Motivating Engineering Mathematics Education with Game Analysis Metrics

David I. Schwartz
Rochester Institute of Technology
Department of Information Technology
GCCIS, 70-2509
+1 585-475-5521
dis[at]it[dot]rit[dot]edu

Abstract

In this paper, I develop a system for computational analysis of games that uses scoring functions to motivate engineering mathematics education. Although many modern videogames have abandoned points as an archaic form of representing victory (or defeat) conditions, scoring can help represent a game as a dynamic simulation. The paper derives a functional mathematical relationship between temporal game state and score, which provides a foundation for addressing computational issues of games. Scoring functions can provide analytical tools for game analysis by measuring continuous and discrete game state. These tools may assist with game design, analysis, and balancing. Linking game creation with computational analysis could provide an excellent context to integrate mathematics at early stages of education. Moreover, the addition of theory might attract more engineering educators to provide rigor to the emerging academic field of game design. The paper concludes with proposed research into classes of scoring functions. If certain genres of games and difficulty levels yield similar scoring functions, concepts of topology could further connect education of engineering mathematics theory with game design and analysis.

1. Introduction

Although studying theoretical applications of game scores might not seem an obvious choice for motivating engineering mathematics education, this section provides background on the concept and draws various connections.

1.1 Scoring in Games

Starting with pinball machines into which people fed coins, the "classic age" of arcade videogames engaged players in a similar fashion. Players fed quarters into machines, hoping to beat high scores. As games matured and moved to personal computers and home consoles, victory conditions shifted from amassing points to achieving goals often tied into a plot or story. However, interest in points and scores remains, usually seen in the growing casual game market (e.g., *PopCap Games*[1]), *retrogaming*[2], and on-line rankings (e.g., Xbox Live's *Achievements*[3]).

Although this paper focuses on applying numerical scoring systems, current games that lack scoring might still shift points to other representations. For example, role-playing games involve levels and

"loot." A game interface may also provide meters and monitors that shield internal quantitative systems. Thus, seemingly abandoned scores might simply "lurk" inside a game.

1.2 Education and Games

In recent years, many academic programs have incorporated games, ranging from introductory courses to merging with the major. Much of this research on engineering mathematics integration focuses on programming, software engineering, artificial intelligence, and graphics. However, game design courses do not usually broach engineering mathematics theory, including mathematical foundations. Although path finding, artificial intelligence, and game physics use theory and mathematics, scoring can involve many mathematical concepts as well, as discussed in this paper.

1.3 Linking Scoring to Engineering mathematics

As discussed in Section 1.1, games may involve a large degree of numerical computation through scoring either internally or externally. This paper demonstrates that scores can offer applications of discrete and continuous mathematics, thus applying introductory concepts learned by students. In fact, studies of scoring functions already occur in computational analysis, e.g., computational genomics along and other data-intensive fields[4].

A score can rate player progress, measuring a rudimentary state of a game. A score can therefore reduce an extremely complex system into a simpler computational model. In this paper, Sections 2 and 3 develop mathematical definitions of points, scores, and game state. Then, Section 4 applies scoring functions to game design and analysis, demonstrating a variety of mathematical and computational applications for education. This work offers a possible avenue for engineering mathematics and engineering educators looking for a more theoretical approach to discuss game design and analysis.

2. Scoring Theory

This section introduces the notion of treating game state as a temporal sequence of play using mathematical definitions for points and scores.

2.1 Games and Theory

Game theory is the mathematical study of decision making[5]. By studying a system that allows decisions that seek victory conditions, one can study strategies for dealing with competitive systems[6]. In a similar fashion, the study of algorithmic game complexity involves searching for moves/strategies that will win a game. Researchers study turn-based games (e.g., chess as a classic example), developing algorithms for building and searching game trees.

2.2 Games as Simulations

Game theory mathematically represents certain kinds of games, and the study of algorithmic game complexity involves a deep study of algorithms. However, when confronting a "realistic" game, the multitude of choices, especially in "real-time" games, can bog down both approaches. Although a theoretician might wish to explore the mathematical and computational complexity, a practitioner might find the theory too computationally expensive for design and implementation. For educational purposes, students learning about game design and development early in academic studies lack the proficiency and theoretical depth.

Instead, consider the notion of studying games as *simulations*. A game can simulate a real-world system, creating an abstraction of reality. Figure 1 demonstrates one example in which Tic Tac Toe simulates warfare[6].

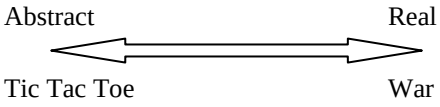

Figure 1. Abstraction and Combat Simulation

In fact, a simulation is a "meta-game" in which the victory condition is the accuracy of the simulation. Abstraction of reality relates to one aspect of a game definition, a game "artificial conflict"[5].

2.3 Temporal Game State

Considering a game as a collection of components, a set can provide a rudimentary description of a game[6], as shown in Equation 1:

$$Game = \{Entities, Events, Rules, Setting, User\ Interface\} \quad (1)$$

One could decompose the set further, e.g., entities representing players, avatars, agents, and other forms of actors/characters. Each component can change in time, which provides an overall temporal game state (TGS) (renamed from "game state"[5]). The TGS uses a sequence of machine-time steps, real time, or turns to distinguish the state of a game. For example, although a board game may result in pieces having identical positions after a turn (e.g., a player backtracks a particular move), the game has still progressed in time, producing a sequence of events.

Unlike the study of algorithmic game complexity, TGS helps to involve a notion of gameplay, whereby something changes at each time step of the game. Even if a game allows skipped turns or stationary characters, actual play time still advances—thus, the term *temporal* added to game state[5] to distinguish from other properties (e.g., spatial).

Figure 2 presents one way to visualize TGS. As time advances, some measure or assessment can represent the game state at any particular time, or $G(t)$. This measure can include any aspect of the game's components (e.g., location, attributes, score).

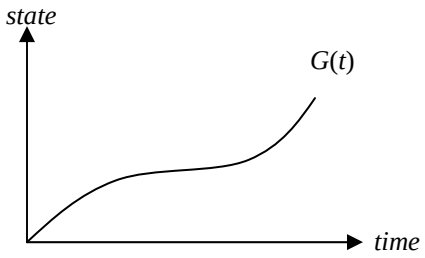

Figure 2. Temporal Game State

Although shown as continuous, $G(t)$ is typically discrete, especially given clock cycles of computers/consoles along with turn-based games. However, as time steps shrink in (approximated) real-time games, calculus provides a convenient model, especially to engage engineering mathematics students early in their studies.

This functional representation of a game mathematical represents a game apart from game theory. Section 3 explores scoring as one form of measure of $G(t)$.

2.4 An Aside: Saved Games

Note that a *saved game* is effectively a TGS, which represents the current state of all aspects of the game so that the player can restart at certain "points". A saved game is a "snapshot" of the component states stored for later retrieval. Even a board game has a natural save-game feature—players can either record piece positions or simply put the board with pieces somewhere safe[8].

3. Game Metrics

This section introduces points, scores, and scoring functions with a mathematical foundation. Game design literature lacks theory of points and scoring—even Wikipedia's current articles lack detail[8]. Providing a theoretical foundation is part of the process of trying to appeal to theory-oriented educators.

3.1 Points

Borrowing a related notion from education, teachers can grade assignments and tests with *points*, though no uniform standard can enforce the degree, type, and quality of work associated with one "standard point." However, a point measures some degree of worthiness of an answer.

When a player reaches a victory condition, albeit brief or partial, a game can assign a point to the player's action. For example, hitting a target, matching a note, avoiding an attack, and numerous other game mechanics can earn points. Depending on the game's design, missed/incorrect actions may yield no points or perhaps a penalty. In these cases, an action yields a numerical value, a *point p*, which typically belongs to the set of integers, Z, as shown in Equation 2:

$$p \in \mathbf{Z} \tag{2}$$

A point is an atomic measure of success. For other numerical systems, like decimal values, a designer could normalize the points to integer—though assigning irrational values as scores might yield an interesting twist to game design.

3.2 Scoring Functions

In a game that measures success quantitatively, points accumulate (or deduct) in *scores* as a player progresses (or regresses). A function that assigns points to a given state of some object (real or abstract) is a *score function* (also, *scoring function*)[5]. As stated previously, scoring functions appear in computational genomics, economics, databases, and other computational fields[4].

A game's scoring function can "measure some aspect of the game state"[5]. Instead of choosing a particular aspect or component from Equation 1, this paper's study of scoring functions defines the measure of

temporal game state in entirety with a single mapping. Work on game balancing with "challenge functions" uses a similar approach, whereby a score (or other internal measure) can represent a game state[9]. By dynamically adjusting difficulty, a game can adapt to different player abilities[9, 10].

Adapting the formalism and state[4, 9], a score function at time t assigns a score s to a temporal game state g:

$$s\colon g \to \mathbb{Z} \tag{3}$$

where $s \in Z$ and $g = G(t)$. Thus, a score can provide a basic measure of temporal game state. From Section 2, $G(t)$ represents an abstract measure of temporal game state—*state* is an arbitrary representation of G's components (Equation 1) at a specific time t. Thus, in Figure 2, *state* can replace *score*, using Equation 3.

For discrete systems (e.g., turn-based play), we can instead refer to time t_i, score s_i, and g_i, or just $G(t_i)$ using piecewise functions. As shown in Figure 3, as TGS advances from t_i to t_{i+1}, the score advances in a discrete fashion as well. To represent temporally discrete games, one could use points and/or line segments in Figure 2 instead of a continuous plot.

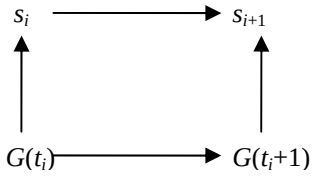

Figure 3. Scoring for Discrete Gameplay

4. Educational Applications

This section applies the theoretical concepts from Sections 2 and 3 to game design and analysis education. Again, this section provides a "roadmap" to entice educators and motivate students.

4.1 Implementing Score Functions

As discussed in Section 1.1, games that do not include quantitative scoring can still involve numerical points. For example, a developer can link internal tracking of health to points. Akin to diagnostic and profiling tools, a scoring function can quantify and expose internal details of the game. By working on "scoring modules," students would gain experience with software development tools and approaches.

4.2 Score Function Classes

Reducing a game to a 2-D function of time vastly simplifies issues of algorithmic game complexity. However, the large search-space shifts to choosing scoring functions. Then again, this tradeoff should still help students, because choosing a heuristic function resembles the process of game balancing. Adjusting

quantitative values for difficulty closely resembles adjusting scoring functions[9], perhaps leading to multiple versions as shown in Figure 4.

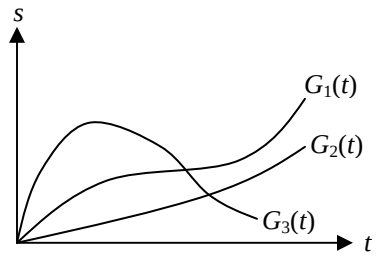

Figure 4. Multiple Score Functions

Although tweaking a simulation may resemble engineering more than engineering mathematics, students would now have a computational model for testing and analysis. Students may see how engineering mathematics also has an experimental/experiential aspect akin to physics and chemistry at an introductory level. Moreover, the model would also avoid combinatorial expenses given the simplifications, thus allowing for a realistic game.

4.3 Quality of Choices

In teaching scoring functions to students, one could further simplify G(*t*) to pose simple "base cases," as in a linear equation:

$$G(t) = qt \tag{4}$$

where q is an constant, defined later in this section. One could also simplify G(*t*) even further with a constant k:

$$G(t) = k \tag{5}$$

Figure 5 shows Equations 4 and 5. These simple functions hide a elegant reasoning about games and gameplay. For example, starting with Equation 5, if $k = 0$, then the player gains no score at each time increment. Thus, $G(t) = 0$ represents one of these situations:

- A player makes no choices.
- Game components do not change.
- A penality matches a gain of points.

If k is non-zero, then a player has an initial score or penalty.

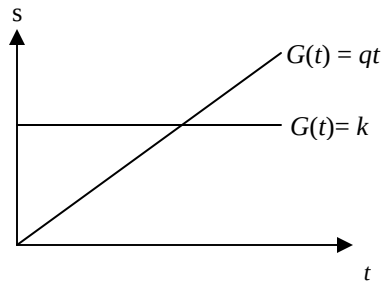

Figure 5. Basic Scoring Functions

In Equation 4, $G(t)$ measures score per unit time. As score increases in the same amount of time as another score, we have a *rate* of improving score. Using this observation, q can express a *quality of decision*, or change of score per time increment. Using an approximation of continuous change, we can express this relationship in Equation 6, below:

$$\frac{ds}{dt} = q \tag{6}$$

If a player can progressively gain (or lose) relatively larger (or smaller) amounts of points for the same time increment, the score accelerates (or decelerates). Equation 7 represents changing quality, below:

$$\frac{d2s}{dt2} = p \tag{7}$$

where p stands for *improvement*. For example, a player who can achieve a greater change of points for the same amount of time (or effort) increases the rate of q.

These notions provide a direct link to calculus and system modeling. Although relatively simplistic, an educator can apply mathematical modeling at an introductory level within the context of gameplay and game balancing[10]. Discrete functions and turn-based games have a similar development via "deltas" and piecewise analysis.

4.5 Optimization and Strategy

Extending the notion of quality of decision, an educator could introduce optimization with scoring functions. In Figure 4, different functions might have different local minima and maxima, perhaps leading to hills and valleys, depending on the game and function. Using multidimensional scoring functions could make the analysis more complex. Even so, having this computational model helps demonstrate how a game involves system thinking and searching solution spaces.

5. Summary and Conclusions

Although many games no longer use points, scoring functions can provide an excellent context to demonstrate engineering mathematics theory and mathematics in games. Points and scores can represent a distilled game state, and thus, a functional representation of a game. Given a simulation perspective, a

game changes discretely or continuously by approximation within a unit of time. In either case, a scoring function can measure the temporal game state.

Although game theory and the study of algorithmic game complexity offer similar "in-roads" to games and engineering mathematics theory, they do not easily handle realistic games a student might wish to build and/or study. Reducing a game's representation into the proposed two-dimensional functional form addresses both practicality and theory. As shown in this paper, a variety of educational concepts would benefit from application and study of scoring functions:

- Describing the relations between points, scores, and measures of game state can demonstrate applications of discrete and continuous mathematics.
- Discussing notions of gameplay concerning player actions and increasing/decreasing scores can involve a direct application of calculus at a first-year level. These applications connect to adaptive game balancing, as well.
- Building tools for internal/external score tracking can introduce notions of software analysis. Akin to experimental analysis (e.g., profiling), students would effectively build data acquisition systems into their games.

Introducing hill climbing, simulated annealing, and optimization introduces important algorithms and applications. Given the proposed single-valued scoring functions, a student could more easily grasp these concepts while still analyzing a complex game.

Showing students a mathematical representation, albeit reduced, offers an excellent connection to fundamental concepts. By motivating students to apply these concepts, educators could introduce engineering mathematics theory in a relatively easy and applicable fashion.

6. Future Work

Although other fields have used scoring functions, applying these notions for game design and development education are relatively new. As discussed in this paper, game designers have not abandoned scores, especially in casual games. Also, research into game balancing[9, 10] has begun to address how scoring and game state can advance the field of game design.

The next step is incorporating scoring functions into early game design and development courses that involve engineering mathematics students. As part of the design process (both game and software), students would need to iterate over a variety of parameters to determine useful functions. Notions of complexity analysis might help students bound scoring functions and further introduce engineering mathematics theory. But, the choice of specific scoring functions remains heuristic. Future research may show that certain classes of game genres yield classes of scoring functions. Perhaps scoring functions belong to established topological spaces, thus spurring deeper mathematical analysis and linking to more advanced theory.

When viewed as a collection of components, a game's functional model could expand into multidimensional functions. Although the simplicity of a single score would be lost, a more complex model would offer greater insight into how a game changes state. Deciding which game components and states should contribute (and to what degree) to a score needs further work.

Finally, educators may wish to introduce mathematical concepts from a game design and analysis perspective. Ultimately, scoring could provide an essential link between theoretical fundamentals and software implementations.

7. Acknowledgements

I would like to express my gratitude to the Air Force Research Laboratory (AFRL) in Rome, New York for providing a Visiting Faculty Research Professorship for the summer of 2007 and a subsequent extension grant. Research into wargame design and development methods as part of my summer research inspired the concepts I propose in this paper. My AFRL mentor, David O. Ross, and his colleague, Bruce Rubin, were very helpful and encouraging of this work.

8. References

[1] http://www.popcap.com, accessed Oct. 2007.

[2] http://www.vintagecomputing.com, accessed Oct. 2007.

[3] http://www.xbox.com/en-US/support/systemuse/xbox360/livefeatures/achievements.htm , accessed Oct. 2007.

[4] Hsu, D. F, Y-S. Chung, B. S. Kristal, 2006., Combinatorial Fusion Analysis: Methods and Practices of Combining Multiple Scoring Systems, in *Advanced Data Mining Technologies in Bioinformatics* (ed. H-H. Hsu), Idea Group, Chapter 3.

[5] Salen, K. and E. Zimmerman, 2004. *Rules of Play: Game Design Fundamentals,* The MIT Press.

[6] Dresher, M., 1981. The Mathematics of Games of Strategy: Theory and Applications, Dover Publications.

[7] Schwartz, D. I., K. Locke, D. O. Ross, and M. Emeny, 2007. The Future of Wargaming: A Componentized Approach, to appear in *Proceedings of the 2007 Huntsville Simulation Conference* (ed. J. Gauthier).

[8] http://en.wikipedia.org; see articles on “High score,” “Saved game,” “Score (game),” and “Game Balance,” accessed Oct. 2007.

[9] Demasi, P., and Cruz, 2002. A. Online Coevolution for Action Games, in *Proceedings of The 3rd International Conference on Intelligent Games And Simulation*, pp. 113-120.

[10] Hunicke, R., and Chapman, V., 2004. AI for Dynamic Difficulty Adjustment in Games, *Challenges in Game Artificial Intelligence AAAI Workshop*, pp. 91-96.

Biography

David I. Schwartz, Ph.D., a 1999 graduate of the State University of New York at Buffalo, published two textbooks on introductory computing skills while completing his dissertation in civil engineering. Publishing three books simultaneously sparked Cornell University's interest. So, in the summer of 1999, Schwartz accepted a lecturer position in the Department of Computer Science to teach computer programming and develop new introductory courses. Recognizing the academic potential of games, Schwartz founded the Game Design Initiative at Cornell (GDIAC) in the spring of 2001. Soon after, he designed Cornell Library Collaborative Learning Computer

Laboratory (CL3), which started hosting GDIAC courses in August 2004. In May 2006, these efforts established Cornell's Minor in Game Design offered by the College of Engineering, the first formal Ivy-League games program.

In the summer of 2007, Schwartz joined the Rochester Institute of Technology's Game Design & Development program as an assistant professor. Dr. Schwartz currently collaborates with the Air Force Research Laboratory on wargame design theory and software development. Schwartz also consults for the budding upstate New York game development company Made-for-Motion, which focuses on motion-controlled game development.