EVO-CBG - An Evolutionary System for Automatically Generating Character Behaviours for Game Environments

James Patten and Conor Ryan

Biocomputing and Developmental Systems Group,
Department of Computer Science and Information Systems,
University of Limerick, Ireland
{james.patten,conor.ryan}@ul.ie

Abstract. In this paper we discuss the need to extend the standard types of character behaviours found in game environments if we are to create new and more compelling gaming experiences. We propose using techniques from Evolutionary Algorithms and research from Game Design to create a system that can help game designers extend standard types of behaviours. This system automatically produces behaviours that are designed to optimize predefined parameters in the game environment. We outline experiments conducted using an implementation of this system to produce behaviours for the game Ms. Pac-Man, along with providing an overview of the results obtained. Finally we discuss these results and the potential they show for our system to help game designers not only create character behaviours, but also diagnose the effect that adding or removing certain mechanics will have on the overall gaming experience.

1 Introduction

Modern digital video games[1] provide a vast array of diverse virtual environments for end-users (players) to interact with. Games are much more than merely a medium for entertainment. Apart from providing entertainment, games have been used to tackle social issues such as poverty, conflict and various other serious topics. They are also widely used as a medium for training and education.

The powerful and advanced graphics, audio and physics simulation systems found in modern games provide some of the key ingredients required to produce compelling and diverse gaming experiences. As highlighted by Rouse [1] and Crawford [2], one of the key elements that needs to be present, if something is to be considered a game, is "interaction". The player, in the context of a game, must be able to interact with the components in the game environment. The game system is required to provide the player with feedback that makes clear the effect their actions have on the game environment and its components.

[1] Simply referred to as "games" in the remainder of this paper.

Another key element that needs to be present if something is to be considered a game is "conflict". This conflict provides the player with an obstacle to overcome, a task to master. Often in games, with some form of underlying narrative, it is a Non Player Character (NPC) that provides this conflict, in the form of an adversarial relationship with the player. NPCs also often take the role of companion to the player, helping guide them to discover story elements and initiate events that advance the game narrative.

Regardless of the specific role of the NPC within the game, the duration and depth of interaction that occurs between them and the player influences the level of sophistication required of the NPC behaviour. As was highlighted by Laird [3], very often the conflict between a player and NPC is designed to be violent, as the speed and intensity of violent interactions help masks the NPCs inability to interact with the player on a more fundamental level. For this reason we have chosen to focus our research on developing methods that will help produce more sophisticated NPC behaviours. This research has the potential to allow for new and more intricate interactions between players and NPCs in game environments.

The remainder of our paper can be summarized as follows. Firstly we will give some background to the work outlined in this paper, followed by a brief description of our system for generating character behaviours for game environments. We then provide details of some experiments conducted to test the effectiveness of our system along with the results obtained. Finally we draw some conclusions based on the work we have completed to date.

2 Background

In order to understand the elements that come together to create a compelling gaming experience we must look to the discipline referred to as "Game Design".

Most interactions between players and NPCs are carefully crafted during the development stage of a game. We look to techniques from Evolutionary Algorithms (EAs), namely Genetic Programming (GP), to allow our system automatically generate sets of behaviours to optimize some predefined parameter in the game environment.

2.1 Game Design

While the creation of a game often involves a multitude of disciplines (graphic artists, software developers, audio engineers, AI engineers to name a small few), it is more often than not the responsibility of the game designer to fit the pieces together to craft the overall gaming experience.

In recent years, a number of areas relevant to our work have been the focus of academic research including, but not limited to, the components needed to produce effective game designs, the compelling nature of games and the psychological processes at work in the individual when playing a game. While a thorough discussion of this research is beyond the scope of this paper one of the

key pieces of game design research that influenced the configuration of the system detailed in this paper is the Mechanics, Dynamic and Aesthetics Framework (MDA Framework).

MDA Framework. In the MDA framework Hunicke et. al [4] provide a more formalized overview of the main components that have an influence on the overall experience a player may have in any given game environment. This formal framework attempts to explain the relationships between the mechanics, dynamics and overall aesthetics of a game. While the MDA framework can be considered from the point of view of the player, we are concerned with its application to help understand the role of the NPC behaviours in the gaming experience. In the framework the mechanics are the set of rules that are present in the game environment, the dynamics are the actions that the mechanics give rise to, while the aesthetics is the overall "look and feel" of the gaming experience.

Very often, as was outlined by Sweetser [5], failure to carefully consider design decisions when creating a game can result in unexpected dynamics or emergent behaviours. These emergent behaviours can have a positive or negative influence on the gaming experience. This further highlights the need for careful consideration of the MDA framework when designing NPC behaviours.

2.2 Evolutionary Algorithms

EAs are a set of techniques that outline a means of automatically producing solutions to a wide range of problems. EAs have their foundations in the principles of evolution outlined by Charles Darwin [6]. The EA that our system uses is GP, which was first introduced by Koza [7].

Using predefined elements, found in sets referred to as terminal and function sets, a GP system creates program trees to address a specific problem. In a program tree terminal set elements are represented by nodes with no children (i.e. leaf nodes), while function set elements are represented by nodes with a specific number of connected sub-nodes (i.e. child nodes).

The general evolutionary process, as it relates to GP, can be outlined as follows:

1. Using a combination of the function and terminal set elements, an initial population of individuals, with encoded candidate solutions to an optimization problem, is created.
2. A fitness function evaluates the potential of each individual to satisfy the optimization problem. Those with a higher potential being referred to as "fitter" individuals.
3. A selection function chooses a number of individuals from the population, with fitter individuals having a greater probability of being selected.
4. The genetic materials (program trees) of the individuals selected in 3 are used to produce new individual(s) using reproductive operations such as crossover (combination) and mutation (modification).

5. Steps 2, 3 and 4 repeat until the required number of new individuals have been created.
6. The individuals created in the above steps become the next generation of the population, with the individuals of the old generation being discarded.
7. The process repeats until an individual emerges that has the required level of fitness or a predefined number of generations has been reached.

The real power of EA techniques, such as GP, comes from their ability to tackle problems with very large search spaces in which many traditional brute force methods would fail.

3 System Overview

Our system, EVO-CBG, is designed to automatically generate behaviours for NPCs in a given game environment. The evolutionary process, of the GP subsystem, is driven by the NPCs ability to address some optimization problem. The problem, specified by the game designer in advance to running the system, relates to some element within the game environment. This is discussed in more detail in Sect. 4.

Our system uses the mechanics, defined for a particular game environment, to automatically generate the terminal and function sets used by the GP subsystem. This allows our system to automatically generate behaviours for a number of different game environments.

4 Experiments

To test the effectiveness of our system we used the game Ms. Pac-Man. For the experiments outlined in this paper the Ms. Pac-Man character took the role of the NPC and behaviours generated were to optimize a parameter specific to the Ms. Pac-Man game environment.

We chose this game for a number of reasons:

1. It has a well defined environment and is often used as a test-bed for computational intelligence research.
2. There are a limited and well defined set of game objects (pills, power pills, ghosts) that can be used as parameters to optimize.
3. The minimal set of controller options (Move up, down, left, right or stall) makes diagnosing issues with other parts of our behaviour generation system easier.

In the game the Ms. Pac-Man character moves around a maze gaining points from consuming items such as pills and powerpills. The game also contains a number of adversaries called ghosts. Ms. Pac-Man loses a life if she comes into contact with a ghost in its default state. If all lives are lost the game is over. Ms. Pac-Man can change the state of the ghosts, for a limited period of time, by

consuming a powerpill. In this non-default state points are awarded for coming into contact with a ghost.

The implementation of the game that we used for our experiment is a C# version made available by Flensbak & Yannakakis [8]. For the experiment detailed in this paper, the parameter "Ghosts Eaten" was chosen to be optimized by the system.

The results presented are from a set of 20 distinct runs. Standard GP settings were used, with 500 individuals per population and a maximum of 50 generations in a given run. A graph showing the relationship between the number of ghosts eaten and the generation number, averaged over the 20 runs, can be seen in Fig. 1. As the graph shows, our system gets better at optimizing the parameter, "Ghosts Eaten", as the generation number increases.

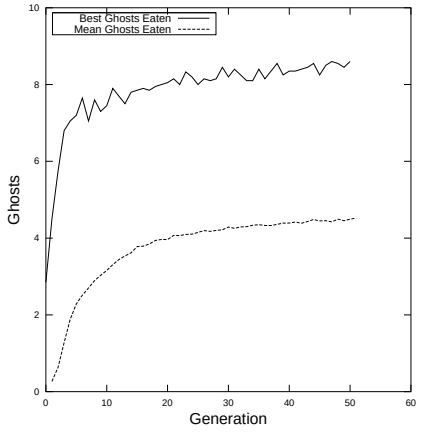

Fig. 1. Graph showing the improvement in the ability of the evolved controller to satisfy the predefined parameter in the Ms. Pac-Man game.

5 Conclusions

As can be seen from the results of our experiments using the game Ms. Pac-Man, our system does evolve behaviours that get better at optimizing a predefined parameter as the number of generations in the evolutionary cycle increases. This is an important first step in the development of more intricate behaviours for NPCs in game environments. The next step is to add the ability to specify multiple parameters in the form of ranked priorities. This will allow for more finely tuned NPC behaviours. This is something that we intend to include in future revisions of our system.

As was stated during the introduction of this paper, the advanced level of graphics, audio and physics simulation systems found in modern games presents the opportunity to advance techniques for generating character behaviours. If we wish to create new types of gaming experiences we need these new character behaviour techniques as often the NPC plays a very important role in creating the gaming experience. We believe that our system, which uses a combination of research from game design and EA techniques, can help game designers more quickly and easily develop diverse sets of character behaviours. This will, we feel, lead to the development of behaviours that will add to the overall gaming experience had by a player.

References

1. Rouse, R., Ogden, S.: Game design: theory & practice. Jones & Bartlett Learning (2005)
2. Crawford, C.: The art of computer game design. Osborne/McGraw-Hill (1982)
3. Laird, J.: Using a computer game to develop advanced AI. Computer (2001) 70–75
4. Hunicke, R., LeBlanc, M., Zubek, R.: MDA: A formal approach to game design and game research. In: Proceedings of the AAAI Workshop on Challenges in Game AI. (2004) 04–04
5. Sweetser, P.: An emergent approach to game design. PhD thesis, Ph. D Thesis. Available via http://www.itee.uq.edu.au/penny/publications (2005)
6. Darwin, C.: On the origin of species by means of natural selection, or the preservation of favoured races in the struggle for life. New York: D. Appleton (1859)
7. Koza, J.: Genetic programming: on the programming of computers by means of natural selection. The MIT press (1992)
8. Flensbak, J., Yannakakis, G.: Ms. pacman ai .net. `http://mspacmanai.codeplex.com` (August 2008)