Embedding a Problem Graph into Serious Games for Efficient Traversal Through Game Space

Vidya Bommanapally
Computer Science Department
University of Nebraska Omaha

Mahadevan Subramaniam
Computer Science Department
University of Nebraska Omaha

Abhishek Parakh
Computer Science Department
Kennesaw State University

***Abstract*—Serious games have been widely used as medium of instruction in various domains. A serious game is composed of various exercises a player has to accomplish in order to achieve the final goal. Each exercise is designed using a set of concepts that a player has to achieve proficiency in. The design of problems is of greater importance in such scenarios than the physical game design. However, traversal through the game is challenging for users with no gaming experience. Spatial distribution of these exercises in the game may influence the learning potential of a player, especially when the exercises have a conceptual dependency associated with them. Our approach in this project is to map the problem graph on to a selected environment in the game for efficient traversal through the game space. Towards this goal, we plan to explore existing environments from Unreal Engine with possible locations for exercises. We will study the theory of graph embedding used to map multi-threaded programs to high performance architecture topologies and explore their adaptation to the problem of exercise distribution in serious game environments.**

***Index Terms*—Graph embedding, serious game design, quantum computing**

I. Introduction

Serious games have become widely used medium of instruction, especially, for online only instruction where students can be more actively engaged in the gaming environment while learning the content. A serious game typically contains targets or exercises that a player has to complete to achieve the learning goal of the level in the game. For this purpose, a set of concepts can be distributed in the game environment that a player has to solve to move further in the game. In most cases, this distribution of concepts and exercises can be random when the concepts are independent of each other. When there is a dependency involved in the concepts that a player has to learn, then a controlled distribution of the exercises helps a player to traverse through the game space to achieve the desired proficiency efficiently. In this project we address this challenge by embedding a problem graph of a serious game to hypercube to preserve concept adjacency and dependencies in a gaming environment.

A hypercube is a n-dimensional structure initially introduced for parallel processing. A hypercube of n-dimension has 2^n nodes as vertices of the cube. Each node of the hypercube is a binary number and the distance between two adjacent nodes is a hamming distance of one bit. Hypercube graphs are used to perform parallel processing by placing multiple processors at each node of the hypercube that performs a subset of the task. Hypercubes are used for parallel processing of sort, and search operations on large data structures quickly. Further, there are studies on embedding graphs onto hypercubes such as a complete binary tree on to a hypercube [13].

In this work in progress, we use the bipartite nature of hypercube to distribute the problem graph of the game onto the game environment. We have used the serious game QuaSim [26] developed for teaching quantum cryptography concepts to students.

II. Background

Serious games have been widely studied in education, industry, rehabilitation, medical illness improvements and many such areas with promising results [14], [31]. Serious games in industry have become popular to train the interns and new employees about company policies and also project training [7]. Studies have been conducted to bridge the gap between teaching and learning through serious games by integrating the serious games with learning management systems. With the increase in serious games for different disciplines studies have also been conducted to standardize the learning and teaching process [11]. Serious games have been used for mental health improvement [10] and studied for different serious game frameworks that best suits the domain. Adaptive learning in serious games is a major concern as each player has different background and learning abilities [29]. Latest artificial intelligence techniques such as deep learning and reinforcement learning techniques have been used to improve performance of players in serious games. Various frameworks for designing serious games have been studied [30] along with analysis of mood and navigation in serious games [9], [16].

On the other hand, hypercubes are used for parallel processing in high performance environments [12] and for scheduling multiple jobs in parallel on each node of the n-dimensional hypercube [5], [28]. Trees such as binary trees can be embedded on the hypercubes for efficient processing [1]. Hypercubes are a type of bipartite graphs as they can only be two colored [6]. The distance between two adjacent nodes in a hypercube is one hamming distance and a hypercube can be defined as union of multiple hypercube subgraphs [8].

This work is funded in part by a Nebraska Collaborative Initiative grant: "Quark: An Intelligent Adaptive Education Platform for Quantum Cybersecurity". Correspondence may be sent to Abhishek Parakh at aparakh@kennesaw.edu.

2023 IEEE Frontiers in Education Conference (FIE) | 979-8-3503-3642-9/23/$31.00 ©2023 IEEE | DOI: 10.1109/FIE58773.2023.10342977

III. Approach

This section provides the strategy used for embedding a problem graph of a serious game onto a hypercube.

A. Hypercubes as Bipartite Graphs

A hypercube is an n-dimensional cube with 2^n nodes. Each node is arranged in the hypercube such that the hamming distance between two adjacent binary number is one. A 2-dimensional hypercube is a square with 4 nodes or vertices, a 3-dimensional hypercube has 8 nodes, and a 4-dimensional hypercube is called a ***tesseract*** and has 16 nodes. Figure 1 pictorially shows the construction of higher dimensional hypercubes from the lower dimensional hypercubes. Each hypercube can be represented using a bipartite graph. The two sets of the nodes in the graph are a partition of the nodes of the hypercube. There is an edge from node u in one set in the graph to node v in another set if nodes u and v are adjacent in corresponding hypercube, i.e., there is an edge connecting them in the hypercube. A 3D hypercube and its corresponding bipartite graph are shown in 2. Each node in an n-dimensional hypercube has n-adjacent nodes. It can be observed that a n-dimensional hypercube has a $n-1$-dimensional hypercube as its subgraph. Hence by repeated decomposition it follows that an n-dimensional hypercube is a combination of bipartite graphs corresponding to its 3D hypercube components. A 4-dimensional hypercube ***tesseract*** and its corresponding bipartite graph combination is shown in figure 3. The nodes of an n-dimensional hypercube are usually indexed by a gray code which is n-bits wide and this results in adjacent nodes of the hypercube to have a hamming distance of 1. As an example, the tesseract in Figure 3 is indexed by 4-bit gray code with adjacent nodes having a hamming distance of 1. The combination of the bipartite graphs corresponding to this hypercube is depicted on the right of Figure 3. The combination of the bipartite graphs consists of 4 sets of 4 nodes each. Three game exercises are assigned to the nodes in each set (one of these nodes is unassigned) are depicted in the figure. These sets correspond to 4 levels in the game ***LA, LB, LC, LD***. The rules used to assign these game exercises to the sets (or game levels) in the bipartite graph combination are discussed later in Section 4.

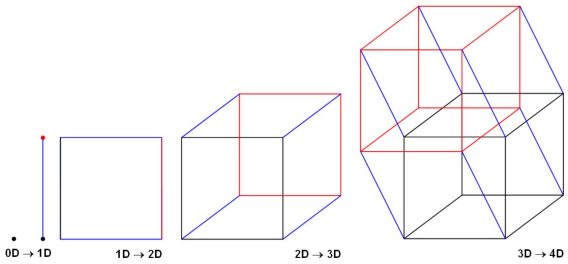

Fig. 1. Hamming Distance

B. Hypercube Mapping to the Problem Graph

An exercise in game space is a subset of two or more concepts that are needed to complete that game level. A

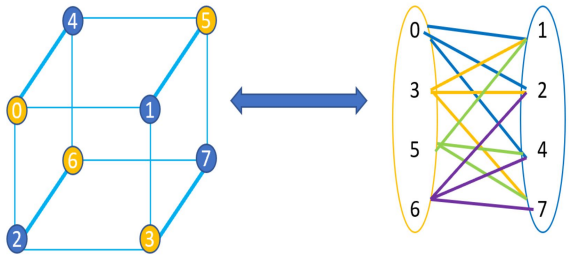

Fig. 2. Example of 3D hypercube to bipartite graph.

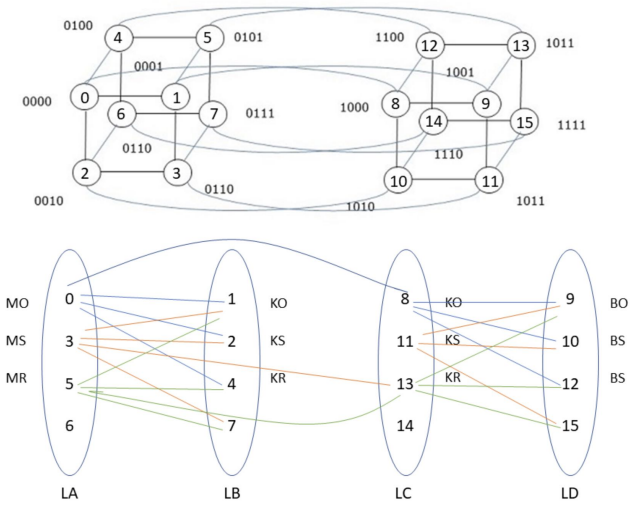

Fig. 3. A 4-dimensional hypercube and corresponding bipartite graph with mapping of the exercises to nodes.

problem graph represents the exercise graph of the game space showing the distribution of concepts and their dependencies in the exercises. Each node in the problem graph is an exercise and a directed edge from one exercise to the other represents dependency between the two exercises in terms of difficulty level or concept dependency meaning concepts in the prior exercise are mandatory to learn the concepts in the later exercise. These concepts can include independent concepts as well as multiple independent trees of concept dependencies. A problem graph is built using the concept dependency graph.

The first factor to consider to map a hypercube to the problem graph is to chose the dimension of the hypercube. We have considered the number of targets or exercises in the game space to chose the hypercube dimension. Since the total number of nodes in the hypercube is a power of 2, the dimension we chose is the nearest power of two as the number of exercises in the game space. For example, if game space has 27 possible exercises, the hypercube dimension to map with the game space is 5 with 32 nodes. The remaining nodes can either be left empty or can have new exercises with a different combination of concepts. The formula to calculate the dimension of hypercube is given by equation 1, where N is the number of exercises in the game space.

$$dim = ceil(log_2(N)) \tag{1}$$

Once the dimension of the hypercube is chosen, the bipartite graph for each 3D hypercube component of the chosen hypercube is generated with adjacent edges to each node. The nodes in a set are *not adjacent* to each other. The adjacency property determines the mapping of exercises to nodes: assign exercises having same combination of concepts or independent concepts and to the same set in the bipartite graph. As mentioned earlier, n-dimensional hypercube consists of subgraphs of $n-1$ dimension. The second set in the pair of bipartite graph are assigned with exercises using the adjacency rule. A connection exists from one node in a set to another node in another set if there is a dependency relation between the nodes of the sets. Usually, after assigning exercises to first set, the exercises assigned to the second set involve concepts that are deeper (in comparison to those in the first set) in the concept dependency graph. Also, the edge from a node in a bipartite subgraph to a node in another bipartite subgraph with a higher most significant bit represents that the exercises in the higher order are involving concepts that are at a deeper level in the concept graph. This kind of mapping helps players to traverse to higher level concepts instead of solving the exercises with concepts that the player has encountered earlier and become proficient in.

As each node in the hypercube has n degrees of freedom, once a player successfully solves an exercise (node) in the graph, there are n choices from which a player can choose to proceed further in the game. Once a player solves an exercise (node) successfully or fails at a node, the edge connecting the current and previous nodes is broken to indicate that the exercises are not available for the player anymore. Each node and their adjacent nodes list are maintained in a database. In case of a failure at a node, the player still has adjacent nodes for the current node that directs them to a lower level concept or a higher level concept. This choice of the next best exercise based on the current performance is done by a dependency algorithm. This next best exercise can be chosen based on an exercise dependency graph and the weighting schema calculated based on the algorithm discussed in the next section.

As the player solves nodes (exercises), the edges in the hypercube get activated showing the navigation points for the player to proceed further in the game. These points are associated with rewards that manifest in the form of game elements such as stair cases, a normal road, an elevator or teleportation.

C. Next Target Algorithm: Preliminary Approach

In order to choose the next target for the player from among the adjacent nodes or the available nodes in the graph that can be reached through teleportation, vertices in the problem dependency graph as shown in figure 4 are assigned with weights. Value of each node is given by,

$$v = \sum \langle 1/|E|, E \in C \rangle \tag{2}$$

where $|E|$ is the number of exercises that each concept in current node is present in and C is the set of concepts

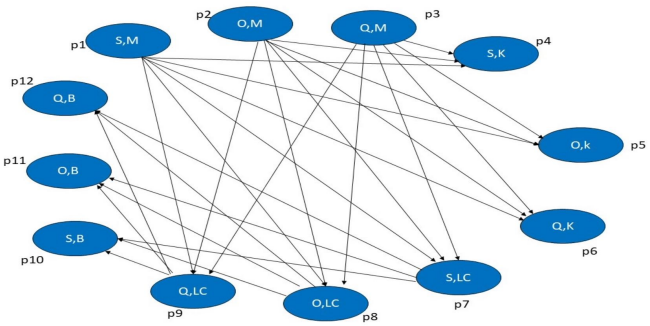

Fig. 4. Distribution of concepts to exercises and the associated problem graph of the polarization level in the QuaSim game. Here Q stands for Opposite Quadrant, S stands for Same Angle, and O stands for Orthogonal Angle of Qubit polarization, B stands for Basis, K stands for Ket representation, and LC stands for Linear Combination.

addressed at the current node or exercise. Each value of the nodes adjacent to the current node are calculated and maximum value is chosen. Following this, a topological sort of the exercises is performed to preserve the dependency between the exercises. The node with the highest value after topological sort is chosen as the next target for the player. The value of a concept in a node is assigned with a zero if the concept is learnt in any of the exercises, else the value is set to 1. A success tuple is maintained that keeps track of the learning progress. The tuple is a Boolean tuple where each component represents the concepts the player should achieve proficiency in by the end of the game. Once all the values in the success tuple are set to 1, the game exits as the player has successfully reached the goal of that level.

IV. Example Evaluation

For preliminary evaluation purposes we have used the game QuaSim [2], [17], [21], [24]–[27], a serious game developed for teaching quantum cryptography and quantum computing. These lay the foundations for future quantum internet [3], [4], [15], [18]–[20], [22], [23]. QuaSim is built using Unreal game engine. The game comprises of five levels each with a learning goal. Here, as a proof-of-concept, we have considered level 1 of the game called Polarization. Figure 5 shows an exercise in the game from polarization level. The game teaches the concept of polarization through set of 12 exercises distributed over a city building in the game. The exercises are built using combination of seven different concepts for representation of qubits such as matrix representation (M), ket representation (K), linear combination (LC), basis (B) over three orientations of qubit such as same angle (S), orthogonal (O), and opposite quadrant (Q). The concept dependency graph of these concepts is shown in the figure 6. The graph shows that in order to learn the concept LC it is necessary to gain efficiency in concept M.

The game QuaSim comprises of 12 exercises to be distributed in the gamespace. Hence, the dimension of hypercube as calculated by equation 1 is 4, which is a 4D hypercube,

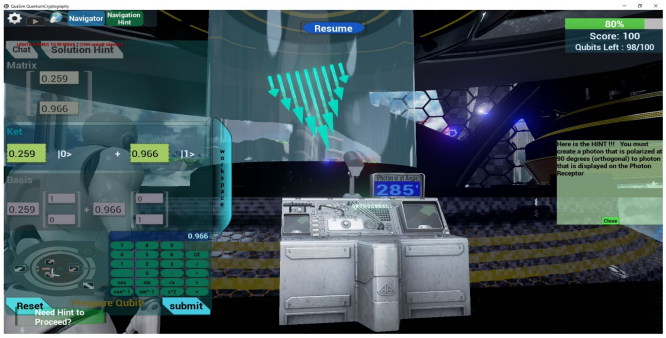

Fig. 5. Sample exercise in polarization level in QuaSim game teaching qubit representations.

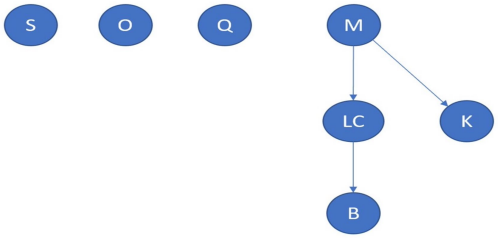

Fig. 6. Concept dependency graph of level 1 exercises in QuaSim.

tesseract as shown in figure 3. The corresponding bipartite graph is shown in the figure 2. Following, the bipartite graph of the 4D hypercube is generated and mapped with the exercises is shown in figure 3. The distribution of concepts over the exercises and the corresponding dependency graph based on the concept dependency graph is shown in figure 4.

Once the player has successfully solved the concepts in the node with index 0, the player has 4 possible nodes (indices 1, 2, 4, and 8 at hamming distance 1) that they can travel to. These nodes are further chosen using the algorithm mentioned above. In case of failure at, say node with index 1, player has outgoing edges from this node, which are already stored into the database. Of the available nodes, the algorithm is used to choose one for the player. In such a case, teleportation can be used to move player automatically to a different node. This helps the player to learn the concepts in an adaptive environment while also providing new concepts through the bipartite mapping maintaining player engagement in the game.

Embedding the problem graph into a hypercube through a bipartite graph enables the game developer to position the game exercise in a systematic way to help the player navigate through the game easily especially for the player with beginner level of gaming experience. Player can spend significant amount of time in learning the concepts rather than exploring the game environment for the exercises.

The game QuaSim comprises of a 4 level building where each level is connected to the other using staircases and an elevator. As per the bipartite graph distribution each level of the building can be distributed with the exercises such that set corresponding to level ***LA*** corresponds to the first level of the building, set corresponding to level ***LB*** and ***LC*** could be present in the same level of the building according to the

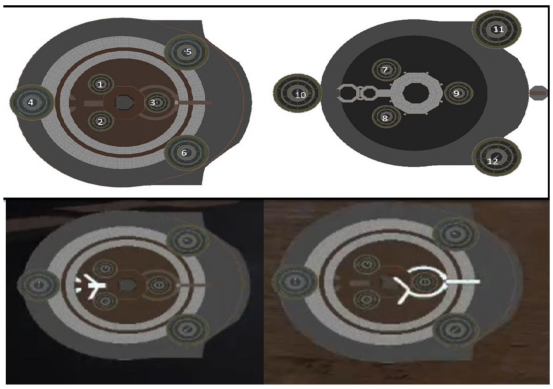

Fig. 7. QuaSim game Level 1 polarization map in top view.Top row shows distribution of exercises in the building, bot-tom row shows navigation points (staircases).

ordering of bipartite graph. In order to reach set corresponding to ***LD*** the player has to go through ***LC*** that is at least one-hop in the hypercube in order to reach ***LD***. Figure 7 shows the map layout of the building in level 1 of the QuaSim game environment in the top view. The two figures on the top row show the exercises distributed in each level of the building where the indices 1, 2, 3 on the map layout shown in the figure correspond to nodes 0, 3, 5 in ***LA*** of bipartite graph. Each inner and outer ring of the map corresponds to each set in the bipartite graph. The two figures on the bottom row in Figure 7 represent the navigation points, here staircase, activated that represent the edge in the hypercube to navigate between the nodes in the graph.

V. Conclusion and Future Work

Hypercubes are widely known to be used in for parallel processing in a high performance environment. Single or multi player games requiring high performance environments can use hypercube graph embedding to improve the game performance. This study is a novel approach that helps game developers embed the problem graph into hypercube for high performance and also systematic distribution of targets in the serious game. In this work we have developed a strategy to map the problem graph on to a serious game using bipartite characteristic of the hypercube. We developed an algorithm to choose the next target that helps players to improve their performance adaptively.

The future goals of the research include being able to embed problem graphs onto various unreal environments and study the player performance for different problem graphs and different environments. One of the limitations of the approach is selecting the dimension of the hypercube as the nodes are increasing exponentially. Choosing a dimension when the number of exercises are say 35, will result in a dimension of 6 which is 64 node, higher than the required nodes. Moreover, here we have assumed that concepts are distributed in the exercises following the dependency graph. Further research direction of this approach would also be to identify a different graph or a mesh network that can aptly embed the problem graph.

REFERENCES

[1] Bhatt, S.N., Chung, F.R.K., Leighton, F.T., Rosenberg, A.L.: Efficient embeddings of trees in hypercubes. SIAM Journal on Computing **21**(1), 151–162 (1992)

[2] Bommanapally, V., Subramaniam, M., Parakh, A., Chundi, P., Puppala, V.M.: Learning objects based adaptive textbooks with dynamic traversal for quantum cryptography. In: iTextbooks@AIED (2020), https://api.semanticscholar.org/CorpusID:221492774

[3] Burr, J., Parakh, A., Subramaniam, M.: Evaluating different topologies for multi-photon quantum key distribution. In: Donkor, E., Hayduk, M., Frey, M.R., Jr., S.J.L., Myers, J.M. (eds.) Quantum Information Science, Sensing, and Computation XIV. vol. 12093, p. 1209309. International Society for Optics and Photonics, SPIE (2022). https://doi.org/10.1117/12.2620057, https://doi.org/10.1117/12.2620057

[4] Burr, J., Parakh, A., Subramaniam, M.: Quantum internet. Ubiquity **2022**(August) (aug 2022). https://doi.org/10.1145/3547493, https://doi.org/10.1145/3547493

[5] Błażewicz, J., Drozdowski, M.: Scheduling divisible jobs on hypercubes†. Parallel Computing **21**(12), 1945–1956 (1995)

[6] Chang, C.H., Lin, C.K., Huang, H.M., Hsu, L.H.: The super laceability of the hypercubes. Information Processing Letters **92**(1), 15–21 (2004)

[7] De Freitas, S., Jarvis, S.: Serious games—engaging training solutions: A research and development project for supporting training needs. British Journal of Educational Technology **38**(3), 523–525 (2007)

[8] Djoković, D.: Distance-preserving subgraphs of hypercubes. Journal of Combinatorial Theory, Series B **14**(3), 263–267 (1973)

[9] Ferguson, C., van Oostendorp, H., Giezeman, G., de Redelijkheid, S., van den Broek, E., Grant, R., Allen, T., Spink, A., Sullivan, M.: Measuring navigation performance in serious games. Proceedings of Measuring Behavior pp. 274–277 (2018)

[10] Fleming, T.M., Bavin, L., Stasiak, K., Hermansson-Webb, E., Merry, S.N., Cheek, C., Lucassen, M., Lau, H.M., Pollmuller, B., Hetrick, S.: Serious games and gamification for mental health: Current status and promising directions. Front. Psychiatry **7**, 215 (2016)

[11] Hamdaoui, N., Idrissi, M.K., Bennani, S.: Serious games in education towards the standardization of the teaching-learning process. pp. 174–181 (2014)

[12] Harary, F., Hayes, J.P., Wu, H.J.: A survey of the theory of hypercube graphs. Computers & Mathematics with Applications **15**(4), 277–289 (1988)

[13] Kapur, D., Subramaniam, M.: Automated reasoning about parallel algorithms using powerlists. In: Alagar, V.S., Nivat, M. (eds.) Algebraic Methodology and Software Technology. pp. 416–430. Springer Berlin Heidelberg, Berlin, Heidelberg (1995)

[14] Michael, D.R., Chen, S.: Serious games: Games that educate, train, and inform (2005)

[15] Mishra, S., Thapliyal, K., Parakh, A., Pathak, A.: Quantum anonymous veto: a set of new protocols. EPJ Quantum Technology **9**(1), 14 (May 2022). https://doi.org/10.1140/epjqt/s40507-022-00133-2, https://doi.org/10.1140/epjqt/s40507-022-00133-2

[16] M.Nazry, N., Romano, D.M.: Mood and learning in navigation-based serious games. Comput. Hum. Behav. **73**(C), 596–604 (aug 2017)

[17] Parakh, A., Chundi, P., Subramaniam, M.: An approach towards designing problem networks in serious games. In: 2019 IEEE Conference on Games (CoG). p. 1–8. IEEE Press (2019). https://doi.org/10.1109/CIG.2019.8848055, https://doi.org/10.1109/CIG.2019.8848055

[18] Parakh, A.: Providing variable levels of security in quantum cryptography. In: Meyers, R.E., Shih, Y., Deacon, K.S. (eds.) Quantum Communications and Quantum Imaging XVI. vol. 10771, p. 107710R. International Society for Optics and Photonics, SPIE (2018). https://doi.org/10.1117/12.2323204, https://doi.org/10.1117/12.2323204

[19] Parakh, A.: Using fewer qubits to correct errors in the three-stage QKD protocol. In: Gruneisen, M.T., Dusek, M., Rarity, J.G. (eds.) Quantum Information Science and Technology IV. vol. 10803, p. 108030A. International Society for Optics and Photonics, SPIE (2018). https://doi.org/10.1117/12.2325891, https://doi.org/10.1117/12.2325891

[20] Parakh, A.: Quantum teleportation with one classical bit. Scientific Reports **12**(1), 3392 (Mar 2022). https://doi.org/10.1038/s41598-022-06853-w, https://doi.org/10.1038/s41598-022-06853-w

[21] Parakh, A., Bommanapally, V., Chundi, P., Subramaniam, M.: (Dec 2020), https://cisse.info/journal/index.php/cisse/article/view/128

[22] Parakh, A., Subramaniam, M.: Bootstrapped QKD: improving key rate and multi-photon resistance. In: Gruneisen, M.T., Dusek, M., Rarity, J.G. (eds.) Quantum Information Science and Technology IV. vol. 10803, p. 1080308. International Society for Optics and Photonics, SPIE (2018). https://doi.org/10.1117/12.2500438, https://doi.org/10.1117/12.2500438

[23] Parakh, A., Subramaniam, M.: Network routing protocols for multi-photon quantum cryptography. In: Deacon, K.S., Meyers, R.E. (eds.) Quantum Communications and Quantum Imaging XIX. vol. 11835, p. 118350L. International Society for Optics and Photonics, SPIE (2021). https://doi.org/10.1117/12.2594891, https://doi.org/10.1117/12.2594891

[24] Parakh, A., Subramaniam, M.: (Aug 2022), https://doi.org/10.53735/cisse.v9i1.142

[25] Parakh, A., Subramaniam, M., Chundi, P.: A framework for incorporating serious games into learning object repositories through experiential learning. In: 55th Hawaii International Conference on System Sciences, HICSS 2022, Virtual Event / Maui, Hawaii, USA, January 4-7, 2022. pp. 1–11. ScholarSpace (2022), http://hdl.handle.net/10125/79978

[26] Parakh, A., Subramaniam, M., Chundi, P., Ostler, E.: A novel approach for embedding and traversing problems in serious games. p. 229–235. SIGITE '20, Association for Computing Machinery, New York, NY, USA (2020)

[27] Parakh, A., Subramaniam, M., Chundi, P., Ostler, E.: A novel approach for embedding and traversing problems in serious games. In: Proceedings of the 21st Annual Conference on Information Technology Education. p. 229–235. SIGITE '20, Association for Computing Machinery, New York, NY, USA (2020). https://doi.org/10.1145/3368308.3415417, https://doi.org/10.1145/3368308.3415417

[28] Saad, Y., Schultz, M.: Topological properties of hypercubes. IEEE Transactions on Computers **37**(7), 867–872 (1988)

[29] Streicher, A., Smeddinck, J.D.: Personalized and Adaptive Serious Games, pp. 332–377. Springer International Publishing, Cham (2016)

[30] Tsikinas, S., Xinogalos, S.: Designing effective serious games for people with intellectual disabilities. In: 2018 IEEE Global Engineering Education Conference (EDUCON). pp. 1896–1903 (2018)

[31] Zhonggen, Y.: A Meta-Analysis of use of serious games in education over a decade. International Journal of Computer Games Technology **2019**, 4797032 (Feb 2019)