BOTS: Harnessing Player Data and Player Effort to Create and Evaluate Levels in a Serious Game

Andrew Hicks
NC State University
890 Oval Drive
Raleigh, NC 27606

ABSTRACT

BOTS is a socially multiplayer online game designed to teach students about introductory computer science concepts such as loops and functions. Using this game, I plan to explore the use of user-generated content (UGC) in game-based tutors, increasing replayability and, ideally, player engagement. BOTS has so far been used for work towards identifying what makes a level or puzzle in the game “good” and how I identify that quality in new submissions, as well as investigating several mechanisms for moderation of submitted content. The use of UGC has the potential to revolutionize how game-based tutors are created, drastically reducing the burden of content creation on developers and educators.

Keywords

Game Based Tutors, Moderation, Player Engagement, Self-Evaluation, Serious Games, Social Games, User Generated Content..

1. INTRODUCTION

1.1 Background

Computer assisted learning, and game-based learning in particular has been shown to be able to be nearly as effective as one-on-one human tutoring [8, 10], however the developers and educators are required to use a great deal of time and expert knowledge [2, 14]. Murray estimated approximately it takes 300 hours to create a single hour of educational content. If concerns for game design, user immersion, and content creation are considered, this time cost would only increase. Additionally, problems created by educators or developers are often presented in a sequence, and once the in-game content is exhausted, the experience is generally over. Replayability is a major component of successful games [20], and games constructed in this way simply cannot be replayable experiences.

According to Scott Nicholson, allowing users to create game content, such as new levels and puzzles, "extends the life of a game and allows the designers to see how creative users can be with the toolkits provided." Many principles from the use of User-Generated Content (UGC) can be used to improve Serious Games by allowing players to set their own goals [16]. Additionally, design patterns for educational games identified by a team at Microsoft Research in [18, 19] indicate that allowing users to create their own challenges is a very powerful motivator.

Previous work done with UGC in serious games showed that level creation increased player motivation, especially for players interested in creativity [6]. By creating games that are solitary, non-replayable experiences, serious games developers are failing to harness the community experience that modern games provide, and may fail to provide ways to refine learned skills outside of rigidly structured areas. Even outside of ITS, software like Scratch, Alice, and other programs often feature communities that highly resemble Steam, Miiverse, and even communities like Wikipedia or YouTube [9, 15], in an effort to provide this type of experience.

1.2 User Generated Content in BOTS

To investigate how best to use UGC in a Serious Game, I present BOTS. BOTS is a socially multiplayer online game built using the Unity 3D Game Engine where players take on the task of programming various robots. Programs take the form of a graphical pseudo-code, where players drag and drop icons representing various commands to direct the robot. Players also can condense a sequence of commands into a single icon by creating a function, and can create loops and conditional statements to further optimize their solutions.

In BOTS, players have to manage several resources to succeed. The size of the players’ programs is restricted to 25 commands. Players also have a limited number of instances of each command to work with. Both of these constraints are designed to encourage players to minimize the repetition of code, using functions and loops where necessary. These are important lessons for novice programmers to learn, and in this environment, can be taught independent of any specific language's syntax.

BOTS also has a collaborative/social aspect. Players can create new puzzles and share them with the game community. Our goal with this feature is to promote a higher level of engagement so that game-based tutors like BOTS can be used as more than a novel substitute for a homework assignment. BOTS should be a full-fledged educational tool that can be used by players throughout their introduction to programming The game was designed using the “Flow of Inspiration” principles outlined in [21]. This creates an environment where players can continually challenge their peers to find better solutions for difficult levels. Previous work has shown that content creators spend more time on their tasks when they have a target in mind [1]. I hypothesize that orienting content-creation as a social task may increase the quality of levels created.

While UGC certainly has a lot to offer for a system like ours, there are also several downsides to it, which can hinder or disrupt game-play. If it is possible to create a system within which users can be trusted to create useful, quality content, then developers will be able to spend more time developing the core of the game, ensuring that the game mechanics are both fun for players and in line with learning objectives. With these advances, we will be able to address many of the NSF's goals for Cyberlearning [17], and BOTS and systems like it may expand to play a more important role in early STEM education.

2. METHODS

In [2], the authors used a Machine Learning approach based on the tagging habits of users to identify low-quality Wikipedia articles. I hope to be able to use similar data-driven methods to analyze user-created levels. Lacking a large installed user-base to tag submitted levels, I work with player solutions as they are submitted, with the first being the author's own solution to the submitted level.

While the quality of a game level is subjective, I developed a set of criteria for a level in our game to be "useful", inspired by the use of design patterns in level design analysis [11]. To identify common design patterns, I examined levels created over the course of several game sessions with three groups. 24 levels were created by the first group, 13 by the second, and 11 by the third. Once I identified these patterns, I detailed how they could impact gameplay, why a level creator could be motivated to create them, and how developers could affect that motivation through game mechanics or incentives.

2.1 Identifying Low-Quality Submissions

Through examination of the existing levels, I identified several patterns of unwanted UGC. Interestingly, these types of levels fall in line with the player behavior types outlined in Bartle's work with online communities [4, 5], **killers, achievers, explorers,** and **socializers.** In the interest of space, I will specifically name only four of these low-quality design patterns here.

- **"Sandbox" levels**, which feature erratically-placed elements. I believe these levels are often created by users who are unfamiliar with the creation interface.
- **"Punisher" levels**, which feature unusually difficult or tedious solution paths, characterized by programs which are trivial but time-consuming to write.
- **"Griefer" levels**, which feature visual obstacles or other abuses of game mechanics, which I believe are intended to frustrate the user. These levels may or may not have solutions.
- **"Trivial" levels**, whose optimal solution is readily apparent to players, and which requires no use of the game's more advanced or difficult concepts to complete.

Based on these classes of unwanted UGC, I developed an evaluation rubric to score levels based on the features they contain. For our purposes, a "high quality" level should:

- Contain an obvious trivial solution
- Contain a different, optimized solution
- Contain structural cues for that optimization
- Contain few unnecessary structural cues
- Take less than 5 minutes for an expert to solve

To explain the last criteria, compare this to a long, completely featureless level in a 2D platform game. The task itself is not providing difficulty, but achieving what should be a simple goal has become unnecessarily obtrusive [12, 13]. I used the above criteria to evaluate levels in the next part of our investigation.

2.2 Moderating User Submissions

To see how different game mechanics affected the levels created, we implemented several different types of moderation which we believed could discourage players from submitting low-quality UGC. Students at a STEM-related after-school program played BOTS for one hour under one of three conditions, and we examined the levels they created using the rubric I had previously developed. The three types of moderation investigated were as follows:

Condition 1: Unrestricted Level Submission

There is no filtration process in place and the puzzle must only pass the base conditions of each level having a starting point and goal. If the level has those conditions, it will be made public and immediately available for play as soon as the participant submits it.

Condition 2: Self-Evaluation

The participant must first submit a solution for the level they just created before it would be made accessible to the public. I expect that this will reshape the level creation process so that more successful creators will build levels while already having a solution in mind.

Condition 3: Moderator Approval

When a participant submits a level, it will be placed in a queue where an admin can examine the level and determine if it is appropriate to publish. The admin will then reply to the participant either accepting or rejecting that level which was submitted for approval.

2.3 Preliminary Results

After the session, a researcher who was blind to the conditions each level was created under "graded" each level on a simple rubric addressing the criteria discussed above.

	Condition 1	Condition 2	Condition 3
Created	3	4	9
Published	3	2	4
M(Quality)	2	2.5	2.3
M(Published Quality)	2	5	4.5

Figure 1 - Measured quality of submitted levels

I also analyzed the best solutions to these levels using an expert-solver, looking at the difference (in terms of number of commands used) between a naive solution using neither loops nor functions and a master solution using a combination of both techniques.

	Condition 1	Condition 2	Condition 3
Mean	9	11.5	18.6666667
SD	4.58257569	14.0830868	19.4250697
Max	14	32	40
Min	2	0	2

Figure 2 - Differences between naive and expert solutions

Though we were able to collect relatively few levels, the collected data are encouraging. The quality of the published levels in

Condition 2 is similar to that of the levels under Condition 3. Interestingly, in both conditions where some form of moderation is present, the average quality of all levels, including those left incomplete or unpublished. is slightly higher. Though I have a very small sample size in this study, I hope to be able to investigate these effects with a larger group of players.

3. FUTURE WORK

In addition to replicating the above experiment with a larger group of students, I have already begun an investigation of how to further use student data to moderate and evaluate submitted levels. Being able to assess UGC in this way allows us to provide meaningful problem orderings even with levels I have not analyzed in depth, as well as provides us with a metric which can be used to reward players for creating specific types of levels, or levels which fill in gaps in content or difficulty. In the future, I will experiment with different methods of directed level creation using the information gained, to see if level creation can be better integrated into the system as a learning activity in and of itself.

4. ACKNOWLEDGEMENTS

Thanks to my advisor, Dr. Tiffany Barnes, and to the developers who have worked on BOTS so far, including Veronica Catete, Monique Jones, Andrew Messick, Thomas Hege, Michael Clifton, Vincent Bugica, Victoria Cooper, Dustin Culler, Shaun Pickford, Antoine Campbell, and Javier Olaya. This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No 0900860 and Grant No 1252376.

5. REFERENCES

[1] Aleahmad, T., Aleven, V., and Kraut, R. Open community authoring of worked example problems. In Proceedings of the 8th international conference on International conference for the learning sciences (ICLS'08), Vol. 3. 3-4.

[2] Anderka, M., Stein, B., and Lipka, N. Predicting quality flaws in user-generated content: the case of wikipedia. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval* (SIGIR '12). ACM, New York, NY, USA, 981-990.

[3] Bayliss, J. D. Using games in introductory courses: Tips from the trenches. In *Proceedings of the 40th ACM Technical Symposium on Computer Science Education*. (SIGCSE '09). ACM, New York, NY, 337-341

[4] Bartle, R. A.. Hearts, clubs, diamonds, spades: Players who suit MUDs. *Journal of MUD research*, 1-19, 1996.

[5] Bartle, R. A. *Designing Virtual Worlds.* Boston, MA: New Riders / Pearson Education. 2004

[6] Boyce, A., Doran, K., Pickford, S., Campbell, A., Culler, D., and Barnes, T. BeadLoom Game: Adding Competitive, User Generated, and Social Features to Increase Motivation. In *Proceedings of the Foundation of Digital Games* (FDG '11). ACM, New York, NY, USA, 243-247.

[7] Carmel, D., Roitman, H., and Yom-Tov, E. 2012. On the Relationship between Novelty and Popularity of User-Generated Content. *ACM Trans. Intell. Syst. Technol.* 3, 4, Article 69 (September 2012)

[8] Chaffin, A., Doran, K. Hicks, D., and Barnes, T. Experimental evaluation of teaching recursion in a video game. *In Proceedings of the 2009 ACM SIGGRAPH Symposium on Video Games* (Sandbox '09), Stephen N. Spencer (Ed.). ACM, New York, NY, USA, 79-86.

[9] de Kereki, I.F. Scratch: Applications in Computer Science 1. In *Proceedings of Frontiers in Education Conference*, (FIE 2008). 22-25 Oct. 2008

[10] Eagle, M., Barnes, T. Experimental evaluation of an educational game for improved learning in introductory computing. In *Proceedings of the 40th ACM technical symposium on Computer science education* (SIGCSE '09). ACM, New York, NY, USA, 321-325.

[11] Hullett, K. and Whitehead, J. Design patterns in fps levels. In *Proceedings of the Fifth International Conference on the Foundations of Digital Games* (FDG '10) ACM, New York, NY, USA, 2010.

[12] Juul, J. In search of lost time: on game goals and failure costs. In Proceedings *of the Fifth International Conference on the Foundations of Digital Games* (FDG '10) ACM, New York, NY, USA, 2010.

[13] Juul, J. Easy to use and incredibly difficult: on the mythical border between interface and gameplay. In *Proceedings of the Sixth International Conference on the Foundations of Digital Games* (FDG '11) ACM, New York, NY, USA, 2011.

[14] Murray, T. An overview of intelligent tutoring system authoring tools: Updated analysis of the state of the art. In *Authoring Tools for Advanced Learning Environments*. T. Murray, S. Blessing, and S. Ainsworth (Eds.) Chapter 17, 491-544. Dordrecht, the Netherlands: Kluwer Academic Publishers.

[15] Malan, D, Leitner, H, Scratch for budding computer scientists. In *Proceedings of the 38th SIGCSE technical symposium on Computer science education* (SIGCSE '07). ACM, New York, NY, USA, 223-227.

[16] Nicholson, S. A User-Centered Theoretical Framework for Meaningful Gamification. In *Games+Learning+Society* (GLS 2012), Madison, WI.

[17] NSF Publication. Cyberlearning: Transforming Education. www.nsf.gov/pubs/2010/nsf10620/nsf10620.htm

[18] Plass, J., and Homer, B. Educational Game Design Pattern Candidates. White Paper, Institute for Games for Learning, 2009 http://g4li.org/research

[19] Plass, J., and Homer, B. Learning Mechanics and Assessment Mechanics for Games for Learning. White Paper, Institute for Games for Learning, 2011 http://g4li.org/research

[20] Prensky, M. Computer Games and Learning: Digital Game-Based Learning. In *Handbook of Computer Games Studies*. Cambridge MA, MIT Press; 2005

[21] Repenning, A., Basawapatna, A., and Koh, K. H. Making university education more like middle school computer club: facilitating the flow of inspiration. In *Proceedings of the 14th Western Canadian Conference on Computing Education* (WCCCE '09). ACM, New York, NY, 9-16.