Selected Topics

Daniel E. Bruce

October 12, 2011

Contents

Chapter 1

Introduction

This document covers a selection of the most relevant previous work related to my Thesis with the working title “Automated Synthesis of NPC AI with ADATE”, on the topic of creating AI for game Non-Playing Characters with evolutionary programming, using the ADATE system.

There are two main areas that have to be covered, specifically the topics of Artificial Intelligence and Automatic Programming.

The work existing on the topic of Artificial Intelligence in games can be roughly divided between work describing various implementation techniques, and work describing how to make game AI more entertaining within the constraints of existing implementation techniques.

The line between these two subtopics is very fuzzy and gradual, so some judgment has been employed when putting the work into one category or the other, in some cases the choice is purely symbolical, such as when the work describes a new implementation technique created for the express purpose of increasing entertainment value (for example Khoo et al., 2002).

There is also some work on the more specific subtopic of using Automatic Programming to improve game AI, either for skill or entertainment value, which relates directly to the topic of my thesis.

Chapter 2

AI techniques

There is constant work on new AI techniques, both to make AI more efficient, and to make AI seem smarter or more human. Familiarizing oneself with some of these will allow better ability to choose an implementation technique that fits with the goal of allowing automatic programming.

There is some differences between the techniques used in "traditional" AI and game AI, primarily based on the performance of the various methods and their focus on ability to solve problems and be entertaining, respectively. The two areas seem to be converging, however, as traditional AI is starting to shift from the goal of being "really smart" to making computers seem more human, have an ability to learn and adapt, and maybe act like assistants, as detailed in Ramos et al. (2008).

When it comes to games (both video games and traditional games), the biggest difference in how to do the AI comes from whether or not the players have perfect information and whether there is chance involved. A perfect-information game will require a completely different approach to AI than a hidden-information or a stochastic game, and this will heavily influence what type of AI is utilized, as mentioned in Schaeffer and Van den Herik (2002, pg. 4-5).

2.1 Traditional AI techniques

As noted in Munakata (2008, chap. 1) there are six main areas of AI techniques, namely:

1. Symbolic AI (also "traditional AI")
2. Neural networks
3. Genetic algorithms

4. Fuzzy systems
5. Rough sets
6. Chaos

Symbolic AI is the umbrella term for the traditional methods dating from the field's inception, focusing on abstracting the world and applying logic and rules to reach decisions.

The following two methods (Neural Networks and Genetic Algorithms) are different reactions to the insufficiencies of Symbolic AI by attempting to more closely modelling biological processes.

The remaining three methods are more recent developments which attempt to attack the problem from a different angle, collectively called "soft AI" due to their focus on not giving "hard answers" and use of uncertainty. These methods are less relevant to the topic of my thesis, and as such will not be covered extensively.

2.1.1 History

Before going further it would be useful to go through the history of Artificial Intelligence, to see its roots, and how closely it has developed alongside the field of computer science itself. (Buchanan, 2002; Luger, 2005)

The field has traces back to antiquitiy with greek myths about artificial beings, such as the golems and homunculus, and to the writings of many ancient scholars (among others, Aristotle and Euclid) on the topic of reasoning, logic and the mind. Further there have been many people working on mechanical automaton, which tried to emulate human behaviour.

There was early work on intelligent machines from the very point where there existed computers, but computer AI as a field didn't really come into existance until 1950, through a pair of papers.

1. A very important paper was published by by Alan Turing named "Computing machinery and intelligence" (Turing, 1950), where the he defined the "Turing Test" as a way to test whether a machine was truly intelligent.
2. Claude Shannon publishes a paper on programming a computer to play chess, by representing it as a search problem. (Shannon, 1950)

The term "artificial intelligence" wasn't coined until six years later when John McCarthy used the term for the first AI conference in 1956, where the first running AI program (the Logic Theorist) was demonstrated.

It didn't take long for AI programs to challenge humans at board games, with the first program to challenge a human world champion being made in 1962 by

Arthur Samuel, aimed at the game of checkers and utilizing machine learning to improve its performance.

Further works into problem solving led to SAINT, that solves calculus at a freshman level (Slagle, 1963); ANALOGY, that solves the kind of analogy questions found on IQ tests Evans (1964); and ELIZA, which can simulate conversation (Weizenbaum, 1966).

Later, there was work into “knowledge-based programs” for artificial reasoning, which created programs able to interpret mass spectra of chemical compounds, solve integration problems in math, and play chess good enough to reach a class-C rating.

In 1969, the beginnings of Neural Networks were appearing with Minsky and Papert (1969) which defined Perceptrons, while several papers on natural language understanding were published during the next few years, along with the introduction of “expert systems” using rule-based programming, and the creation of the first computer to make a scientific discovery, the Meta-Dendral learning program (Buchanan et al., 1976).

Through the 70s and into the 80s, an explosion of well-known AI programs occurred. Many “expert systems” were made that were capable of reasoning within a limited space on the same level as a human expert, using the traditional symbolic methods, before Neural Networks become widely used using backpropagation (first introduced in WERBOS (1974)).

Around the 80s, the American Assosication for Artificial Intelligence (AAAI, now named the Association for the Advancement of Artificial Intelligence after a name change in 2007), which hosts many conferences and symposia each year, and supports several journals on AI.

Another such boom happened in the 90s with major advances in all areas of AI, with significant demonstrations in machine learning, intelligent tutoring, case-based reasoning, multi-agent planning, scheduling, uncertain reasoning, data mining, natural language understanding and translation, vision, virtual reality, games, and other topics.

The 90s also had two significant events in games being played by AI, Deep Blue beat Garry Kasparov in 1997, and TD-Gammon was written, that played backgammon at championship-level. In addition, AI was starting to see use in cataloguing the internet.

This brings us to today, with AI seeing use in toys (such as robotic pets), and other forms of entertainment (most notably video games), but is still far away from the goal of creating a human machine.

2.1.2 Symbolic AI

This subfield of AI is also known as “traditional” or “classic” AI, and was the approach that was used during the inception of AI, and is still heavily used today. It is characterized by a top-down focus on logic and reasoning, and relies on a symbolic description of the world, such as a set of rules, and is thus said to be "knowledge-based".

Techniques included under Symbolic AI are knowledge-based systems, logical reasoning, symbolic machine learning, search techniques, and natural language processing.

2.1.3 Neural Networks

2.1.4 Genetic Algorithms

2.1.5 Soft AI

2.2 Game AI techniques

The approach to AI used in video games (as opposed to more traditional board-/card games) is very different from the academic approach, where AI programs can take a long time to reach a decision, might require massive amounts of resources and usually have as a goal to perform as excellently as possible.

In games the AI actors might only have a handful of microseconds of CPU time available to reach a decision lest they impact the performance of the game, and the AIs must fulfill the twin goals of being challenging (but beatable by the majority of players) and entertaining (employ novel methods, and seem human-like).

In addition, game AI is a comparatively recent field when compared to the volume of research on AI work using traditional board and card games, which evolved alongside computer science, as "solving" board games was one of the driving forces behind computer science, as mentioned in Schaeffer and Van den Herik (2002).

2.2.1 History

Before covering specific techniques, it is prudent to go through a history of AI as used in games, to show the field's evolution in contrast to the field of traditional AI which has a long history of strong scientific focus. The majority of this information comes from Tozour (2002).

When it comes to video game AI, the methods employed have been, and still are, marked by the heavy performance requirements and the fact that very little emphasis has been put on AI sophistication until recently, as quipped about in the following quote:

> Even today, game AI is haunted by the ghosts of Pac-Man's Inky, Pinky, Blinky and Clyde. Until very recently, the video game industry itself has done all too little to change this perception.

Continuing on through the article, it is explained that many of the popular early games used very crude AI, basically just a handful of simple rules, with the exception of games that just digitized board games with well established AI research, such as chess.

Sophisticated AI in video games was first embarked upon with turn-based strategy games (such as *Civilization*), then real times strategy games (such as *Age of Empires 2: The Age of Kings* and *WarCraft II*). Further, good AI started showing up in First Person Shooter games (*Half-Life* and *Unreal: Tournament*) that showed tactical ability and the ability to model several actors simultaneously, while *Thief: The Dark Project* had actors that emulated sense of sight and sound in a human-like fashion, and *SWAT 3: Close Quarters Battle* featured randomized AI parameters that allowed each actor to have a slightly different personality every time the game is played.

After that the variety of different AI exploded, with games focusing entirely on watching AI "life" grow, with *SimCity* and *The Sims* being very well known staples, and *Creatures* which is famous for being one of the few games that actually uses a biological model for its "Norns", both modelling biological processes with great detail and using neural networks for the AI (see Grand et al., 1997). Other games fit into the "God games" category, such as *Populous* and *Dungeon Keeper*, alongside *Black & White*, which has the distinction of being the first major game to focus the player's attention entirely on the game's AI capabilities, and including a learning AI, a topic which is considered to be the next "Big Thing" in gaming.

It should also be noted that for all the recent complexity, it is still the case that the game AI community favours simple "traditional" methods implemented through finite state machines, decision trees and rule systems, for their excellent performance and relative simplicity. These are then further augmented to add human-like behaviour, simulating planning and learning (see Isla and Blumberg, 2002; Khoo et al., 2002; Mateas and Stern, 2002; Orkin, 2003).

2.2.2 Traditional game AI

Traditional game AI or "simple AI" is what game AI started out as, and still to this day mostly uses. It's based upon simple rules or simple logical systems, and has traditionally used a sampling of simple techniques, such as simply hard coding the AI in a single routine, utilizing Finite State Machines, or using rule-based systems. In addition, pathfinding has always been a topic with AIs.

The common factor in traditional game AI techniques is that they tend to be static, the NPCs are only capable of the things they were programmed to do beforehand, and has little capability of learning or planning.

Hard-coded AIs

The first game NPCs utilized simple hard-coded AIs, which were basically a short routine that ran every tick of the game with simple behaviours for the opponents, not following any formalized methods for AI. This method is still used to this day, as it's easy to create, and usually many minor opponents in games don't need more sophistication than this method creates.

A good example of this technique is Pac-Man's ghosts, which used simple pathfinding, and a selection of "modes" for the ghosts globally, then four different targeting rules to give each ghost a "personality" as their entire AI (Birch, 2010; Pittman, 2011). Even this very simple AI creates interesting behaviours and engaging gameplay which propelled Pac-Man to one of the most recognized games in video game history, evidence that you don't need fancy high-powered neural nets to create interesting games.

Finite State Machines

Finite State Machines (FSM) are the most used game AI technique, although the FSM used by game developers do not necessarily work the same as the ones described by Computer scientists. They take certain shortcuts which violate the traditional definition of FSM, which make them more applicable to games (Rabin, 2002).

FSM are used to formalize an NPC's behaviour in a simple way, as states and transitions. States correspond to a specific behaviour, whereas a transition correspond to a change in behaviour due to an event in the game. Using this, one can easily map up simple behaviours that allow an NPC to act in a manner that can be deemed "intelligent," as long as the programmer thinks of all the transitions necessary, and makes them seem natural.

This method of AI creation has the benefits of being simple to understand, create and debug, as well as being very versatile by virtue of being a general method that can lend itself to most any problem. Of course, there are downsides to FSM as well. They can easily grow out of hand in more complex AI creations, and they don't have the capability of combining states (so you can't have an NPC be in the *run away* and *attack* states at the same time, to have it do a tactical retreat, without explicitly programming the option in).

For a more thorough coverage of implementing an FSM, one can consult Rabin (2002) or Kirby and Books24x7 (2011, chap. 3).

Rule-based systems

As with FSM, Rule-based systems stem from traditional AI research, but is used in a more loose form in video games (Christian, 2002). The basic idea of rule-based systems is that of a database (the data can be information, actions or other things), where each piece of data has a “matching rule” which is used by the system to infer which pieces of data apply to current situation.

In the context of games, this boils down to series of rules coupled with actions or behaviours, many of which can apply at the same time, which together make up the AI for the given NPC. These are formally named *reaction rules*, and are just one type of rule, with the other major one being *consequent rules*, dealing mostly with information.

Usually there is some method of discerning which rules are the most relevant at the current time, to prevent the NPC from doing too many things at once (many simple AIs only allow the NPC to do a single action per tick), which can range from randomly choosing a rule, to weighting each rule based on its specificity and choosing the most applicable one (Freeman-Hargis, 2002).

The biggest benefit of this is that you can create a good set of behaviour with a comparatively small amount of rules, so you aren’t suspect to “state explosion” as in an FSM, where adding a single new state will result in a cascade of new transitions having to be written to handle every single case where the NPC can switch in and out of the new state. In a rule-based system you just add the new rule and write one matching function for it, and let your system handle the rest.

Of course, the downside in this case is that creating a good AI like this requires a bit more thinking to create a set of actions that acts well in the most common cases, and still handles uncommon and unexpected cases decently well, since one can’t write a rule for every situation. This usually requires a human with some expertise to either formulate or actually code the rules.

More info on writing rule-based systems can be found in Kirby and Books24x7 (2011, chap. 4).

Pathfinding

2.2.3 Towards more human game AI

2.2.4 Advanced game AI

Chapter 3

NPC Entertainment value

When AI is used for NPCs in games, the main goal is not to perfectly emulate a human, nor to create the most skilled opponent possible, but rather to create an opponent that SEEMS human, and possesses behaviour that creates an entertaining experience, without being unduly challenging.

As noted in Yannakakis and Hallam (2004), one of the primary factors in a game's entertainment is having "interactive" opponents, and in that vein here is a slew of work on increasing how human-like game AI is behaving, as a way of increasing entertainment value, which involves planning behaviours, anticipation, learning and adapting (Orkin, 2003, 2004; Spronck, 2005; Yannakakis and Hallam, 2009).

All of this goes into an approach called character-based AI (Isla and Blumberg, 2002), which is gaining grounds in the game industry as a way to give the game's AI agents a reasonable facsimile of human behaviour and personality, or a "character", to make game opponents and allies more entertaining. The term is applied to AI which collect many techniques and behaviours that make the AI seem more human, like the aforementioned planning, learning and adapting, but also the ability to sense patterns, the ability to anticipate, and a certain model of the AI actor's perception that limits its available information to a more "realistic" amount.

There is also work going into methods of quantifying entertainment in games, with articles proposing theories on what makes a game fun (e.g. Koster, 2004; Lazzaro, 2004; Malone, 1981; Read et al., 2002), and others proposing methods for augmenting or optimizing entertainment value in games (e.g. Yannakakis, 2008; Yannakakis and Hallam, 2007, 2009).

Chapter 4

Automatic Programming

Chapter 5

Evolving Game AI

Chapter 6

Conclusion

Bibliography

C. Birch. Understanding pac-man ghost behavior. http://gameinternals.com/post/2072558330/understanding-pac-man-ghost-behavior, December 2010.

B. Buchanan. Brief history of artificial intelligence. In *American Association of Artificial Intelligence*, volume 26, 2002.

B. Buchanan, D. Smith, W. White, R. Gritter, E. Feigenbaum, J. Lederberg, and C. Djerassi. Applications of artificial intelligence for chemical inference. 22. automatic rule formation in mass spectrometry by means of the meta-dendral program. *Journal of the American Chemical Society*, 98(20):6168–6178, 1976.

M. Christian. a simple inference engine for a rule-based architecture. *AI. game. programming. wisdom*, pages 313–320, 2002.

T. Evans. A program for the solution of a class of geometric-analogy intelligence-test questions. Technical report, DTIC Document, 1964.

J. Freeman-Hargis. Rule-based systems and identification trees. http://ai-depot.com/Tutorial/RuleBased.html, 2002.

S. Grand, D. Cliff, and A. Malhotra. Creatures: Artificial life autonomous software agents for home entertainment. In *Proceedings of the first international conference on Autonomous agents*, pages 22–29. ACM, 1997.

D. Isla and B. Blumberg. New challenges for character-based ai for games. In *AAAI Spring-Symposium on AI and Interactive Entertainment*, 2002.

A. Khoo, G. Dunham, N. Trienens, and S. Sood. Efficient, realistic npc control systems using behavior-based techniques. In *Proceedings of the AAAI Spring Symposium on Artificial Intelligence and Interactive Entertainment*, pages 46–51, 2002.

N. Kirby and I. Books24x7. *Introduction to game AI.* Course Technology PTR/-Cengage Learning, 2011.

R. Koster. *A theory of fun for game design.* Paraglyph press, 2004.

N. Lazzaro. Why we play games: Four keys to more emotion without story. In *Game Developers Conference*, 2004.

G. Luger. *Artificial intelligence: Structures and strategies for complex problem solving*. Addison-Wesley Longman, 2005.

T. Malone. What makes computer games fun?(abstract only). In *Proceedings of the joint conference on Easier and more productive use of computer systems.(Part-II): Human interface and the user interface-Volume 1981*, page 143. ACM, 1981.

M. Mateas and A. Stern. A behavior language for story-based believable agents. *IEEE Intelligent Systems*, pages 39–47, 2002.

M. Minsky and S. Papert. *Perceptrons: An introduction to computational geometry*, volume 1988. MIT press Cambridge, MA, 1969.

T. Munakata. *Fundamentals of the new artificial intelligence: neural, evolutionary, fuzzy and more*. Springer-Verlag New York Inc, 2008.

J. Orkin. Applying goal-oriented action planning to games. *AI Game Programming Wisdom*, 2:217–228, 2003.

J. Orkin. Symbolic representation of game world state: Toward real-time planning in games. In *Proceedings of the AAAI Workshop on Challenges in Game Artificial Intelligence*, 2004.

J. Pittman. The pac-man dossier. gamasutra (2009). http://home.comcast.net/~jpittman2/pacman/pacmandossier.html, june 2011.

S. Rabin. Implementing a state machine language. *AI. game. programming. wisdom*, pages 314–320, 2002.

C. Ramos, J. Augusto, and D. Shapiro. Guest editors' introduction: Ambient intelligencethe next step for artificial intelligence. *IEEE Intelligent Systems*, pages 15–18, 2008.

J. Read, S. MacFarlane, and C. Casey. Endurability, engagement and expectations: Measuring children's fun. In *Interaction Design and Children*, pages 189–198. Citeseer, 2002.

J. Schaeffer and H. Van den Herik. Games, computers, and artificial intelligence. *Artificial Intelligence*, 134(1-2):1–8, 2002.

C. Shannon. Programming a computer for playing chess. *Philosophical magazine*, 41(7):256–275, 1950.

J. Slagle. A heuristic program that solves symbolic integration problems in freshman calculus. *Journal of the ACM (JACM)*, 10(4):507–520, 1963.

P. Spronck. *Adaptive game AI*. UPM, Universitaire Pers Maastricht, 2005.

P. Tozour. The evolution of game ai. *AI game programming wisdom*, pages 3–15, 2002.

A. Turing. Computing machinery and intelligence. *Mind*, 59(236):433–460, 1950.

J. Weizenbaum. Elizaa computer program for the study of natural language communication between man and machine. *Communications of the ACM*, 9 (1):36–45, 1966.

P. WERBOS. Beyond regression: new fools for prediction and analysis in the behavioral sciences. *PhD thesis, Harvard University*, 1974.

G. Yannakakis. How to model and augment player satisfaction: A review. In *Proceedings of the 1st Workshop on Child, Computer and Interaction, Chania, Crete, ACM Press*, 2008.

G. Yannakakis and J. Hallam. Interactive opponents generate interesting games. In *Proceedings of the International Conference on Computer Games: Artificial Intelligence, Design and Education*, pages 240–247. Citeseer, 2004.

G. Yannakakis and J. Hallam. Towards optimizing entertainment in computer games. *Applied Artificial Intelligence*, 21(10):933–972, 2007.

G. Yannakakis and J. Hallam. Real-time game adaptation for optimizing player satisfaction. *Computational Intelligence and AI in Games, IEEE Transactions on*, 1(2):121–133, 2009.