A Citizen Science Approach for Sorting Genomes

by

Anjum Ibna Matin

A thesis submitted to
The Faculty of Graduate Studies of
The University of Manitoba
in partial fulfillment of the requirements
of the degree of

Master of Science

Department of Computer Science
The University of Manitoba
Winnipeg, Manitoba, Canada
November 2018

Thesis advisor

Olivier Tremblay-Savard

Author

Anjum Ibna Matin

A Citizen Science Approach for Sorting Genomes

Abstract

The development of citizen science games requires the implementation of different game mechanics to make them challenging, interesting and motivating. These mechanics are often borrowed from commercial video games, but their outcome on the quality of solutions obtained and player motivation are not fully understood.

Citizen science games can be used to solve complex scientific problems. Genome sorting is such a complex problem in genomics which is to find the shortest number of events for transforming one genome to another. Through solving this genome sorting problem we can also learn much about genome evolution. However, the presence of duplicate genes in the genomes makes this problem NP-hard.

In this thesis, we propose a citizen science game *GeSort* which aims at solving the genome sorting problem even in the presence of duplicate genes. Through this game we also analyze the effect of showing a timer, an achievable (top) score and a live leaderboard on players' scores, puzzle completion time and motivation using different versions of our citizen science game *GeSort*.

Contents

List of Figures

List of Tables

Acknowledgments

I would like to express my gratitude to Almighty God for granting me the opportunity to write this thesis.

I would like to thank my supervisor, Dr. Olivier Tremblay-Savard for giving me the opportunity to work under his supervision. I am very grateful for his guidance, suggestions, and feedback throughout the preparation of this thesis. I am thankful to the other members of my committee, Dr. Andrea Bunt, Dr. Noman Mohammed and Dr. Jérôme Waldispühl for the comments and advice they provided.

I am also thankful to my colleagues Rogerio de Leon Pereira and Mardel Maduro for their valuable consults and time. I am equally grateful to all the benevolent faculties of University of Manitoba Computer Science department who shaped my research and academic standing.

Finally, I would like to thank my father, mother, fiance and family for their countless sacrifices and inspirations throughout the years.

This thesis is dedicated to all sacrifices.

Chapter 1

Introduction

Evolution in biology describes the diversification and ancestry of all living entities. The idea of evolution was introduced in the philosophy of Heraclitus (535-475 BC) and established in the sixth century AD [3]. But a little was known back then about how biological evolution occurs, what are the changes it makes in the species. Modern computational biology attempts to answer all these questions. Now research continuously improves our understanding of the genetic changes that have been made into the species through evolution.

Genome evolution refers to the process that changes the structure and the content of a genome over time. During this evolution, genomes keep accruing mutations and these changes affect the gene content and the structure of the whole genome. The events which occur during this evolution can be subdivided into two categories: *rearrangement events* such as inversions, which reorganize the orders of the genes and *content modifying events* which affect the copy number of the genes such as duplications and deletions. Gene duplication is one of the fundamental processes

for genome evolution [4] especially for vertebrate genomes [5; 6; 7; 8]. In general, a significant number of genes in bacteria, archaea and eukaryotes are the result of gene duplication. [9]. Deletions or losses can occur for a single gene or for a segment. Just like other events, deletions can occur as long as it doesn't create a problem for the organism. [5; 6; 7; 8; 10; 11].

The study of genome evolution gives us a better understanding of how the genomes are evolving, how two genomes from different species are related to each other or how distant they are in an evolutionary scenario. Comparing genomes is the first step into a better understanding of how genomes have evolved from ancestral species to species we can observe today. To understand the evolutionary history between genomes, we can infer a sequence of events to transform one genome into another. The *genome sorting problem* is to find the shortest sequence of biological events that can do this transformation. Through solving this genome sorting problem we can also learn much about genome evolution, such as the types of events that are occurring more frequently for example. This problem is simpler when the genome has exactly one copy of each gene and can be solved in polynomial time [12; 13]. However, the usual case is to have multiple copies of certain genes in the genome. In this case, the genome sorting problem becomes NP-hard [14; 15; 16; 17; 18] as there is no one-to-one correspondence between the genes and the source of duplication remains unknown.

Crowdsourcing approaches for solving difficult computational problems are getting popular nowadays. Games like Foldit[19], Phylo[20], Ribo[21], EteRNA [22] illustrate the potential of this technique for solving complex problems in bioinformatics. It also

demonstrates that well-chosen scientific problems can be framed into casual computer games to reach a large pool of contributors. With more than 4 billion users spending on average 6 hours online per day [23], Internet gathers a tremendous amount of human attention. By tapping into this potential, crowdsourcing systems have the potential to provide massive resources to help with solving complex scientific problems.

But for getting the best output from these large pool of players, these human computation games (HCGs) need to be more interesting and captivating. Lack of engagement in these HCGs may result in loosing the players soon after they start playing the game [24; 19]. One way to make these HCGs more captivating is to add game features like a scoring scheme or an award system to motivate the players. This may also lead us to get better results from these HCGs. However, the effects of these game features on motivating the players and on producing quality data have not yet been thoroughly studied.

Recent researches on human computing have started to focus on analyzing the game features and game mechanics [25; 26; 27; 28; 29]. Most of these studies focused on a particular game feature, such as Gaston et al. [30] implemented a three star system on the protein folding game Foldit [19]. Their work [30] illustrates the impact on encouraging the players for getting better results. They showed that with their three star reward system, the players are motivated to perform better and recomplete the levels more often [30]. Siu et al. [26] studied the effects of multiple reward systems on the players performance and experience in HCGs where the players are able to choose the type of reward. They have found that in the version where the players can choose the type of the reward, they performed better than the version

where the rewards were assigned randomly and this resulted in higher correctness of the results and lowering the completion time. These results indicate the necessity of different game mechanics for getting better results from the large number of online players. However, there is still much to learn about certain other game features (timer, on board top score, live leaderboard) and how a combination of them can interact together.

This thesis thus focuses on developing a citizen science game which aims to solve the genome sorting problem with the presence of duplicate genes. One of the major objectives of this game is to get better results from the players. Previous studies [19; 26] showed that adding different game features such as a reward system can lead us to make a HCG more interesting and challenging and this may also lead to get better results. For this reason, this research also focuses on evaluating the effect of different game features (timer, top score and leaderboard) on player's performance and motivation using different versions of this citizen science game. Besides evaluating all these three game features individually, this research also studied the combination of these three game features to find out an effective way to design this HCG.

The research objectives and how this thesis is organized, are given below:

1.1 Research Objectives

This research has two major objectives: first, to analyze the effectiveness of different game features (timer, top score and leaderboard) in a human computing game. Second, to solve the genome sorting problem, which is algorithmically NP-Hard [14; 31; 16; 17; 15; 32] in the presence of duplicate genes, using this game.

After getting a reasonable amount of data from the large pool of players available online then we will analyze the results of the players, their puzzle solving patterns and compare these results with the existing algorithms [2; 33].

1.2 Organization

This thesis is organized as follows:

- Chapter 2 describes the biological and algorithmic background of this research.
- Chapter 3 presents the background of citizen science games.
- Chapter 4 describes the effect of timer, top score and leaderboard on performance and motivation in a citizen science game. This work is submitted to CHI 2019.
- Chapter 5 explains the citizen science game we developed for solving the genome sorting problem. A short paper about this citizen science approach was published in HCOMP 2017 [34].
- Chapter 6 concludes this thesis.

Chapter 2

Bioinformatics Background

2.1 Biological Context

The study of genome evolution gives us a better understanding of how the genomes are evolving, how two genomes from different species are related to each other or how distant they are in an evolutionary scenario. A genome is a set of all the genetic materials of a living organism: it is the entire set of genetic instructions to build and maintain an organism. That's why genomes are often described as the information repository of life. Genomes are made up of chromosomes which are molecules of DNA. Genomes contains genes and proteins. A gene is a segment of a DNA strand which codes for proteins. DNA stands for deoxyribonucleic acid which is a large molecule composed of millions of nucleotides. There are four possible nucleotides: Adenine(A), Guanine(G), Cytosine(C), Thymine(T). To make a protein (the molecules that perform most of the functions in living organisms), another important molecule similar to DNA is needed which is called RNA (ribonucleic acid). RNA is composed of nu-

cleic acids and there are several classes of RNA that can play many different roles. Messenger RNA (mRNA) is responsible for carrying the genetic information copied from DNA. Transfer RNA (tRNA) is responsible for carrying amino acids in protein synthesis. Ribosomal RNA (rRNA) is involved in making up the ribosome and the ribosome makes the protein according to the instructions from the mRNA with the amino acids carried by the tRNA.

DNA molecules are made up of two complementary strands connected together. All the genetic information can be found on any of the two strands. These strands also have an orientation. The normal orientation in most biological processes is 5'-3' (Forward strand). The two complementary strands of DNA are in the opposite orientation relative to each other. This also means that the genes that are found in one strand, are going to be in the opposite direction than the genes that are found on the other strand. Sequence of genes in a chromosome can also be represented as permutations of characters, where each gene is labeled as one character. A *gene order* thus refers to this kind of permutation, representing a sequence of genes as they appear on a chromosome. Gene orders can be signed, in which case genes that are on the forward strand will have a plus sign, and genes on the opposite strand will have a minus sign. Traditionally, the order of the genes was determined by the percentage of recombination between each gene pair. This percentage of recombination tells us how a gene pair is linked to each other and this percentage works as a map unit while mapping the genes on a chromosome. For an example, if we have three genes (X,Y,Z) on a chromosome and the percentages of recombination between each pair of these three genes are: 5% between X and Y, 3% between Y and Z, 8% between X and

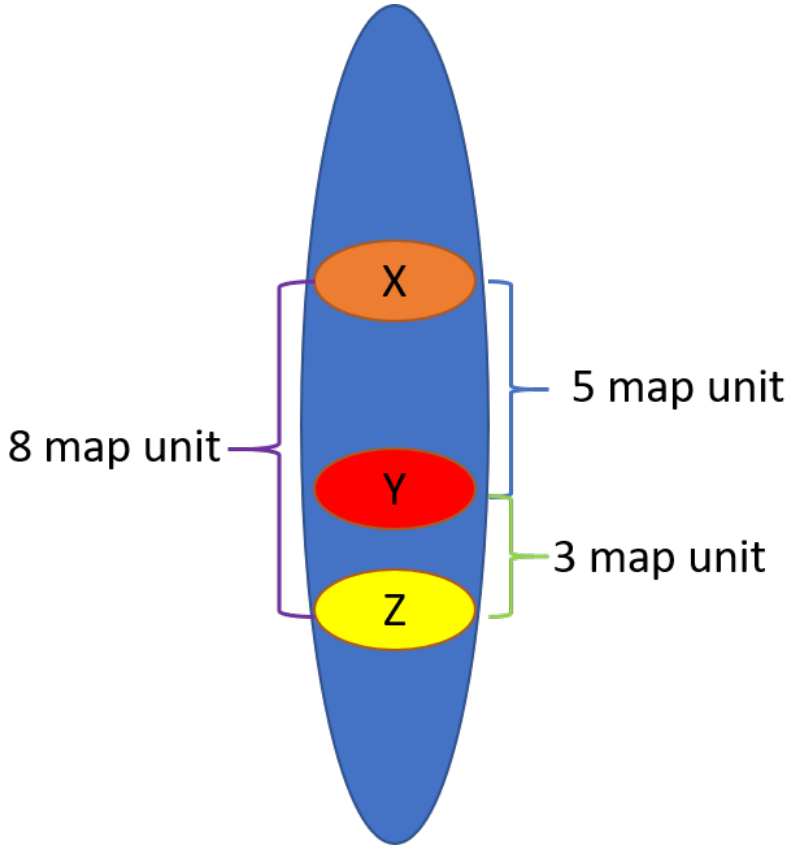


Figure 2.1: Order of X,Y,Z genes in a chromosome.

Z, then the genes can be mapped into the chromosome in the order shown in Figure 2.1.With the reductions of sequencing costs in recent years and the greater availability of fully sequenced genomes, it is now possible to use fully annotated genomes to obtain gene orders.

Genome evolution is the process of changing the structure and content of genomes over time. During this evolution several evolutionary events may occur. The examples of these evolutionary events are given below:

Deletion: Deletion is an event where a part of the chromosome or a sequence of DNA is lost during evolution. An example of deletion is shown in Figure 2.2.

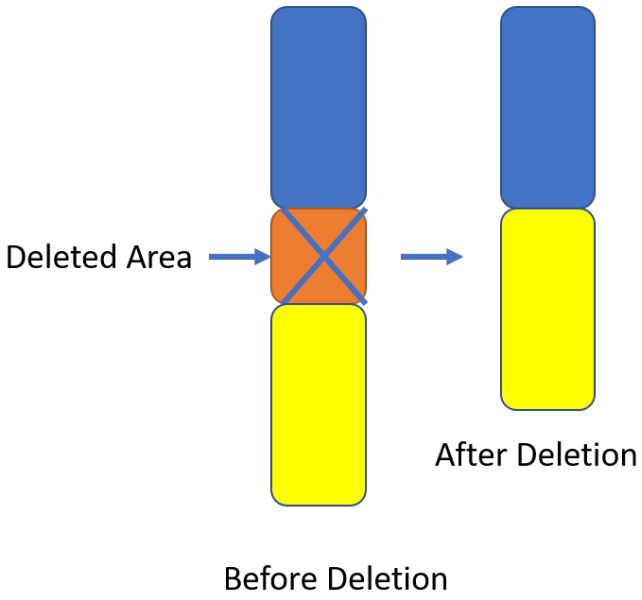


Figure 2.2: Deletion in a chromosome.

Duplication: Duplication is the event where a single gene or a sequence of genes are replicated in a place other than its original place on a chromosome. An example of duplication is shown in Figure 2.3

Inversion: Inversion is the process where a segment on the chromosome is reversed. In bacterial genomes, which have a single circular chromosome, the inversions often occur around the "origin" or "terminus" of replication, which are where DNA replication (copying) starts and end respectively [35]. Usually genes before the terminus are in a positive orientation and genes after the terminus are in a negative orientation. An example of the inversion event is shown in Figure 2.4.

Translocation: In multichromosomal genomes, translocation events are also possible. A translocation is an event that exchanges genetic material at the extremities of two different chromosomes. An example of the translocation between two chromo-

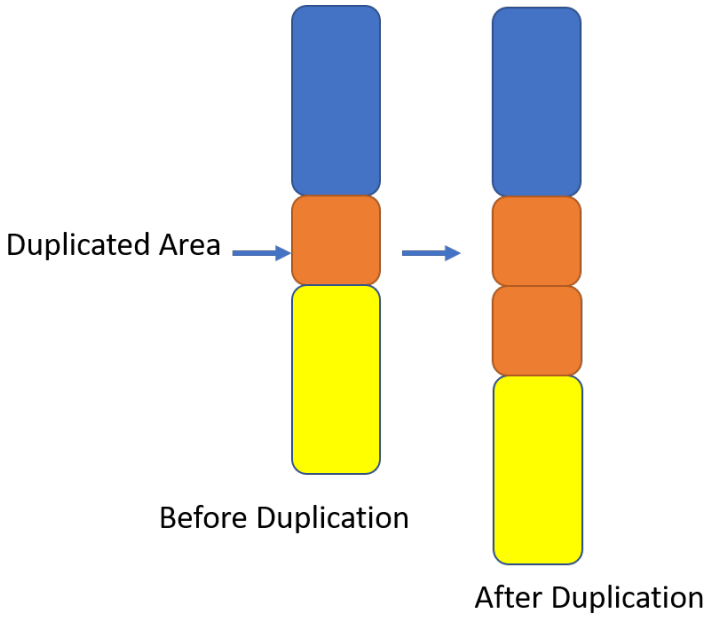


Figure 2.3: Duplication in a chromosome.

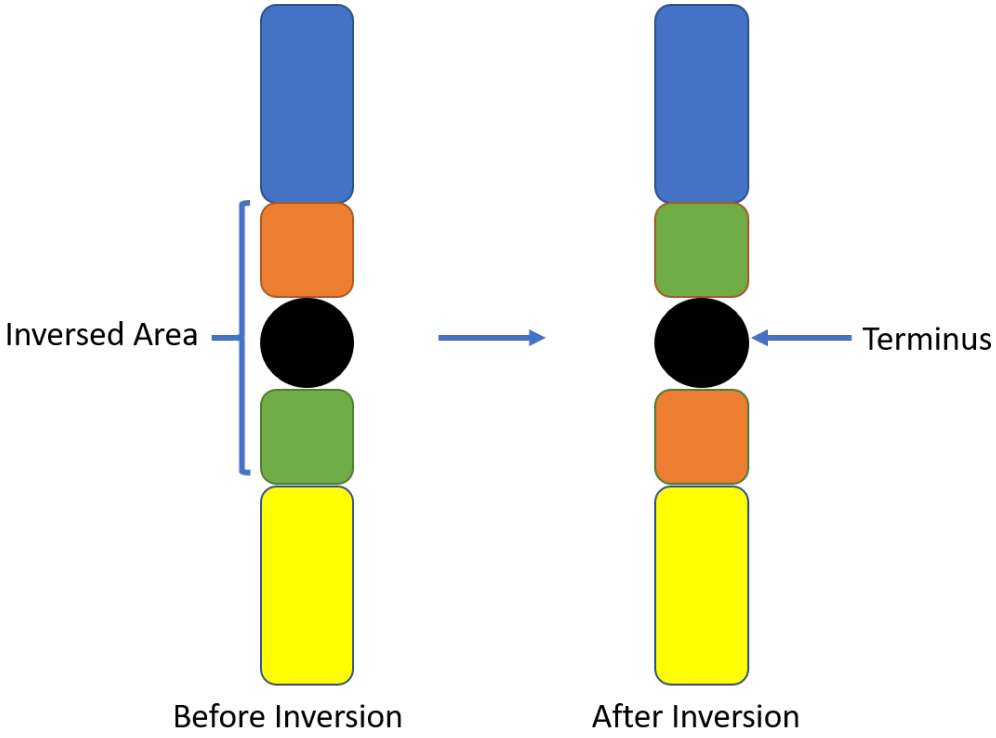


Figure 2.4: Inversion in a chromosome. Here the black circle represents the terminus

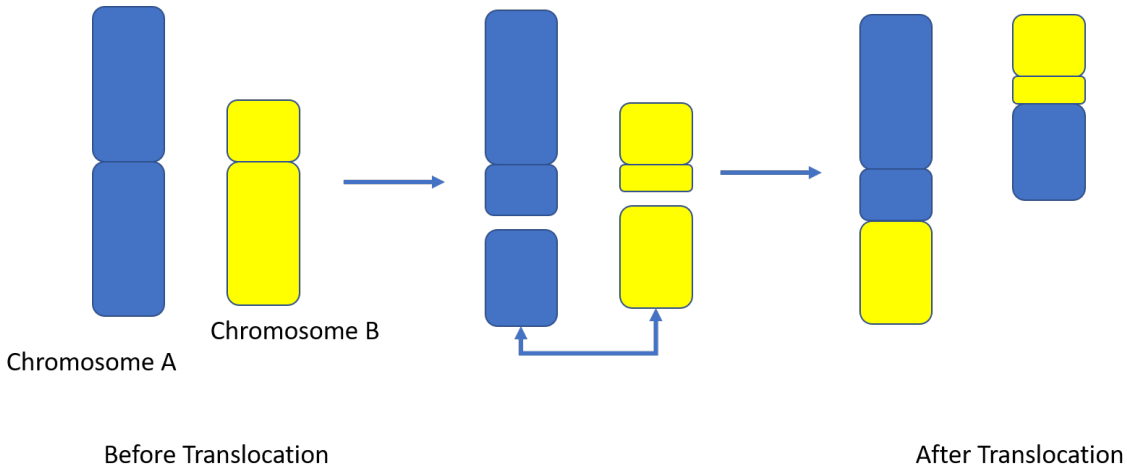


Figure 2.5: Translocation between two chromosomes.

somes is shown in Figure 2.5.

Transposition: Transposition is the process of removing a segment from a chromosome and reinserting it to another place on the same chromosome or on a different chromosome. An example of a transposition event is shown in Figure 2.6.

2.2 Algorithmic Background

In evolutionary molecular biology, the study of genome rearrangements is very important. One of the most studied problems is to compute the minimum number of evolutionary events required to transform one genome into another. This problem is called genome sorting. By sorting two genomes we can also know much about their evolution. For this transformation we need to perform some content-modifying operations such as deletions and duplications and rearrangement operations such as reversals (inversions), transpositions and translocations.[36; 37; 38].

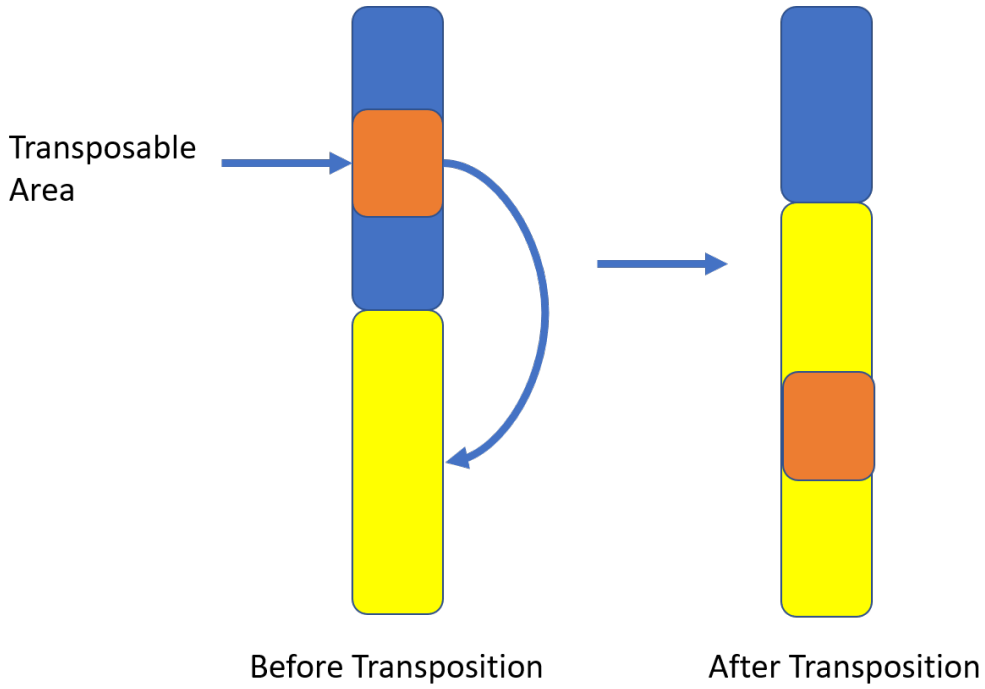


Figure 2.6: Transposition in a chromosomes.

2.2.1 Sorting by Reversals

Sorting a genome by reversals is the problem of finding a shortest sequence of inversions that transforms one genome into another. The position that represents an adjacency that is present in one gene order but not in the other is called a breakpoint. Moreover, a breakpoint is the position in a permutation which needs to be changed by an operation to bring it closer to the other permutation. Kececiglu and Sankoff [37] presented a greedy approximation algorithm which is capable of removing most of the breakpoints from a permutation and at least one breakpoint with every reversal. Their method is a 2-approximation algorithm and the worst-case time complexity of their algorithm is $O(n^2)$. Bafna and Pevzner [39] then introduced a

very important concept of reversal cycle graph which is an edge colored undirected graph. This concept is important because they improved the lower bound for the reversal distance by considering cycles in this graph. Bafna and Pevzner thus gave a 7/4-approximation algorithm and the worst time complexity was also $O(n^2)$. Later Christie [40] presented a 3/2-approximation algorithm for sorting by reversals by introducing a reversal graph.

All these algorithms had the same time complexity until Hannenhalli et al. [12] proved a duality theorem for solving the problem of sorting signed permutations by reversals which is a polynomial time algorithm. This was the first exact polynomial time algorithm for a realistic model of genome rearrangements. Later Tannier et al. [13] proposed an exact method for sorting signed permutations by reversals with a time complexity of $O(n\sqrt{nlogn})$.

2.2.2 Sorting by Transpositions

A transposition is an operation that swaps two adjacent gene blocks in a genome. Similarly to the reversal distance, the transposition distance is the length of the shortest sequence of transpositions that transforms one genome to another and sorting by transpositions is the problem of finding that shortest sequence. Guyer et al. [41] gave a heuristic for sorting by transpositions. Later Bafna and Pevzner [42] introduced the concept of a transposition cycle graph which is a directed edge colored graph. Thus they obtained a polynomial time 3/2-approximation algorithm and the complexity for this problem is unknown [42]. Hartman et al. [43] simplified the problem of sorting by transpositions by proving that the problem of sorting circular permu-

tations by transpositions is equivalent to the problem of sorting linear permutations by transpositions. And it led them to have a 3/2-approximation algorithm with a time complexity of $O(n^{3/2}\sqrt{logn})$. Actually the problem of sorting by transpositions is NP-Hard [44], that's why we only have the approximation algorithms to solve this problem.

2.2.3 Sorting by Reversals and Transpositions

Analysis of genomes evolving by reversals and transpositions leads to a more complete model of genome rearrangements and sorting by reversals and transpositions is the problem of finding the shortest sequence of reversals and transpositions required to transform one genome into another. Walter et al. [45] worked on that problem for linear chromosomes and gave a 3-approximation algorithm. Rahman et al. [46] then introduced a cycle decomposition approach for sorting unsigned permutations by reversals and transpositions which improves the approximation ratio to a 2K-approximation algorithm where K is the approximation ratio of the cycle decomposition algorithm used. Using the best value of K for this problem which was determined by Guohui et al. [47], their algorithm thus had an approximation ratio $2.8386 + \delta$ where $\delta > 0$ and a small positive number.

2.2.4 Sorting by Translocations

Sorting by translocations is the problem of finding the shortest sequence of translocations that is required to transform one multichromosomal genome into another. Kececioglu et al. [48] studied the translocation distance problem for the first time

and they gave a 2-approximation algorithm.

Hannenhalli [49] proved a duality theorem for calculating translocation distance for signed permutations in polynomial time which leads to an algorithm for transforming one genome to another by translocations. The worst case time complexity of that algorithm was $O(n^3)$. Later Bergeron et al. [50] gave a proof that the translocation distance can be computed in linear time and gave a new algorithm for sorting by translocations.

2.2.5 Sorting by DCJ

A double-cut-and-join (DCJ) is an evolutionary model which calculates the distance between two genomes. This model cuts a genome into two pieces either in one chromosome or in two different chromosomes. Then it rejoins the four cut ends in a different order. An example for DCJ operation is shown in Figure 2.7. Yancopoulos et al. [1] introduced DCJ through which the genomic distance can be computed in linear time. Bergeron et al. [51] gave a linear time algorithm to compute the genomic distance via DCJ for linear chromosomes.

These algorithms for calculating the DCJ distance are mainly for signed genomes. Calculating the DCJ distance for sorting unsigned genomes is NP-Complete [52]. Jiang et al. [52] devised a 1.5-approximation algorithm for sorting unsigned linear multichromosomal genomes by the DCJ operations.

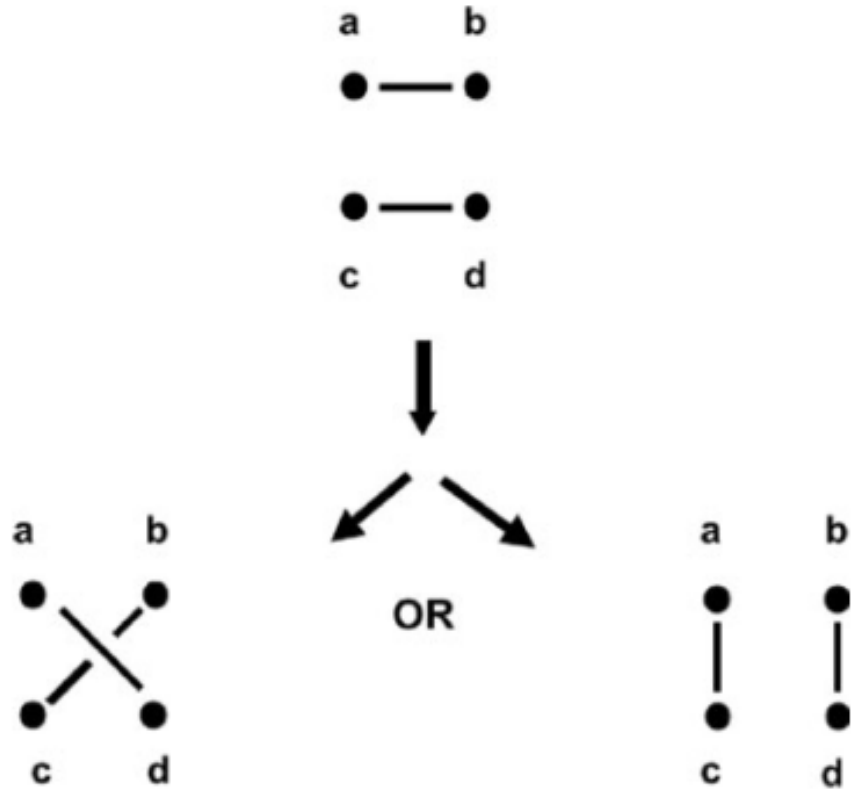


Figure 2.7: An example of general DCJ operation. Figure taken from Yancopolos et al.[1]

2.2.6 Sorting with Unequal Gene Content

All of these algorithms stated above assumed that each gene appears at most one time in each genome, that is there is no duplicate genes in the genomes. Calculating genome rearrangement distances in the presence of duplicated genes is NP-hard [14; 31; 16; 17; 15; 32]. As a result, sorting the genomes in this particular case also becomes NP-hard. Chen et al. [15] described the problem of computing the inversion distance for sorting genomes with duplicate genes and they gave a greedy heuristics. Shao et al. [53] studied the edit distance for sorting genomes with duplicate genes and gave

an approximation algorithm with an approximation ratio of $1.5 + \epsilon$. Later Shao et al. [32] studied the similar problem under the DCJ model and using an integer linear programming (ILP) approach they gave an exact algorithm for computing the DCJ distance for sorting genomes with duplicate genes. [20; 19; 54].

2.2.7 Gene Order Alignment

The gene order alignment problem is to find an alignment between the two gene orders that can be interpreted by a minimum number of evolutionary events (rearrangements and content-modifying operations). The presence of duplicate genes also makes this problem NP-Hard [33; 55; 56]. Benzaid et al. [55] introduced their [55] heuristic DAlign which is a dynamic programming approach for the Duplication-Loss Alignment of two genomes, is limited to only duplication and loss. Holloway et al. [33] presented an exact pseudo-boolean linear programming algorithm to search for the optimal alignment in the presence of duplicate genes but this is also limited to only duplication-loss model and the running time of this algorithm is exponential in the worst case. Tremblay-Savard et al. [2] introduced OrthoAlign, a polynomial time heuristic which considers both content modifying (duplications, losses) and rearrangement events (inversion) and their algorithm [2] is faster than the previously developed algorithms [33; 57].

As these problems are difficult computational tasks and algorithmically NP-Hard, through a citizen science approach we might have an idea about how the human brain can solve these problems and this might lead to a solution for solving these algorithmically hard problems.

Chapter 3

Citizen Science

Citizen science is the process where people are recruited to contribute in a scientific analysis [58; 59]. It is a collaboration of some scientific research projects with the public who are recruited to participate. Prestopnik et al. [60] describes it as a social computing system where human participants and computing technologies work together to produce some results which is could not be obtained by a single person. A citizen science approach can be useful when a large amount of data is needed for some research purpose. This citizen science approach is getting popular day by day among the researchers. However, there are some challenges remaining such as collecting quality data, recruiting participants and maintaining a large enough community of participants.

The concept of citizen science was developed in the 19th Century. Alan irwin [61] introduced the term *citizen science* in his work of finding out the significance of the participation of non expert people in social experiments. It indicated a new approach to incorporate more people in solving complex scientific problems or solving

social decision making issues. Later this approach was defined as a technique to collect scientific data to enhance scientific literacy [62]. According to the European Commission, *Green Paper* - citizen science is the process of engaging citizens with scientific research activities where people contribute to scientific research with their creative ideas and knowledge[63].

Citizen science actually engages people in scientific projects which are difficult to be done by the scientists themselves and to gather a large amount of data [64]. There are several types of citizen science projects which include monitoring wildlife [65; 66], image classification [67; 68; 69], interpreting old records [70; 67], games with a purpose (GWAP) [20; 54; 71; 22; 19] and so on. Based on the participants' involvement, the citizen science projects can divided into three categories: contributory (participants contributing in data collection), collaborative (citizens have participation in analyzing data) and co-created (citizens participating in every section of a project) [62; 67; 70]. Another classification was proposed by Wiggins et al. [59] based on the objective of the study: action projects (volunteer initiated projects for arising local concerns) [72], conservation projects (projects on how to manage natural resources) [73],virtual projects (online based scientific projects) [74] and educational projects (projects related to educational curriculum) [75].

Citizen science and crowdsourcing are becoming more and more popular for solving problems in molecular biology and bioinformatics. Protein structure prediction is such a problem. If the proteins are not in their native structure, they can not perform their function properly. Thus proteins fold into their original structure to perform their functionality. When a polypeptide chain folds into its native 3D structure to become

a biologically active protein, this event is called as protein folding. Cooper et al. [19] solved the problem of predicting protein structure by introducing Foldit, a multiplayer online game which is engaging online players to solve this hard prediction problem. With Foldit, players work collaboratively to develop new strategies and algorithms.

Multiple sequence alignment (MSA) is the problem of finding an alignment that maximizes the sum of similarities for all pairs of biological sequences in a pairwise alignment problem. Kawrykow et al. [20] developed a human computing game named Phylo which is solving the MSA problem. The main idea of Phylo is to convert the MSA problem into a color matching game and get the results. In their game they used DNA sequences which are represented by colored blocks, where one color is assigned to each of the four possible nucleotides. Then the players need to align the sequences by moving the blocks around and thus get a score on the resulting alignment. The target of Phylo here is to find the alignments maximizing matches and minimizing mismatches and gaps.

Improving the accuracy of RNA alignments is another challenging and NP-Complete problem [21]. Waldispühl et al. [21] introduced the human computing game Ribo which aims at improving the accuracy of the alignment of RNA sequences considering structural information.

Citizen science game can also help us in gathering quality data which was discussed by Loguercio et al. [76] in their citizen science game Dizeez. Dizeez is an online quiz game built with the purpose of structuring knowledge of gene-disease associations [76].

Other crowdsourcing platforms like Eyewire [77], Open Phylo [54], Galaxy Zoo

[71], EteRNA [22] are also solving different scientific problems and these are good examples of how citizen science approaches can help solving difficult computational problems.

Chapter 4

Effect of Timer, Top Score and Leaderboard on Performance and Motivation in a Citizen Science Game

4.1 Abstract

The development of citizen science games requires the implementation of different game mechanics to make them challenging, interesting and motivating. These mechanics are often borrowed from commercial video games, but their outcome on the quality of solutions obtained and player motivation are not fully understood. We analyze the effect of showing a timer, an achievable (top) score and a live leaderboard on players' scores, puzzle completion time and motivation using different versions of a citizen science game. We show that presenting a top score on a puzzle results in better solutions, but at the expense of completion time, whereas the presence of a timer

has the opposite outcome. As for the live leaderboard, we have observed an almost significant interaction effect with the timer. This work offers guidance for citizen science game developers about what to expect from these different game mechanics, and how they can promote certain player behaviors.

4.2 Introduction

Human computing (process of recruiting humans to help solve problems that are complex for computers), *citizen science* (recruitment of the general public to help collect and analyze scientific data) and *crowdsourcing* (practice of obtaining information from a large crowd of people, typically online) have developed rapidly in the last decade. Those practices are now being considered as valid options to solve various types of problems which can benefit from human abilities. Platforms such as Amazon Mechanical Turk [78] or Microworkers [79] have been created for the distribution of crowdsourcing tasks, where the tasks are completed by human "workers" in exchange of a monetary compensation. There is also an increasingly high number of human computing, citizen science and crowdsourcing games being deployed, which aim to solve problems from many differents areas, such as molecular biology [19; 20; 80], health [81], ecology [82; 83], neuroscience [84], astronomy [85; 86], quantum physics [87] and deep learning [88]. Participants playing these games generally do it for free, which can be an advantage for requesters who need to distribute a large amount of *human-intelligence tasks*. However, there is a major trade-off: the game needs to be interesting and motivating enough to attract and retain a large enough player-base, and finding the right balance between accurate data collection

and a fun gaming experience is an extremely difficult task [26].

One way to make human computing and citizen science games more captivating is for them to emulate commercial video games by incorporating several game mechanics, such as scoring systems, timers and leaderboards. However, it is not clear how efficient these game features are at motivating the players and if they can potentially interfere with the quality of the solutions produced. Research in human computing, citizen science and crowdsourcing games is now focusing more and more on the analysis of game features and game mechanics [25; 26; 27; 28; 29]. The objective is to learn more about which game mechanics work well in the context of these *games with a purpose*, and how to implement them successfully.

In this paper, we present a study of three game mechanics in the context of a citizen science game in genomics: a timer (setting a time limit on a puzzle), a top score (representing a score that is achievable), and a live leaderboard (points are rewarded for completing puzzles, and the player's progress is shown on the live leaderboard). The citizen science game asks players to transform one sequence of tokens (genes in this context) into another by applying a series of possible operations (inversions, duplications and deletions). The goal of the game is to find the shortest sequence of operations that can transform the sequence into the other.

Using different versions of the same citizen science game, we evaluate the effect of showing a timer, a top score and a live leaderboard on the players' scores, puzzle completion time and motivation. More specifically, we study how all these mechanics interact together by testing all the possible combinations of them being present or not. Moreover, we design a points system, which is linked to the live leaderboard,

that is different depending on the presence of the other game mechanics (top score or timer). When the top score is present, the points system rewards players for reaching or beating the top score, and when the timer is present, extra points are given for each remaining second on the timer after completing the puzzle. We evaluate if the implementation of different point systems for the leaderboard somehow increases the effect of the other two mechanics.

4.3 Related Work

Several recent studies have been devoted to the analysis of game mechanics in human computing games. Siu *et al.* [25] have investigated competitive and collaborative scoring mechanisms in a human computing game and found that competitive scoring is better at engaging players than collaborative scoring, while yielding results that are just as accurate. Siu and Riedl [26] have observed that offering a choice of rewards to players improves task completion and player experience. Another study on a *Super Mario Bros* based human computing game has shown that players in the collaborative multiplayer version produced better solutions than players in the competitive multiplayer version, although the latter was found to be more challenging [27]. Tremblay-Savard *et al.* [29] have studied how a market system can be used in a collaborative human computing game to promote cooperation, and how a skill and a challenge system can guide players into accomplishing actions that are beneficial to the collaborative system.

To the best of our knowledge, only Phylo [20] and FoldIt [30] have implemented a game mechanic that gives the player some idea of what an achievable score (or

number of moves) can be on any given level. In Phylo, players have to produce an alignment of sequences that is at least as good as a machine-computed alignment to proceed to the next level. This score that must be attained is called the "par" [20] and it is guaranteed to be achievable. However, the "par" score is not always a good representation of the best score that can be obtained by a player, which results in some puzzles being harder than others simply because they offer a "lower possiblility of improvement" [89]. The current version of Phylo also shows the top score [90], which corresponds to the best score achieved by a player on any specific level, although the authors have not studied the effect of presenting it to the players. FoldIt ran an online experiment on a three-star system for their tutorial levels [30]. Instead of being linked to the score of a folded protein, the star system was linked to the total number of moves used by the players to complete each level. Players would get three stars if they completed the level using an "ideal" number of moves, and lost stars for using additional moves. The authors found that the three-star system resulted in players taking more time per move, producing solutions with fewer moves and retrying the levels more often. However, they did not observe any significant benefit on the retention of players.

Timers are often used in cooperative human computing games, such as the ESP game [91], Peekaboom [92], TagATune [93] and Ask'nSeek [94], to assure that both cooperating players complete the levels in a reasonable time. Timers have also been used in the context of collaborative leisure, such as the Story Mashup game [95; 96], to make sure that "players cannot stall the game dynamics" [96]. In single-player human computing games, timers are used mostly as an attempt to make them more

interesting and challenging. For example, a timer was initially integrated in the first version of Phylo as one of the mechanisms designed to "increase the entertaining value of the game" [20], although it has been retired from the current online version [90]. Lomas *et al.* [97] have studied the effect of giving players different time limits to complete levels in an online *Battleship-style* game where players must estimate numbers on a number line to sink enemy ships. They observed that giving more time to the players increased success (*i.e.* accurate estimates) and engagement (measured as a combination of time spent in the game and number of levels attempted), but noted that the online nature of the experiment did not allow to gain much qualitative insight into why the players responded in this manner. Denisova and Cairns [98] explored the effect of dynamic difficulty adjustment on player experience and performance by manipulating the speed of a timer to make a shooting game more or less challenging depending on player performance. They observed that players in the dynamically-adjusted timer condition felt more immersed in the game and obtained scores that were less variable than the players in the control condition with a normal timer.

Leaderboards are one of the most common features of human computing games, as they are often used as a gamification technique to stimulate the more competitive participants and encourage them to perform better and eventually get to the top of the leaderboard [99; 100; 101]. Several studies have shown that different personality types (extroverted vs introverted for example) respond differently to leaderboards and have called for more carefully designed and personalized leaderboards [102; 103; 99]. Bowey *et al.* [104] have shown that manipulating a leaderboard to simulate success increases the player's perception of competence, autonomy, presence, enjoyment and positive

Figure 4.1: The game interface for the timer, top score and leaderboard condition. The interface can be separated into three panels: panel A (green box) is the live leaderboard, panel B (red box) is the player information panel, and panel C (yellow box) is the game panel.

affect. The authors have also noted that it would be interesting to see if it would be even more effective to display a leaderboard in real-time, instead of only presenting the leaderboard at the end of the game [104]. While in most human computing games the leaderboards are only accessible in a menu or shown after completing a level [105; 29; 19; 106; 84; 87; 107; 92; 108], there are relatively few examples of live leaderboards, or in other words, leaderboards that are continuously present on the screen (TagATune being one of them [93]).

4.4 System overview

4.4.1 Description of the citizen science game

The citizen science game that was used in this study is a puzzle game that was designed in our lab to solve a difficult problem in genomics: finding the shortest sequence of operations that can transform one ordered sequence of genes into another. The game is intended to be accessible to casual players. As such, the sequences of genes are simply represented as sequences of colored shapes (see Figure 4.1). We used six distinct shapes and three different colors (safe for color blindness) to represent the different genes in the sequences.

The goal of the game is to find the minimum number of operations that can transform the mutable sequence of colored shapes into the target sequence. Each move (i.e. operation) applied by the player increases the score by 1, and similarly to golf, the objective is to complete each puzzle with the minimum score.

The operations that are available to the players correspond to biologically relevant evolutionary events: duplications (copying selected colored shapes to another location), deletions (removing the selected colored shapes) and inversions (inverting the order of the selected colored shapes). There is an additional constraint on inversions events, which is motivated by biology: the inversions must occur around a "pivot" in the sequence of genes, represented as a black dot near the center of the sequence (this corresponds to the *terminus of replication* in bacterial genomes). In other words, this means that inversion operations must be applied to a selection of colored shapes that includes the black dot to be valid.

4.4.2 Implementation of the three game mechanics

Timer:

To assure that a reasonable time limit was set for each level, three lab members with varying degrees of experience with the game played all the levels of all datasets (see section 4.6.2 for a description of the datasets), and the average completion time was calculated for each level. The time limit on each level was set to the average time to which 45 to 90 seconds were added, depending on the level's difficulty (*i.e.* +45s for levels 1-2, +60s for levels 3-4, + 75s for level 5 and +90s for level 6). The timer starts counting down as soon as the level is loaded and if it is incomplete when all the time has expired, the participant is prompted to restart the level.

Top score:

The top score aims to represent the lowest score achieved by a player on the level. In this study, we set the top score to the number of events that were simulated to produce each level to which we added 1 (see section 4.6.2 for a description of the datasets). This was done to simulate a real top score, which would not necessarily be optimal, which leaves some room for the players to improve.

Leaderboard:

The leaderboard was populated with 9 simulated players in addition to the current player. Points were assigned to the simulated players in such a way that each real participant could potentially reach the top of the leaderboard during each game session. The point system for the leaderboard assigns 50 points for completing a

level, and additional points depending on the presence/absence of the timer and the top score. Table 4.8 presents the distribution of points for each of the possible game conditions.

Table 4.1: Distribution of points for each condition when the leaderboard is present. 'T' represents timer and 'TS' represents top score, and time represents the remaining time in seconds.

Conditions	**Calculation of points**
T:OFF , TS:OFF	50
T:ON , TS:OFF	$50 + 1 \times \text{time}$
T:OFF , TS:ON	$50 + \max\{50 \times (\text{TS - score} + 1), 0\}$
T:ON , TS:ON	$50 + 1 \times \text{time} + \max\{50 \times (\text{TS - score} + 1), 0\}$

4.4.3 Game interface

The game was built in Unity 3D using C#. As shown in Figure 4.1, the game interface can be divided into three parts: the live leaderboard (A), the player information panel (B) and the game panel (C).

A: Leaderboard

This panel contains a live leaderboard showing the standings of the players according

to their points. After completing each level, the players are rewarded with points and their ranking is updated. The top three positions have a gold, silver and bronze medal associated with them (similarly to the colors that were used in [104]) and the current position of the player is highlighted in light blue.

B: Player information panel

There are five components in the player information panel. On the left-hand side, there is a pause button. Upon clicking on the pause button, a pause menu will show up and from there the player can go to main menu, restart the level or resume the game. The current level number is displayed next to the pause button. The timer appears in the center. On the right-hand side of the screen, the current score (number of moves played on the level) and top score (under the "best" label) are displayed.

C: Game panel

In the game panel, the puzzle is shown at the top: the first row is the *target* sequence (not modifiable) and the second row is the *mutable* sequence. The players need to transform the mutable sequence to the target sequence by performing any of the three operations, which can be applied by first selecting colored shapes and clicking on one of the buttons shown below: dup (duplication), del (deletion), inv (inversion). Once the two rows of sequences are matched completely, the "Level completed" panel will appear and will prompt the players either to go to the next level or restart the current level to achieve a better score.

4.5 Research Questions

Considering the two main dependent variables, which are the score obtained by a player and the completion time for each level (puzzle), and the player motivation (as determined by qualitative analysis), we aim to answer these research questions:

RQ1: *What is the effect of setting a time-limit for each puzzle, using a timer?*

RQ2: *What is the effect of showing the top score?*

RQ3: *What is the effect of showing the live leaderboard coupled with a context-dependent point system?*

RQ4: *How will these three different game mechanics (timer, top score and live leaderboard + point system) interact together?*

4.6 Experiments

4.6.1 Experimental Design

We used a 2x3 mixed factorial design where we have two within-subjects factors (top score, leaderboard) and one between-subject factor (timer) which can all be either on or off. The timer was chosen to be a between-subject factor to avoid a potentially pervasive effect where players being introduced to a timer would think that they always need to complete levels as quickly as possible, even when it is not

there anymore. The dependent variables are the score obtained and the time required to complete each level.

24 participants were recruited and divided into two groups of 12 (one group with the timer present all the time, and the other without). After reading and signing the consent form, the players were asked to complete a short demographic questionnaire (with questions on age, gender, average game time per week, etc.).

Within each group of 12, the participants played all the different combinations (on/off) of the top score and live leaderboard. This resulted in four different conditions in total, which are presented below:

1. Top score on, leaderboard on.
2. Top score on, leaderboard off.
3. Top score off, leaderboard on.
4. Top score off, leaderboard off.

Each participant was playing the game for the first time, and completed an in-game tutorial before starting the experiment, which consisted of four game sessions (one for each condition). Each game session contained six different levels of increasing difficulty (see how the four distinct datasets of six levels were generated in the next subsection), with the first level being a training level allowing the players to get accustomed to the new game condition. The four conditions were played in different orders by the participants, following a Latin-square, to avoid sequence effects. Moreover, the four datasets were shuffled between participants to assure that the same dataset is not always associated with the same game condition.

After playing each condition, a short interview was conducted to get their opinion on the last game condition played. A longer interview was conducted at the very end of the experiment to get the player's overall opinion on the game and to ask which condition they found more engaging and fun. Players' performance in the game (time to complete a level and scores) was also recorded for all the players, for each level of each condition.

4.6.2 Generation of the datasets

We used synthetic datasets for this experiment, created using a level generator which simply creates an original sequence (the target) and modifies it by applying random events to obtain the mutable sequence. We created four distinct datasets (one for each game condition) consisting of 6 levels of increasing difficulty. To increase the difficulty, we generated longer sequences of colored shapes and simulated more events between the target and the mutable sequence. Each corresponding level in each of the four datasets were simulated using the same parameters, to assure a similar level of difficulty between datasets. We verified the differences between the datasets using an ANOVA and found no significant difference in scores and completion time between them.

4.6.3 Participant demographics

Out of the 24 participants, 16 were male and 8 were female. The average age was 25.8 and the average video gaming time was 5.7 hours/week. All the participants were university students and most of them were at the undergraduate level.

4.7 Results

4.7.1 Quantitative analysis

A mixed factorial ANOVA was conducted to check for main effects of the timer, the top score and the leaderboard on players' score and time to complete each level. We first present the results on the score dependent variable, and then the results for the completion time dependent variable.

Analysis on dependent variable: Score

The descriptive results for the score obtained on a level is shown in Table 4.9, and represented graphically in Figure 4.2. Recall that in this game the goal is to keep the score minimum, so lower scores correspond to better performance. Significant differences in mean scores between timer on and off (as determined by simple effects of timer from table 4.4) are shown in bold. Mean scores are always lower when the timer is off, with the lowest mean obtained in the first condition (top score and leaderboard both on). This can be explained by the fact that the top score gives players an idea of what an achivable score can be, and the combination of the point system with the leaderboard further encourages players to achieve a low score.

The highest mean score (7.47) occurred in the third condition, with the timer, when the top score is off but the leaderboard is on. This can be explained by the presence of the timer which was playing a role in the points calculation. Players trying to complete the level faster to get more points, coupled with the absence of the top score in this condition, probably encouraged the players to complete the levels as

quickly as possible without too much consideration for the score.

The presence of the timer and the absence of the top score seem to result in a greater variability in the scores obtained, as reflected by the higher standard deviations. There tends to be less variability in the scores when the timer is off and the top score is shown.

The mixed factorial ANOVA results for the main effects of the timer, the top score and the leaderboard, and their main interaction effects are shown in Table 4.3. The main effect for the timer yielded an F ratio of $F = 5.121$, $p<.05$, indicating a significant difference on the players' scores between showing a timer and not showing it. The main effect of the top score is also significant with an F ratio of $F = 4.843$, $p<.05$. For the leaderboard, we did not get a significant main effect on the score dependent variable. Interestingly, we observed an almost significant interaction effect of the leaderboard with the timer with an F ratio of $F = 3.27$, $p = .07$. We did not

Table 4.2: Descriptive results for score for all four conditions of two groups (Timer:ON and Timer :OFF) shown as mean (SD). 'TS' corresponds to top score and 'LB' corresponds to leaderboard.

Conditions	**Timer:ON**	**Timer:OFF**
1) TS:ON , LB:ON	**7.07(3.209)**	**5.77(1.934)**
2) TS:ON , LB:OFF	6.73(2.761)	6.20(1.920)
3) TS:OFF , LB:ON	**7.47(3.582)**	**6.15(2.791)**
4) TS:OFF , LB:OFF	7.17(3.206)	6.45(3.159)

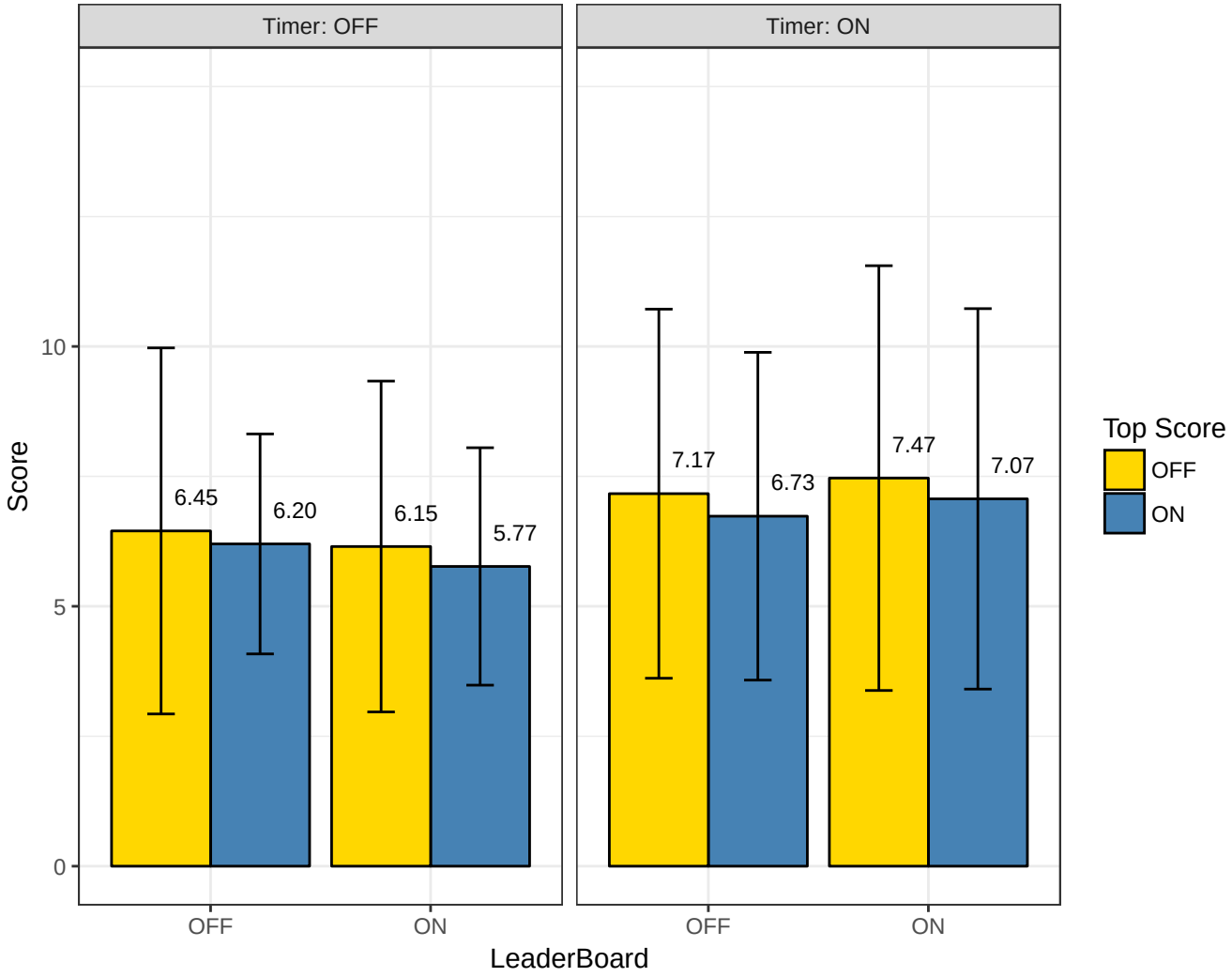


Figure 4.2: Mean scores and standard deviations according to the presence of all three independent variables (timer, top score and leaderboard).

find any other interaction effect.

Table 4.4 shows the simple effects of the timer with all four combinations of the other two within-subject factors. We can see that there is a significant simple effect of timer when both the top score and leaderboard is on ($F = 7.221$, $p < .05$) and also when the top score is off and the leaderboard is on ($F = 5.044$, $p < .05$). These simple effects, and by the same token the interaction between the timer and the leaderboard,

Table 4.3: Mixed factorial ANOVA results for the score dependent variable. The main effects of the timer (between-subject factor) , the top score (within-subject factor) and the leaderboard (within-subject factor), and their main interaction effects are shown. Significant codes (Sphericity assumed): p<.05 '*'.

Factors	**F (1,118)**	**p value**
Timer (Between)	5.121	**.025 ***
Top score (Within)	4.843	**.03 ***
Leaderboard (Within)	.08	.895
Top score * Timer	.90	.765
Leaderboard * Timer	3.27	.07
Top score * Leaderboard	.021	.886
Top score * Leaderboard * Timer	.058	.811

are clearly observable in Figure 4.3, as reflected by the larger differences in mean score between the timer on or off when the leaderboard is on. This tends to indicate that on one hand, players are effectively encouraged by the point system and leaderboard to complete the levels faster in the timer condition, which comes at the expense of their score on the levels, and on the other hand, they are encouraged to focus entirely on their score when the timer is not present.

Table 4.4: Simple effects of Timer with all four combinations of the other two factors (top score (TS) and leaderboard (LB)) shown. Significant codes : p<.05 '*'. and dv = score.

Conditions	**Conditions of Timer**	**F (1,118)**	**p value**
TS:ON, LB:ON	ON/OFF	7.221	**.008***
TS:ON, LB:OFF	ON/OFF	1.509	.222
TS:OFF, LB:ON	ON/OFF	5.044	**.027***
TS:OFF, LB:OFF	ON/OFF	1.521	.220

Table 4.5: Descriptive results for mean completion time (SD), for all four conditions of two groups (Timer:ON and Timer :OFF). 'TS' corresponds to top score and 'LB' corresponds to leaderboard.

Conditions	**Timer:ON**	**Timer:OFF**
1) TS:ON , LB:ON	**66.69(34.22)**	**104.46(57.85)**
2) TS:ON , LB:OFF	**71.38(34.16)**	**112.41(51.67)**
3) TS:OFF , LB:ON	**65.66(32.72)**	**95.05(68.23)**
4) TS:OFF , LB:OFF	**66.74(27.06)**	**95.22(65.94)**

Analysis on dependent variable: Time

The descriptive results for the level completion time is shown in Table 4.5 and also represented graphically in Figure 4.4. Significant differences between mean com-

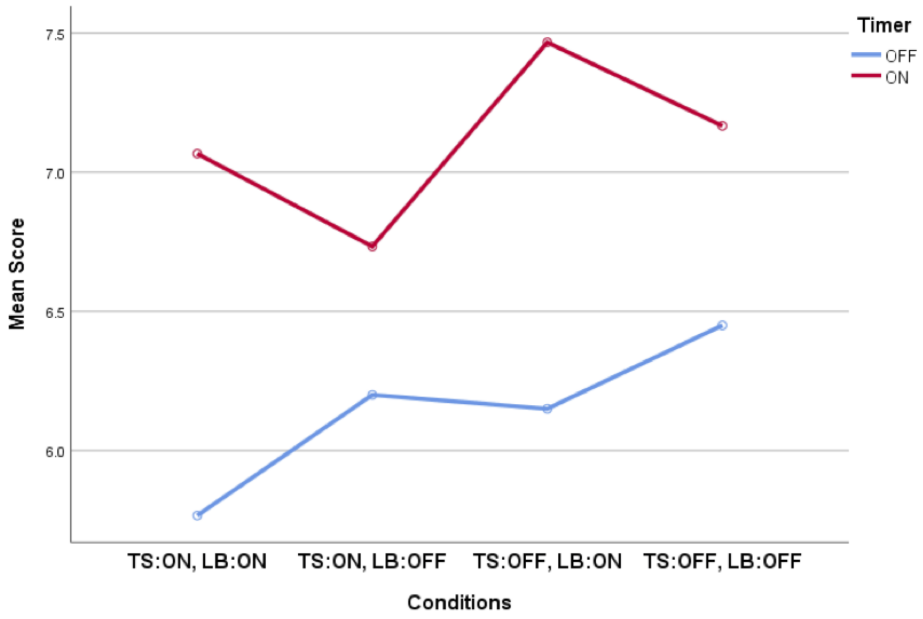


Figure 4.3: Interaction plot for the score dependent variable. Red line represents timer on, and blue line represents timer off. All four combinations of the two within-subject factors (top score (TS) and leaderboard (LB)) are shown on the x axis.

pletion times (as determined by simple effects of timer from table 4.6) are shown in bold. This clearly indicates that with the presence of the timer, the average completion times were significantly lower than without the timer. The variability of the completion times is also much higher when the timer is absent, as demonstrated by the standard deviations which are approximately twice as large compared with the ones of the timer condition.

Mean completion times are similar between all four possible conditions when the timer is present. However, the completions times are significantly higher when the top score is on and the timer is off, which indicates that players are taking more time in those conditions as they try to reach or beat the top score.

The mixed factorial ANOVA results for the main effects of the timer, the top score

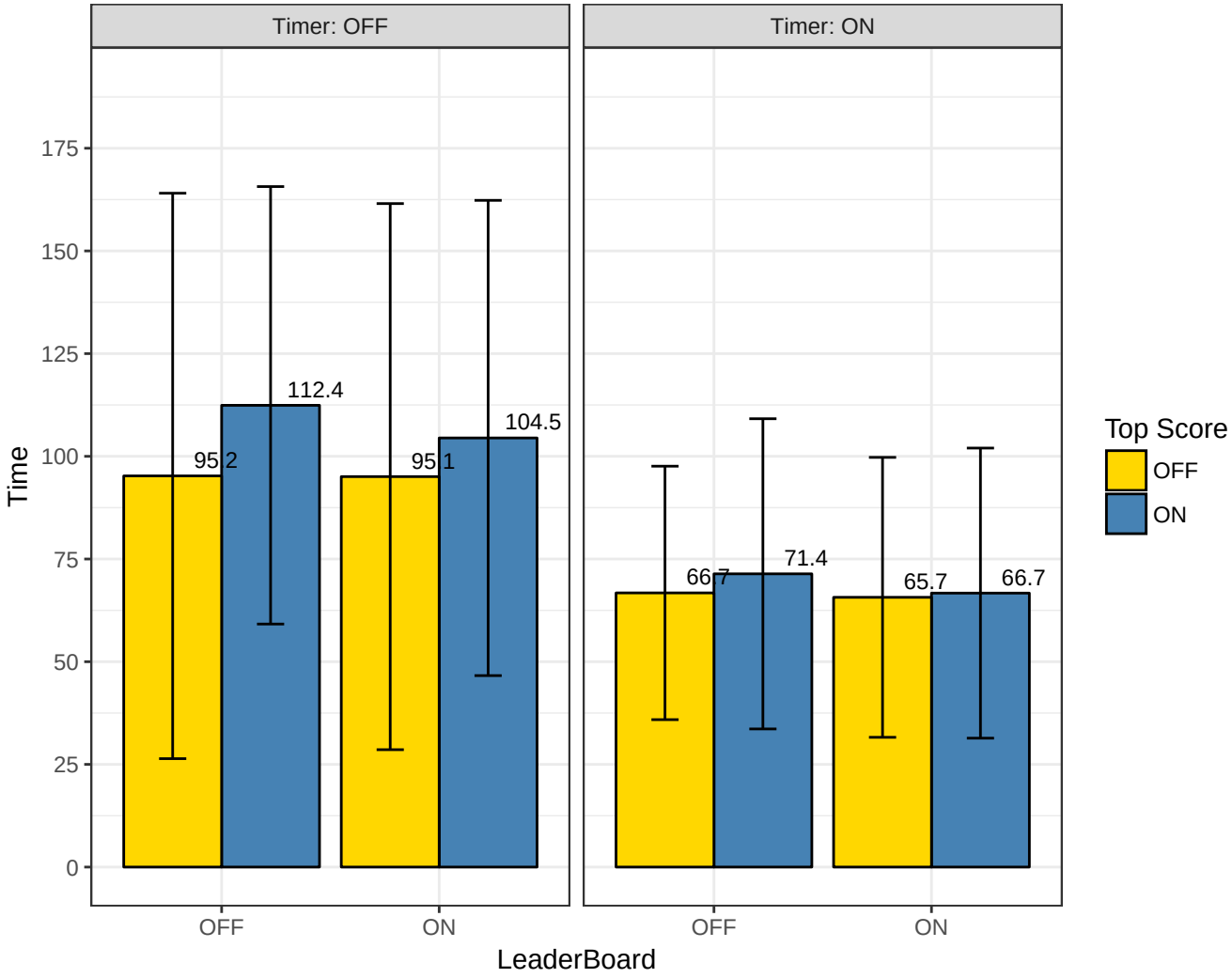


Figure 4.4: Mean level completion times (in seconds) and standard deviations according to the presence of all three conditions (timer, top score and leaderboard).

and the leaderboard, and their main interaction effect on mean level completion time is shown in Table 4.7. The main effect for the timer yielded an F ratio of F = 28.622, p<.05, indicating a significant difference between showing a timer and not showing it. The main effect of the top score is also significant with an F ratio of F = 5.869, p<.05. Similarly to the previous analysis for the score dependent variable, we did not obtain a significant main effect of the leaderboard on the level completion time.

Table 4.6: Simple effects of Timer with all four combinations of the other two factors (top score and leaderboard) shown. Significant codes : p<.05 '*'. Here dv = time.

Conditions	**Conditions of Timer**	**F (1,118)**	**p value**
TS:ON, LB:ON	ON/OFF	18.948	**.000***
TS:ON, LB:OFF	ON/OFF	26.319	**.000***
TS:OFF, LB:ON	ON/OFF	9.046	**.003***
TS:OFF, LB:OFF	ON/OFF	9.579	**.002***

We did not find any significant interaction effect between the different independent variables, which can also be observed in the interaction plot of Figure 4.5.

Table 4.6 describes the simple effects of the timer with all four combinations of the other two within subject factors. This result shows a significant simple effect (p<.05) of timer for all the four conditions of the top score and the leaderboard. Clearly, the level completion time is greatly influenced by the presence or absence of the timer.

Restart times We also looked into the restart times of the participants. We have found a significant effect on restart times between the two groups of the participants (92 times for Timer:ON and 19 times for Timer:OFF). There was no significant effect among the within factor conditions for restart times.

Table 4.7: Mixed factorial ANOVA results for the level completion time dependent variable. The main effects of timer (between-subject factor), top score (within-subject factor) and leaderboard (within-subject factor), and their interaction effects are shown. Significant codes (Sphericity assumed): p<.05 '*'.

Factors	**F (1,118)**	**p value**
Timer (Between)	28.622	**.000 ***
Top score (Within)	5.869	**.017 ***
Leaderboard (Within)	1.075	.302
Top score * Timer	2.481	.118
Leaderboard * Timer	.031	.861
Top score * Leaderboard	.483	.488
Top score * Leaderboard * Timer	.064	.800

4.7.2 Qualitative analysis

As mentioned earlier, each of the 24 participants was interviewed after each of their game session. In the interview sessions after each specific game session, the participants were asked:

- "What was the most interesting/fun part of your last game session?"

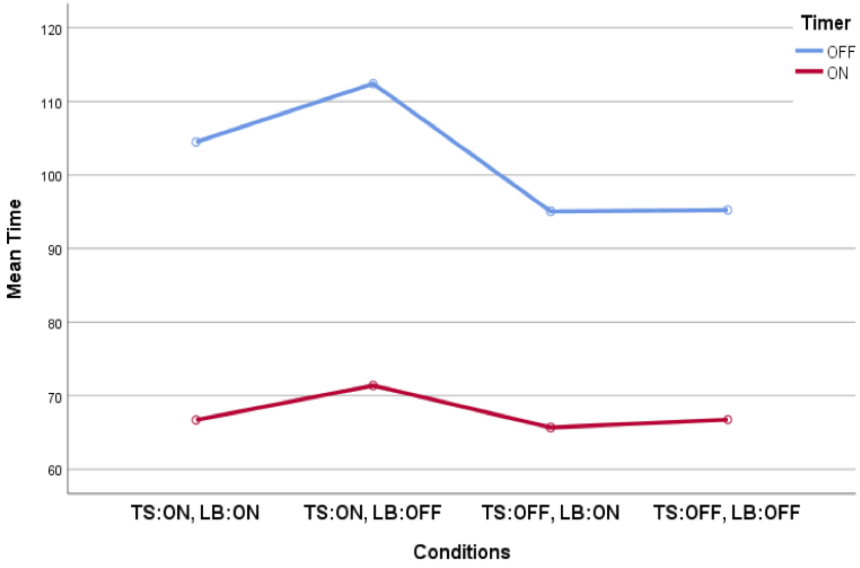


Figure 4.5: Interaction plot for the level completion time dependent variable. All four combinations of the two within-subject factors (top score (TS) and leaderboard (LB)) are shown on the x axis.

- "What was the most challenging part in your last game session? Why it was challenging?"
- "Was there something that you disliked about the last session?"

A longer interview was conducted at the end of experiment. During this final interview, participants were asked to compare the different game sessions, and were asked what they liked and disliked about each of the three game mechanics in the context of this game and in other games in general.

The following sections describe the different comments and views of the participants for each of the three game mechanics.

Top score:

Most participants mentioned that they were paying attention to the top score, and it forced them to think more about their moves.

"I paid more attention to the top score." (P6,P13,P17,P19)

"Top score has an effect; it forced me to think more." (P1,P2,P6,P23)

Most participants mentioned that seeing the top score made the game more challenging.

"The most challenging part was to beat the top score." (P1,P2, P3,P5,P14,P15,P17,P20,P24).

However, some players noted that they appreciated seeing it, because it gave them some directions and allowed them to know how well they were performing.

"The top score is challenging, but there are some directions." (P3)

"The top score is interesting, because you know the optimal solution." (P9,P13)

Some participants didn't appreciate losing the top score after playing a session where it was present.

"I missed the top score." (P15).

"I disliked not seeing the best possible score." (P19)

When we asked the participants about their general appreciation of top scores in other games, they mentioned both the positive and negative sides of it. On one hand, the majority of participants mentioned that seeing the top score makes the game more addictive, more challenging and more enjoyable.

"Any person would be addicted to try and beat the top score." (P24)

On the other hand, some players mentioned that it might cause some frustration too.

"In any game with the top score, you know there is always somebody who can do crazy things, which most people can't do." (P3)

"Up to some point it's okay if you can beat it, otherwise it's frustrating." (P14)

Participant 24, who mentioned how addictive the top score can be, also mentioned: *"I feel frustrated when I can't reach the top score" (P24)*

Leaderboard:

The participants enjoyed seeing the live leaderboard, and they mentioned it as their favorite game feature several times.

"Seeing the position, how you rank among people, the reward system is interesting." (P3,P4,P5,P7,P12,P14,P17,P20)

"I like to see my name going higher in the leaderboard." (P15,P21)

Some participants mentioned that climbing up in the leaderboard was challenging and motivating at the same time.

"The most challenging part was to climb up in the leaderboard." (P1)

"You try to do better to go to higher rank." (P3)

"I think I tried harder when the leaderboard was on." (P20)

A few participants didn't like the leaderboard, or didn't even pay attention to it.

"I didn't like the leaderboard." (P13)

"I didn't pay attention to the leaderboard" (P6)

Without the leaderboard the participants were unable to see their progress. When they were playing the game session without the leaderboard some mentioned that it felt like it was less interesting.

"No motivation because no leaderboard, no progress." (P4)

"I was trying to improve but I couldn't compare with others now" (P10)

But a few participants thought that there is no need of having a live leaderboard which you can see all the time. In some cases, it might even be distracting.

"It's good but you don't need it there all the time. " (P6)

"It disrupted my concentration." (P23)

Timer:

Participants who played with the presence of the timer felt challenged to complete the level within a given time limit and to avoid a force restart. It kept their completion time low, but at the expense of a higher number of moves as we've seen previously. Especially in the game session where there were both the top score and the leaderboard, it became more challenging for them.

"The top score and the timer was the most challenging part in my last game session (Which was with the timer, the top score and the leaderboard). (P2) "

"keeping pace with time left is hard, it actually requires time to think." (P4,P14)

We have seen in the interviews that some people appreciated the added challenge of the timer.

"Timer is good, its adds more challenge, makes it more interesting."(P16)

"It keeps you more focused." (P18)

"It's good, stressful in a good way." (P6)

"It's good, because you are thinking of different strategies in a short amount of time and without it I would get bored." (P24)

On the other side, we also have found some participants who did not like the timer. Some mentioned that the timer was adding extra pressure and forced them to restart the levels when they could not complete within the given time.

"I didn't like the timer, it added pressure." (P2, P12)

Some players pointed out that for them, a puzzle game is supposed to be relaxing, and the timer is creating the opposite effect.

"I like the idea of giving a time limit, but for a relaxation game you actually don't need a timer." (P4)

"I don't like a timer for puzzle game because it is meant for relaxation." (P20)

Favorite game session

All of the participants were asked "Out of the four game sessions which one was your favorite?" The results are presented in the pie chart of Figure 4.6. Recall that players were assigned to either the timer or no timer group, so they could only choose between the presence or absence of top scores and leaderboards. 46% of the participants preferred to have the conditions both with the leaderboard and the top score, followed by 31% for the top score without leaderboard and 23% for only the leaderboard. None of the participants voted for the version which was without the top score and the leaderboard. In other words, the majority of participants (46% + 31% = 77%) preferred to see the top score, either with or without the leaderboard.

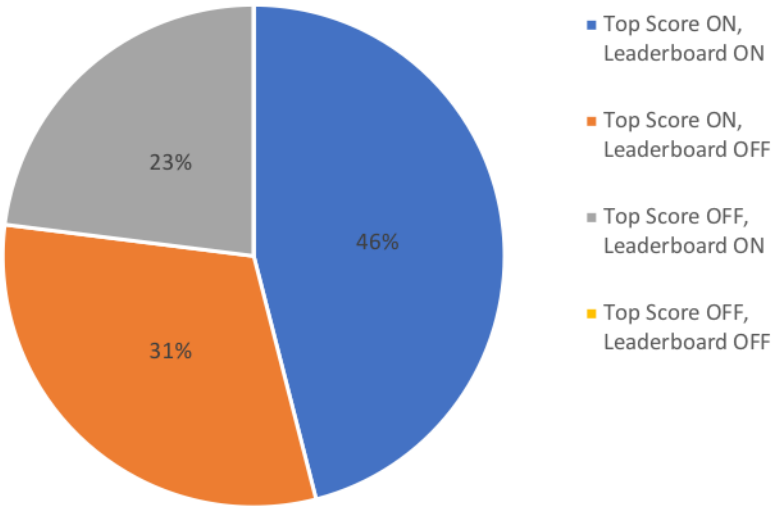


Figure 4.6: Conditions preferred by the participants

4.8 Discussion

We have shown that both the timer and the top score had significant main effects on the two dependent variables (score and completion time for each level).

The timer was making players complete the levels faster (although they had to restart the levels more often), but at the same time the scores obtained were worse on average. From a human computing point of view, the advantage of collecting results faster is mitigated by the poorer quality of the solutions produced. From our qualitative analysis, we obtained mixed opinions about having a timer or not having it. In some cases, we have seen that the timer can make a human computing game more engaging and challenging, which can promote player motivation and ultimately retention. However, others might want to play these puzzle games as a way to relax and the timer is clearly having the opposite effect of "adding pressure". Our results

suggest that timers should be used with caution in single-player citizen science games that require solving puzzles. A reasonable option could be to make timers optional, or having different games modes (one with a timer, another without) and let the players choose.

The effect of the top score on player performance was the complete opposite. The top score was effective at forcing the players to produce the best possible solution, but this came at the expense of the completion time. The qualitative analysis showed that the top score makes the game more challenging and enjoyable in most cases. It is effective at giving the players an idea of what an achievable score can be. As such, players make a better effort of finding the best moves. The only downside of the top score, which has been mentioned in the interviews, is the feeling of frustration that comes with the inability to reach or beat the top score. It is interesting to note that all of our top score were actually beatable by design, and it still created some frustration for the participants. One could imagine that in a real online game environment, where the top score represents the best score achieved by a player, this feature could potentially be even more frustrating if the top score becomes unbeatable (once the optimal solution is found). One solution for this would probably be to retire the puzzles that have reached "optimality" from the game (for example, after a puzzle hasn't been improved in some time, it should be retired). Overall, the top score was shown to be the favorite and most motivating feature of the participants, and it helped in collecting better solutions at the same time.

Although from our quantitative research we are observing that the leaderboard does not have a significant main effect, if we look at our qualitative analysis we can

also see that most of the participants enjoyed the presence of the leaderboard. They also mentioned the leaderboard as a source of motivation. However, for a puzzle game, a live leaderboard may work as a distraction too and some participants mentioned that it is not always necessary. Yet our dummy leaderboard does not reflect the wide range of possibilities of having a leaderboard in a citizen game, as will be discussed in the next section.

Interestingly, we did not observe significant interactions between the different mechanics, only an almost significant interaction between the timer and the leaderboard, reinforced by the point system.

4.9 Limitations

One of the limitations of this study is that we have used a dummy leaderboard with 9 simulated player scores. This leaderboard does not represent a real online game context, in which the top players would be unreachable. Also, since each experiment was completed in a lab environment in roughly 45 minutes, it is difficult to reproduce and analyze the benefits that a leaderboard could have on a long term experiment. Perhaps connecting the leaderboard with social media so that the players can play with their real friends could be more effective too.

Secondly, we did not test different locations for the game features on the screen. Locations of the game features help the players to focus on a particular point which could make a difference on their performance. A player mentioned for example that the leaderboard was distracting, so maybe it was taking too much space on the screen. A future study could investigate the proper sizes and locations for showing the leader-

board, the timer and the top score.

Thirdly, for the timer we only tried with a fixed time limit for every participants. We might get different results if we used many different versions of the time limit (in one version we could use a shorter time limit, where in the other version we could try by giving the players a longer time limit).

4.10 Conclusion

The goal of this study was to study the effect of different game mechanics (the timer, the top score and the leaderboard) on players performance and motivation in a human computing game. We chose these three mechanics because these are the most common features of many human computing games. From our results we observed that the timer and the top score had significant effects on completion time and scores. The top score results in better solutions, but at the expense of the completion time, and the timer has the opposite outcome. The leaderboard wasn't significantly affecting the scores and completion time, but we did observe an almost significant interaction effect with the timer on the score (the scores being worst when the timer and leaderboard are present). Qualitative results have shown that showing a top score was the most appreciated and motivating feature.

4.11 Supplementary Materials

4.11.1 Anova Result on Datasets

As each of our participants played four conditions, so we had different datasets for each of the conditions. The difficult level of these datasets were equal. To test these, we have conducted an anova analysis which is shown in table 1 and table 2. We can see from the results that, these datasets (A,B,C,D) have no significant effects over the dependent variables score and time. So we can say these datasets are of equal difficulty.

Factors	**F value**	**p value**
Datasets (A,B,C,D)	0.292	0.831

Table 4.8: Results from Anova on Datasets: p<.05 '*'. dv = score.

Factors	**F value**	**p value**
Datasets (A,B,C,D)	0.063	0.979

Table 4.9: Results from Anova on Datasets: p<.05 '*'. dv = time.

4.11.2 Normalized Difference with top score

Figure 4.7 describes the normalized difference of score (difference of score between the player's score and the top score for each level / the top score for each level). Clearly the condition with the top score on, the leaderboard on and the timer off wins here. And this difference is always lower when the top score is on and the timer is off. Again with the presence of the timer this difference is going higher because the players tried to finish their level first to avoid a force restart in the expense of a higher number of operations.

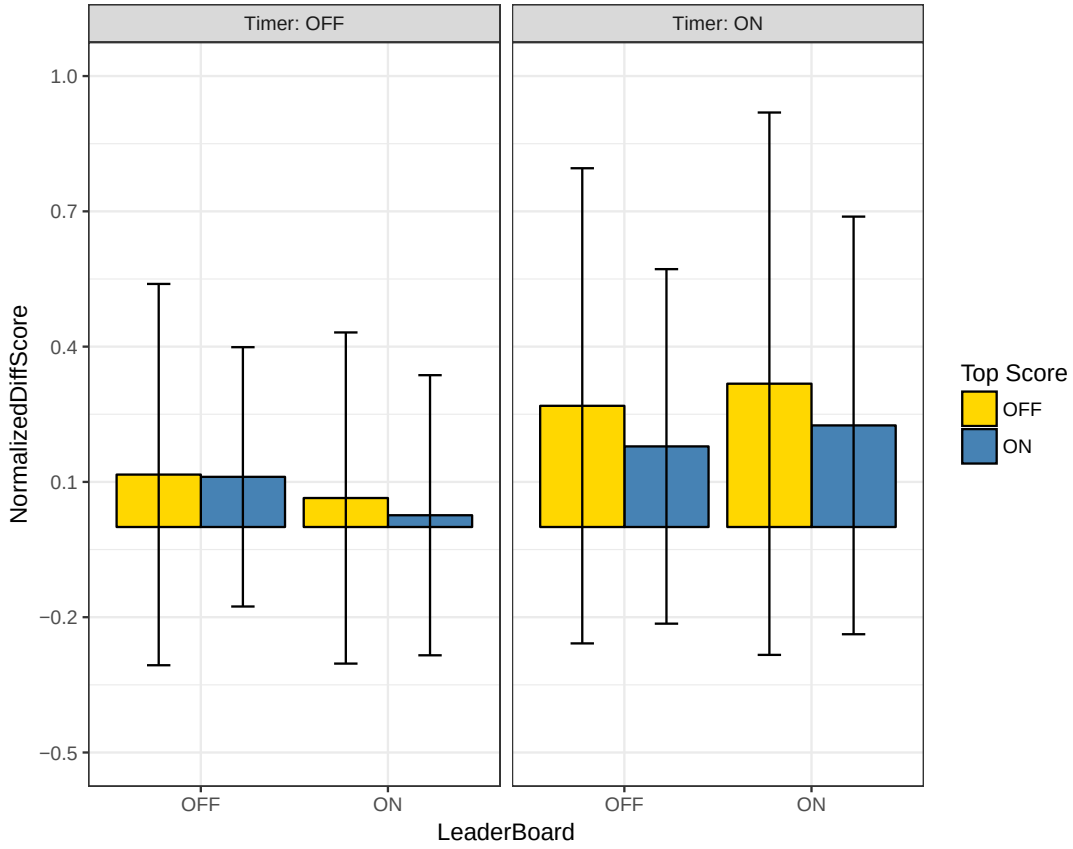


Figure 4.7: Mean normalized scores (normalized difference between the top score and player's score) and standard deviations according to the presence of all three independent variables (timer, topscore and leaderboard).

4.11.3 Normalized Difference with Target Time

Similar like Figure 4.7, Figure 4.8 describes the normalized difference of time (difference of time between the player's completion time and the target time for each level / the target time for each level). As we had two groups (one group played with the timer, another group played without the timer), the participants who played with the timer were faster than the other group who played without the timer. And the time difference is always higher in the absence of the top score. Because the players were thinking of getting the optimal solution when the top score is there and resulted in a higher completion time. From Figure 4.8, we are also seeing, the error is bar is higher when the timer is off. This is because, when the timer is off, participants who were good took less time than the other for thinking about the minimal moves.

4.11.4 Timer × Leaderboard × Score interaction on Score

The interaction graph of Timer × top score × Leaderboard on dependent variable Score is shown in Figure 4.9.

For the timer, we can see from the graph that the score is always higher when the timer is on. But when the timer is off, it's going lower. Again the score is always lower when the top score is on. But the most interesting thing we can see, the score is going lower even when the leaderboard is on and the timer is off but it's going higher when the timer is on and the leaderboard is on. The reason is, in this condition the points for the leaderboard was calculated based on top score and target time. So the players tried to maintain both the score and time when the leaderboard, the top score and the timer all are on. And this resulted in the score going a bit higher. Thus we

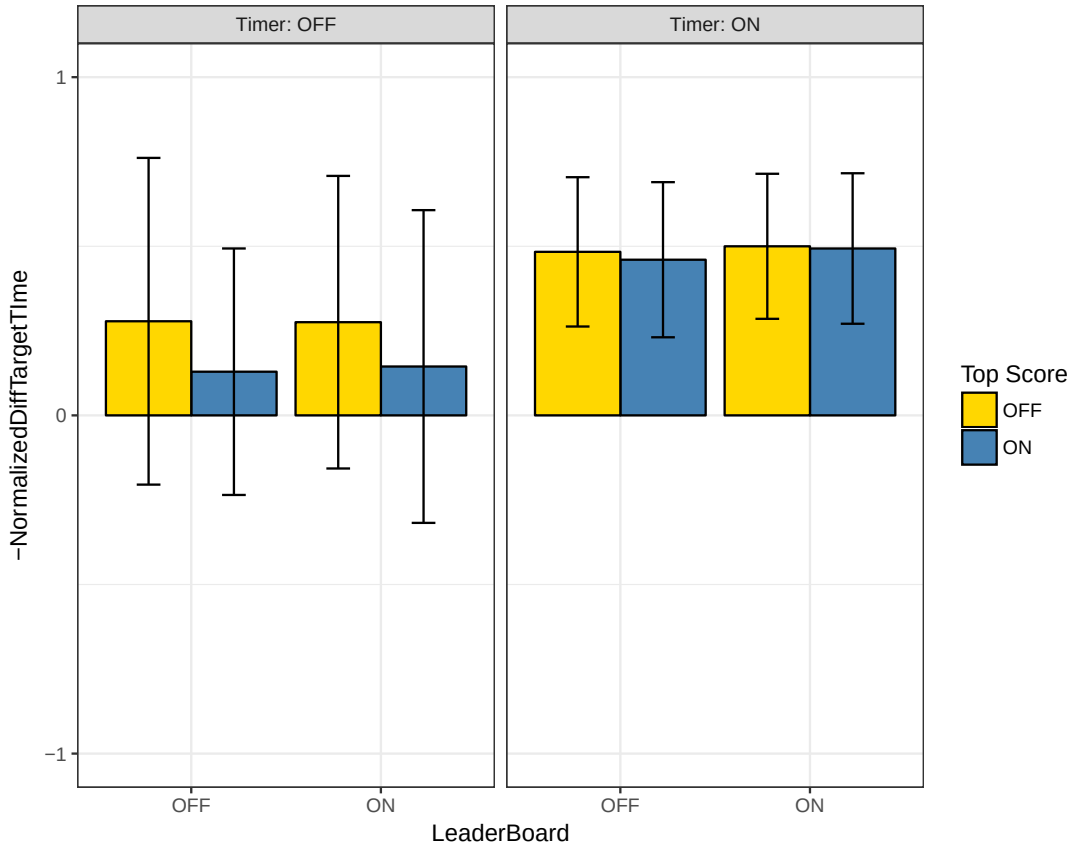


Figure 4.8: Mean normalized time (normalized difference between the target time and player's completion time) and standard deviations according to the presence of all three independent variables (timer, topscore and leaderboard).

can observe at a certain degree some effect of the leaderboard (and point system).

4.11.5 Timer × Leaderboard × Score interaction on Time

The interaction graph of Timer × top score × Leaderboard on dependent variable Time is shown in figure 4. There is clearly a visible effect of the timer on time. The participants with the timer, took much less time than the participants without the timer. When the top score is on, the participants were taking a bit more time then when the top score is off. A reason for that is the participants were thinking a bit

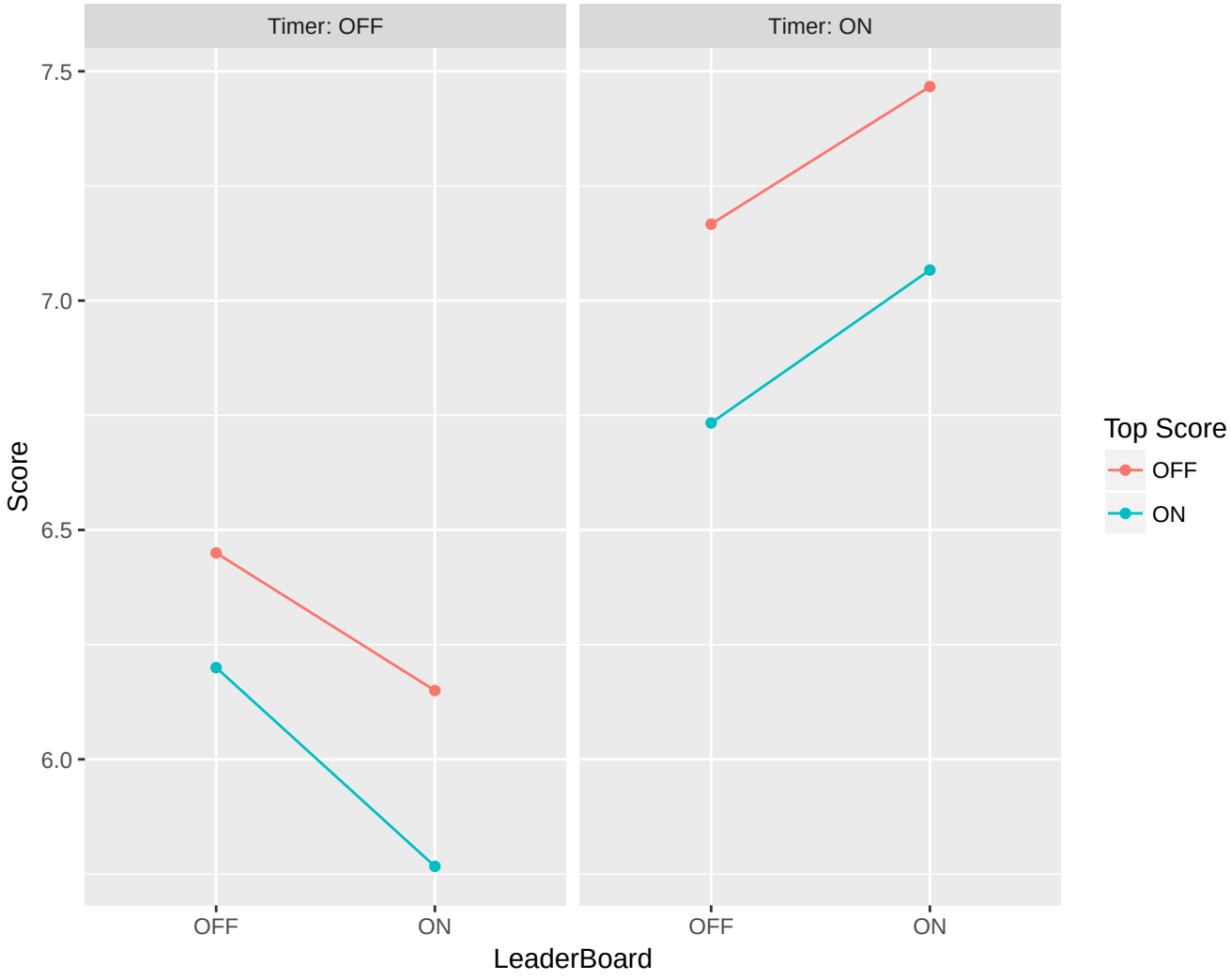


Figure 4.9: Interaction graph of Timer × Leaderboard × Score where dv=Score

more when trying to reach the top score. Here again, we can see a slight effect of the leaderboard (and point system). As time was a factor to calculate the points for the participants who played with the timer, so when the leaderboard is on, the time is decreasing even when the top score is on.

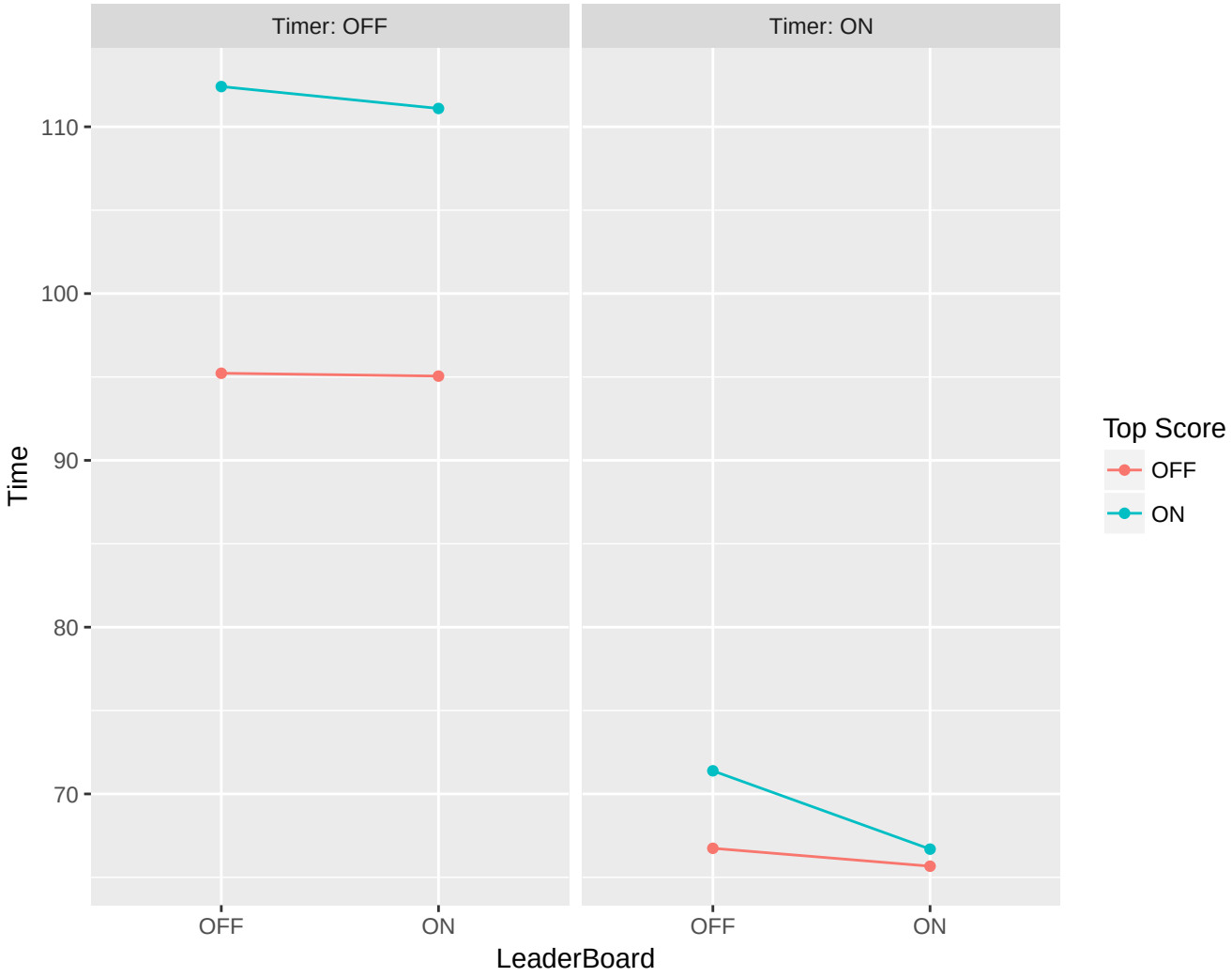


Figure 4.10: Interaction graph of Timer × Leaderboard × Score where dv=Time

4.11.6 Who are good players?

From our questionnaires, we collected the information about how many hours per week the participants are spending time in playing video games, how they evaluate their strength in puzzle solving games etc. These data along with the quantitative results we have obtained from them are showing that participants who are playing

video games more hours in a week tended to perform better in this game. The average total score of those who play games more than 10 hours/week was 117.25, whereas for those who play less than 10 hours/week it was 135.6. Figure 4.11 is showing how this game time is affecting their score with a correlation value r = -0.67. This regression model with $r^2 = 0.45$ shows us 45% of the variation in score can be explained by a linear relationship with the the players game time (hours/week). And comparing to the r value what was described in [29] we can say score has a moderate correlation with game time of participants.

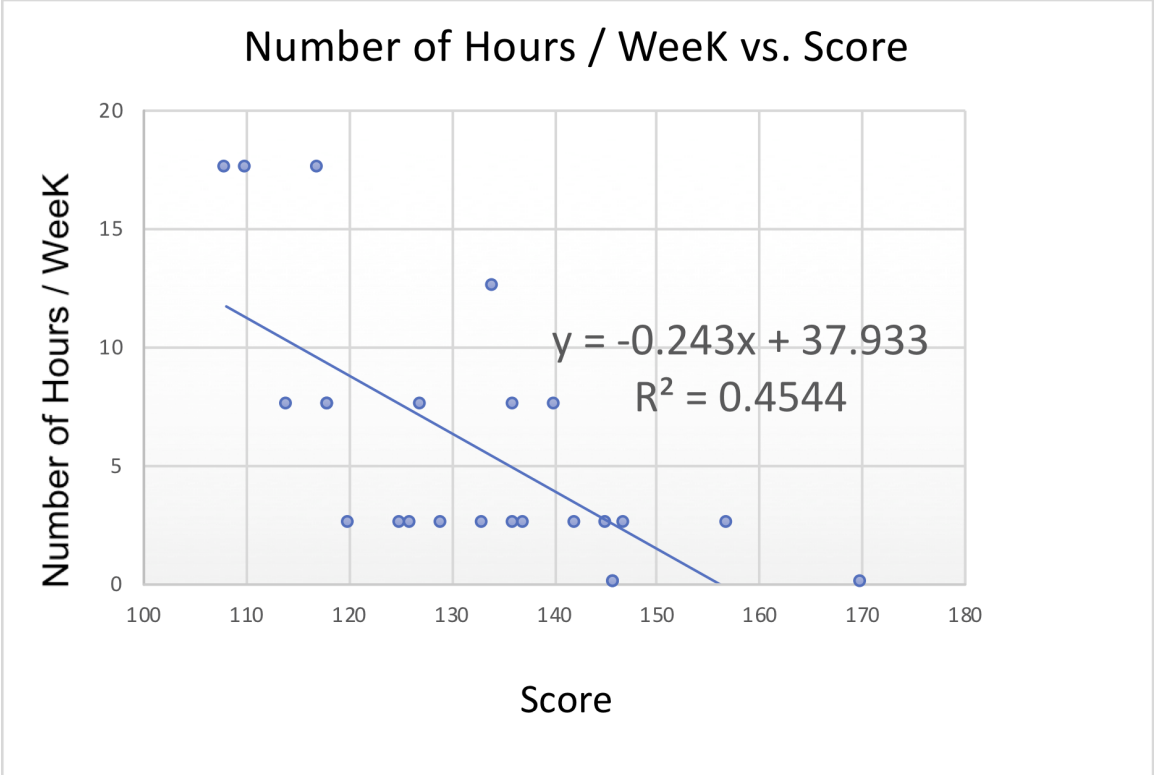


Figure 4.11: Correlation of game time (hours/week) vs. Score

Then we tried to find the correlation between the score of the players and their puzzle solving skill. Figure 6 shows us that there is also a moderate correlation with

a $r^2 = 0.47$ (r = -0.69) between the self evaluation of the puzzle solving skills of the players and the score.

We also tried to find out the correlation in age vs. score, age vs. time, number of hours /week vs. time, puzzle solving skill vs. completion time but we did not find any strong correlation in them.

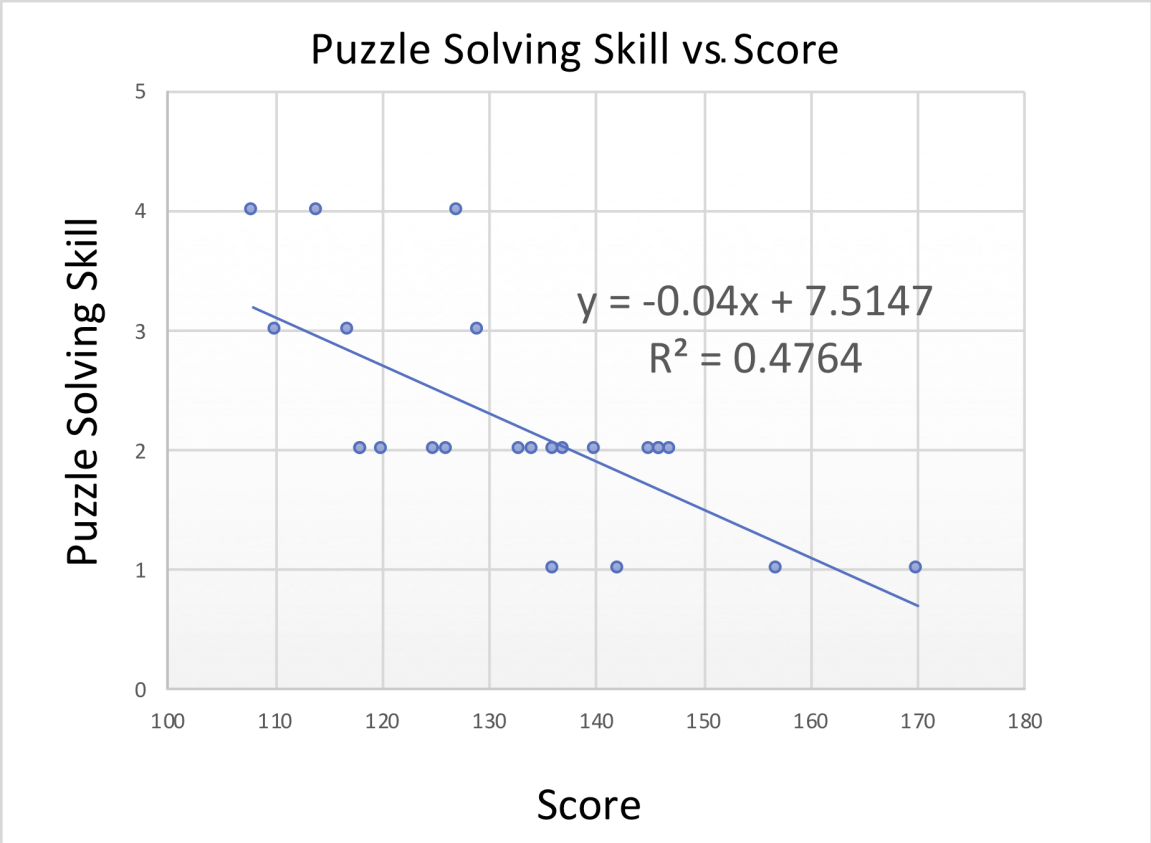


Figure 4.12: Correlation of Puzzle Solving Skill vs. Score

Chapter 5

GeSort: A citizen science approach for sorting genomes

5.1 Genome Sorting Problem

In the last decades the genome sorting problem has received a lot of attention from the comparative genomics community. Given two genomes (represented by gene orders), the goal is to find the shortest sequence of biological events to transform one genome into the other. When both genomes have exactly one copy of each gene, the problem is simpler. For example, in this context (one copy of each gene), sorting genomes by reversals (inversions of segments of genes) can be solved in polynomial time [12; 13]. However, it is often the case that genomes have multiple copies of certain genes. In this case, the genome sorting problem becomes NP-hard in the case of sorting by reversals[18], or sorting by reversals and duplications [15] for example.

5.2 Objective

As a first step towards the development of new algorithmic methods to study genome evolution in highly divergent genomes with duplicate genes, our goal is to develop a new crowdsourcing and human computing game that will ask players to find optimal evolutionary scenarios transforming one genome into another. Our game takes the form of a puzzle game and will be targeting both casual players and biology students, similarly to Phylo [20]. In addition to obtaining optimal evolutionary scenarios from the players, we will record every move that the players make. This will allow us to analyze the strategies employed by the players to solve those problems, and this information will then be used as inspiration to develop new algorithmic methods.

5.3 Overview of the Citizen Science Game

5.3.1 Game overview

The goal of our citizen science game "GeSort" is to find out the minimum number of events needed to transform one sequence of ordered genes to another using some biological operations. These sequences of genes are simply represented by colored shapes in the game (see Figure 5.1).We used six distinct shapes and four different colors (safe for color blindness) to represent the different genes in the sequences. Thus the goal for the players is to find out the minimum number of events to transform one row of colored shapes into another. Each move (i.e. operation) applied by the player increases the score by 1, and similarly to golf, the objective is to complete each

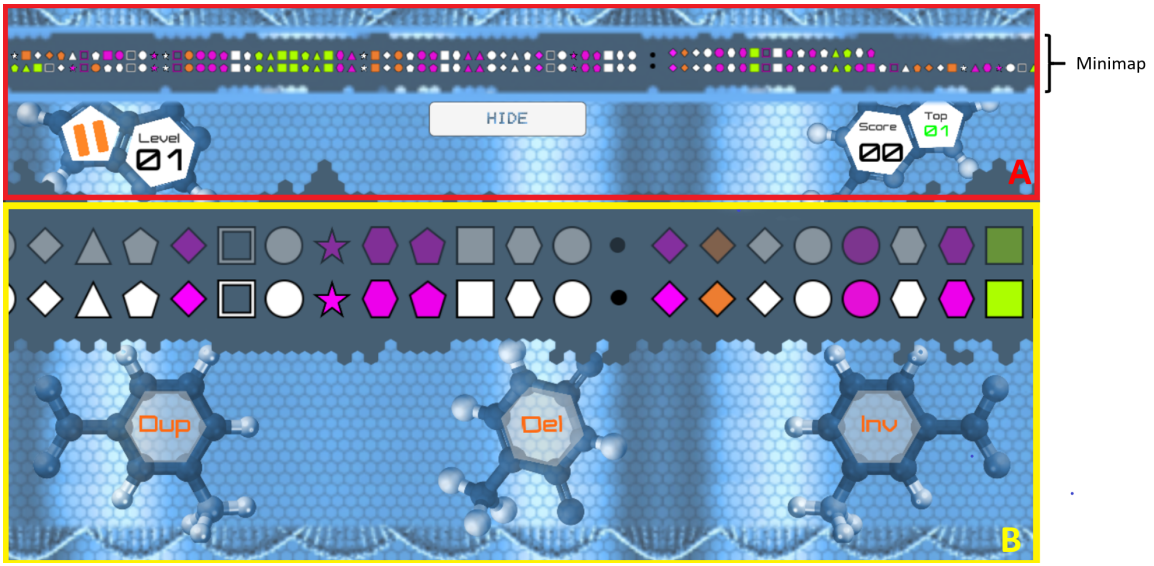


Figure 5.1: Game interface of *GeSort*.

puzzle with the minimum score.

Similarly to the game version that was presented in chapter 4, this main version considers the same the biological events: duplications (copying selected colored shapes to another location), deletions (removing the selected colored shapes) and inversions (inverting the order of the selected colored shapes). Inversion events occur mostly around the terminus of replication, which is located approximately in the middle of the genome. Consequently, only inversion events of segments of genes that are overlapping the terminus (represented by a black dot in the game) will be considered as valid moves.

5.3.2 Interface

Figure 5.1 shows the interface of *GeSort*. Unlike the version that was presented in the chapter 4, this main version of the game has only two sections: the information

panel (A) and the game panel (B).

A: Information Panel

There are five components in the information panel. On the left hand side we have the pause button and the current level number. If we press this button the game will be paused and a pop-up menu will show up. From there we can resume the game again or go back to main menu. In the middle we have the button for showing the minimap. Upon pressing on this one, the players can see a minimap of the whole puzzle on the top of the screen which is shown in Figure 5.1. This minimap helps the player to navigate through the puzzle when it is big. They can move the blocks and go to any place in the puzzle by clicking on a position in the minimap. They can also hide it if they don't need the minimap. On the right hand side of the panel we have the score (the current score of the player for the current puzzle) and the top score (the top score made by any player for this particular puzzle).

B: Game Panel

The game panel is similar to the version we presented in chapter 4. The only difference is in this version we have four different colors instead of three to present the genes. So there are two rows of colored shapes. The first one is the target one and the second one is the mutable sequence, which can be modified by the player. All the buttons for the three operations (duplications, deletions and inversions) are at the bottom of the screen.

5.3.3 Interactive tutorial

GeSort also has an interactive tutorial. This tutorial is added for a better un-

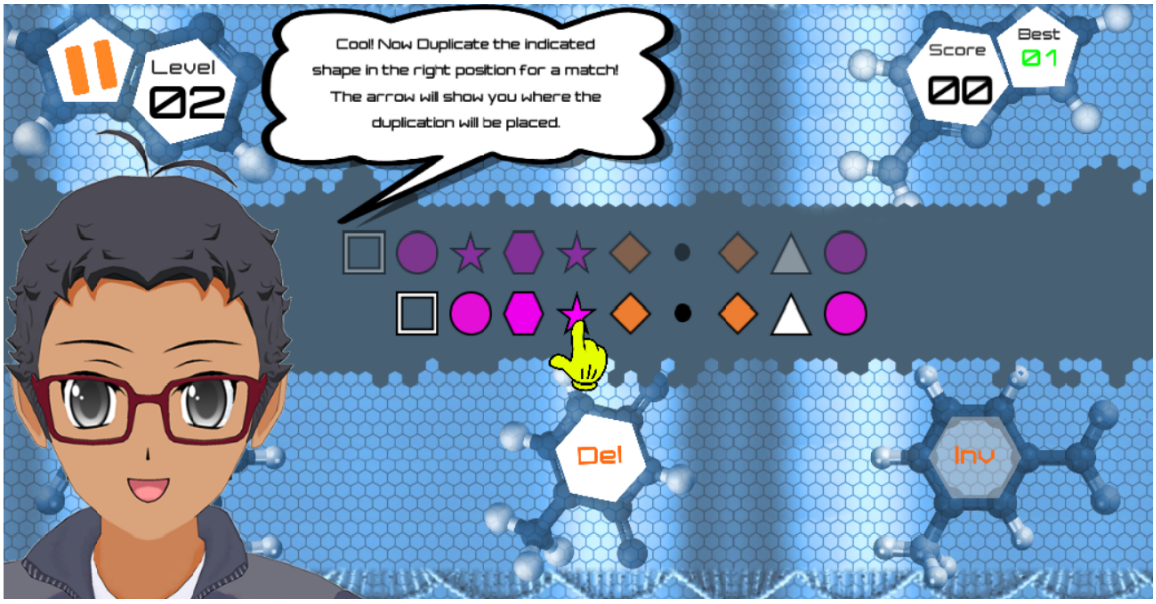


Figure 5.2: *GeSort* tutorial.

derstanding of the game. A screenshot of the tutorial is shown in Figure 5.2. With this interactive tutorial the players can understand how to play this game, how to perform each of the operations and how to complete a puzzle with fewer moves.

The players can solve the puzzle in different ways in this game but as the goal of the game is to make the fewest number of moves for solving a puzzle, it is necessary for the players to have the knowledge about how they can solve the puzzle with fewer moves. This tutorial thus helps the players to understand all the basics of the game before going to the main game session. The players can play the tutorial as many times as they want.

5.4 Data processing

5.4.1 Generating Synthetic Datasets

For the first few levels of this game we have used a synthetic dataset. We created a level generator which simply creates an original sequence (the target) and modifies it by applying random events to obtain the mutable sequence. These levels are used to train the players at the beginning of the game. It starts with small sequences and gradually increases the number of genes in the sequences. As the sequences in the real datasets are bigger, it would be confusing and discouraging for the players to immediately start with big puzzles. Thus this synthetic datasets help the player to familiarize them with puzzles that are gradually getting bigger before getting to the real datasets in the puzzle.

For showing the top score for each level, we have used the original number of moves from the level generator and adds 2 with it. We added this to motivate the players so that they can beat the initial top score and find the game interesting. However when a players makes fewer moves than this, it will be getting updated immediately.

5.4.2 Preparing Real Datasets

For now we have used *Bacillus* genome data for creating the puzzles of this game. We have followed a phylogenetic tree to make the pairs of the genomes. The phylogenetic tree we have followed is shown in Figure 5.3. We are interested in sorting genomes that are close in the tree, i.e. the genomes that form "cherries" (two leaves connected to the same internal node). We first took all the cherries which are con-

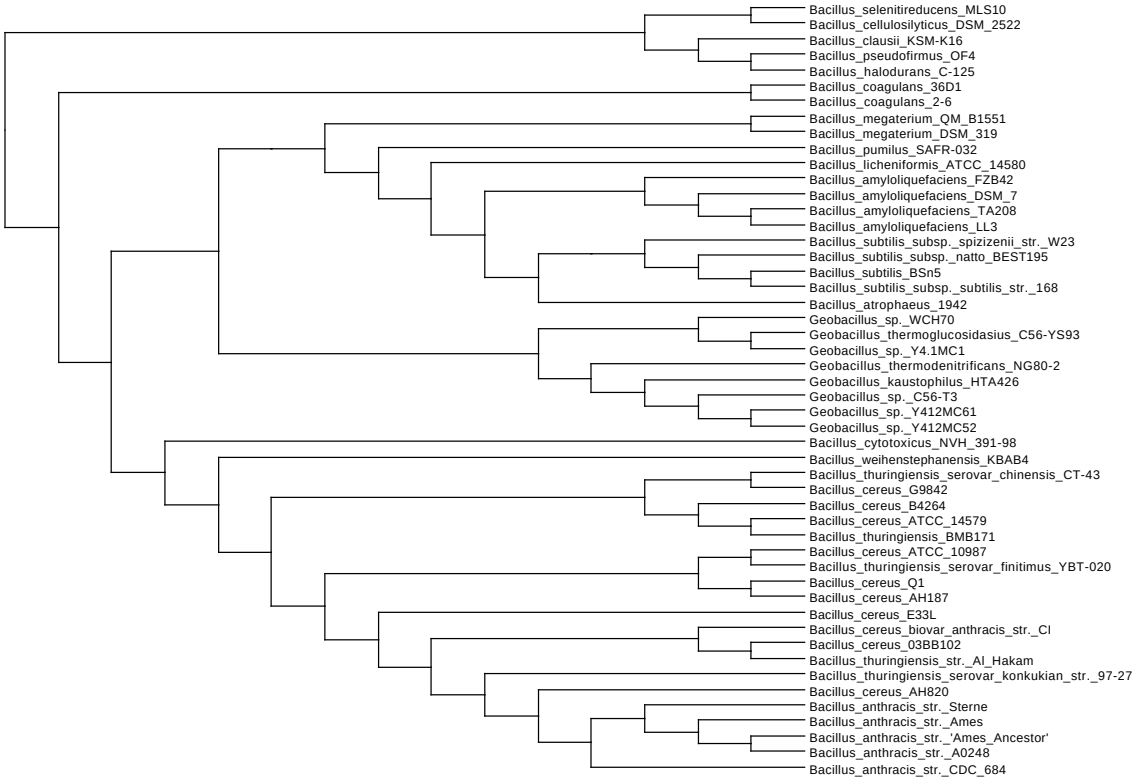


Figure 5.3: Phylogenetic tree of the 50 studied *Bacillus* strains [2]

nected together as a pair. Then we took the cherries which are connected with the most recent common ancestor of another two cherries and then moved to the ancestors accordingly. The datasets are stored in a text file. Pairs of genomes constitute puzzles, and each genome is represented by two rows. The first row is a header and contains information about the top score, the genome name and a label that identifies it as a target or a mutable genome. The second row consists of the full sequence of genes representing the genome.

Before pushing this data into the database we ran OrthoAlign[2] for each of the pairs to find out the top score (lowest number of operations required) according to

OrthoAlign. Then we also added 2 to this value and used this to set the initial top score of each puzzle. The reason for adding the 2 is to motivate the players so that they can beat the initial top score. This top score keeps being updated when a player completes the puzzle with fewer moves. When a player starts the game, it searches for the lowest number of moves for the puzzle loaded and shows it to the player.

5.5 Methods

5.5.1 Puzzle distribution system

Figure 5.4 shows an overview of the puzzle creating system of GeSort. As we calculated the top score primarily by OrthoAlign[2], we organized the dataset in a ascending order according to the number of moves, since these numbers also define the difficulty for solving a puzzle. Then we divide the dataset into five difficulty levels (very easy, easy, medium, hard and very hard). Once a player starts for the first time, the level is loaded from the very easy section. Depending on their progress, the difficulty level is dynamically changed to preserve the same level of challenge. If the player keeps scoring better then this system will keep loading from the higher difficulty level section. On the other hand, if the player makes more moves than the tolerance level (tolerance level is 30% more moves than the top score) it will go down to the lower difficulty level.

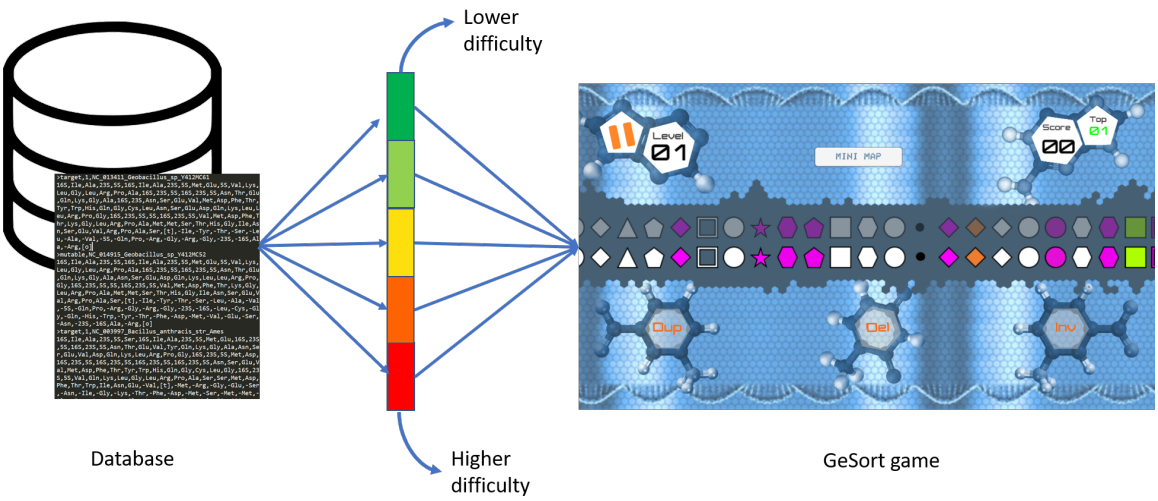


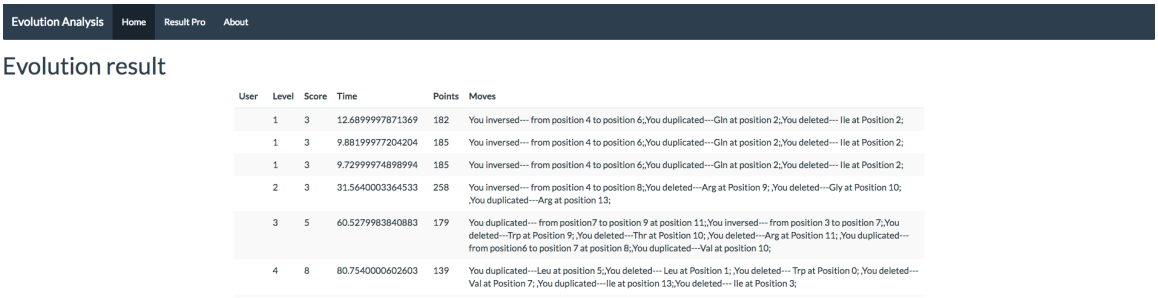
Figure 5.4: Creating puzzles from *Bacillus* genome data

5.5.2 Storing data

When the players start playing this game as a guest or with their login credentials, we keep storing the data. We store their login credentials, the level they are playing, the genome names, score, completion time and the moves they made to solve the puzzle. Here in the puzzle each gene in the genomes is represented by an unique shape. When we are entering results into the database, we are converting each shape back to its original gene name. For storing the data we have used a MongoDB server.

5.5.3 Analyzing data

We have a different website other than the website for the players where all the data is visible to us (evolgame.cs.umanitoba.ca:4000). Figure 5.5 shows the website where we can see the results of the players. We can analyze the data also by genome name. We can see the evolution of the scores for each puzzle over time. After getting



User	Level	Score	Time	Points	Moves
	1	3	12.6899997871369	182	You inversed--- from position 4 to position 6;,You duplicated---Gln at position 2;,You deleted--- Ile at Position 2;
	1	3	9.88199977204204	185	You inversed--- from position 4 to position 6;,You duplicated---Gln at position 2;,You deleted--- Ile at Position 2;
	1	3	9.72999974898994	185	You inversed--- from position 4 to position 6;,You duplicated---Gln at position 2;,You deleted--- Ile at Position 2;
	2	3	31.5640003364533	258	You inversed--- from position 4 to position 8;,You deleted---Arg at Position 9;, You deleted---Gly at Position 10; ,You duplicated---Arg at position 13;
	3	5	60.5279983840883	179	You duplicated--- from position7 to position 9 at position 11;,You inversed--- from position 3 to position 7;,You deleted---Trp at Position 9;, You deleted---Thr at Position 10;, You deleted---Arg at Position 11;, You duplicated--- from position6 to position 7 at position 8;,You duplicated---Val at position 10;
	4	8	80.7540000602603	139	You duplicated---Leu at position 5;,You deleted--- Leu at Position 1;, You deleted--- Trp at Position 0;, You deleted--- Val at Position 7;, You duplicated---Ile at position 13;,You deleted--- Ile at Position 3;

Figure 5.5: GeSort Result Analysis

a reasonable amount of results for each pair of genomes then we will compare players' scores with the score from OrthoAlign [2] and DupLoss [33] for these particular pairs of genomes. We will also compare the events that are inferred to better understand if and how the players were able to get similar or better evolutionary scenarios.

5.6 Evaluation

To solve the genome sorting problem we need to find the shortest sequence of events that can transform one gene order into another. From our results, we will find out the least number of events needed for sorting each pair of genomes and run the same datasets on DupLoss [33] and OrthoAlign [2]. Here is an example of how we are going to evaluate our result:

5.6.1 Running on DupLoss:

We will run both our synthetic datasets and real datasets on DupLoss [33]. For an example, Figure 5.6 shows us the output of DupLoss for one synthetic genome pair. The genome pair we used is :

```
Detailed Results
****************
   Genome X         Genome Y   Operation
   ========         ========   =========
    Leu_1     ---    Leu_10
    Asn_2     ---    Asn_11
      .       ---    Ile_12    Duplication of Ile_12 to Ile_12 copied from Ile_17 to Ile_17
    Gln_3     ---    Gln_13
    Ile_4     ---      .       Loss of Ile_4
      .       ---    [t]_14    Loss of [t]_14
    His_5     ---    His_15
    [t]_6     ---      .       Loss of [t]_6
    Gln_7     ---      .       Loss of Gln_7
    Met_8     ---    Met_16
      .       ---    Ile_17    Loss of Ile_17
    Gln_9     ---    Gln_18

Total Cost = 6.0

Genome X: Leu , Asn , Gln , Ile , His , [t] , Gln , Met , Gln
Genome Y: Leu , Asn , Ile , Gln , [t] , His , Met , Ile , Gln
>Ancestor
Leu,Asn,Gln,Ile,[t],His,[t],Gln,Met,Ile,Gln
```

Figure 5.6: Example of output of DupLoss

Genome X: Leu,Asn,Gln,Ile,His,[t],Gln,Met,Gln

Genome Y: Leu,Asn,Ile,Gln,[t],His,Met,Ile,Gln

In the output we can see, it shows the total number of events and the type of events.

5.6.2 Running on OrthoAlign:

For OrthoAlign [2] we have used the same genome pair and got the following result shown in Figure 5.7.

It also shows us the number of events and the type of events.

5.6.3 Comparing number of moves:

We also used the same genome for playing our HCG GeSort and below there is an example show in Table 5.1 of how we are going to compare our result with DupLoss

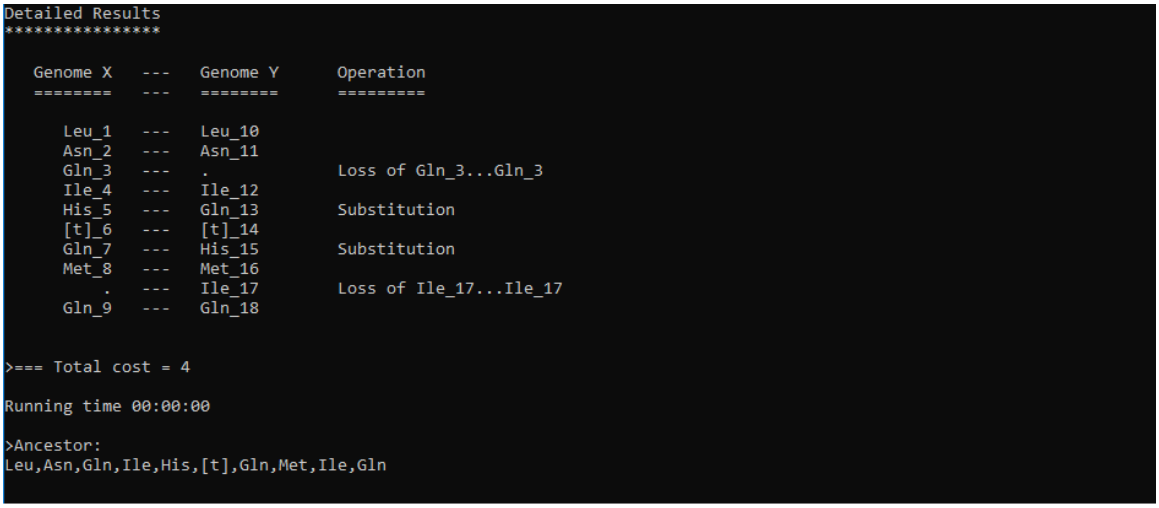


Figure 5.7: Example of output of OrthoAlign

and OrthoAlign:

Table 5.1: Comparing scores of OrthoAlign , DupLoss and GeSort. Here Training 1 is the pair of genomes discussed above.

Genome Pair	**DupLoss**	**OrthoAlign**	**GeSort**
Training 1	6	4	3

5.6.4 Comparing type of moves:

We will also compare the type of moves we get from GeSort with the type of moves from DupLoss and OrthoAlign. For the pair of genomes given above, here is an example shown in Table 5.2 of how we are going to compare the types of moves.

Table 5.2: Comparing type of moves of OrthoAlign , DupLoss and GeSort. Here Training 1 is the pair of genomes discussed above.

Method	**Genome Pair**	**Duplication**	**Deletion**	**Substitution**	**Inversion**
DupLoss	Training 1	1	5	0	0
OrthoAlign	Training 1	0	2	2	0
GeSort	Training 1	1	1	0	1

5.7 Discussion

GeSort will soon be available online (evolgame.cs.umanitoba.ca) and we will start recruiting players soon. This citizen science game is designed with effective game features which we have been discussed in the chapter 4. An interactive tutorial makes this game more interesting and easier for the players to understand about how to play this game. Another important feature of this game is the minimap which helps the players to navigate through a large genome (as the real datasets have large genomes). Thus we believe that *GeSort* can be an entertaining game for all level of players (from non-expert to expert). At this point, the next step will be to analyze and evaluate the data we will get from the players.

Chapter 6

Conclusion

In this thesis, we proposed a citizen science game *GeSort* which aims to solve the genome sorting problem. This is a completely new approach for solving this problem. More importantly, here we illustrated that we can turn this problem in an entertaining and intuitive casual game where both the expert and non-expert players can participate. From the study discussed in chapter 4, we have also observed that the players do not need scientific background knowledge to play this game.

For getting the best output from a HCG, introducing different game features is important. In this thesis, we demonstrated the effect of different game features (timer, top score and leaderboard) on players performance and motivation in a HCG (Chapter 4). We have found that both the timer and the top score had significant main effects on the two dependent variables (score and completion time for each level). With a timer, the players completed the levels faster but at the expense of poor scores. If we think from the perspective of a citizen science game, getting more optimal results is one the most important purposes. On the other hand, the timer also makes a HCG

more interesting and challenging. We have found both the positive and negative sides of adding a timer from our qualitative analysis. As previous studies [26] showed that making a reward system optional can have positive effect on players performance and experience, so for the timer, an optimal solution could be making this as an optional feature for this game. The players can play with the timer for getting some extra rewards. We have seen a connection between the timer and the reward system (in our case it is the point system and the leaderboard) as we have found an almost significant interaction effect between the timer and the leaderboard. The effect of the top score is the opposite of the effect of the timer. We have seen that showing a top score in the game can motivate the players and results in making fewer moves to complete the puzzles. In our study, in most of the cases, the score of the players were better than the average in those sessions where the top score was present. It forces the players to have the best solution. They were more engaged in the game when they saw a top score for a certain level. Showing a top score in the game also gives the players an idea about the optimal solution. From both our quantitative and qualitative analysis, the top score score was shown to be the most favorite and encouraging feature. Although we did not have any significant effect of the leaderboard, from our qualitative analysis we have found that showing a leaderboard can also increase the motivation of the players. As because of the limited time for the study we could not have observed all the possibilities of the leaderboard, we believe having a leaderboard for a longer version this game could give us more insights.

After determining an effective game design for *GeSort*, we are about to publish the game in online for getting the results from a large number of players. We have

made a model on how we can compare our results with the existing algorithms [2; 33]. In this part of the thesis we mainly focused on our method for solving the genome sorting problem and creating the model for analyzing and evaluating the data we will get from the players. In future, we are looking forward to have a reasonable amount of data from the players so that we can compare our results. We will analyze the strategies employed by the players to solve these puzzles. We will also compare the events for a better understanding on the evolutionary scenarios made by the players.

Bibliography

[1] Sophia Yancopoulos, Oliver Attie, and Richard Friedberg. Efficient sorting of genomic permutations by translocation, inversion and block interchange. *Bioinformatics*, 21(16):3340–3346, 2005.

[2] Olivier Tremblay-Savard, Billel Benzaid, B Franz Lang, and Nadia El-Mabrouk. Evolution of trna repertoires in bacillus inferred with orthoalign. *Molecular biology and evolution*, 32(6):1643–1656, 2015.

[3] Izabela Makałowska, Igor B Rogozin, and Wojciech Makałowski. Genome evolution. *Advances in bioinformatics*, 2010, 2010.

[4] Susumu Ohno. The enormous diversity in genome sizes of fish as a reflection of natures extensive experiments with gene duplication. *Transactions of the American Fisheries Society*, 99(1):120–130, 1970.

[5] Tine Blomme, Klaas Vandepoele, Stefanie De Bodt, Cedric Simillion, Steven Maere, and Yves Van de Peer. The gain and loss of genes during 600 million years of vertebrate evolution. *Genome biology*, 7(5):R43, 2006.

[6] James A Cotton and Roderic DM Page. Rates and patterns of gene duplication

and loss in the human genome. *Proceedings of the Royal Society of London B: Biological Sciences*, 272(1560):277–283, 2005.

[7] Evan E Eichler and David Sankoff. Structural dynamics of eukaryotic chromosome evolution. *science*, 301(5634):793–797, 2003.

[8] Richard A Gibbs, Jeffrey Rogers, Michael G Katze, Roger Bumgarner, George M Weinstock, Elaine R Mardis, Karin A Remington, Robert L Strausberg, J Craig Venter, Richard K Wilson, et al. Evolutionary and biomedical insights from the rhesus macaque genome. *science*, 316(5822):222–234, 2007.

[9] Jianzhi Zhang. Evolution by gene duplication: an update. *Trends in ecology & evolution*, 18(6):292–298, 2003.

[10] Michael Lynch and John S Conery. The evolutionary fate and consequences of duplicate genes. *Science*, 290(5494):1151–1155, 2000.

[11] Jeffery P Demuth, Tijl De Bie, Jason E Stajich, Nello Cristianini, and Matthew W Hahn. The evolution of mammalian gene families. *PloS one*, 1(1):e85, 2006.

[12] Sridhar Hannenhalli and Pavel A Pevzner. Transforming cabbage into turnip: polynomial algorithm for sorting signed permutations by reversals. *Journal of the ACM (JACM)*, 46(1):1–27, 1999.

[13] Eric Tannier and Marie-France Sagot. Sorting by reversals in subquadratic time. In *Combinatorial pattern matching*, pages 1–13. Springer, 2004.

[14] David Bryant. The complexity of calculating exemplar distances. In *Comparative Genomics*, pages 207–211. Springer, 2000.

[15] Xin Chen, Jie Zheng, Zheng Fu, Peng Nan, Yang Zhong, Stefano Lonardi, and Tao Jiang. Assignment of orthologous genes via genome rearrangement. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, 2(4):302–315, 2005.

[16] Cedric Chauve, Guillaume Fertin, Romeo Rizzi, and Stéphane Vialette. Genomes containing duplicates are hard to compare. *Computational Science–ICCS 2006*, pages 783–790, 2006.

[17] Guillaume Blin and Romeo Rizzi. Conserved interval distance computation between non-trivial genomes. In *International Computing and Combinatorics Conference*, pages 22–31. Springer, 2005.

[18] David A Christie and Robert W Irving. Sorting strings by reversals and by transpositions. *SIAM Journal on Discrete Mathematics*, 14(2):193–206, 2001.

[19] Seth Cooper, Firas Khatib, Adrien Treuille, Janos Barbero, Jeehyung Lee, Michael Beenen, Andrew Leaver-Fay, David Baker, Zoran Popović, et al. Predicting protein structures with a multiplayer online game. *Nature*, 466(7307):756–760, 2010.

[20] Alexander Kawrykow, Gary Roumanis, Alfred Kam, Daniel Kwak, Clarence Leung, Chu Wu, Eleyine Zarour, Luis Sarmenta, Mathieu Blanchette, Jérôme

Waldispühl, et al. Phylo: a citizen science approach for improving multiple sequence alignment. *PloS one*, 7(3):e31362, 2012.

[21] Jérôme Waldispühl, Arthur Kam, and Paul P. Gardner. Crowdsourcing Rna Structural Alignments With an Online Computer Game. *Biocomputing 2015*, pages 330–341, 2014.

[22] https://eterna.cmu.edu.

[23] https://https://www.globalwebindex.net.

[24] Erik Andersen, Yun-En Liu, Richard Snider, Roy Szeto, Seth Cooper, and Zoran Popović. On the harmfulness of secondary game objectives. In *Proceedings of the 6th International Conference on Foundations of Digital Games*, pages 30–37. ACM, 2011.

[25] Kristin Siu, Alexander Zook, and Mark O Riedl. Collaboration versus competition: Design and evaluation of mechanics for games with a purpose. In *FDG*, 2014.

[26] Kristin Siu and Mark O Riedl. Reward systems in human computation games. In *Proceedings of the 2016 Annual Symposium on Computer-Human Interaction in Play*, pages 266–275. ACM, 2016.

[27] Kristin Siu, Matthew Guzdial, and Mark O Riedl. Evaluating singleplayer and multiplayer in human computation games. In *Proceedings of the 12th International Conference on the Foundations of Digital Games*, page 34. ACM, 2017.

[28] Kristin Siu, Alexander Zook, and Mark O Riedl. A framework for exploring and evaluating mechanics in human computation games. In *Proceedings of the 12th International Conference on the Foundations of Digital Games*, page 38. ACM, 2017.

[29] Olivier Tremblay-Savard, Alexander Butyaev, and Jérôme Waldispühl. Collaborative solving in a human computing game using a market, skills and challenges. In *Proceedings of the 2016 Annual Symposium on Computer-Human Interaction in Play*, pages 130–141. ACM, 2016.

[30] Jacqueline Gaston and Seth Cooper. To three or not to three: Improving human computation game onboarding with a three-star system. In *Proceedings of the 2017 CHI conference on Human Factors in Computing Systems*, pages 5034–5039. ACM, 2017.

[31] Zhixiang Chen, Bin Fu, Jinhui Xu, Boting Yang, Zhiyu Zhao, and Binhai Zhu. Non-breaking similarity of genomes with gene repetitions. In *CPM*, pages 119–130. Springer, 2007.

[32] Mingfu Shao and Bernard ME Moret. Comparing genomes with rearrangements and segmental duplications. *Bioinformatics*, 31(12):i329–i338, 2015.

[33] Patrick Holloway, Krister Swenson, David Ardell, and Nadia El-Mabrouk. Ancestral genome organization: an alignment approach. *Journal of Computational Biology*, 20(4):280–295, 2013.

[34] Akash Singh, Chris Drogaris, Elena Nazarova, Mathieu Blanchette, Jérôme

Waldispühl, Anjum Ibna Matin, Mardel Maduro, and Olivier Tremblay-Savard. A human-computation platform for multi-scale genome analysis. *Association for the Advancement of Artificial Intelligence (www. aaai. org). Retrieved from https://www. humancomputation. com/2017/papers/100_ human-computation-platform. pdf*, 2017.

[35] Elisabeth RM Tillier and Richard A Collins. Genome rearrangement by replication-directed translocation. *Nature genetics*, 26(2):195, 2000.

[36] Sridhar Hannenhalli. *Polynomial-time algorithm for computing translocation distance between genomes*, pages 162–176. Springer Berlin Heidelberg, Berlin, Heidelberg, 1995.

[37] John Kececioglu and David Sankoff. Exact and approximation algorithms for sorting by reversals, with application to genome rearrangement. *Algorithmica*, 13(1):180–210, 1995.

[38] David Sankoff. Genome rearrangement with gene families. *Bioinformatics*, 15(11):909–917, 1999.

[39] Vineet Bafna and Pavel A Pevzner. Genome rearrangements and sorting by reversals. *SIAM Journal on Computing*, 25(2):272–289, 1996.

[40] David A Christie. A 3/2-approximation algorithm for sorting by reversals. In *SODA*, pages 244–252, 1998.

[41] Scott A Guyer, Lenwood S Heath, and John Paul C Vergara. Subsequence and run heuristics for sorting by transpositions. 1997.

[42] Vineet Bafna and Pavel A Pevzner. Sorting by transpositions. *SIAM Journal on Discrete Mathematics*, 11(2):224–240, 1998.

[43] Tzvika Hartman and Ron Shamir. A simpler and faster 1.5-approximation algorithm for sorting by transpositions. *Information and Computation*, 204(2):275–290, 2006.

[44] Laurent Bulteau, Guillaume Fertin, and Irena Rusu. Sorting by transpositions is difficult. *SIAM Journal on Discrete Mathematics*, 26(3):1148–1180, 2012.

[45] Maria Emilia MT Walter, Zanoni Dias, and Joao Meidanis. Reversal and transposition distance of linear chromosomes. In *String Processing and Information Retrieval: A South American Symposium, 1998. Proceedings*, pages 96–102. IEEE, 1998.

[46] Atif Rahman, Swakkhar Shatabda, and Masud Hasan. An approximation algorithm for sorting by reversals and transpositions. *Journal of Discrete Algorithms*, 6(3):449–457, 2008.

[47] Guohui Lin and Tao Jiang. A further improved approximation algorithm for breakpoint graph decomposition. *Journal of Combinatorial Optimization*, 8(2):183–194, 2004.

[48] John D Kececiogluy and R Raviz. Of mice and men: Algorithms for evolutionary distances between genomes with translocation. In *Symposium on Discrete Algorithms*, volume 604, page 613, 1995.

[49] Sridhar Hannenhalli. Polynomial-time algorithm for computing translocation

distance between genomes. *Discrete applied mathematics*, 71(1-3):137–151, 1996.

[50] Anne Bergeron, Julia Mixtacki, and Jens Stoye. On sorting by translocations. *Journal of Computational Biology*, 13(2):567–578, 2006.

[51] Anne Bergeron, Julia Mixtacki, and Jens Stoye. A new linear time algorithm to compute the genomic distance via the double cut and join distance. *Theoretical Computer Science*, 410(51):5300–5316, 2009.

[52] Haitao Jiang, Binhai Zhu, and Daming Zhu. Algorithms for sorting unsigned linear genomes by the DCJ operations. *Bioinformatics*, 27(3):311–316, 2011.

[53] Mingfu Shao and Yu Lin. Approximating the edit distance for genomes with duplicate genes under DCJ, insertion and deletion. *BMC Bioinformatics*, 13(Suppl 19):S13, 2012.

[54] Daniel Kwak, Alfred Kam, David Becerra, Qikuan Zhou, Adam Hops, Eleyine Zarour, Arthur Kam, Luis Sarmenta, Mathieu Blanchette, and Jérôme Waldispühl. Open-phylo: a customizable crowd-computing platform for multiple sequence alignment. *Genome biology*, 14(10):R116, 2013.

[55] Billel Benzaid, Riccardo Dondi, and Nadia El-Mabrouk. Duplication-loss genome alignment: Complexity and algorithm. In *International Conference on Language and Automata Theory and Applications*, pages 116–127. Springer, 2013.

[56] Riccardo Dondi and Nadia El-Mabrouk. Aligning and labeling genomes under

the duplication-loss model. In *Conference on Computability in Europe*, pages 97–107. Springer, 2013.

[57] Sandro Andreotti, Knut Reinert, and Stefan Canzar. The duplication-loss small phylogeny problem: from cherries to trees. *Journal of Computational Biology*, 20(9):643–659, 2013.

[58] Jeffrey P Cohn. Citizen science: Can volunteers do real research? *AIBS Bulletin*, 58(3):192–197, 2008.

[59] Andrea Wiggins and Kevin Crowston. From conservation to crowdsourcing: A typology of citizen science. In *System Sciences (HICSS), 2011 44th Hawaii international conference on*, pages 1–10. IEEE, 2011.

[60] Nathan R Prestopnik and Kevin Crowston. Gaming for (citizen) science: Exploring motivation and data quality in the context of crowdsourced science through the design and evaluation of a social-computational system. In *2011 Seventh IEEE International Conference on e-Science Workshops*, pages 28–33. IEEE, 2011.

[61] Alan Irwin, Susse Georg, and Philip Vergragt. The social management of environmental change. *Futures*, 26(3):323–334, 1994.

[62] Rick Bonney, Caren B Cooper, Janis Dickinson, Steve Kelling, Tina Phillips, Kenneth V Rosenberg, and Jennifer Shirk. Citizen science: a developing tool for expanding science knowledge and scientific literacy. *BioScience*, 59(11):977–984, 2009.

[63] Socientize Consortium et al. Green paper on citizen science. citizen science for europe. towards a better society of empowered citizens and enhanced research, 2013.

[64] Kyle W Willett, Chris J Lintott, Steven P Bamford, Karen L Masters, Brooke D Simmons, Kevin RV Casteels, Edward M Edmondson, Lucy F Fortson, Sugata Kaviraj, William C Keel, et al. Galaxy zoo 2: detailed morphological classifications for 304 122 galaxies from the sloan digital sky survey. *Monthly Notices of the Royal Astronomical Society*, 435(4):2835–2860, 2013.

[65] Elizabeth Howard, Harlen Aschen, and Andrew K Davis. Citizen science observations of monarch butterfly overwintering in the southern united states. *Psyche: A Journal of Entomology*, 2010, 2010.

[66] Yolanda Wiersma. Birding 2.0: Citizen science and effective monitoring in the web 2.0 world* ornithologie 2.0: la science citoyenne et les programmes de suivi à l'ère d'internet 2.0. *Avian Conservation and Ecology*, 5(2):1–9, 2010.

[67] Siamak Faridani, Bryce Lee, Selma Glasscock, John Rappole, Dezhen Song, and Ken Goldberg. A networked telerobotic observatory for collaborative remote observation of avian activity and range change. *IFAC Proceedings Volumes*, 42(22):56–61, 2009.

[68] Alex C Williams, John F Wallin, Haoyu Yu, Marco Perale, Hyrum D Carroll, Anne-Francoise Lamblin, Lucy Fortson, Dirk Obbink, Chris J Lintott, and James H Brusuelas. A computational pipeline for crowdsourced transcriptions

of ancient greek papyrus fragments. In *Big Data (Big Data), 2014 IEEE International Conference on*, pages 100–105. IEEE, 2014.

[69] G Rodighiero, E Daddi, I Baronchelli, A Cimatti, A Renzini, H Aussel, P Popesso, D Lutz, P Andreani, S Berta, et al. The lesser role of starbursts in star formation at z= 2. *The Astrophysical Journal Letters*, 739(2):L40, 2011.

[70] Andrew Hill, Robert Guralnick, Arfon Smith, Andrew Sallans, Rosemary Gillespie, Michael Denslow, Joyce Gross, Zack Murrell, Tim Conyers, Peter Oboyski, et al. The notes from nature tool for unlocking biodiversity records from museum records through citizen science. *ZooKeys*, (209):219, 2012.

[71] Kate Land, Anže Slosar, Chris Lintott, Dan Andreescu, Steven Bamford, Phil Murray, Robert Nichol, M Jordan Raddick, Kevin Schawinski, Alex Szalay, et al. Galaxy zoo: the large-scale spin statistics of spiral galaxies in the sloan digital sky survey. *Monthly Notices of the Royal Astronomical Society*, 388(4):1686–1692, 2008.

[72] Ria Follett and Vladimir Strezov. An analysis of citizen science based research: usage and publication patterns. *PloS one*, 10(11):e0143687, 2015.

[73] Carolyn Rosevelt, M Los Huertos, C Garza, and HM Nevins. Marine debris in central california: Quantifying type and abundance of beach litter in monterey bay, ca. *Marine pollution bulletin*, 71(1-2):299–306, 2013.

[74] Daniel Clery. Galaxy zoo volunteers share pain and glory of research, 2011.

[75] J Kelemen-Finan, C Knoll, and U Proebstl-Haider. Citizen science—really cool

or just stupid? lay monitoring as contribution to the environmental education of young people. *Naturschutz und Landschaftsplanung*, 45(6):171–6, 2013.

[76] Salvatore Loguercio, Benjamin M. Good, and Andrew I. Su. Dizeez: An Online Game for Human Gene-Disease Annotation. *PLoS ONE*, 8(8), 2013.

[77] https://eyewire.org.

[78] Amazon Mechanical Turk Inc. Amazon mechanical turk, 2018.

[79] Inc. Weblabcenter. Microworkers, 2018.

[80] Jeehyung Lee, Wipapat Kladwang, Minjae Lee, Daniel Cantu, Martin Azizyan, Hanjoo Kim, Alex Limpaecher, Snehal Gaikwad, Sungroh Yoon, Adrien Treuille, et al. Rna design rules from a massive open laboratory. *Proceedings of the National Academy of Sciences*, 111(6):2122–2127, 2014.

[81] Graham Alvare and Richard Gordon. Ct brush and cancerzap!: two video games for computed tomography dose minimization. *Theoretical Biology and Medical Modelling*, 12(1):7, 2015.

[82] Anne E Bowser, Derek L Hansen, Jocelyn Raphael, Matthew Reid, Ryan J Gamett, Yurong R He, Dana Rotman, and Jenny J Preece. Prototyping in place: a scalable approach to developing location-based apps and games. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1519–1528. ACM, 2013.

[83] Anne Bowser, Derek Hansen, Yurong He, Carol Boston, Matthew Reid, Logan Gunnell, and Jennifer Preece. Using gamification to inspire new citizen sci-

ence volunteers. In *Proceedings of the first international conference on gameful design, research, and applications*, pages 18–25. ACM, 2013.

[84] Jinseop S Kim, Matthew J Greene, Aleksandar Zlateski, Kisuk Lee, Mark Richardson, Srinivas C Turaga, Michael Purcaro, Matthew Balkam, Amy Robinson, Bardia F Behabadi, et al. Space–time wiring specificity supports direction selectivity in the retina. *Nature*, 509(7500):331, 2014.

[85] CCP. Eve online: Project discovery, 2017.

[86] François Bouchy, Maxime Marmier, and Oliver Turner. Light-curve classifications scrutinized by unige astronomers, January 2018.

[87] Jens Jakob WH Sørensen, Mads Kock Pedersen, Michael Munch, Pinja Haikka, Jesper Halkjær Jensen, Tilo Planke, Morten Ginnerup Andreasen, Miroslav Gajdacz, Klaus Mølmer, Andreas Lieberoth, et al. Exploring the quantum speed limit with computer games. *Nature*, 532(7598):210, 2016.

[88] Devin P Sullivan, Casper F Winsnes, Lovisa Åkesson, Martin Hjelmare, Mikaela Wiking, Rutger Schutten, Linzi Campbell, Hjalti Leifsson, Scott Rhodes, Andie Nordgren, et al. Deep learning is combined with massive-scale citizen science to improve large-scale image classification. *Nature biotechnology*, 2018.

[89] Akash Singh, Faizy Ahsan, Mathieu Blanchette, and Jérôme Waldispühl. Lessons from an online massive genomics computer game. In *HCOMP*, pages 177–186, 2017.

[90] Chris Drogaris, Akash Singh, Elena Nazarova, Lance Zhou, Alex Kawrykow,

Gary Roumanis, Alfred Kam, Mathieu Blanchette, and Jérôme Waldispühl. Phylo, 2018.

[91] Luis Von Ahn and Laura Dabbish. Labeling images with a computer game. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 319–326. ACM, 2004.

[92] Luis Von Ahn, Ruoran Liu, and Manuel Blum. Peekaboom: a game for locating objects in images. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 55–64. ACM, 2006.

[93] Edith Law and Luis Von Ahn. Input-agreement: a new mechanism for collecting data using human computation games. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1197–1206. ACM, 2009.

[94] Amaia Salvador, Axel Carlier, Xavier Giro-i Nieto, Oge Marques, and Vincent Charvillat. Crowdsourced object segmentation with a game. In *Proceedings of the 2nd ACM international workshop on Crowdsourcing for multimedia*, pages 15–20. ACM, 2013.

[95] Jürgen Scheible, Ville H Tuulos, and Timo Ojala. Story mashup: design and evaluation of novel interactive storytelling game for mobile and web users. In *Proceedings of the 6th international conference on Mobile and ubiquitous multimedia*, pages 139–148. ACM, 2007.

[96] Ville H Tuulos, Jürgen Scheible, and Heli Nyholm. Combining web, mobile

phones and public displays in large-scale: Manhattan story mashup. In *International Conference on Pervasive Computing*, pages 37–54. Springer, 2007.

[97] Derek Lomas, Kishan Patel, Jodi L Forlizzi, and Kenneth R Koedinger. Optimizing challenge in an educational game using large-scale design experiments. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 89–98. ACM, 2013.

[98] Alena Denisova and Paul Cairns. Adaptation in digital games: the effect of challenge adjustment on player performance and experience. In *Proceedings of the 2015 Annual Symposium on Computer-Human Interaction in Play*, pages 97–101. ACM, 2015.

[99] Yuan Jia, Yikun Liu, Xing Yu, and Stephen Voida. Designing leaderboards for gamification: Perceived differences based on user ranking, application domain, and personality traits. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, pages 1949–1960. ACM, 2017.

[100] Juho Hamari, Jonna Koivisto, and Harri Sarsa. Does gamification work?–a literature review of empirical studies on gamification. In *2014 47th Hawaii international conference on system sciences (HICSS)*, pages 3025–3034. IEEE, 2014.

[101] Richard N Landers, Kristina N Bauer, and Rachel C Callan. Gamification of task performance with leaderboards: A goal setting experiment. *Computers in Human Behavior*, 71:508–515, 2017.

[102] Alexandra Eveleigh, Charlene Jennett, Stuart Lynn, and Anna L Cox. "i want to be a captain! i want to be a captain!": Gamification in the old weather citizen science project. In *Proceedings of the first international conference on gameful design, research, and applications*, pages 79–82. ACM, 2013.

[103] David Codish and Gilad Ravid. Personality based gamification-educational gamification for extroverts and introverts. In *Proceedings of the 9th CHAIS Conference for the Study of Innovation and Learning Technologies: Learning in the Technological Era*, volume 1, pages 36–44, 2014.

[104] Jason T Bowey, Max V Birk, and Regan L Mandryk. Manipulating leaderboards to induce player experience. In *Proceedings of the 2015 Annual Symposium on Computer-Human Interaction in Play*, pages 115–120. ACM, 2015.

[105] Greg Walsh and Jennifer Golbeck. Curator: a game with a purpose for collection recommendation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 2079–2082. ACM, 2010.

[106] Seth Cooper, Adrien Treuille, Janos Barbero, Andrew Leaver-Fay, Kathleen Tuite, Firas Khatib, Alex Cho Snyder, Michael Beenen, David Salesin, David Baker, et al. The challenge of designing scientific discovery games. In *Proceedings of the Fifth international Conference on the Foundations of Digital Games*, pages 40–47. ACM, 2010.

[107] Simone Hantke, Florian Eyben, Tobias Appel, and Björn Schuller. ihearu-play: Introducing a game for crowdsourced data collection for affective computing.

In *Affective Computing and Intelligent Interaction (ACII), 2015 International Conference on*, pages 891–897. IEEE, 2015.

[108] Irene Celino, Dario Cerizza, Simone Contessa, Marta Corubolo, Daniele DellAglio, Emanuele Della Valle, and Stefano Fumeo. Urbanopoly–a social and location-based game with a purpose to crowdsource your urban data. In *2012 International Conference on Privacy, Security, Risk and Trust and 2012 International Confernece on Social Computing*, pages 910–913. IEEE, 2012.