan recognition and learning methods can be broken down into several categories; deductive (finds a plan by comparing the situation of the expected behavior for known plans), abductive (finds a plan by making assumptions that are sufficient to explain the observations), probabilistic (tries to find the most expected outcome of a situation based on statistics and expected probabilities), and case-based (Just cased based reasoning) techniques, learning by observation/Imitation learning (analyses traces, or recorded events within a game, the system could learn from the player behavior instead of needing programmers to specify it's behavior manually) and learning from demonstration (Imitation learning with annotations for context).

Research areas

Overall AI research, although robust and varied by multiple techniques and approaches still has many areas in which it can grow. Comparing industry AI (which has mixed opinions from gamers of all experience levels) and academic AI and why there's a gap by comparing goals, incentives and risk. Financial incentive for a lot of industry seems to be a problem, RTS games are seen more of a multi-player genre where people are expected to play against human opponents but there's a lot to be gained by experimenting with various AI that utilize different techniques. There's also multi-scale AI & cooperation, getting different aspects of RTS AI to work together under a hierarchy of some sorts, just like the system which units in an RTS game work under. AI is a very active field in Computer Science and as reliance in Automation grows, making better, more robust AI will be crucial.

Conclusions

This paper went over several reviewed techniques for developing AI for RTS games. The two major areas for these techniques are Tactical-Decision and Strategic Decision Making systems, all of the techniques varying but also applied and worked with other techniques by researchers using open source programs. There was also a brief overview of plan recognition and learning, as well as potential research areas. The AI field is growing rapidly with more and more usage is applicable due to growth of automation, and although RTS AI has been developed quite a bit, because of its unique and challenging domain and space problem, can be developed into something a lot better than what it currently is and then by extension help other sub fields. Two big ways to accomplish that is by closing the gap between academia and the games industry (despite both groups having different goals there is a common ground of creating new, exciting and challenging AI that will get attention of players of all skills) and further cooperation between different researchers and scholars (one noteworthy example is the AI competition for StarCraft).

Works Cited

Abdelrahman Elogeel, Andrey Kolobov, Matthew Alden, and Ankur Teredesai. 2015. Selecting Robust Strategies in RTS Games via Concurrent Plan Augmentation. In Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems (AAMAS '15). International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 155-162.

Buro, Michael. "Call for AI research in RTS games." Proceedings of the AAAI-04 Workshop on Challenges in Game AI. 2004.

Chen Si, Yusuf Pisan, and Chek Tien Tan. 2014. A Scouting Strategy for Real-Time Strategy Games. In Proceedings of the 2014 Conference on Interactive Entertainment (IE2014), Karen Blackmore, Keith Nesbitt, and Shamus P. Smith (Eds.). ACM, New York, NY, USA, , Article 24 , 8 pages. DOI=10.1145/2677758.2677772 http://doi.acm.org/10.1145/2677758.2677772

Quake Isn't a Shooter | Game Design Short Talk. Perf. Alex Mandryka. Game Whispering, 2014.

Robertson, G. (2014). A Review of Real-Time Strategy Game AI. Association for the Advancement of Artificial Intelligence.

Stene, Sindre Berg. "Artificial intelligence techniques in real-time strategy games-architecture and combat behavior." (2006).

Wintermute, Sam, Joseph Xu, and John E. Laird. "SORTS: A human-level approach to real-time strategy AI." *Ann Arbor* 1001.48 (2007): 109-2121.