Computational Intelligence and Game Balance

Mihail Morosan
Centre for Doctoral Training in Intelligent Games and Game Intelligence
School of Computer Science and Electronic Engineering
University of Essex, UK
mmoros@essex.ac.uk

Riccardo Poli
School of Computer Science and Electronic Engineering
University of Essex, UK
rpoli@essex.ac.uk

Daniel Kudenko
Department of Computer Science
University of York, UK
kudenko@cs.york.ac.uk

Abstract

Games have an ongoing requirement to exhibit the ability to react to player behaviour. Changes to their mechanics and available tools are constantly required. This is to keep their audience both entertained and feeling as if their strategic choices and in-game decisions have value. Game designers spend a lot of time both gathering data and analysing it to introduce minor changes that bring their game closer to a state of balance, a task with a lot of potential not completely tapped by the literature. Work is being done to better understand the best methods of automatically altering video games to create the most enjoyable experiences for players.

Introduction

Artificial intelligence has the potential to aid humans in almost every area available, from high-confidence predictions of the weather for agriculture, to better diagnosis of medical conditions and better design of hardware and software. Finding new ways to further research of video games and game AI, or, alternatively, using video games to further the area of AI or human understanding of the world, is the main goal of our research.

Designers spend weeks or months fine-tuning their games to be enjoyable by their targeted demographic, sometimes never managing to finish the task of balancing their games, as it might not even be possible or desirable to do so. Computational intelligence has a track record of successfully finding patterns in data that humans might not have immediately noticed, or speeding up tasks that humans might consider monotonous or time-consuming. Our research seeks to apply these techniques to find how humans and artificial intelligence can work together in the area of game design and game balance.

Game balance

The literature presents different notions of "balance", perhaps because different games, or even genres, naturally lead to (or need) slightly different versions of the concept. The most generic definition, supplied by Schreiber (Schreiber 2010), is that game balance "is mostly about figuring out what numbers to use in a game".

For the purpose of our research, a definition was required. A balanced game is a game where, over many playthroughs, none of the available tools or game mechanics are objectively more desirable or undesirable to players, resulting in an environment where there are no dominant strategies and no game mechanic is redundant.

Methodology

Video Games

StarCraft Work started using a commercial title, *StarCraft*, to highlight the potential use in industry. *StarCraft* is a real-time strategy game from Blizzard Entertainment, known for its e-sports scene, but more importantly, its AI development community. The game involves managing one of three distinct races, each with its own available units, strategies and strengths.

Experiments were done to control the dominance of a given strategy using evolutionary algorithms. The results are promising and have prompted further work in better understanding how games impact the field of genetic evolution. Genetic algorithms have proven to be capable of finding solutions that fit a game designer's requirements, even though they don't have an understanding of the game or its mechanics.

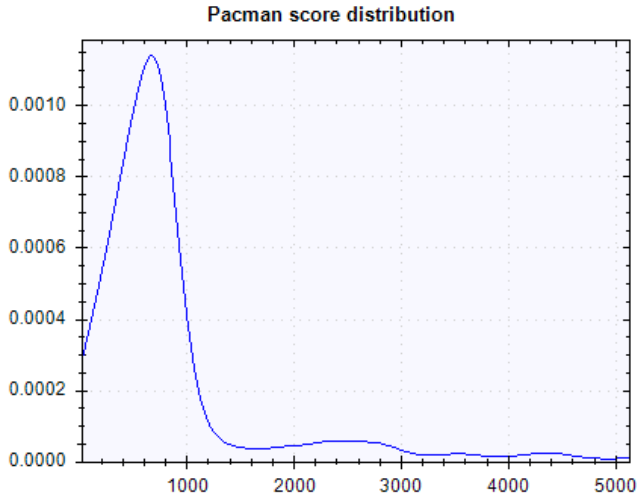

Figure 1: Sample distribution of scores achieved by a rules-based *Ms PacMan* agent, highlighting high variance

Ms PacMan A classic arcade game, *Ms PacMan* is a common choice for AI research due to its perceived simplicity, yet potential for complex work to be done. It is an asymmetric game, where one player, controlling the PacMan, attempts to collect as many pills while being chased by hostile ghosts. When controlled by the computer, the ghosts have slightly predictable, yet affected by stochastic elements, behaviour. This means that even deterministic PacMan agents will not achieve the same scores over many games (Figure 1).

Given access to the source code and, as a result, the ability to run simulations hundreds of times faster than with *StarCraft*, as well as the task to better understand how a game's stochastic elements impact the evaluation of various parameters or intelligent agents, *Ms PacMan* was a great choice to do further research on.

AI techniques employed

Evolutionary Algorithms and Genetic Programming Genetic programming (GP) makes use of evolutionary algorithms to build solutions for problems with complex definitions (Poli, Langdon, and McPhee 2008). Simple tasks can include function regression (Koza 1994), while complex tasks could include entire behaviour trees for survival strategies (Sipper 2008). By using methods found in nature, particularly natural selection, evolutionary algorithms take many generations of solutions, test them, then discard those deemed to be not good enough. This results in ever-improving solutions.

Given a different random starting seed, GP is extremely likely to offer completely different solutions every time it is run, as the search space for many tasks is immense. Many have attempted to improve the algorithm over time (Fernandez-de Vega et al. 2004), to minimise the generation of weak members from the get-go, but a general solution is not possible, as each task that GP tries to tackle brings its own complexities and requirements to the table.

One interesting application is in the evolution of entire neural networks (Stanley, Bryant, and Miikkulainen 2005) (Fogel 2001). Rather than collecting immense amounts of data for the training of a neural network, evolution can just search for one that is close enough to the game's requirements. This allows for agents that are potentially just as good, but with a lot less effort in data collection.

Some of the work done with *StarCraft* and *Ms PacMan* might benefit controlling noise in sample-based fitness functions and, by extension, evolutionary algorithms, with applications beyond video games.

Monte-Carlo Tree Search (MCTS) MCTS is a relatively recent contender in the AI world, taking the lead in many AI competitions early in its lifetime. It began simply enough by becoming competent in board games such as Go (Chaslot et al. 2006), but soon enough it established a foothold and got employed for video games as well (Champandard 2014).

One of the algorithm's main strengths is its flexibility in how much time it needs. It can end its search after 1 second or after 30 milliseconds. The longer it is left to calculate, however, the better the solution it can offer. It does however require, at this time, that there exists a relatively accurate and fast way of simulating what the player's various actions would do to the game state, as MCTS often requires hundreds and thousands of moves to be planned in the future.

Some of the best agents for *Ms PacMan* make use of MCTS in some manner and are being used in our research due to their performance. These agents are used alongside GA and GP to solve the task of automated game balancing.

Results

Results so far do show a lot of promise, as the methodology proposed could not only be useful in balancing a game itself, but could aid in calibrating the performance of other intelligent agents to displaying a desired level of skill and expertise. Instead of evolving game parameters, one could evolve the AI parameters and run a virtually unchanged set of fitness tests, hoping to optimise it to behave at a required level, for instance to create different difficulty levels for games.

The *StarCraft* study has highlighted that genetic algorithms are viable for use in game design, complementing human intuition and offering a tool for future problem-solving of complex scenarios. They still require optimisations of their own to become usable in the industry and these experiments have highlighted many of the areas that need improving.

Game balance should also be studied and better understood, as it impacts player enjoyment in ways that few have clearly defined. Developers constantly want their players to keep playing their games, and will resort to constantly updating their games with new content, some of which might break the game. Automating many of the processes used can only result in quicker access to information and, potentially, better decisions from the game designers. This paper represents a first step in this direction.

References

Champandard, A. J. 2014. Monte-Carlo Tree Search in TOTAL WAR: ROME II's Campaign AI.

Chaslot, G.; Saito, J. T.; Uiterwijk, J. W. H. M.; Bouzy, B.; and van den Herik, H. J. 2006. Monte-Carlo strategies for computer go. In *Belgian/Netherlands Artif. Intell. Conf.*

Fernandez-de Vega, F.; Gil, G. G.; Pulido, J. A. G.; and Guisado, J. L. 2004. Control of bloat in Genetic Programming by means of the Island Model. In *Parallel Probl. Solving from Nat. - PPSN VIII*, volume 3242, 263–271.

Fogel, D. B. 2001. *Blondie24: Playing at the Edge of AI*. Morgan Kaufmann.

Koza, J. R. 1994. Introduction to genetic programming. In *Adv. Genet. Program.* 21–42.

Poli, R.; Langdon, W.; and McPhee, N. 2008. A field guide to genetic programming. *books.google.com*.

Schreiber, I. 2010. Game Balance Concepts.

Sipper, M. 2008. Evolving Game-Playing Strategies with Genetic Programming. *ERCIM News* 64:28–29.

Stanley, K. O.; Bryant, B.; and Miikkulainen, R. 2005. Evolving Neural Network Agents in the NERO Video Game.