



HackR

0	1	0	0	1	0
---	---	---	---	---	---

Hack Research 2018




RGV

Proceedings of the 1st Hack Research Hackathon
University of Texas - Rio Grande Valley

HackR 2018

Tim Wylie, Editor



Copyright © 2018 for the individual papers by the papers' authors. Copying permitted only for private and academic purposes. This volume is published and copyrighted by its editors.

Preface

Hack Research is largely an experiment in trying to address the shortcomings of most hackathons while also trying to get students more involved in research. Thus, rather than software skills, the competition focuses on algorithmic and theoretical skill development. The majority of the problems are actual open problems of interest in several areas of Computer Science.

With any new event, there is a concern that no one will participate. When the event is pushing research and critical thinking, the worry is justified that students may not be interested in devoting their weekend to a lot of work in areas that are unfamiliar, and that require skills beyond their current education.

Our fears turned out to be unfounded as students showed both the willingness and tenacity to push themselves and get involved in several areas of research. After the event, some students are continuing to pursue projects they began or to get involved with the professors whose problems they were excited by. This is the best outcome we could hope for: that collaborations might be established that provide students opportunities for growth and development.

We have received a lot of feedback and learned a great deal in order to improve for next year, and we are grateful the students were understanding of our inexperience and gave us constructive input. With some refinements to the format and timing, we can greatly improve the student participation and experience. Our continuing focus will be on involving students in research and giving them opportunities to succeed.

The success of the event was due to the contributions of many people, and I have tried my best not to miss anyone in the acknowledgements section. I do want to thank Robert Schweller here for his help in organizing and directing the event. Overall, *HackR* was more successful than expected and we are all looking forward to 2019.

1-2 December, 2018
Edinburg, TX, United States

Tim Wylie, Editor

Acknowledgements

This event would not have been possible without the help of many people of whom we are extremely grateful to know. Our families were supportive and sacrificed a great deal for us to pursue these outreach activities, and we are thankful for them and their patience. We owe a lot to Lisa Moreno and Emmett Tomai, whose administrative help made all the tedious aspects of the process simple on our end.

Several people contributed a lot of time and effort whom I must mention. The entire ASARG group posed problems and helped during parts of the event. Austin and Samantha Luchsinger were instrumental to the success of the event. Ari Gutierrez helped, posed problems, and supplied games. Jose Balanza-Martinez helped get the food orders. All the faculty that contributed problems also came and stayed most of the time to help out. This list consists of Marzieh Ayati, Emmett Tomai, and Jingru Zhang. Dongchul Kim and Hongkai Yu also came to help setup, clean, and judge.

We are thankful for our sponsors who gave us the support we needed to make the event a reality. The Computer Science department at UTRGV donated several prizes and handouts, as well as the support of faculty and staff. Hiten Patel, as the CEO of Nerdvana, was an invaluable partner. Not only was he willing to sponsor and host the event, but he also contributed experience and ideas in scheduling and running the event. Finally, we are thankful for the support from the National Science Foundation in our efforts to explore theoretical Computer Science and to increase undergraduate research participation.

Robert Schweller, Tim Wylie

Sponsors



<http://www.utrgv.edu/csci>



<http://www.nerdvana.io>



**The National
Science Foundation**

<http://www.nsf.gov>

Organization

HackR 2018 was organized by the Algorithmic Self-Assembly Research Group (ASARG) with the help of the Department of Computer Science at the University of Texas Rio Grande Valley. The event was graciously hosted by Nerdvana in Edinburg, TX. Further, the event was supported in part by National Science Foundation Grant CCF-1817602.

Directors

Robert Schweller	robert.schweller@utrgv.edu	University of Texas Rio Grande Valley
Tim Wylie	timothy.wylie@utrgv.edu	University of Texas Rio Grande Valley

Program Committee

Marzieh Ayati	marzieh.ayati@utrgv.edu	University of Texas Rio Grande Valley
Dongchul Kim	dongchul.kim@utrgv.edu	University of Texas Rio Grande Valley
Emmett Tomai	emmett.tomai@utrgv.edu	University of Texas Rio Grande Valley
Hongkai Yu	hongkai.yu@utrgv.edu	University of Texas Rio Grande Valley
Jingru Zhang	jingru.zhang@utrgv.edu	University of Texas Rio Grande Valley

Volunteers

Samantha Luchsinger
Hiten Patel

ASARG Members

Jose Balanza-Martinez
David Caballero
Angel Cantu
David Dittman
Mauricio Flores
Tim Gomez
Bryan Guerra
Ari Gutierrez
Austin Luchsinger

Results

Hack Research 2018 was experimental for many reasons, and thus we restricted registration to keep it small. The event had 44 students participate in the 24-hour event. Of those students, a remarkable 23 students submitted a paper to compete for the prizes. In total, there were ten submissions. Another couple of teams decided not to submit despite being close to finishing. Five professors contributed problems, and seven faculty attended some portion with several staying the majority of the time in order to encourage and assist students.

Winners

There were about \$500 worth of prizes to distribute to the top teams. Due to varying team size and the quality of the papers, five teams were awarded prizes.

1. First Place
 - One-player Constraint Inequality is Easy*
 - An Investigation of the Existence of a Perfect Play in a Four Player 9×9 Grid Pentago Game
2. Second Place
 - Topic: Bioinformatics and Computational Biology
3. Third Place
 - Ultimate Tic-Tac-Toe
 - Celtic! Knot so Easy

**Graduate Team*

Table of Contents

Ultimate Tic-Tac-Toe	1
Jose Balanza, David Caballero, Angel Cantu, Mauricio Flores, Alvaro Rios . . .	
Celtic! Knot so Easy	3
Eden Canales, Robert Michael Alaniz, Luis Romo, Richard Marvin	
Topic: Bioinformatics and Computational Biology	5
Taranau Mou, Paulo Uchoa, Luis Romo	
An Investigation of the Existence of a Perfect Play in a Four Player 9×9 Grid Pentago Game	6
Domingo Martinez Jr., Jose Oscar Mendoza, Arianna Arriaga, Maria Romero .	
One-player Constraint Inequality is Easy	7
Cameron Chalk, Boya Wang	
Adaptive Design for Adaptive Learning	9
Mario Salinas	
Network Concurrency Over Online Video Games	11
Antonio Adame	
Computational Biology	13
Daniel Acevedo	
Bioinformatics and Computational Biology	14
Austin Karingada, Odette Marin	
A Special Case for the K-Center Problem on Uncertain Data	NI
Daniel Ruiz	

Some submissions are not included (NI) due to ongoing research on the topic.

Ultimate Tic-Tac-Toe *

Jose Balanza David Caballero Angel Cantu Mauricio Flores Alvaro Rios

1 Introduction

We investigate the complexity of a variation of tic-tac-toe in which several concurrent sub-games are played, and the outcomes of the sub-games play a larger game of tic-tac-toe. For the generalized version ($n \times n$), we show that it is NP-Hard to determine if player 1 can force a win in a given instance of a game. This game has much different rules than standard tic-tac-toe. In a 3x3 game, player 1 chooses the initial play location. If a player plays in section x of section y , then the other players next move must be in section y . See Figure 1a. When a section is won by a player by placing 3 of their shape in a row, then that section is considered won by that player. see Figure 1b. When 3 sections are won by a player in a row, then that player wins the game. see Figure 3a.

2 Preliminaries

Game A game is a $n^2 \times n^2$ grid split into n^2 $n \times n$ sections. Two players take turns placing X's and O's according to the position the last player played.

Main Board The main board is the larger $n \times n$ board in which a symbol is placed by winning a section. Winning the main board wins the game

Subboard A subboard is one of the n^2 smaller boards contained in the mainboard. The subboard is where players place their symbols.

Instance An instance of a game is considered as the $n^2 \times n^2$ grid, the symbols that have been played, and the last position played by player 2 (O).

Without loss of generality we will consider player 1 to be X, and player 2 to be O. X will have the first move of the game.

3 Complexity

We will now look at the complexity of the problem, given an instance of a game, can X force a win. We will show that this problem is NP-Hard by a reduction from a version of the HAMPATH problem in which the starting vertex is specified. We will create an instance

of Ultimate Tic-Tac-Toe in which X can force a win if and only if there exists a Hamiltonian Path within the given graph.

Given a graph with n vertices, we will consider an $10n \times 10n$ board of Ultimate Tic-Tac-Toe. We will first consider the idea of being able to control the locations that moves will be played with the starting configuration of the board. Consider the current play section of the board having only one spot playable. See Figure 4a. The next play section is determined by the location of the empty spot in that section. If we change the location this empty spot in our starting configuration, then we can also change the section that the 2nd move will be played in. We will say that subboard A points to B if there is one play location in A , and playing that location makes the next move play in B . We can combine these in order to create a determined path of where all moves will be made. Given a HAMPATH problem we will start the construction of our board by having a subboard for each vertex in a line such that the only option left for X to win is to win each of these subboards. See Figure 6a. These vertex subboards will be "connected" by encoding the locations of all of this vertex's neighbors in the playable spots in our subboard, i.e. a vertex with 3 neighbors will have a corresponding vertex subboard with 3 empty spots. If one of these spots is chosen a following move will have to be played in one of its neighbors. Every vertex subboard will have a corresponding *buffer* gadget. This is a gadget composed of three subboards that prevents any vertex subboard from being "traversed" more than once. We will represent a subboard "pointing" to another subboard with an arrow. A subboard that has more than 1 empty spot can point to more than one subboard. This will be represented with multiple arrows. Every vertex will have buffer, made of 3 subboards that points to it. See Figure 2. In Figure 2 Buf-A2 and Buf-A3 are simply pointer subboards that point to the vertex subboard corresponding to A. Buf-A1 has added functionality. This subboard will have 2 open spots, pointing to Buf-A2 and Buf-A3. Because of the order of our moves, it will always be O's turn on Buf-A1. Buf-A1 is created in a way that If the 2 empty spots in Buf-A1 are filled with O's then O will win the game. An edge from B to A will mean that an empty spot in B will point to Buf-A1. For every edge from X to Y, we will add an empty spot in X such that it points to the buffer of Y. See Figure 6b.

This system is restricted in such a way that it will always be X's turn when a variable subboard is in play,

*Department of Computer Science, University of Texas Rio Grande Valley

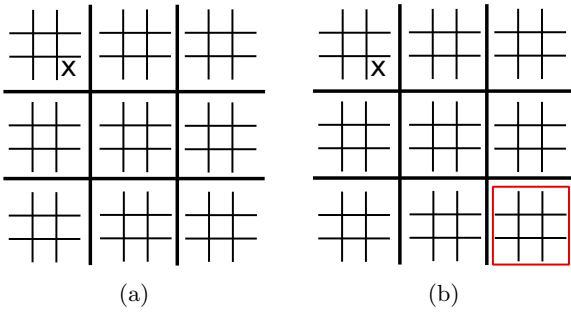


Figure 1: (a) X plays in section 9 of section 1. (b) O must now play in section 9

and it will always be O's turn when the first buffer subboard is in play. This holds our constraints of a buffer only being able to be traversed once.

Theorem 1 *Given an instance of the HAMPATH problem, we can construct an instance of Ultimate Tic-Tac-Toe problem such that X can force a win if and only if a there exists a solution to the HAMPATH problem*

Proof. We can show that X can only force if there exists a solution to the corresponding HAMPATH problem by following the flowchart under the assumption that any buffer can not be traversed more than once. A buffer being traversed more than once is equivalent to a vertex being traversed more than once. Since X needs all vertex subboard victories in order to win the game, there must exist a way to place an X in all of these subboards, but a if a vertex is travelled through more than once, than O is given a win due to the functionality of the buffer. A solution to the HAMPATH problem is the order that X would need to win the vertex subboards.

The HAMPATH problem also only has a solution if it is possible for X to force a win from this given configuration. If X can force a win, it must have placed an X in all the vertex subboard, and it must have done so without ever crossing a buffer gadget more than once. The order that X won the vertex subboard is the solution to the HAMPATH problem. □

4 Future Work

Despite the common game theory standards, we believe this problem to be in NP, we plan to show NP membership and therefore NP-Completeness.

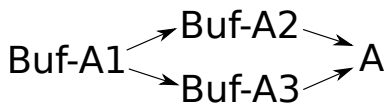


Figure 2: The 3 subboard buffer that points to variable A

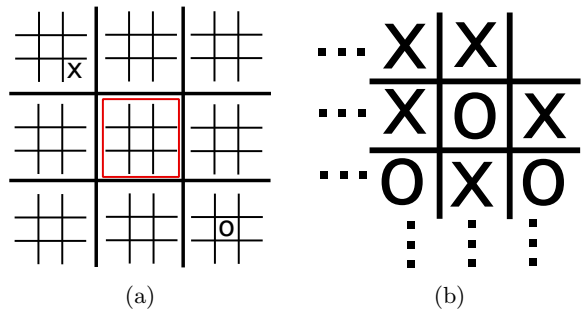


Figure 3: (a) O plays in section 5 of section 9, X must now play in section 5. (b) A subboard with one empty spot at the top right.

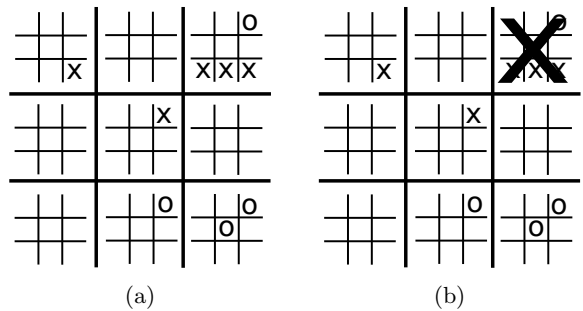


Figure 4: (a) X has 3 in a row in section 3. (b) X gets placed on the main board.

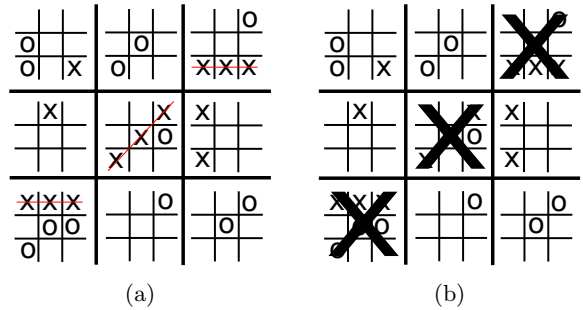


Figure 5: (a) X has 3 sections won. (b) X wins the game.

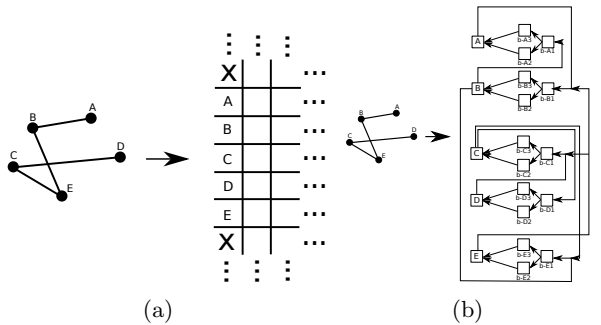


Figure 6: (a) The vertex subboards from a 5 vertex HAMPATH problem. (b) The flowchart of an example HAMPATH problem.

Celtic! Knot so Easy

Eden Canales *

Robert Michael Alaniz †

Luis Romo ‡

Richard Marvin §

Abstract

We research the properties and aspects of the 2009 game made by Cameron Browne, "Celtic!" We examine properties such as number of possible processes, optimal strategy, and aspects of the game pieces (which we will refer to as tiles) themselves. We propose using the tiles not as game pieces, but as aspects of an equation and attempt to correlate the usage of each tile by its mathematical properties to the attempts of victory.

1 Introduction

Celtic! is a game developed by Cameron Browne in which the goal of the game is to have a higher number of your color-specific tiles to be attached to a knot on a 5x5 board when there are no more moves available to be made by either player. Each player begins with 10 blue or orange tiles that connect or tie the exits between other pieces. There are 5 neutral white tiles, 1 used to begin the game, and 4 capable of being used at any turn by any player[1].

There is no previous work to look into besides the creator's original description. As our first steps to understanding Celtic! to its fullest extent, we propose to look at the game less as a strategical placement game and aim to look at the pieces for use as terms to an equation, constantly calculating the number of exits (paths) branching from our tiles and attempting to correlate the numbers to victory and optimal strategy.

We briefly highlight the definitions in Section 2 and results of our work in Section 3. We then conclude in Section 4 and point towards the general research goals for this work.

2 Preliminaries

The game, deceptively difficult, beckons that we look at the individual tiles as complicated pieces and even as different actions depending on how they're tilted and used to direct players of the game. We analyze several features of the tile pieces as well as concepts when playing the game.

*Department of Computer Science, UTRGV,

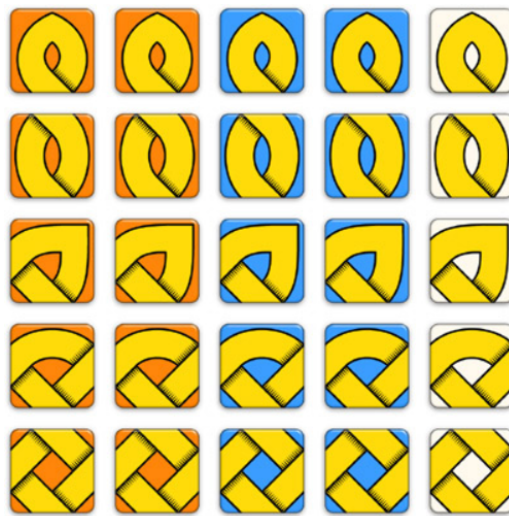
†Department of Computer Science, UTRGV,

‡Department of Computer Science, UTRGV,

§Department of Computer Science, UTRGV,

2.1 Tiles

For each tile, we examine 3 total attributes: E, its net exit effect which is the effect a tile will have on the number of exits on the board, its individual number of exits K, and the total number of tiles owned by each player is R, their resources. At the end of each player's turn, E will be added to the total amount of exits on the board, and in the case that the tile touches more than 1 exit, there's a condition that removes a number of those exits in total from the board for both players. The possibilities for the condition are: 1) If the amount of exits touching the tile are equal to K, subtract K from the total amount of exits. 2) The amount of exits touching the tile are still less than K in which case you subtract K - exits touching K from the total amount of exits. The game begins with 4 shared neutral tiles, but those are not made note of.



The total amount of playable tiles in the game.

2.2 Optimization

An aspect of the game explored was optimization and attempts to consistently gain the upper hand and overall win each game. An aspect of explored optimization is "disruption," coined to denote the usage of tiles that have either 0 or -1 E that force opponent players to lose exits to work with or to follow premeditated exit paths determined by the E and K of the board. Allowing opponents to have more overall paths allows for a greater number of factors

3 Our Results

Through trial and error, the consistently successful strategy was the "disruption" strategy. This strategy constantly does one of two main things: closes a knot early, causing the players to have to look for new openings or causes a knot to be impossible to finish within our 5x5 board resulting in a draw. It should be noted that this strategy doesn't cause a guaranteed victory but instead decreases the players chance of losing.

4 Conclusion

Finding a strategy or algorithm to guarantee a victory proved to be a challenging endeavor, as instead a way to minimize chances of losing was found. Players must use the disruption strategy on their opponent while also avoiding "mimicking" the actions of the other player. When the actions of both players are similar to each other in values E and K, the best case scenario for the first player to mimic the other is a draw. Though we made progress building a framework for a strategy in "celtic!" our framework raises more questions that should be further researched. Questions such as: What is the optimal amount, if any, of total exits for a player? Does it matter if the total exits are odd or even? Does a specific cardinal point from the origin affect the gameplay? Is it better to forfeit your turn in some cases to keep the board the same? How much of an effect can the neutral resources have? In order to help us understand these concepts further and answer these questions, further research with A.I. would most likely be optimal.

References

[1] Cameron Browne. Celtic! rules. <http://cambolbro.com/games/celtic/>.

Topic: Bioinformatics and Computational Biology

Tarana Mou *

Paulo Uchoa †

Luis Romo ‡

Abstract

We performed statistical testing to find the pattern of protein expression and phosphorylation between basal and luminal type cancers. In order to find pattern between basal and luminal type cancers, we have performed independent two tailed t-test between basal and luminal cancers. We have observed that the amount proteins highly expressed in basal and luminal types of cancer are almost similar while the luminal type cancer had almost six folds more proteins highly phosphorylated than the amount of the basal cancer.

Introduction

Human diseases can arise from different molecular pathogenic processes which can be characterized by mutations, methylations, protein expression, protein phosphorylation etc. Analysis and finding pattern of the molecular pathogenic processes could be the key steps to reveal the disease causes, diagnosis and prognosis. Understanding the vital facts of the disease could direct us the proper prevention and cure methods. Here, we performed statistical testing to find the pattern of protein expression and phosphorylation between basal and luminal type cancers.

Method

In to find pattern between basal and luminal type cancers, we have performed independent two tailed t-test between basal and luminal cancers. We performed T-test for each protein in Basal vs Luminal. Got the significantly highly expressed proteins in Basal and luminal. Again Performed the Independent Two-Tailed T-test for protein phosphorylation. Got the significantly highly phosphorylated proteins in Basal and also in luminal. Two different datasets from protein expression and protein phosphorylation contexts were used to conduct the statistical test. The alternative hypothesis was accepted for a 99 percent confidence level.

Result

We have detected 165 proteins had a significantly high expression in basal type cancer while 179 protein had

a significantly high expression in luminal type cancer. On the other hand, 203 proteins had significantly high phosphorylation in basal type cancer, while 1241 proteins had high expression in luminal type cancer. The significantly highly expressed and phosphorylated proteins in basal and luminal type cancers are listed in the results folder.

Conclusion

From the experiment, we have obtained the proteins those could play role to direct a cancer to basal or luminal types. Also, we have observed that the amount proteins highly expressed in basal and luminal types of cancer are almost similar while the luminal type cancer had almost six folds more proteins highly phosphorylated than the amount of the basal cancer.

*University of Texas, Rio Grande Valley

†University of Texas, Rio Grande Valley

‡University of Texas, Rio Grande Valley

An Investigation of the Existence of a Perfect Play in a Four Player 9x9 Grid Pentago Game

Domingo Martinez Jr. * Jose Oscar Mendoza † Arianna Arriaga ‡ Maria Romero §

Abstract

We present a solution to the question whether there exists a first player win strategy (perfect play) for the 9x9 grid board used for Pentago Multiplayer. We attempt to prove through contradiction that a perfect play for the first player is not feasible. Using the pigeonhole and the prisoners dilemma principles we hypothesize there is no single player perfect play strategy. However the game can be lengthened through combined efforts against the first player who can be seen to have the upper hand because of the first player’s ability to fill in the center 3x3 grid with 3 pebbles whereas the other players can only place 2.

1 Introduction

The version of Pentago considered is the 9x9 grid for 4 players. Pentago is an abstract strategy game whose objective is to connect five pieces of the same color in a horizontal, vertical, or diagonal line. A maximum number of four players can share the grid at a time. Each turn a player must place a game piece and rotate a 3x3 grid by 90 degrees. The rotation becomes mandatory when there is at least one pebble in each of the 9 3x3 grids where the pebble is not in the center. The game ends when a player connects five or there are no empty spaces resulting in a draw.

We briefly highlight some related work, and then provide the definitions and results of our work. We then conclude and point towards the general research goals for this work.

2 Our Results

Theorem 1 *The center 3x3 grid provides the most opportunities to win.*

Proof. $4^9 - 3 = 262,141$ is the total number of possible configurations by using only rotations. The center 3x3

grid has the most adjacent 3x3 grids with a total of 8. The other grids either have a total of 3 or 5 adjacent grids. □

Theorem 2 *The first player is the only player capable of having 3 pebbles in the center 3x3 grid regardless of the other players’ moves.*

Proof. If every player attempts to place a pebble in the center 3x3 grid. By the pigeonhole principle only the first player will be able to place 3 pebbles while the other players can only place 2. □

3 Alliance Strategy

We infer that there is no perfect play for player 1 because of the strategies we considered for players 2 and 3 to prevent player 1 from winning. If player 4 also aims to prevent player 1 from winning, then player 1 is more unlikely to win.

There are various ways to achieve this including allowing a player other than Player 1 to win.

In order for a player to win, there must be a minimum of five pieces of their designated color already in place, which may or may not require a rotation to connect them.

After testing various strategies, we haven’t identified an infallible strategy to ensure player 1 victory despite their advantage of placing their piece first. Player 1 can only make one grid rotation at a time during a move. To prevent a possible win, an alliance is essential to ensure any player aside from player 1 wins. It is impossible for player 1 to force an alignment of their pieces because the other players have 3 opportunities to block and rotate the corresponding 3x3 grids where player 1 may win.

4 Conclusion

Our current results suggest a perfect play for player 1 is unlikely. Player 1 is at a disadvantage in the event of a mutual alliance involving at least two other players in a single match.

*Department of Computer Science, University of Texas Rio Grande Valley, domingo.martinez01@utrgv.edu

†Department of Computer Science, University of Texas Rio Grande Valley, jose.mendoza05@utrgv.edu

‡Department of Computer Science, University of Texas Rio Grande Valley, arianna.arriaga01@utrgv.edu

§Department of Computer Science, My University of Texas Rio Grande Valley, maria.romero01@utrgv.edu

One-player Constraint Inequality is Easy

Cameron Chalk *

Boya Wang †

Abstract

We show, given an $n \times n$ board, deciding if the one-player constraint inequality game is solvable is in \mathcal{P} . Further, we give a polynomial-time algorithm which outputs a winning strategy iff the game is winnable.

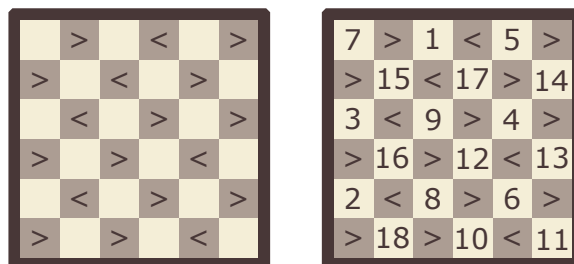
1 Introduction

In [1], the authors give the following simple puzzle in one-dimension: given a line of n cells, where every even cell has an inequality, insert numbers $1, \dots, \frac{n}{2}$ (using each number once) into the odd number cells such that every inequality is satisfied left-to-right. In this work, we study a natural generalization, the two-dimensional one-player constraint inequality game (CIG). Given an $n \times n$ chess board, where dark squares have inequalities, use each of $1, \dots, \frac{n^2}{2}$ exactly once to fill in the light squares such that inequalities are satisfied left-to-right and top-to-bottom. See Figure 1 for an example board and solution.

We show that in some cases, the game is not winnable—not all light squares can be filled in while satisfying the inequalities. We show that a game is not winnable if and only if there exists a cycle of inequalities, in the natural representation of the board as a directed graph. Such a theorem allows us to give efficient algorithms to both decide if a given board is winnable, and if it is indeed winnable, output a winning strategy.

2 Definitions

Definition 1 ($n \times n$ CIG board) For n an even integer, an $n \times n$ CIG board is a set of points $(x, y) \in \mathbb{Z}^2$ with $1 \leq x, y \leq n$. A point (x, y) is a dark square if x is odd and y is even, or if x is even and y is odd. Otherwise, a point is a light square. Each dark square is assigned one inequality, $<$ or $>$. An example 6×6 CIG board is shown in Figure 1a.



(a) A 6×6 CIG board B . (b) A full, valid configuration of B .

Figure 1

Definition 2 (Full, valid board configuration)

Given an $n \times n$ CIG board B , a full configuration of B is a bijective function C mapping light squares to numbers one through $\frac{n^2}{2}$. I.e., $C : \{1, \dots, \frac{n^2}{2}\} \rightarrow \{(x, y) \mid (x, y) \text{ is a light square}\}$. A full configuration C is a valid configuration if \forall black squares (x, y) , if (x, y) is assigned $<$ (resp., $>$), then $C((x-1, y)) < C((x+1, y))$ (and $C((x, y+1)) < C((x, y-1))$) (resp., $C((x-1, y)) > C((x+1, y))$ and $C((x, y+1)) > C((x, y-1))$). A full, valid configuration is depicted in Figure 1b.

Definition 3 (Solvable) An CIG board B is solvable if there exists a full, valid board configuration of B .

Definition 4 (Directed board graph) Given an CIG board B , we define the directed board graph $G_B = \{V, E\}$ where each $v \in V$ maps uniquely to one light square, and for vertices u and v mapping to (x, y) and (w, z) respectively, $(u, v) \in E$ iff one of the two following predicates is true: (1) $w = x + 2, z = y$ and the dark square $(x + 1, y)$ is assigned $>$, or (2) $w = x, z = y - 2$ and the dark square $(x, y - 1)$ is assigned $>$. An example board graph is depicted in Figure 2a.

Denote the light square mapped to by vertex $v_i \in V$ as (x_{v_i}, y_{v_i}) . Note that a full configuration C is valid iff for all edges $(v_i, v_j) \in E$, C must satisfy $C((x_{v_i}, y_{v_i})) > C((x_{v_j}, y_{v_j}))$.

The upper-left graph G_L of an CIG board contains all the vertices v_i in G_B such that y_{v_i} is odd, and includes all edges connecting those vertices.

The lower-right graph G_R of an CIG board all the vertices v_i in G_B such that y_{v_i} is even and, includes all edges connecting those vertices.

*Department of Electrical and Computer Engineering, The University of Texas at Austin, ctchalk@utexas.edu

†Department of Electrical and Computer Engineering, The University of Texas at Austin, bywang@utexas.edu

We point out the upper-left and lower-right graphs to show that G_B is not connected, but always has exactly two components, which is relevant to the implementation details for the graph algorithms used in the solutions later. The upper-left and lower-right graphs are depicted in Figure 2b.

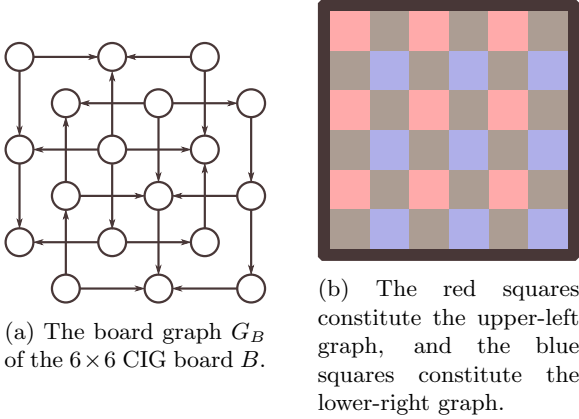


Figure 2

3 Our Results

Lemma 1 *Given an CIG board B , if the board graph $G_B = \{V, E\}$ has a cycle, the board is not solvable.*

Let $c = \{v_1, v_2, \dots, v_{k-1}, v_k, v_1\}$ be a cycle in G_B . By transitivity of $>$, this cycle implies a valid, full configuration must satisfy $C((x_{v_1}, y_{v_1})) > C((x_{v_1}, y_{v_1}))$, which is unsatisfiable, and thus there does not exist a valid, full configuration of B and B is not solvable.

Lemma 2 *Given an CIG board B , if the board is not solvable, then the board graph G_B has a cycle.*

Proof. We prove the contrapositive statement: if G_B does not have a cycle, then B is solvable. If G_B does not have a cycle, there exists a topological ordering $\{v_1, v_2, \dots, v_{\frac{n^2}{2}}\}$ of its vertices. Design a full configuration C as such: assign $C((x_{v_1}, y_{v_1})) = \frac{n^2}{2}$, assign $C((x_{v_2}, y_{v_2})) = \frac{n^2}{2} - 1$, etc. Informally, simply assign values in descending topological order. Recall by definitions that a full configuration C is valid iff for all edges $(v_i, v_j) \in E$, C must satisfy $C((x_{v_i}, y_{v_i})) > C((x_{v_j}, y_{v_j}))$. We have assigned each $C((x_{v_i}, y_{v_i}))$ as larger than each $C((x_{v_j}, y_{v_j}))$ which appear later in the topological ordering, therefore no inequalities are unsatisfied and thus, C is valid. \square

Lemma 3 *An CIG board B is solvable iff the board graph G_B does not have a cycle.*

Proof. Direct from Lemmas 1 and 2. \square

Theorem 4 *Deciding if an $n \times n$ CIG board B is solvable takes $\mathcal{O}(n^2)$ time.*

Proof. We give an algorithm for the decision problem. Construct G_B from B in $\mathcal{O}(n^2)$ time. Decide if G_B has a cycle in $\mathcal{O}(V + E) = \mathcal{O}(n^2)$ time. If G_B has a cycle, output *NO*. If G_B does not have a cycle, output *YES*. Proof of correctness follows from Lemma 3. \square

Theorem 5 *Given an $n \times n$ CIG board B , there exists an $\mathcal{O}(n^2)$ -time algorithm which outputs a winning strategy if the board is solvable, and outputs “not solvable” otherwise.*

Proof. We prove existence by giving the algorithm. Construct G_B from B in $\mathcal{O}(n^2)$ time. Decide if G_B has a cycle in $\mathcal{O}(V + E) = \mathcal{O}(n^2)$ time. If G_B has a cycle, output “not solvable”. If G_B does not have a cycle, topological-sort G_B in $\mathcal{O}(V + E) = \mathcal{O}(n^2)$ time. Assign numbers to vertices in G_B in descending topological order in $\mathcal{O}(n^2)$ time. Proof of correctness follows from the proof of Lemma 2. \square

4 Conclusion

We have shown that the decision problem of determining if an $n \times n$ CIG board is solvable is in \mathcal{P} . We have provided polynomial-time algorithms to (1) decide if a board is winnable and (2) output a winning strategy if the board is winnable. We leave the two-player variants of the game open as future work.

Acknowledgements

The authors would like to thank Tim Wylie for suggesting the open problem and helpful discussions, and Nerdvana and the HackResearch organizers for hosting a the first of many great, research-oriented hackathons.

Post-workshop Addendum

We note that the proofs herein do not utilize the two-dimensionality of the CIG board—the proofs apply just as well to the k -dimensional generalization. In fact, the proofs assume no structure whatsoever on the board graphs. Thus, the theorems discussed apply to deciding solvability and returning a solution for arbitrary sets of inequalities of the form $a < b$ or $a > b$ with similar constraints on variable assignments.

References

- [1] M. L. Anany Levitin. *Algorithmic Puzzles*. Oxford University Press, 198 Madison Avenue, New York 10016, 1 edition, 2011.

Adaptive Design for Adaptive Learning

Mario Salinas *

Abstract

This paper observes if video games can serve as a tool for learning while having fun instead of it simply being a form of entertainment. The majority of modern games already have players learning small tasks that teach them certain elements that can be used in-game only, but what if these elements can be utilized outside of the game using the same principles? A game could revolve around a certain set of tools in order to teach an individual an actual real world task or indirectly teach them skills they otherwise would have never gained because the difficulty of the activity is too high. All of this can be done by adaptive learning, where an individual is given a playing field with all the tools they need in order to learn anything at a pace of which they choose via a video game.

1 Introduction

Game Design is a complex and time consuming process, and the task would be daunting to most individuals. So how could we make game design approachable and fun for just about anyone? How could we translate level design, testing and editing into a game itself? Introducing: Super Mario Maker! Released in 2015 on the WiiU, it gave individuals the easy-to-pick-up-and-master tools in order to create any kind of level they desired. Outside of it being a game, it served as a great learning and creativity tool for adults and children. It taught them that creating a level isn't very easy, and that there's a lot of trial and error behind every great game level or world. I personally played the game for two years, and my vision as a game designer definitely changed while I learned from my own mistakes. It was also possible to share the level with friends or watch others play it over a streaming service like Twitch, which allowed me to observe human behavior and decisions during play. After much observation, I was able to adapt and utilize the tools given to me properly because the game allowed me to learn and improve upon my design. The differences were definitely there when comparing my first iteration of a level and the final product. In conclusion, I learned a lot over time while still having fun. So what other games share similar principles? Why haven't these principles transferred over to other areas such as our education system? Children could greatly benefit from such a system, where they are given a canvas to adapt and

learn with the tools provided to them.

2 Further Possibilities

There are already games that set out to do what was stated above, such as Minecraft. The game could be used to create circuits, CPU's and other aspects of electrical engineering. Since that topic can be a bit difficult for a younger audience, elementary schools have actually adopted the game to teach math to elementary students. According to author Anton Petrov, in his paper titled "Using Minecraft in Education: A Qualitative Study on Benefits and Challenges of Game-Based Education", two elementary teachers by the names of Larry and Deb use Minecraft as a means of education in their classes. Both teachers have reported great success in motivating their students to work hard and reach curriculum expectation while using the video game. Each teacher approached each class differently though, and one had their students keep a math journal for the activities in Minecraft. These journals verbalized their thinking process in the game, and it also helped them with their overall writing abilities and math vocabulary. Here's a quote from Deb, "I used it with my third graders once, teaching language and math. I like my students leading the way, and I had them keep a math-Minecraft journal. At first it was hard for them, for instance: How many bricks do you need to build a house? What is the area of the inside of the house? Is a rectangular house bigger than a square house? How many blocks did you need? How many blocks does one tree get you? I find this knowledge a lot more helpful, because you need to know those things in the game." This type of knowledge can be extended upon other aspects of the game, as well as their math abilities and real world activities such as counting money for instance. How many pennies does it take to make a dollar and etc. But this type of gained knowledge must have some sort of incentive to motivate an individual to take what they learned to the next step. A paper by Dondlinger mentions another work by Lee, Luchini, Michael, Norris, and Soloway (2004): "a math facts game for second graders deployed on handheld computers encouraged learners to complete a greater number of problems at an increased degree of difficulty. Learners playing the handheld game completed nearly three times the number of problems in 19 days as those using paper worksheets. Learners using the handheld game also voluntarily increased the level of difficulty in the game as they continued to play."

*Department of Computer Science, UTRGV, mario.salinas01@university.edu

3 Taking it Even Further

It doesn't stop there of course. The learning process in video games can go beyond that of the standardized education realm, but let's stop for a moment and evaluate how video games can be designed to facilitate learning. There's a paper written by Mary Jo Dondlinger, and it states that there's a difference from edutainment and education: "The main characteristic that differentiates edutainment and video games is interactivity, because, the former being grounded on didactic and linear progressions, no place is left to wandering and alternatives". So what should a game designer strive for if they wish the player to adapt to their game? Adaptive Design. Design a game that adapts to the player, and the player will act accordingly since they will see that there are many other possibilities after learning that a number of small tasks can lead to an even bigger outcome. They will want to strive for more, and they will wonder where the ceiling of possibilities end within the game. Game design with aspects like these in mind require a lot of extra effort of course, since the designer must be ready to program their game to simulate a number possible outcomes that a player will attempt. The paper I mentioned above by Mary Jo also states: " In contrast, educational video games require strategizing, hypothesis testing, or problem-solving, usually with higher order thinking rather than rote memorization or simple comprehension. Characteristics of such games include a system of rewards and goals which motivate players, a narrative context which situates activity and establishes rules of engagement, learning content that is relevant to the narrative plot, and interactive cues that prompt learning and provide feedback. " Which takes me back to a game like Super Mario Maker, where the player can set their own goals and proceed at their own pace. They can stop once they've reached their desired point and ask another to play their level. While watching that player go through their level, they may notice a flaw in their design when their friend was able to bypass an obstacle in an unintended way. So the designer must ask themselves, " What can I do to allow this not to happen? What did I do wrong? " Then it's back to the drawing board.

4 A Game Idea

Probably the best challenge to take on when designing a video game with the goal of teaching the player in mind, is to find an activity that is usually intimidating for those that know very little about it. Then, you should find ways to make it less intimidating and fun. We can take an activity like cooking for instance, an activity that is scary for many due to a number of reasons. Or we can push the idea, and have the player manage a small restaurant. There can be a character creator, as well as an editable biography for said character to

give the player a reason to care for their character. We can also allow the player to name their character and their staff. Now we have a number of individuals that the player needs to care for and manage, as well as a restaurant that they can design and manage as they please. The player can start off with a certain budget, but their budget can grow as they learn how to manage their restaurant properly. They can start off with simple dishes/ingredients, but the larger their budget grows, the better and larger their ingredient/dish list becomes. Though, there can be obstacles where the player may experience setbacks if their restaurant is poorly managed, so they will have to reevaluate what they're doing wrong. Aside from the management aspect, you will have the fun element where you simulate the cooking aspect of the restaurant. Dishes can be prepared or taught to the player in numerous ways, and they will have all the tools they need from the get go in order to put them out. All the dishes can be judged accordingly: the time of which it was completed, if the right ingredients were used, if new ingredients are compatible with the dish, if the correct temperature of the meat was met etc. All these different aspects have potential to motivate the player to strive and learn.

5 Conclusion

It is clear from the number of topics on the subject, that video games and their design do affect learning. A number of design elements such as narrative context, rules, goals, rewards, multisensory cues, and interactivity play a large role in the learning outcome. It's also possible to improve upon those designs that have been proven effective in order to reach a better fun learning experience. As technology grows, and more individuals become interested in the field, we will no doubt see a greater number of games that invoke adaptive learning.

References

- [1] Anton Petrov. Using Minecraft in Education. A Qualitative Study of Benefits and Challenges of Game-Based Education. 2014.
- [2] Erin Shaw, Minh Tuan La, Richard Phillips, and Erin B. Reilly. PLAY Minecraft! Assessing secondary engineering education using game challenges within a participatory learning environment. In Proc. of the *121st ASEE Annual Conference & Exposition*, 2014.
- [3] Lee, J., Luchini, K., Michael, B., Norris, C., and Soloway, E. (2004). More than just fun and games: Assessing the value of educational video games in the classroom. Paper presented at the CHI '04 Extended Abstracts on Human Factors in Computing Systems, Vienna, Austria.
- [4] Mary Jo Dondlinger (2007). Educational Video Game Design: A Review of the Literature. Presented by the Department of Technology & Cognition, College of Education, University of North Texas.

Network Concurrency Over Online Video Games

Antonio Adame *

Abstract

During a fast paced multiplayer experience, it is vital that there are precise algorithms in place to combat server lag, cheating as well as timing. For this research project, I set out to find an efficient implementation for this problem. Using timing techniques, for sending data about deterministic events from server to client and vice versa, we can see how these simple techniques fare in greater volume scenarios.

1 Introduction

Multiplayer video game design requires high speed communications between the client and the server. When designing these communications, we always want to assume that we cannot trust the client. Authoritative server implementations are what is most commonly used when you wish to solve problems like this. Authoritative servers, in short, take care of positioning the players around the game area. This makes it possible to stop the client from attempting to make invalid moves, since the server acts as authority to move the player. For example, if player was at $p(10,10)$, and we pressed the right key to move to the right, we would send that data to the server. The server would then know that we have pressed the right key, so the server then returns $p(11,10)$ and the client then moves. This however, can pose a problem when we have an animation that can take a while to complete. For this example, we will assume we have an animation that lasts 1000ms with a server lag of the same length. If a call takes 1000ms to make a round trip and we play the animation as soon as we press the right key, we will see the player do the animation without moving, and once the response from the server is received, the player will move. Evidently, this causes a mismatch of the player's expected movement versus what actually happens (Figure 1).

For my implementation, I did not use animations. However, client side prediction and synchronization can be applied to collision detection in making sure that all players see each other in the correct positions and states are changed at the same time for every player. This gets increasingly difficult when dealing with several players, as the servers requires more computation in order to efficiently show every player's precise movements and

state changes (in the event of having two players of different teams collide with a third player almost at the same time, more on this later).

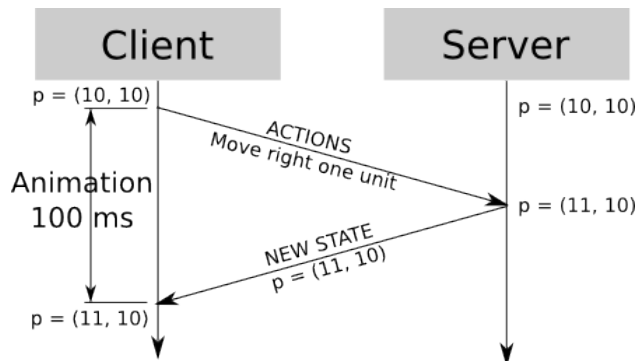


Figure 1: client side prediction.

2 Our Results

For my implementation, I simulated client connections to a local server. The game relies on the server relaying every active users position and color in the game, while each player attempts to "catch" the other players in order to turn them to their side (in the case of this game, if player A collides with player B, player B will turn the same color as player A, therefore now they are in the same team and should now attempt to turn every other player into their team until no other color remains). Since this server was hosted locally, it was difficult to test this with real server latency, so I had to rely solely on making my implementation as efficient as I could. Using common methods for collision detection in Javascript, I was able to allow the server to let the client know if he had collided with any of the users currently in the game. After this, there had to be some coordination between the server and the client using asynchronous functions in Node.js in order to transmit the new color of the collided player from the server to all of the clients.

3 Conclusion

In all, my implementation required a bit more testing in order to fully prove and test several cases. However I do feel that I was able to discover implementations for server and client communications. Furthermore, the algorithms I used for collision detection could have been a

*Department of Computer Science, My University, antonio.adame01@utrgv.edu

bit more sophisticated, as I simply measured each player's position with an offset on their radius (players are represented by circles). When checking for collisions, my algorithm would compare the requested player with every other user in game, since this is a polynomial time algorithm, it is not very efficient. With more time, a possible optimization would be to only worry about checking collisions with other players who are nearer to me. While still polynomial time in theory, this would cut the computation cost significantly depending on player volume.

<http://www.gabrielgambetta.com/client-side-prediction-server-reconciliation.html>

<https://www.github.com/aadame3311/NC-in-videogame-implementation>

Computational Biology

Daniel Acevedo *

Abstract

Cases of the disease are often considered a single outcome, and assumed to share a common etiology. However, evidence indicates that many human diseases arise and evolve through a range of heterogeneous molecular pathologic processes. That's why one drug can improve the symptoms of a patient while making the other patient with the same disease worse. For this reason, even for one disease, there might be multiple subtypes (sub-categories). For example, there are five main molecular subtypes of breast cancer. For example, Luminal which tend to grow slowly and have the best prognosis and Basal which is more common to happen with a mutation in a gene called BRCA1 and it is more common between younger women and have worse prognosis. We introduce machine learning to try to identify patterns between the data and type of disease that is caused.

1 Introduction

The data given varies. We look at protein interaction between other proteins. We look at the intensity of expression of proteins in different patients. Protein Phosphorylation is looked at and we focus on the intensity of how much each protein locations has been phosphorylated.

To classify this data to either a basal or luminal we have to find a model that can predict a output with multiple features and associate the different types of behavior of different proteins that lead up to one of these choices. For this project we chose to use a support vector machine(SVM). A support vector machine is great for clustering data, they are used for linear classification but also efficiently perform non-linear classification. With out sparse data that doesn't correlate it would be time consuming for even a SVM to train a non linear classifier so we use something called a kernel trick. What this does is implicitly maps input into high-dimensional feature spaces.

2 Our Results

Results are still pending. Training the network is still in progress.

github → github.com/daniel235/breastCancer

*Department of Computer Science, University of Texas Rio Grande Valley, daniel.acevedo01@utrgv.edu

3 Method

We used various number manipulation libraries and scientific tools to prepare the data for our network. We used python as our programming language and used sci-kits library to get our support vector machine. The data was broken up into pairs. We organized the name of the protein and each of the numbers as a pair and then added a separate array to distinguish what type of cancer was the result of this pair. We placed 4 dimensions and they all produced one output either basal or luminal. The first two dimension belonged to the protein phosphorylation data and the last two belonged to the protein expression data. These two were the only data sets the had a result the other we plan to add in after training on the current data.

4 Conclusion

In Conclusion we believe any clustering algorithms such as KNN, naive bayes, etc would be worth looking into so we can compare performance of the different models. In the future we hope to optimize the model to attain correlations that can be used in other research.

References

- [1] Patel, S. (2017, May 03). SVM (Support Vector Machine) - Theory Machine Learning 101 Medium. Retrieved from <https://medium.com/machine-learning-101/chapter-2-svm-support-vector-machine-theory-f0812effc72>.

Bioinformatics and Computational Biology

Austin Karingada *

Odette Marin †

Abstract

One of the problems that we are trying to see within the data set is to see whether there is a correlation between basal and luminal cancer types. We look into the data sets and saw on average, basal phosphorylation is lower than luminal phosphorylation.

1 Introduction

In the work, we used python to extract and calculate data in order to see whether there is a correlation between the data. We calculated the average phosphorylation between all basal patients and luminal patients. We also used Pearson method from a python library in numpy in order to find correlation between all patients.

2 Our Results

Our results were that luminal had higher phosphorylation than basal by calculating the average for every protein in each patient but separated them by what type of cancer they had. We also were in the middle of calculating the Pearson correlation values but was short on time so the results are still inconclusive on that part.

3 Conclusion

There were some technical errors throughout the process, one was when we were trying to calculate the data, it kept showing errors in which we didn't know why this was happening until later. The error was that the data was all strings and that strings don't do calculations so by the time we got around to do the calculations, we ran short and had really little data.

*Department of Computer Science, My University,
austin.karingada01@university.edu

†Department of Computer Science, My University,
odette.marin01@university.edu

Author Index

Acevedo, Daniel, 13
Adame, Antonio, 11
Alaniz, Robert Michael, 3
Arriaga, Arianna, 6

Balanza, Jose, 1

Caballero, David, 1
Canales, Eden, 3
Cantu, Angel, 1
Chalk, Cameron, 7

Flores, Mauricio, 1

Karingada, Austin, 14

Marin, Odette, 14
Martinez Jr., Domingo, 6
Marvin, Richard, 3
Mendoza, Jose Oscar, 6
Mou, Taranau, 5

Rios, Alvaro, 1
Romero, Maria, 6
Romo, Luis, 3, 5
Ruiz, Daniel, vi

Salinas, Mario, 9

Uchoa, Paulo, 5

Wang, Boya, 7

