A Graph-based Approach for Adaptive Serious Games

Ryan Hare, *Student Member, IEEE*, Nidhi Patel, Ying Tang, *Senior Member, IEEE,* and Pankti Patel
Electrical and Computer Engineering
Rowan University, Glassboro, USA

Abstract—**In higher education, researchers have recently focused on exploring automated, personalized instructional systems to enhance students' learning experiences. Motivated by this, we propose a personalized instructional system using a straightforward graph system to offer an educational game that is effective for students and intuitive for developers. Our system uses a directional graph, called an action graph, for representing solutions to in-game problems based on possible player actions. Through our proposed algorithm, a serious game integrated with our system would both detect player errors and provide personalized assistance to direct a player in the direction of a correct solution. To verify system performance, we present comparison testing on a group of students engaging in the game both with and without AI. Students who played the AI-assisted game showed an average 20% decrease in time needed and average 58% decrease in actions taken to complete the game.**

Keywords— Adaptive game, action graph, personalized system, educational game

I. Introduction

A number of recent reports make it clear that the classic one-size-fit-all teaching method is not universally effective given the wide range of preferred learning styles and personality traits found in students [20], [21]. While most students benefit more from guided learning, it is extremely challenging for instructors to offer specific guidance, particularly if the guidance needs to be tailored to individual student needs. However, technological advancement has made it possible to explore the interplay of learning, student behavior, and student attributes. These advancements have led to several recent research efforts to explore and develop serious games for learning [15]-[19]. By integrating pedagogical content into a virtual environment, both students and educators are provided with a solid structure for contextual learning. Additionally, easy access to student data through such virtual environments provides a wide range of opportunities to understand what truly happens when students are stuck on a problem. Based on that wide range of student data, accurate and relevant scaffolding and support can then be easily offered.

A perusal of current literature provides a number of works that take advantages of serious games for adaptive learning systems. For example, one work from Papadimitriou *et al.,* (2019) presented a fuzzy logic-based approach to dynamically adjust quiz questions and game content based on a player's performance in an escape room game written in HTML [8]. Another approach by Hussaan *et al.,* (2011) focused instead on generating educational scenarios tailored to players based on a detailed player profile that was populated prior to game intervention [9]. Takahashi *et al.,* (2018), meanwhile, focused more on a social aspect of serious games by dynamically selecting dialogue for characters in the game to create more human-like conversations and statements to say to the player [10].

Adaptive methods in serious games often make use of the game environment by modifying it or even generating new environments. For example, Gombolay *et al.,* (2019), proposed a method for generating sequences of tutorials from test results, applying hidden Markov models to generate sequences automatically [11]. One approach by Mitsis *et al.,* (2020), even focused on dynamically generating the behavior of in-game characters, targeting behavior that would further enhance a player's learning using genetic algorithms [12]. Even when not modifying content directly, games can instead provide hints to players. For example, González-González *et al.,* (2019) proposed a system that observed the action history of players to adaptively recommend future exercises [13].

Although most approaches described above demonstrated proficiency in improving student learning, they also are often very specific to the domain of choice, limiting potential applications in other serious games. But, all serious games generally share a similar structure, in which students must perform sequential decision-making processes in order to find the right solution to a presented problem. And as there are correct sequences of actions to arrive at such a solution, we can then define a solution graph that represents all acceptable sequences of actions that arrive at a solution. With this remark, this paper develops a graph-based heuristic approach to guide the design of an adaptive serious game. In particular, the paper makes the following contributions: We provide a graph-based framework by which solutions to in-game problems from educational serious games can be represented. Using this framework, we develop a graph search algorithm to both detect errors in a player's in-game actions and to provide assistance to a player when needed, guiding them toward the correct solution. Finally, we test the proposed system through a case study in which we compare student opinions and in-game performance on a game with and without the proposed system.

The rest of the paper is organized as follows: Section II overviews our proposed methods, including action graphs and our heuristic algorithms. Section III provides our case study for an adaptive serious game, followed by conclusions in Section IV.

II. Action-Graph-Based Heuristic for Adaptation

As players are engaged in a serious game to solve a problem, they usually associate sequences of actions with game scenarios, creating a "best" decision at any given moment that leads to a specific destination/objective. For a given problem, an action graph can be built beforehand by a domain expert to represent all possible action sequences that lead to the destination/objective. As players learn to play the game, their actions are logged as an action model. When superimposing the action model on top of the solution graph, it is easy to compare

This work was supported in part by the National Science Foundation under Grant 1913809, Grant 1610164, and Grant 2121277.

the player action sequence with the action graph and recognize the differences. Such diagnosis allows the system to easily spot the wrong actions players take and provide appropriate guidance to correct the player. In this way, the players can more easily grasp the problem presented to them and reach the objective quicker. Following this line of thinking, this section presents the solution-action tree structure first in Section II.A. The heuristic algorithm for error detection and personalized guidance is then presented in Section II.B.

A. Action Graph

To represent and search the possible solution space within a serious game, we propose representing the space of possible solutions as an action graph. An action graph is a rooted tree data structure where each node represents a single action that a player could take within a given serious game. In this way, such a graph can be formally defined as $G = (N, E)$, where N is a set of nodes and E is a set of directionally-connected edges:

- $N = \{n_1, n_2, \dots, n_j\}$ The set of nodes that comprise the action graph. The single root node from which all directional connections originate is defined as $n_1 \in N$.
- $E = \{e_1, e_2, \dots, e_j\}$ The set of directional edges, each of which connects from some source node $n_s \in N$ to some destination node $n_d \in N$.
- $f_p(e)$ is a function that, given an edge $e \in E$, returns the source node from which the given edge originates, also known as the parent node.
- $f_c(e)$ is a function that, given an edge $e \in E$, returns the destination node to which the given edge travels, also known as the child node.
- $C(n_i) = \bigcup\{n \in N \mid \exists e \in E\colon n_i = f_p(e), n = f_c(e)\}$ is a function that, for a given node $n_i \in N$, returns a set of all nodes $n \in N$ that are child nodes of n_i.
- $L = \bigcup\{n \in N \mid \nexists e \in E\colon n = f_p(e)\}$ is the set of all leaf nodes $n \in N$, defined as all nodes that do not have edges originating from them.

The action graph for a specified game is defined in such a way that following any sequence of in-game actions from the root node to a leaf node will result in a complete, correct solution applied to the relevant in-game problem. Thus, by extension, any unique sequence of nodes that starts at n_1 and traverses to a leaf node represents a unique solution to the given in-game problem.

B. Heuristic Algorithm

With the action graph described above, we seek to implement error detection, adaptive student guidance, and solution checking all through one combined algorithm. The proposed algorithm uses a list of actions that a student has taken in a game, traversing the graph to both check that the student is still on track toward a solution and, in the case of an error, providing assistance to push the student in the direction of the "nearest" solution. This approach is formalized in Algorithm 1.

Algorithm 1 then returns a flag if the player has made an error and returns the list of actions the player must take to move from their current state to the nearest solution. To fully integrate this low-level algorithm into the system and provide personalized student assistance, we also must consider the player's overall performance. If we provide hints and assistance on every single error a player makes, they have little opportunity to learn on their own. Thus, we consider a performance score.

Algorithm 1: Action graph searching

Inputs: Action graph for target in-game problem, G
Ordered list of player actions, P, where $p[i]$ represents the i-th element of P and $p[1]$ is the first action associated with the root node, $n_1 \in N$

1: **Initialize** $n = n_1 \in N$
$error = false$
2: **For** $i = 2, i = i + 1, i < |P|$ **do:**
  a. **If** $p[i] \in C(n)$ **do**:
    i. $n =$ the node $\in C(n)$ that represents $p[i]$
  b. Else if $p[i] \notin C(n)$ **do:**
    i. $error = true$
    ii. Break
  c. End if
3: **End for**
4: Starting at $n \in N$, **depth first search** to find the shortest path to a leaf node.
5: Parse shortest path into action list P^*, which is the list of actions the player must take to reach the nearest solution.
6: **Return** $error$ and P^*

The performance score is a numerical indicator of a student's performance in the game, with higher values indicating more negative performance. The specific method for computing performance score depends on implementation, though typically, the score would be derived from performance indicating variables such as number of incorrect answers, time taken to make a move, number of past errors, or other relevant values. Finally, Algorithm 2 shows the top-level method for solution checking and providing assistance.

Algorithm 2: Player assistance and solution checking

Inputs: Action graph for target in-game problem, G
Player's prior performance score, ω
Minimum performance threshold, ω_{min}

1: **Initialize** player action list $P = \emptyset$ and $\omega = 0$
2: **While** problem is not solved **do:**
  a. **If** P is updated from the system **do:**
    i. Receive list of player actions P from system
    ii. Call **Algorithm 1** with G and P as inputs to get $error$ and P^*
    iii. Compute new ω using errors and other player data
    iv. If $\omega \geq \omega_{min}$ do:
      1. Use P^* to provide hints to the player.
    v. End if
  b. End if
3: **End while**

III. Case Study

To better visualize the educational impact of our proposed method, this section presents a case study from an in-classroom

Fig. 1. (a). Water Purification introduction scene; (b) Pipe challenge scene

implementation of our proposed action graph model. To measure system effectiveness, we implemented our proposed system and algorithms into a game called *Algae City*, and specifically into the Water Purification module. Section III.A briefly overviews *Algae City* and the Water Purification module, while Section III.B shows results from our case study applying the game with and without learning guidance from the proposed system.

A. *Algae City - Water Purification*

Algae City is an interactive educational game built in the Unity engine that focuses on educating students in various applications for algae. The game is designed for middle school students (age 11-13). While most students in the target audience will be aware of algae as a common photosynthetic organism, the game focuses on educating them in less common and more significant applications for algae. *Algae City* does so by prompting students to explore algae as an environmentally friendly solution to problems such as pollution, energy generation, vehicle fueling, and skin cosmetics. Specifically, the game is composed of 4 modules: water purification, algae reactors, algae fuel, and algae mart, each of which deliver different contents pertaining to the overall purpose of the game [14].

In our case study, we implemented the proposed system and adaptive support process into the water purification module. In the water purification module, students control a character in a virtual city park environment, where they immediately see a polluted lake. A cartoonish in-game character then prompts players to solve the lake pollution using algae, as shown in Fig. 1(a). Players are then tasked to solve a simple puzzle by placing pipe components from a given selection into a grid, shown in Fig. 1(b). The full problem to solve is to connect the input valve (left) to the algae scrubber (middle) and then to the output valve (right). As players solve levels of this game, the task is made more difficult through increasing grid size, different choices of pipe segments, additional algae scrubbers, and obstacles in the grid preventing pipe placement. Table 1 provides the full list of pipe segments that students use to solve the in-game problems. Taking the list of possible pipe segments, we can then derive an action graph for the first level of the pipe game, as shown in Fig. 2.

TABLE I. LIST OF PIPE SEGMENTS

Pipe	**Pipe name**	**Acronym**	**Quantity of Pipes**
	Elbow Pipe	Elb	2
	Elbow- Elbow Pipe	ElbElb	2
	Elbow-Straight Pipe	ElbStr	4
	Elbow-Elbow Curve Pipe	ElbElb C.	1
	Straight Pipe	Str	1
	Straight-Straight Pipe	StrStr	2

As shown in Fig. 2, a new player is initialized with node n_1 as their starting position with no prior actions taken. The space of possible solutions in the game level is then shown over the rest of the action graph with $N = \{n_1, n_2, \dots, n_{93}\}$. When starting, it can be seen in Fig. 2 that all possible actions lead to possible solutions with the exception of the elbow-elbow curve pipe. Taking an example list of player actions, P:

$$P = [Elb, Str, StrStr, Elb, ElbElb]$$

The algorithm would then traverse nodes $n_1 \rightarrow n_2 \rightarrow n_9 \rightarrow n_{19}$ before finding that no child nodes of n_{19} have the Elb action. In this situation, the system would then enable the error flag and stop traversing the graph. Algorithm 2 would then return a $true$ error flag and a path to the nearest solution, P^*:

$$P^* = [n_{30}, n_{42}, n_{52}, n_{62}, n_{69}, n_{76}, n_{84}]$$

The provided path would then be used to inform our integrated hint system. However, since this is the student's first error, they would likely not be provided immediate assistance. Similar to a human tutor, the goal of the system is to give students space to solve issues and overcome impasses through their own effort and learning processes. As such, the game does not provide immediate assistance but rather waits until the player's performance score exceeds a certain threshold.

Algae City collects student data as they play including number of movements, time taken to move, and number of errors, combining them to derive the performance score. When a student's performance score exceeds a certain threshold, Algorithm 2 will actually begin to provide assistance to the player using P^*. Say, for example, the student from the earlier

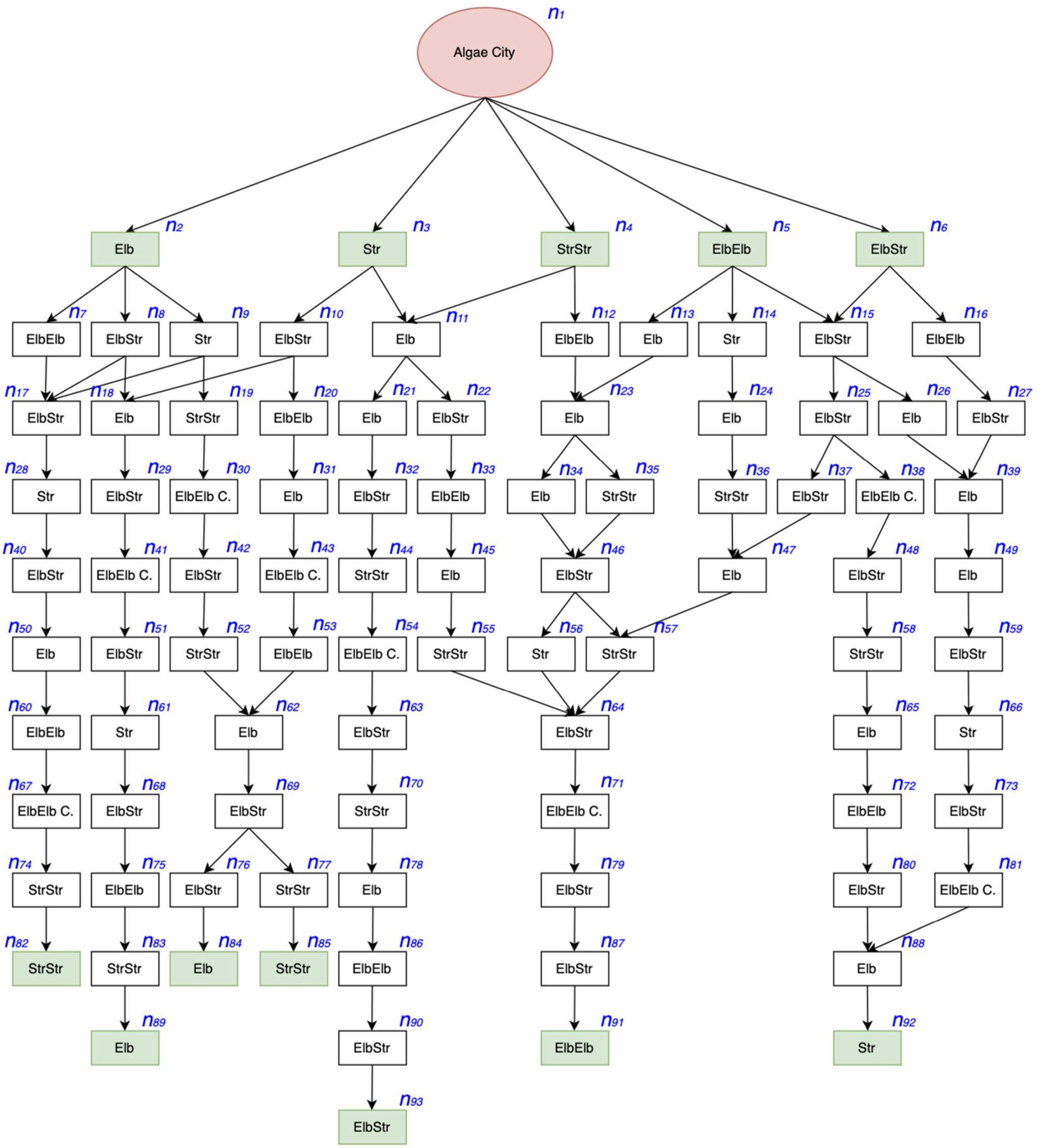

Fig. 2. The action graph of *Water Purification* representing all possible solutions to the problem based on what pipes are placed in what order.

example continues to play through and produces the following sequence of actions:

$$P = [Elb, Str, StrStr, ElbElb\ C., ElbStr, StrStr, ElbStr]$$

The system would then traverse from $n_1 \rightarrow n_2 \rightarrow \cdots \rightarrow n_{52}$ before encountering another error and enabling the error flag. Like before, P^* is returned, this time as follows:

$$P^* = [n_{62}, n_{69}, n_{76}, n_{84}]$$

This time, the student's performance score exceeds the threshold due to the additional error. The student would then be provided a hint that recommends they explore the action represented by n_{62}, which is to place an elbow pipe in their solution. And while the pipe game is a specific case of this issue, many such problems in educational serious games can be decomposed into simple sequences of actions, making the system very flexible for many games and problems.

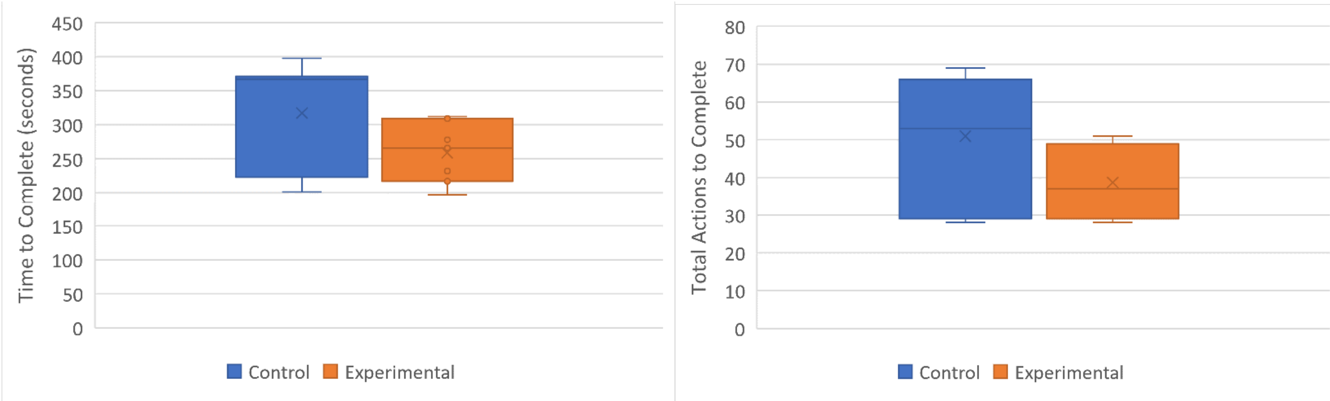

Fig. 3. Comparison between experimental control groups in (a) time taken; and (b) actions taken.

B. Comparison Results

Our comparison case study used a focus group of 14 students to compare the educational effectiveness of our serious game. The 14 students were randomly split between Game A, the control group without adaptive guidance, and Game B, the experimental group with adaptive guidance. All 14 students that participated in the case study had similar educational standing in terms of GPA. As the students played through *Algae City's* water purification module, we recorded the time taken and number of actions required for them to complete the pipe puzzle game. Recorded values are shown in Table 2.

From Table 2, it can be observed that the experimental group using Game B showed overall faster and more efficient completion of the pipe game. On average, students who used Game B completed the game in 28 actions, compared to 51 actions from students using Game A. Additionally, students in the experimental group completed the game in 259 seconds, on average, compared to 317 seconds for the control group. Fig. 3 visualizes this data to further show the impact on completion time and actions taken.

The proposed system provides automated educational hints and guidance to students as they play. The reduction in time and actions needed for players to complete the game demonstrates positive initial results for the system's ability to impact players' educational experiences.

TABLE II. DATA COLLECTED FROM STUDENTS IN THE CONTROL (GAME A) AND EXPERIMENTAL (GAME B) GROUPS

Game A (Control)			Game B (Experimental)		
Player	**Time (s)**	**Movements**	**Time (s)**	**Movements**	**# Hints**
1	371	66	266	49	2
2	398	69	312	51	3
3	201	28	197	29	1
4	289	49	232	33	2
5	371	63	309	44	3
6	367	53	278	37	2
7	223	29	217	28	1
Avg	*317*	*51*	*259*	*28*	*1*

The graph-based approach can be applied to any serious game provided that the developers are able to create an appropriate action graph. When varying the learning context of serious games, the educational performance is ultimately up to the design of hints and the design of the game itself. The proposed system is designed to augment a player's experience with personalized guidance toward a solution. The actual form of that guidance and the problem presented in the game would vary the performance of the system. However, given an appropriate action graph, the system is always able to provide the player with hints toward a solution, and as such, it is expected that the player would always reach a solution faster than in an unguided system.

The chosen learner model, as well, could impact the system's educational effectiveness. Our performance scale model accounted for several aspects of a player's performance (such as time taken and number of errors made) to determine if the player needed assistance. Depending on the context of other serious games, other variables may need to be accounted for to achieve comparable educational effectiveness. In general, it is key to provide the system with a solid metric of student performance to make informed decisions on when to provide assistance and when to allow the player to self-learn. Given such a metric, the system should then always be able to leverage the action graph at appropriate times to guide a player toward a correct solution.

IV. Conclusion

This paper proposes a personalized instruction system to implement adaptive hints and educational support in serious games. In the proposed system, students are given individualized support based on their gameplay. Our proposed system uses a directional tree structure called an action graph to represent solutions to in-game problems as sequences of actions. Then, using our provided algorithm, a serious game can easily be integrated with personalized hints that guide students toward solutions when they make errors.

Survey results and comparison testing from our focus group shows that the game with the personalized instruction system proves more effective in terms of both time and effort needed for students to complete segments of the game. Furthermore, student surveys from our case study indicate positive overall reception of the adaptive assistance system. Future work will focus on continuing to test the system on different narrative games and gather additional real data by comparing playing games with AI-assisted segment and without AI segment.

References

[1] G. Eason, B. Noble, and I. N. Sneddon, "On certain integrals of Lipschitz-Hankel type involving products of Bessel functions," Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955. *(references)*

[2] J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[3] I. S. Jacobs and C. P. Bean, "Fine particles, thin films and exchange anisotropy," in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.

[4] K. Elissa, “Title of paper if known,” unpublished.
[5] R. Nicole, “Title of paper with only first word capitalized,” J. Name Stand. Abbrev., in press.
[6] Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, “Electron spectroscopy studies on magneto-optical media and plastic substrate interface,” IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].
[7] M. Young, The Technical Writer’s Handbook. Mill Valley, CA: University Science, 1989.
[8] S. Papadimitriou, K. Chrysafiadi, and M. Virvou, 'Evaluating the use of fuzzy logic in an educational game for offering adaptation', CITS, 2019.
[9] A. M. Hussaan, K. Sehaba, and A. Mille, 'Tailoring Serious Games with Adaptive Pedagogical Scenarios: A Serious Game for Persons with Cognitive Disabilities', IEEE 11th International Conference on Advanced Learning Technologies, 2011.
[10] T. Takahashi, K. Tanaka, and N. Oka, “Adaptive mixed-initiative dialog motivates a game player to talk with an NPC,” Proceedings of the 6th International Conference on Human-Agent Interaction, 2018.
[11] M. C. Gombolay, R. E. Jensen, and S.-H. Son, “Machine learning techniques for analyzing training behavior in serious gaming,” IEEE Transactions on Games, vol. 11, no. 2, pp. 109–120, 2019.
[12] K. Mitsis, E. Kalafatis, K. Zarkogianni, G. Mourkousis, and K. S. Nikita, 'Procedural content generation based on a genetic algorithm in a serious game for obstructive sleep apnea', IEEE Conference on Games, 2020.
[13] C. S. González-González, P. A. Toledo-Delgado, V. Muñoz-Cruz, and P. V. Torres-Carrion, "Serious games for rehabilitation: Gestural interaction in personalized gamified exercises through a recommender system," Journal of Biomedical Informatics, vol. 97, p. 103266, 2019.
[14] Y. Tang, K. Jahan, K. Trinh, G. Gizzi, and N. Lamb, “Board 50: Algae City - an interactive serious game,” 2018 ASEE Annual Conference and Exposition Proceedings, Jun. 2018.
[15] J. Fletcher, “Adaptive instructional systems and digital tutoring,” in *International Conference on Human-Computer Interaction*. Springer, 2019, pp. 615–633.
[16] L. Morgenthaler and M. D. Barrett, “Core to the learning day: The adaptive instructional system as an integrated component of brick-and-mortar, blended, and online learning,” *Adaptive Instructional Systems. Design and Evaluation*, Cham: Springer International Publishing, 2021, pp. 370–381.
[17] T. E. Bannan B, P. H, P. R, and C. J.L, Sensor-Based Adaptive Instructional Systems in Live Simulation Training, 2020, vol. 12214.
[18] J. Liang, Y. Tang, R. Hare, B. Wu and F. -Y. Wang, "A Learning-Embedded Attributed Petri Net to Optimize Student Learning in a Serious Game," in IEEE Transactions on Computational Social Systems, DOI: 10.1109/TCSS.2021.3132355.
[19] D. Hooshyar, M. Yousefi, and H. Lim, “A systematic review of data-driven approaches in player modeling of Educational Games,” *Artificial Intelligence Review*, vol. 52, no. 3, pp. 1997–2017, 2017.
[20] H. Siy, B. Dorn, C. Engelmann, N. Grandgenett, T. Reding, J.-H. Youn, and Q. Zhu, “Sparcs: A personalized problem-based learning approach for developing successful computer science learning experiences in middle school,” 2017 IEEE International Conference on Electro Information Technology (EIT), 2017.
[21] A. J. Christopher Franzwa, Ying Tang and T. Bielefeldt, “Balancing fun and learning in a serious game design,” *International Journal of Game-Based Learning*, vol. 4, pp. 37–57, 2014.