Review of Teaching Recursion using Educational Games

Moegamat Ra-eez Stenekamp

Computer Science Honours 2019

University of Cape Town

stnmoe001@myuct.ac.za

Abstract

This review will investigate the teaching of the difficult concept of recursion in Computer Science and software development, which many students have difficulty in understanding. Serious/ educational games will then be explored to see how they could be used to aid teaching, particularly with regards to recursion. Game generation and how game generation can be used to generate serious games to aid the teach recursion will then be explored.

1. Introduction

Recursion is an extremely important topic that every programmer should know. It is necessary to solve certain problems, especially when it is used as the conceptual model to solve abstract problems, as described by Wu, Dale and Bethel [23]. Even though it is so important, it is a difficult topic for many students to grasp due to it being so different from the iterative way that they would have been used to learning [20].

Games have been used to aid teaching for decades, however this has become more prevalent in recent years with the increasing number of serious and educational games being developed. Serious games are games which have an alternate focus other than entertainment. When a serious game's focus is on learning and education, then it becomes an educational game [22]. There have even been serious games created that help teach the difficult task of recursion. These games have yielded good results showing their effectiveness in teaching this subject [3,4].

Game generation is extremely useful for creating customisable games [5]. These games can be serious/ educational games, which are useful as they allow teachers who have no experience in coding to be able to create games quickly and with their own customisations [17].

2. Recursion

Recursion is one of the harder topics for an introductory level programmer to grasp. This was tested by Milne and Rowe [20]. This test involved comparing the difficulty students found when learning all the topics generally taught to those in the first year of learning programming. The results revealed recursion being among the top six most challenging topics. Tarbak, Royden, Stephan and Herbst [21] believed that the reason students seem to struggle learning recursion initially is due to recursion requiring a whole new form of thinking to the standard iterative way of thinking students would have been accustomed to.

2.1 Teaching Recursion

Ginat and Shifroni [10] pointed out the fact that those just starting Computer Science tend to struggle with getting a good understanding of recursion and that there are multiple solutions already existing to help aid the teaching of recursion. Ginat and Shifroni found that a big problem in the teaching of recursion stemmed from a limited understanding of the basic computing model. They went about the approach of teaching recursion by focussing on declarative and abstract representations recursion. Ginat and Shifroni found that this tied in nicely with the way that Computer Science is generally taught. This allowed students to understand recursion faster as it followed a similar way of thinking that they had been introduced to when taught iteration. This addresses Stephan and Herbst's [21] belief stated above (in section 2 – Recursion) that the reason students seem to struggle with the recursion was due to their iterative way of thinking they were introduced to.

Wilcocks and Sanders [14] took an alternative approach to Ginat and Shifroni's approach [10] described above (in section 2.1 – Teaching Recrusion). This alternative approach they proposed was that the optimal way of teaching recursion to students was to use visualisations to help show the students a better way of picturing and visualising recursion and the whole process of a recursive function with recursive calls. Wilcocks and Sanders' study provided evidence that the visualisation approach both worked and was better than the alternative approaches they tested. The alternative approaches they tested against varied in terms of the models used and presented to students.

3. Teaching Using Games

Stapleton [22] explains how some games have shifted their focus from entertainment to being used to aid teaching. This has become more prevalent in recent years and it is used in all different fields, not only Computer Science and tech related fields. Sanford, Ulicsak and Facer [19] found that there was a great divide between students and teachers, with only 28% of teachers evaluated who play video games for leisure compared to the 82% of the evaluated students who play video games for leisure. This study further showed how students respond better to games and learning with games, indicated that it would be worthwhile to incorporate learning (particularly in difficult topics) with games into the teaching curriculum or at the very least for teachers to incorporate it into their lessons where appropriate.

3.1. Serious/ Educational Games

Serious games are games that, unlike the more traditional view of a game, are focussed on another goal that is not entertainment. Educational games are a form of serious games that are focussed around teaching and education, as Backlund and Hendrix [1] describe.

3.2. Proven Effectiveness

Stapleton [22] shows how games are effective when aiding teaching as they provide a more learner-cantered approach to education, rather than the usual teacher-cantered approach of regular teaching. Hays' study [2] supports Stapleton's findings [22] as Hays' work [2] also says that games are an effective way of aiding teaching, however he expands further stating that the effectiveness of educational games vary. This study explains that this range of effectiveness heavily depends on what is being taught and how the game teaches it.

Backlund and Hendrix [1] evaluated the results from various studies testing the effectiveness of educational games in learning over learning without educational games. Figure 1 (below) shows these results, which depict 29 out of 40 of the studies had positive results with educational games being effective. Only 2 of the 40 studies said that the effectiveness was negative. (The remaining studies had neutral outcomes with 2 of them having unclear results).

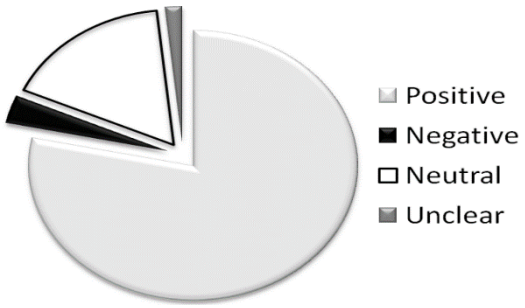

Fig 1: Pie Chart Showing Effectiveness of Educational Games in Learning by Backlund and Hendrix [1]

It was shown in a later study done by Backlund and Hendrix [4] that the reason why many other projects utilising educational games fail at teaching the material, is that they are not hands on enough for the students. This shows that the cause for many studies yielding negative results in Backlund and Hendrix's previous writing [1], could be because of the studies' chosen approach not being hands on enough.

3.3. Should Games Aid Teaching Programming Related Topics

Vahldick, Mendes and Marcelino [17] showed how serious games are a great option to implement to aid in the teaching of Computer Science. Various games found online that claim to support the development of competencies in introductory computer programming learning were tested and evaluated by Vahldick, Mendes and Marcelino [17]. This study found that all the games evaluated aided the teaching of the content in one way or another, however the level of the students they were aimed at varied as well as the effectiveness of their aid in teaching Computer Science. Although some of the games found barely had any effect in the learning process, it was still found to be of some help. This suggests that serious games are a great option when it comes to aiding in teaching Computer Science as even the ones of lesser quality are still of some aid in the learning process.

Khenissi, Essalmi and Jemni [7] tested the use of educational games to increase the motivation of students with regards to programming language education. They also made use of a technique in which it makes the content seem less mundane by representing the data in another format, allowing the users to not feel like it is coursework being taught in a classroom. This was achieved by having different functionalities (such as drag and drop technology and avatars) that are used appropriately with the type of learning game used. Figure 2 (below) shows Khenissi, Essalmi and Jemni's results from this in the form of a table [7]. In this table '-' means that the game functionality is not appropriate for a particular leaning game. In this table '+' was used to show the level of effectiveness of a functionality with regards to a leaning game (the amount of '+' refers to the level of its effectiveness respectively).

Learning games / Game functionality	MARVIN	Crossword	Shooting	Prog&Play	Circuit Game	Bomberman	RPLG	IRPG
Personalization	-	-	-	-	-	-	-	+++
Online availability	-	-	-	-	-	-	-	+++
Immediate feedback	+++	++	+++	+	+++	+	-	+++
Drag and drop technology	-	-	-	-	+++	-	-	+++
Self explanation	-	-	-	-	+++	-	-	+
Avatars provide helps	+++	-	+	-	-	-	-	+++

Fig 2: Table Showing the Effectiveness of Various Game Functionalities on Different Learning Games

3.4. Implementation of Games Aiding the Teaching

There are many ways games were implemented to aid the teaching of recursion. Kazimoglu, Kiernan, Bacon and Mackinnon [11] took an approach focussed on deep learning to help develop computational thinking to aid in the expansion of user's knowledge of programming. Their study stated that other serious games that are currently out there did not consider deep learning (for computational thinking specifically), so that was their focus. Su and Cheng [18] created a serious game geared towards teaching mobile development using the programming language Lua. It was created to be played on the mobile platform. This took a unique approach in that it focussed on getting aspects from casual games to make the game more entertaining for the users, which was very different to Kazimoglu, Kiernan, Bacon and Mackinnon's [11] focus on deep learning described earlier.

Distasio and Way [15] took a different approach to Su and Cheng [18] and Kazimoglu, Kiernan, Bacon and Mackinnon [11]. Distasio and Way's approach [15] focussed on making computer game development accessible to less skilled/ early stage programmers, since it can be quite a technical, challenging and daunting field to someone new to creating games. This is achieved by creating a base and simple game, and then the programmers learning would be challenged to try and do tasks ranging from simply changing the image of a character in the game for the more beginner programmers, to changing how one of the non-player characters move for the slightly more advanced programmers.

3.5. Teaching Recursion using Educational Games

An educational game was created by Chaffin, Doran, Hicks and Barnes [3,4] that has the goal of teaching recursion. When they tested and compared the knowledge of students that used their game and those who had not, they found very noticeable results which show those who used the game had a far greater understanding of recursion than those who did not. Although they would have liked a larger sample size, the results were clearly noticeable. Another example of an educational game teaching recursion was a game in which the user must create recursive programs to accomplish the various goals, as described in the studies by Telssler, Beth and Lin [12] and the study that expanded on that by Lee, Shan, Beth and Lin [13]. After testing this they found that the students had both enjoyed playing the game and had a better understanding of recursion afterwards. The students said they felt more confident in their ability to work with recursion in a review afterwards.

4. Game Generation

Game generation is software that will create a game when it is run. These may vary, from randomly generated games, to games that get created with specific user input. Treanor, Blackford, Mateas and Barnes [5] show that it is possible to create a game generation tool where the users can create their own custom games. This gives the users some control over the design of their games. This also incorporates forms of customisation as the users can upload their own images to the game objects itself. There is also the ability to keep modifying the generation process until a satisfactory generation is found. This also promotes reuse, as new games can be generated allowing for multiple unique experiences.

4.1. Game Generation with Customisations

Khalifa, Perez-Liebana, Lucas and Togelius [8] present an example of level generation not game generation, however the principles still apply. There are 3 main types of level generation: random, constructive and search-based. In their implementation the user can choose which type of level generation they want, this principle could be applied to a game generator, in which the user would be able to choose among some choices which could affect how the game is generated. Lando [9] shows an example of game generation with this concept of core functions and customisations. In this project the game was split up into two key notions, handlers and assignments. Handlers are the users movements rules (such as whether the character can jump or double jump, the directions they are allowed to move etc), which the user customises and sets before the game is generated. Assignments are tasks that the player will have to accomplish in the game, these are dependent on the movement restrictions the players have (for example an assignment would have to be obtainable given the set of movements allowed by the handler). This allows the user to customise and set the core function (the movement in the handler) and then the game would be generated around their specifications.

Bouzid, Khenissi and Jemni [6] created a game generator that was simple to use and allowed customisation without needed programming knowledge. Users can create things such as custom avatars before the game gets generated, allowing for a form of customisation to make the game

more of a unique experience while simultaneously making the game a more enjoyable experience for the user to play.

4.2. Generating Serious Games

Eaachak, Belhabibe and Bouhorma [16] created a serious game generator that generates a serious game over a web-based service in which the user (presumably a teacher) would be able to input their own information, and then a serious game would be developed using those parameters. The purpose of this was to make it so that teachers would be more inclined to use serious games for educational purposed by making it easier and more accessible for them to do so. The users would not need any programming knowledge, and it would be less time consuming that making it themselves if they would know how to program.

4.3. Game Generators that Generate Educational Games Teaching Recursion

There are no studies found on this particular topic, suggesting that this is an open field for research that has not been documented before.

5. Conclusion

Educational games are an extremely effective teaching aid as they help students engage with the content while keeping their motivation high. These games yield extremely successful results when being used to aid the teaching of Computer Science and programming languages. In this field recursion is one of the topics that students find the most difficulty in understanding. Games that are created to help teach recursion have been extremely successful since they allow the user to engage with the topic more and have a better visualisation of how recursion works. The huge success of these games suggest that this is the most optimal solution for students to use when learning recursion.

Game generation shows the ability for customisability and reuse, however there are currently no game generators for games that teach recursion.

Therefore there is a need for a game generator for generating educational games that aid teaching recursion as these will give the teacher the power to customise the game they create to suit their lesson, even if they do not have programming experience, while creating an educational game that allows the students to engage with and get a better understanding on the difficult topic of recursion.

6. References

[1] Backlund, P., & Hendrix, M. (2013). *Educational games - Are they worth the effort? A literature survey of the effectiveness of serious games. 2013 5th International Conference on Games and Virtual Worlds for Serious Applications (VS-GAMES)*.doi:10.1109/vs-games.2013.6624226

[2] Hays, R. T. (2005). *The effectiveness of instructional games: A literature review and discussion (No. NAWCTSD-TR-2005-004). Naval Air Warfare Center Training Systems Div Orlando Fl.*

[3] Chaffin, A., Doran, K., Hicks, D., & Barnes, T. (2009). *Experimental Evaluation of Teaching Recursion in a Video Game (pp. 79-86).*

[4] Chaffin, A., Doran, K., Hicks, D., & Barnes, T. (2009). *Experimental evaluation of teaching recursion in a video game. In Proceedings of the 2009 ACM SIGGRAPH Symposium on Video Games* (pp. 79-86).

[5] Treanor, M., Blackford, B., Mateas, M., & Bogost, I. (2012). *Game-o-matic: Generating videogames that represent ideas. In Proceedings of the The third workshop on Procedural Content Generation in Games* (p. 11).

[6] Bouzid, Y., Khenissi, M. A., & Jemni, M. (2015). *Designing a game generator as an educational technology for the deaf learners. In 2015 5th International Conference on Information & Communication Technology and Accessibility (ICTA)* (pp. 1-6). IEEE.

[7] Khenissi, M. A., Essalmi, F., & Jemni, M. (2013). *Presentation of a Learning Game for Programming Languages Education. 2013 IEEE 13th International Conference on Advanced Learning Technologies*.doi:10.1109/icalt.2013.100

[8] Khalifa, A., Perez-Liebana, D., Lucas, S. M., & Togelius, J. (2016). *General video game level generation. In Proceedings of the Genetic and Evolutionary Computation Conference 2016* (pp. 253-259).

[9] Landro, A. C. S. (2007). *A game generator-the framework for an educational system development game*.

[10] Ginat, D., & Shifroni, E. (1999). *Teaching recursion in a procedural environment---how much should we emphasize the computing model? The Proceedings of the Thirtieth SIGCSE Technical Symposium on Computer Science Education - SIGCSE '99*.doi:10.1145/299649.299718

[11] Kazimoglu, C., Kiernan, M., Bacon, L., & Mackinnon, L. (2012). *A Serious Game for Developing Computational Thinking and Learning Introductory Computer Programming. Procedia - Social and Behavioral Sciences, 47, 1991–1999*.doi:10.1016/j.sbspro.2012.06.938

[12] Tessler, J., Beth, B., & Lin, C. (2013). *Using cargo-bot to provide contextualized learning of recursion. In Proceedings of the ninth annual international ACM conference on International computing education research* (pp. 161-168).

[13] Lee, E., Shan, V., Beth, B., & Lin, C. (2014). *A structured approach to teaching recursion using cargo-bot. In Proceedings of the tenth annual conference on International computing education research* (pp. 59-66).

[14] Wilcocks, D., & Sanders, I. (1994). *Animating recursion as an aid to instruction. Computers & Education, 23(3), 221–226*. doi:10.1016/s0360-1315(05)80013-2

[15] Distasio, J., & Way, T. (2007, June). *Inclusive computer science education using a ready-made computer game framework. In ACM SIGCSE Bulletin* (Vol. 39, No. 3, pp. 116-120).

[16] Eaachak, L., Belhabibe, A., & Bouhorma, M. (2015). *Towards a new concept of serious games generator. 2015 International Conference on Electrical and Information Technologies (ICEIT)*. doi:10.1109/eitech.2015.7162973

[17] Vahldick, A., Mendes, A. J., & Marcelino, M. J. (2014). *A review of games designed to improve introductory computer programming competencies. 2014 IEEE Frontiers in Education Conference (FIE) Proceedings*.doi:10.1109/fie.2014.7044114

[18] Su, C. H., & Cheng, C. H. (2015). *A mobile gamification learning system for improving the learning motivation and achievements. Journal of Computer Assisted Learning*, *31*(3), 268-286.

[19] Sandford, R., Ulicsak, M., & Facer, K. (2006). *Teaching with Games: using computer games in formal education. Futurelab, Bristol*.

[20] Milne, I., & Rowe, G. (2002). *Education and Information Technologies, 7(1), 55–66*. doi:10.1023/a:1015362608943

[21] Turbak, F., Royden, C., Stephan, J., & Herbst, J. (1999). *Teaching recursion before loops in CS1. Journal of Computing in Small Colleges, 14(4), 86-101*.

[22] Stapleton, A. J. (2004, September). *Serious games: Serious opportunities. In Australian Game Developers" Conference, Academic Summit, Melbourne*.

[23] Wu, C. C., Dale, N. B., & Bethel, L. J. (1998, March). *Conceptual models and cognitive learning styles in teaching recursion. In ACM SIGCSE Bulletin (Vol. 30, No. 1, pp. 292-296).*