Dynamic game difficulty balancing

From Wikipedia, the free encyclopedia

Dynamic game difficulty balancing, also known as **dynamic difficulty adjustment** (DDA) or **dynamic game balancing** (DGB), is the process of automatically changing parameters, scenarios and behaviors in a video game in real-time, based on the player's ability, in order to avoid them becoming bored (if the game is too easy) or frustrated (if it is too hard). The goal of dynamic difficulty balancing is to keep the user interested from the beginning to the end and to provide a good level of challenge for the user.

Traditionally, game difficulty increases steadily along the course of the game (either in a smooth linear fashion, or through steps represented by the levels). The parameters of this increase (rate, frequency, starting levels) can only be modulated at the beginning of the experience by selecting a difficulty level. Still, this can lead to a frustrating experience for both experienced and inexperienced gamers, as they attempt to follow a preselected learning or difficulty curve. Dynamic difficulty balancing attempts to remedy this issue by creating a tailor-made experience for each gamer. As the users' skills improve through time (as they make progress via learning), the level of the challenges should also continually increase. However, implementing such elements poses many challenges to game developers; as a result, this method of gameplay is not widespread.

Contents

Dynamic game elements

Some elements of a game that might be changed via dynamic difficulty balancing include:

- Speed of enemies
- Health of enemies
- Frequency of enemies
- Frequency of powerups
- Power of player
- Power of enemies
- Duration of gameplay experience

Approaches

Different approaches are found in the literature to address dynamic game difficulty balancing. In all cases, it is necessary to measure, implicitly or explicitly, the difficulty the user is facing at a given moment. This measure can be performed by a heuristic function, which some authors call "challenge function". This function maps a given game state into a value that specifies how easy or difficult the game feels to the user at a specific moment. Examples of heuristics used are:

- The rate of successful shots or hits
- The numbers of won and lost pieces
- Life points
- Evolution
- Time to complete some task

... or any metric used to calculate a game score.

"parameters manipulation" to some mechanisms to modify the behavior of the non-player characters (NPCs) (characters controlled by the computer and usually modeled as intelligent agents). This adjustment, however, should be made with moderation, to avoid the 'rubber band' effect. One example of this effect in a racing game would involve the AI driver's vehicles becoming significantly faster when behind the player's vehicle, and significantly slower while in front, as if the two vehicles were connected by a large rubber band.

A traditional implementation of such an agent's intelligence is to use behavior rules, defined during game development. A typical rule in a fighting game would state "punch opponent if he is reachable, chase him, otherwise". Extending such approach to include opponent modeling can be made through Spronck *et al.*'s dynamic scripting,[2][3] which assigns to each rule a probability of being picked. Rule weights can be dynamically updated throughout the game, accordingly to the opponent skills, leading to adaptation to the specific user. With a simple mechanism, rules can be picked that generate tactics that are neither too strong nor too weak for the current player.

Andrade *et al.*[4] divides the DGB problem into two dimensions: competence (learn as well as possible) and performance (act just as well as necessary). This dichotomy between competence and performance is well known and studied in linguistics, as proposed by Noam Chomsky. Their approach faces both dimensions with reinforcement learning (RL). Offline training is used to bootstrap the learning process. This can be done by letting the agent play against itself (selflearning), other pre-programmed agents, or human players. Then, online learning is used to adapt continually this initially built-in intelligence to each specific human opponent, in order to discover the most suitable strategy to play against him or her. Concerning performance their idea is to find an adequate policy for choosing actions that provide a good game balance, i.e., actions that keep both agent and human player at approximately the same performance level. According to the difficulty the player is facing, the agent chooses actions with high or low expected performance. For a given situation, if the game level is too hard, the agent does not choose the optimal action (provided by the RL framework), but chooses progressively less and less suboptimal actions until its performance is as good as the player's. Similarly, if the game level becomes too easy, it will choose actions whose values are higher, possibly until it reaches the optimal performance.

Demasi and Cruz[5] built intelligent agents employing genetic algorithms techniques to keep alive agents that best fit the user level. Online coevolution is used in order to speed up the learning process. Online coevolution uses pre-defined models (agents with good genetic features) as parents in the genetic operations, so that the evolution is biased by them. These models are constructed by offline training or by hand, when the agent genetic encoding is simple enough.

Other work in the field of DGB is based on the hypothesis that the player-opponent interaction—rather than the audiovisual features, the context or the genre of the game—is the property that contributes the majority of the quality features of entertainment in a computer game.[6] Based on this fundamental assumption, a metric for measuring the real time entertainment value of predator/prey games was introduced, and established as efficient and reliable by validation against human judgment.

Further studies by Yannakakis and Hallam[7] have shown that artificial neural networks (ANN) and fuzzy neural networks can extract a better estimator of player satisfaction than a human-designed one, given appropriate estimators of the **challenge and curiosity** (intrinsic qualitative factors for engaging gameplay according to Malone)[8] of the game and data on human players' preferences. The approach of constructing user models of the player of a game that can predict the answers to which variants of the game are more or less *fun* is defined as *Entertainment Modeling*. The model is usually constructed using machine learning techniques applied to game parameters derived from player-game interaction and/or statistical features of player's physiological signals recorded during play. This basic approach is applicable to a variety of games, both computer[7] and physical.

Uses in recent video games

The video game Flow was notable for popularizing the application of mental immersion (also called *flow*) to video games with its 2006 Flash version. The video game design was based on the master's thesis of one of its authors, and was later adapted to PlayStation 3.

The 2008 video game Left 4 Dead uses a new artificial intelligence technology dubbed "The AI Director".[9] The AI Director is used to procedurally generate a different experience for the players each time the game is played. It monitors individual players performance and how well they work together as a group to pace the game, determining the number of zombies that attack the player and the location of boss infected encounters based on information gathered. Besides pacing the Director also controls some video and audio elements of the game to set a mood for a boss encounter or to draw the players' attention to a certain area.[10]

In 2009, the game Resident Evil 5 employed a system called the "Difficulty Scale", unknown to most players as the only mention of it was in the Official Strategy Guide. This system grades the players performance on a number scale from 1 to 10, and adjusts both enemy behavior/attacks used and enemy damage/resistance based on the players' performance (such as deaths, critical attacks, etc.). The selected difficulty levels lock players at a certain number, for example, on Normal difficulty one starts at Grade 4, can move down to Grade 2 if doing poorly, or up to Grade 7 if doing well. The grades between difficulties can overlap.[12]

In the match-3 game Fishdom, the time limit is adjusted based on how well the player performs. The time limit is increased should the player fail a level, making it possible for any player to beat a level after a few tries.

In the 1999 video game Homeworld, the number of ships that the AI begins with in each mission will be set depending on how powerful the game deems the player's fleet to be. Successful players have larger fleets because they take fewer losses. In this way, a player who is successful over a number of missions will begin to be challenged more and more as the game progresses.

In the video games Fallout: New Vegas and Fallout 3, as the player increases in level, tougher variants of enemies, enemies with higher statistics and better weapons, or new enemies will replace older ones to retain a constant difficulty, which can be raised, using a slider, with experience bonuses and vice versa in Fallout 3. This can also be done in New Vegas, but there is no bonus to increasing or decreasing the difficulty.

The Mario Kart series of games feature items during races that help the individual driver get ahead of opponents. These items are distributed based on a driver's position in a way that is an example of Dynamic Game Difficulty Balancing. For example, a driver near the bottom of the field is likely to get an item that will increase their speed, whereas a driver in first or second place can expect to get these kinds of items rarely.

See also

- Difficulty level
- Nonlinear gameplay
- Game balance
- Game artificial intelligence
- Flow (psychology)

References

1. ^ Robin Hunicke, V. Chapman (2004). "AI for Dynamic Difficulty Adjustment in Games". *Challenges in Game Artificial Intelligence AAAI Workshop*. San Jose. pp. 91–96.
2. ^ Pieter Spronck (https://web.archive.org/web/20111212020340/http://www.spronck.net/) from Tilburg centre for Creative Computing
3. ^ P. Spronck, I. Sprinkhuizen-Kuyper, E. Postma (2004). "Difficulty Scaling of Game AI". *Proceedings of the 5th International Conference on Intelligent Games and Simulation*. Belgium. pp. 33–37.
4. ^ G. Andrade, G. Ramalho, H. Santana, V. Corruble (2005). "Challenge-Sensitive Action Selection: an Application to Game Balancing". *Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT-05)*. Compiègne, France: IEEE Computer Society. pp. 194–200.
5. ^ P. Demasi, A. Cruz (2002). "Online Coevolution for Action Games". *Proceedings of The 3rd International Conference on Intelligent Games And Simulation*. London. pp. 113–120.
6. ^ G. N. Yannakakis, J. Hallam (13-17 July 2004). "Evolving Opponents for Interesting Interactive Computer Games". *Proceedings of the 8th International Conference on the Simulation of Adaptive Behavior (SAB'04); From Animals to Animats 8*. Los Angeles, California, United States: The MIT Press. pp. 499–508.
7. ^ *a b* G. N. Yannakakis, J. Hallam (18-20 May 2006). "Towards Capturing and Enhancing Entertainment in Computer Games". *Proceedings of the 4th Hellenic Conference on Artificial Intelligence, Lecture Notes in Artificial Intelligence*. Heraklion, Crete, Greece: Springer-Verlag. pp. 432–442.
8. ^ Malone, T. W. (1981). "What makes computer games fun?". *Byte* **6**: 258–277.
9. ^ "Left 4 Dead" (https://web.archive.org/web/20111212020340/http://web.archive.org/web/20090316000000/http://l4d.com/info.html) . Valve Corporation. Archived from the original (https://web.archive.org/web/20111212020340/http://www.l4d.com/info.html) on 2009-03-16. http://web.archive.org/web/20090316000000/http://l4d.com/info.html.
10. ^ "Left 4 Dead Hands-on Preview" (https://web.archive.org/web/20111212020340/http://www.left4dead411.com/left-4-dead-preview-pg2) . *Left 4 Dead 411*. http://www.left4dead411.com/left-4-dead-preview-pg2.
11. ^ Newell, Gabe (21 November 2008). "Gabe Newell Writes for Edge" (https://web.archive.org/web/20111212020340/http://www.next-

Further reading

- Hunicke, Robin (2005). "The case for dynamic difficulty adjustment in games". *Proceedings of the 2005 ACM SIGCHI International Conference on Advances in computer entertainment technology*. New York: ACM. pp. 429–433. doi:10.1145/1178477.1178573 (https://web.archive.org/web/20111212020340/http://dx.doi.org/10.1145%2F1178477.1178573) .
- Byrne, Edward (2004). *Game Level Design*. Charles River Media. p. 74. ISBN 1584503696.
- Chen, Jenova (2006). "Flow in Games" (https://web.archive.org/web/20111212020340/http://www.jenovachen.com/flowingames/abstract.htm) . http://www.jenovachen.com/flowingames/abstract.htm.

External links

- ~~Dynamic Difficulty Adjustment (https://web.archive.org/web/20111212020340/http://www.gameontology.org/index.php/Dynamic_Difficulty_Adjustment) - Game Ontology Wiki~~ dead link

Retrieved from "http://en.wikipedia.org/w/index.php?title=Dynamic_game_difficulty_balancing&oldid=445906345"
Categories: Video game gameplay | Game design
