Game theory for computer games design

Abstract

Designing and developing computer games can be a complex activity that may involve professionals from a variety of disciplines. In this article, we examine the use of game theory for supporting the design of game play within the different sections of a computer game, and demonstrate its application in practice via adapted high-level decision trees for modelling the flow in game play and payoff matrices for modelling skill or challenge levels.

Keywords: Computer game, game theory, game play, modelling, design

Introduction

Modern computer games may be composed of hundreds of thousands, or even millions of lines of program code (Ampatzoglou and Stamelos, 2010). Typically, modern computer games will involve a variety of discrete and analogue inputs from the player that will control an often rich and complex multimedia system, with many games now being online. When designing a new computer game it may be difficult to communicate design concepts to the variety of professionals involved. It can be particularly difficult to communicate the game play, or flow of the computer game to the different professionals involved (Hunicke et al; 2004, Dormans, 2011; Smith et al, 2010; Treanor et al, 2012). O'Hagan et al (2014) commented that there is no single model that serves as a best practice process model for computer game development and it is a matter of deciding which model is best suited for a particular game.

In this paper, we examine how game theory can be applied to computer games design to model the game play within particular sections of a computer game. In particular, we examine how game theory can be applied to create adapted high-level decision trees that pictorially represent the main game play pathways and incorporate symbolic representations of the fixed, portable and interactive game objects within a section of a computer game. In addition, we examine how payoff matrices can be used to support the design of the level of skill and challenge of the different sections of a computer game.

The rationale for applying game theory to computer games design, and in particular, for developing techniques for representing game theory based game play is that it can be particularly difficult to model, describe and document the actual game play within a computer game. That is to identify, model, document and communicate the main choices (or decisions) available to a game player and the interactions that will alter the game play based upon the inputs made be the game player, and the level of skill or challenge of different computer game segments. Game theory provides a more formalised mechanism for examining what actually takes place when a computer game is played. This provides a basis for a more formalised description and representation of the game play. Although formal detailed mathematical decision trees would be overly complex and unwieldy even for the simplest of computer games, they can be adapted to represent higher level views of game pathways, and by the incorporation of symbolic representations of game objects, can visually represent in a simple and easy to understand manner the game play within a section of a computer game. In addition, high-level payoff matrices can be used to support the design of different game levels in terms of the skill and challenge of the different sections within a computer game.

Literature review

Game theory

Game theory was initially developed as a branch of economics (Von Neumann and Morgenstern, 1944). Game theory concerns the mathematical study of decision making. Game theory was developed to model how individuals behave in specific circumstances that resemble simple kinds of games. Game theory can be used to

examine the relationships between decisions and outcomes (Saken and Zimmerman, 2004). A game in the context of game theory can be described as a contest (or play) amongst adversaries (or players) operating under constrains (or rules) for an objective (winning) (Stenros, 2016; Ellington et al, 1982). Game theory can be used to structure and analyse problem situations. Formal modelling of a situation as a game can involve determination of the players, their options, and consideration of strategies and preferences (Qureshi et al, 2012).

Game theory includes the concept of utility, which can be described as a mathematical measure of player satisfaction (Von Neumann and Morgenstern, 1944). In game theory, for all the outcomes that a decision might have, a utility can be assigned to that decision. Another game theory concept is that of the saddle point property which concerns the choices of game players that lead to the same result (Von Neumann and Morgenstern, 1944). A further game theory concept is that of the payoff matrix which maps the decision making process into a grid structure. One axis of the grid represents one player's decision. The other axis of the grid represents the other player's decision. The cells within the grid represent the outcomes reached depending on which decisions were made (Saken and Zimmerman, 2004).

Computer games design approaches

Various computer games design approaches have been developed. These include: flowcharts (Akcaoglu, 2016); storyboards (Frossard et al, 2012; Moreno-Ger, 2007); topological maps or graphs, with nodes representing scenes and edges representing the transitions between them (Gold, 2004), state transition models (LaMothe, 2002); and UML use case and class diagrams (Tenzer and Stevens, 2007; Ampatzoglou and Stamelos, 2010; Ampatzoglou and Chatzigeorgiou, 2007). Decision trees (Jones, 2008) can be used to design game play where the different paths in the decision tree relate to different paths that can be chosen by a player, or the different outcomes that can result from the actions of the game player (Woodcock, 1999). A decision tree is a branching tree-style diagram that can outline the set of possible actions and decisions that a player could make in a game. Decision trees can model how players move through the space of possibilities of a computer game (Saken and Zimmerman, 2004). Game flow design (Taylor et al, 2006) can be used to design the overall flow of the game play in sections of computer games. A variety of different approaches to designing the game flow of computer games have been proposed. Hunicke et al (2004) developed the MDA framework (Mechanics, Dynamics, and Aesthetics) which enables gameplay to be described in terms of game mechanics that describe the particular components of the game, at the level of data representation and algorithms. Dynamics describe the run-time behaviour of the mechanics acting on player inputs and each other's outputs over time. Aesthetics concern the emotional responses evoked in the player. Dormans (2011) proposed the Machinations framework that uses diagrams similar to Petri nets to represent the flow of tangible and abstract resources through a computer game. The diagrams comprise of elements such as pools, drains, gates, sources, converters and traders as well as connector symbols for representing game flow. Smith et al (2010) developed the Ludocore logical "game engine" based upon event calculus, that links game rules to formal logic used by automated reasoning tools in artificial intelligence. Treanor et al (2012) proposed the concept of micro-rhetorics, which are patterns of game mechanics and beliefs about static visuals and sounds, that can be used to represent the ideas that form the

foundation of representation for a computer game. Martens (2015) created the Ceptre methodology, which can be used for understanding games based on linear logic, a formal logic concerned with resource usage that can support rapid prototyping for experimental game mechanics.

Game theory for computer games design

Game theory examines the relationships between decisions and outcomes. The interactivity between actions and outcomes can be used to model game play (Davis, 1983). A game theory view of the design of a computer game can involving viewing a computer game as a series of strategic decisions made by the player. The game play within a computer game can be ordered and structured by rules. Rules can constitute the inner form or organization of a computer game, and rule schemas can provide analytic tools that mathematically dissect computer games (Saken and Zimmerman, 2004). Game theory can be used to develop the 'rules of play' for a computer game. Decision trees that can formally mathematically represent the pathways within a computer game can potentially help computer games designers understand how game players can move through the space of possibilities within the different parts a computer game.

Research methodology

Theoretical research was utilised to develop an approach for applying game theory to computer game design via an adapted decision tree approach and payoff matrices. The approach developed aimed to support the computer game design process for computer games that involve a variety of player choices, a variety of game pathways, and a variety of game outcomes (for example, first person shooter and puzzle games).

The approach for computer game design that was developed aimed to support the process of computer game development from the conceptual design stage to the physical program design stage. The approach aimed to reduce the complexity of computer game inputs, pathways and rules to a manageable level, where sections of a computer game can be diagrammatically represented to indicate the main pathways, main player choices and main outcomes, and where levels of skill and challenge can be modelled.

Incorporating symbols for inanimate game objects, portable game objects, and interactive game objects into a decision tree structure can support a more visual representation of the gameplay within a given section of a computer game. High-level payoff matrices can assist in designing levels of skill or challenge. In commercial practice, the computer games design process may typically involve higher level storyboarding and detailed program design, with less emphasis on the mid-level of actual gameplay design.

Research Results

Game theory for computer games design

Game theory can be applied to computer game design to model the game play within a particular section of a computer game. The game play could be represented by a formal mathematical decision tree. However, for even the most simple of games, e.g. noughts and crosses, a full formal mathematical decision tree would be complex and unwieldy. For the game of noughts and crosses using a fixed frame of spatial reference the number of branches on a full formal mathematical decision tree would be 9x8x7x6x5x4x3x2x1 assuming that all possible game play moves were used (even those that adopted no strategy). The complexity of this decision tree could be slightly reduced using an initially non-fixed frame of spatial reference, in the sense that for the first move in noughts and crosses due to symmetry (or equivalence) it would not make any difference, if for example, any of the four corners was chosen. A similar concept of equivalence of decision tree branch could be applied to more complex computer games design, for example, if in a first person shooter game, the game play represented by the choice of two similar weapons was equivalent. Visually the choice of weapons would be different, however, in terms of interactions with other game objects they would be the same (or equivalent).

Adapted decision trees for computer games design

In terms of applying game theory to computer games design, when creating decision trees (or game trees), it is the game interactions (or main decision points) that are of most interest. Game interactions could be with fixed inanimate objects (that can be graphically represented within a decision tree by boxes) or with animate objects (that can be graphically represented within a decision tree by ovals) (Taylor et al, 2006), or portable game objects (that can be graphically represented within a decision tree by diamonds).

Traditional decision trees can be used to determine the likelihood of events occurring e.g. the likelihood of a head or tail in a game of heads or tails. In complex computer games, the likelihood of success (or completion of a particular game segment) would be unwieldy to calculate in a formal mathematical decision tree sense. However, it can be possible to represent the likelihood of success in terms of the parameters required for success. Typically, in most computer games, using the control set provided by the game controller the main parameters of the actions employed by the game player are choice (in terms of which button to press), speed (in terms of how quickly a button is pressed or a joystick is moved) and accuracy (in terms of how precisely a joystick is moved in a particular direction or directions).

The major limitation of formal mathematical decision trees is the nature of their construction, in terms of the order of decisions. Unless a computer game was to be played in a strict order, then attempting to create a decision tree for a whole game would be inappropriate and unwieldy. However, although ordering may be inappropriate on a larger scale, on the smaller scale of game segments, or mission segments, decision trees can provide a useful framework for designing game play.

Each branch on the adapted decision tree represents a set of game states (for example, amount of ammunition and health status in a first person shooter game), and a set of game inputs (for example, the speed and direction of movement towards a game object). Thereby the complexity and size of the decision tree for game design of a given game segment can be reduced to a manageable size and complexity. Figure 1

represents the design of a simple example of a very small segment of a hypothetical game scenario. This shows the game objects with which the player may interact, and the nature of the interaction in terms of (in this instance) how fast and how accurately the game characters (Opponent 1 and Opponent 2) will respond.

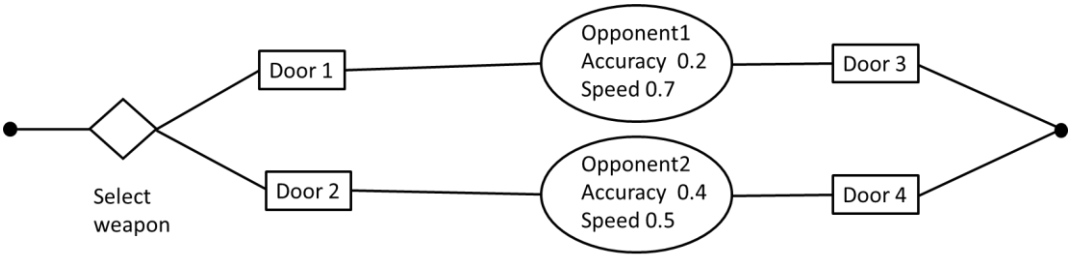

Figure 1. Example of adapted decision tree for computer game design.

Game theory concept of utility for computer games design

The game theory concept of utility (which can be described as a mathematical measure of player satisfaction) can be incorporated into computer games design by considering player satisfaction (in terms of actual game play) (Boyle et al, 2012) as relating to computer games in which there is an appropriate variety of choices (as represented by the different pathways within the adapted decision tree structures) and an appropriate level of challenge (as represented by factors such as the speed and accuracy of player inputs) in order to complete the different computer game segments (Jeffries, 2011).

Game theory concept of the saddle point property for computer games design

The game theory concept of the saddle point property concerns the choices of game players that lead to the same result (Von Neumann and Morgenstern, 1944). A saddle point can be viewed as an optimal solution to a computer game. For example, if in a first person shooter game, the choice of a particular type of weapon (e.g. a heavy machine-gun) meant that the player would almost always easily complete that particular section of the game, then the player might always make that choice and lessen the meaning and variety of the game play. Careful construction of the adapted decision tree structures can attempt to avoid creating saddle points (or simple shortcut routes) for the player in the computer game design.

Some outcomes in a computer game can potentially be more determined by game design rather than player input (for example, the availability of spare ammunition in a first person shooter game). Such elements of game design can be used to set the 'level' of the game play, and relate to the probability of a player successfully completing a particular game play path through the decision tree for a given game segment. Computer games may be designed so that players have to seek out different styles of play and game paths to win, or games may be designed so that players are forced down particular game pathways. More advanced forms of game play could assess the player's 'ability level' based upon previous game play and alter the level of difficulty to further challenge the player.

Payoff matrices for computer games design

Rollings and Morris (2004) described the concept of game balance, which included three different categories: Player/player balance that concerns making a multiplayer game fair so that each player gets no other special advantage but their skill; Player/gameplay balance that involves ensuring that the player's learning curve is matched by rewards that keep them playing; and Gameplay/gameplay balance that relates to features within the game being balanced against each other. The game theory concept of the payoff matrix can be used to model the player's decision making process into a grid structure in order to analyse, document and communicate the skill or challenge levels within the sections of a computer game. A payoff matrix can be described as a visual representation of all the possible outcomes that can occur when two individuals have to make a strategic decision. One axis of the payoff matrix can represent the player's decision. The other axis of the payoff matrix can represent the software based opponent player. The cells within payoff matrix represent the outcomes that depend upon the player and software based opponent player decisions. For simple games such as those were the player selects one choice out of a range of choices, the payoff matrix could be fairly straightforward. As with the application of decision trees for computer game design, using formal mathematical payoff matrices for every possible decision and action taken by a computer game player in more complex games would be unwieldy and unfeasible. However, it is practicable to use high-level payoff matrices that model more generic level decisions made by players within a given segment of a computer game. For example, in one of the first computer games, Pong (a table tennis simulation game) (Dickey, 2005) the accuracy of the movements made by the player affected the outcome of the game, as ball contact with the outer edges of the ping pong bat returned the ball at smaller (and more difficult) angles. Thus, a payoff matrix for this computer game could model high and low player accuracy outcomes as generic player decisions. Rollings and Morris (2004) commented that even when payoff matrices are only a very abstract model of a game, they can be useful in balancing different elements of the game design.

As an example, within Figure 1, a payoff matrix could be used to determine what skill level (or level of challenge) would be required with regard to the game play involving either of the two software based opponents (or players). The elements of the payoff matrix could either be estimated by the computer game designer, or could be more statistically determined by conducting experiments with actual game players in a test environment to determine distributions of outcomes (Fullerton, 2008). This can essentially examine the probability of success of game players as they engage in the game play of the different sections of the computer game. Playtesting is something that a computer games designer can perform throughout the entire design process of a computer game in order to gain an insight into whether or not the computer game is achieving player experience goals (Fullerton, 2008). A simple way to ensure balance in a computer game could be by exact symmetry, which means that players would have exactly the same weapons, manoeuvers, hit points, etc., however typically asymmetries are often necessary for reasons of computer game realism or aesthetics (Rollings and Morris, 2004). For multi-player on-line computer games, statistical analysis of the strategies used by a large number of players via payoff matrices could be used to better understand the typical strategies used by game players, which could then be used to inform game design. In summary, payoff matrices may initially be

proposed by a game designer, in terms of the variables against which the outcomes will be measured, and initial estimates of the likely outcome probabilities. However, in order to achieve appropriate gameplay experience for a variety of game players, it would be practicable to undertake experimentation in terms of adjustments of the variables involved and detailed playtesting, in order to ensure that the payoff matrices represent a statistically representative assessment of the gameplay.

Software based opponent player (or players)	Player		
		High speed / low accuracy	High accuracy / low speed
	High speed / low accuracy	Likely draw	Player wins
	High accuracy / low speed	Player loses	Likely draw

Figure 2. Payoff matrix example to illustrate how different factors could be used to determine and document the skill or challenge level of the sections within a computer game.

Figure 2 shows a simplified payoff matrix that could be proposed by a game designer to determine what skill level (or level of challenge) would be required with regard to the game play involving either of the two software based opponents (or players) from Figure 1. In terms of the skill level or challenge level of the different sections of a computer game, the payoff matrices could be used to apply the concept of the Nash equilibrium (Nash, 1951) in which the player (and the software based opponent player) would be assumed to be aware of the equilibrium strategy of the other player, and neither player would have anything to gain by changing strategy. An equilibrium state could represent an "average" game player. Novice, and skilled player levels could then be set in terms of, for example, the speed and accuracy of the player inputs that would lead to outcomes more favourable or less favourable to the player.

Software based opponent player (or players)	Player		
		High speed / low accuracy	High accuracy / low speed
	High speed / low accuracy	50, 50	20, 80
	High accuracy / low speed	80, 20	50, 50

Figure 3. Simple example payoff matrix for the skill or challenge level of a section within a computer game.

Figure 3. shows a simple example payoff matrix based upon an analysis of different games played by actual game payers during playtesting for a given section of a computer game. The first number in each cell is the payoff (or probability of success) for the software opponent and second number in each cell is the payoff (or probability of success) to the player. If the player and software opponent use the same strategy (high accuracy / low speed or high speed / low accuracy) then the player will win 50% of the time and the software opponent will win 50% of the time. However, if the player adopts a high accuracy / low speed strategy and the software opponent adopts a high speed / low accuracy then the player will win 80% of the time. Similarly, if the software opponent adopts a high accuracy / low speed strategy and the player adopts a high speed / low accuracy then the software opponent will win 80% of the time.

A payoff matrix can be used to predict the choices or actions of the players. Here we assume that if there is a dominant strategy, then the player would choose that strategy. A dominant strategy is a best response to every strategy of the other player. In the example above, the two strategies available are S1 - 'High accuracy / low speed' and S2 - 'High speed / low accuracy'. The game player would typically always choose the 'High accuracy / low speed' strategy (S1) over the 'High speed / low accuracy' strategy (S2) for the following reasons:

Firstly, if the software based opponent player's choice is S2 then the best response from the game player would be S1. Because the game player wins.

Secondly, if the software based opponent player's choice is S1 then the best response from the game player would be S1. Because a 'likely draw' is better than the game player losing.

So regardless of the software based opponent player's choice, the game player would typically always choose S1, the 'High accuracy / low speed' strategy. Similarly, we can say that the software based opponent player would always choose the S1 'High accuracy / low speed' strategy. We can predict that both players typically would choose strategy S1 for this example.

Not all computer games can be predicted in this manner, since there may be no dominant strategies. When neither player in a two-player game has a dominant strategy, we should expect players to use strategies that are the best responses to each other. Suppose that Player 1 chooses a strategy S and Player 2 chooses a strategy T. We say that this pair of strategies (S, T) is a Nash equilibrium if S is a best response to T, and T is a best response to S. However, there are also computer games that may have no Nash equilibria at all. For such computer games, we can make predictions about players' behaviour by enlarging the set of strategies to include the possibility of randomisation.

Overall, the adapted high-level decision trees can describe and visually document the formal space of possibilities (that is actions and outcomes) within the sections of a computer game. Appropriate design via the adapted high-level decision trees can increase the utility (or player satisfaction) of a computer game by attempting to ensure that appropriate levels of variety and challenge of gameplay are provided. Appropriate design can also decrease saddle points that can undermine meaningful (and hopefully interesting) gameplay by allowing holes or gaps in the design that allow players to easily (too easily) complete game sections. High-level payoff matrices can assist in designing the skill or challenge level of different sections of a computer game.

Conclusion

In this paper, we have demonstrated how the concepts of game theory can be applied to the design of computer games to design game play paths, difficulty levels, and potentially more advanced adaptive forms of game play. Game theory can provide a formal approach to understanding decision making in a computer game environment, which can support the computer game design process.

In order to design computer games that offer appropriate levels of challenge and variety of game play choices and pathways, game theory can be utilised to help design the overall structure of the game play experienced by the player. The variety of game pathways and possible game actions implies that complete formal mathematical specification would be unwieldy at best. A simple game of noughts and crosses can illustrate the variety of pathways that even a simple computer game can entail. Game theory can be applied to computer game design in a practical and usable manner by using adapted high-level decision tree structures to outline the main game pathways, the main decision points (in terms of player choices within segments of the computer game), and how the player actions relate to the probability of success (or completion) for a given game segment. By incorporating an abstraction of the probability of success within the decision trees, it is possible to outline different levels of gameplay (e.g. novice, average, skilled levels) overall, but also the skill level (or likelihood of success) for different game paths. High-level payoff matrices can be used to model the skill or challenge level of different game segments. In addition, by recording and analysis of the level of player ability, it would be possible to develop more adaptive

gameplay, where the skill of the player is assessed in previous game sections and future game sections could be altered accordingly.

Overall, adapted decision trees can provide a more formal understanding of the nature of game play, and a decision tree based computer game design approach can provide a straightforward and useful way of developing, documenting and communication the structure of the segments of a computer game. High-level payoff matrices can be used to model and communicate skill and challenge levels in computer game segments. Hopefully the approach to applying game theory to computer games design via the use of adapted high level decision trees incorporating symbolic representation of computer game objects and payoff matrices to model skill and challenge levels can be of use to computer games designers and developers, when moving between the storyboarding and program design phases of computer games design.

References

Akcaoglu, M. (2016) Design and implementation of the Game-Design and Learning Program, TechTrends, 60, 2, 114-123.

Ampatzoglou, A., Chatzigeorgiou, A. (2007) Evaluation of object-oriented design patterns in game development, Information and Software Technology, 49, 5, 445-454.

Ampatzoglou, A., Stamelos, I. (2010). Software engineering research for computer games: A systematic review, Information and Software Technology, 52, 9, 888-901.

Boyle, E., Connolly, T., Hainey, T., Boyle, J. (2012) Engagement in digital entertainment games: A systematic review, Computers in Human Behavior, 28, 3, 771-780.

Davis, M. (1983) Game theory: a nontechnical introduction, Basic books, New York, USA.

Dickey, M. (2005) Engaging by design: How engagement strategies in popular computer and video games can inform instructional design, Educational Technology Research and Development, 53, 2, 67-83.

Dormans, J. (2011) Simulating Mechanics to Study Emergence in Games, in Proceedings of the Artificial Intelligence in the Game Design Process Workshop at the Seventh Artificial Intelligence and Interactive Digital Entertainment Conference, October 10 - 14, 2011, Stanford University, California, USA, pp. 2-5.

Ellington, H., Addinall, E., Percival, F. (1982) A handbook of game design, Kogan Page, London, England.

Frossard, F., Barajas, M., Trifonova, A. (2012) A Learner-Centred Game-Design Approach: Impacts on Teachers' Creativity, Digital Education Review, 21, 13-22.

Fullerton, T. (2008) Games Design Workshop: A playcentric approach to creating innovative games, Morgan Kaufmann, Burlington, MA, USA.

Gold, J. (2004) Object oriented game development, Addison Wesley, Harrow, UK.

Hunicke, R., LeBlanc, M., Zubek, R. (2004) MDA: A formal approach to game design and game research, in Proceedings of the AAAI Workshop on Challenges in Game AI, July 25 – 26, 2004, San Jose McEnery Convention Center, San Jose, California, USA, pp. 17 - 22.

Jeffries, K. (2011) Skills for creativity in games design, Design Studies, 32, 1, 60-85.

Jones, M. (2008) Artificial Intelligence: A Systems Approach, Infinity Science Press, Hingham, MA, USA.

LaMothe, A. (2002) Tricks of the Windows game programming gurus, Sams Publishing, Indianapolis, Indiana, USA.

Martens, C. (2015) Ceptre: A language for modelling generative interactive systems, in Proceedings of Eleventh Artificial Intelligence and Interactive Digital Entertainment Conference, November 14 – 18, 2015, University of California, Santa Cruz, California, USA, pp. 51 – 57.

Moreno-Ger, P., Sierra, J., Martínez-Ortiz, I., Fernández-Manjón, B. (2007) A documental approach to adventure game development, Science of Computer Programming, 67, 1, 3-31.

Nash, J. (1951) Non-cooperative games, Annals of Mathematics, 54, 286–295.

O'Hagan, A., Coleman, G., O'Connor, R. (2014) Software development processes for games: a systematic literature review, In Proceedings of European Conference on Software Process Improvement, Springer, Berlin, pp. 182-193.

Qureshi, N., Mushtaq, H., Aslam, M., Ahsan, M., Ali, M., Atta, M. (2012) Computing Game Design with Automata Theory, International Journal of Multidisciplinary Sciences and Engineering, 3, 5, 13-21.

Rollings, A., Morris, D. (2004) Game Architecture and Design: A New Edition, New Riders Publishing, Indianapolis, Indiana, USA.

Saken, K., Zimmerman, E. (2004) Rules of Play - Game Design Fundamentals, The MIT Press, Cambridge, Massachusetts, USA.

Smith, A., Nelson, M., Mateas, M. (2010) Ludocore: A logical game engine for modelling videogames, in Proceedings of the IEEE Symposium on Computational Intelligence and Games, 18 - 21 August, 2010, IT University of Copenhagen, Copenhagen, Denmark, pp. 91-98.

Stenros, J. (2016) The Game Definition Game: A Review, Games and culture,

https://doi.org/10.1177/1555412016655679

Taylor, M.J., Gresty, D., Baskett, M. (2006) Computer Game Flow Design, ACM Computers in Entertainment, 4, 1.

Tenzer, J., Stevens, P. (2007) GUIDE: Games with UML for interactive design exploration, Knowledge-Based Systems, 20, 7, 652-670.

Treanor, M., Schweizer, B., Bogost, I., Mateas, M. (2012) The micro-rhetorics of Game-O-Matic, in Proceedings of the ACM International Conference on the Foundations of Digital Games, May 29 - June 01, 2012, Raleigh, NC, USA, pp. 18-25.

Von Neumann, J., Morgenstern, O. (1944) Theory of Games and Economic Behavior, Princeton University Press, Princeton, New Jersey, USA.

Woodcock, S. (1999) Game AI: The State of the Industry, Game Developer Magazine, 3, 33, 35-43.