A Crowdsourced Gameplay for Whole-Genome Assembly via Short Reads

https://doi.org/10.3991/ijoe.v16i08.14821

G. Gamage, I. Perera, D. Meedeniya (✉)
University of Moratuwa, Moratuwa, Sri Lanka
dulanim@cse.mrt.ac.lk

Anuradha Welivita
EPFL (École polytechnique fédérale de Lausanne), Lausanne, Switzerland

Abstract—Next-generation sequencing has revolutionized the field of genomics by producing accurate, rapid and cost-effective genome analysis with the use of high throughput sequencing technologies. This has intensified the need for accurate and performance efficient genome assemblers to assemble a large set of short reads produced by next-generation sequencing technology. Genome assembly is an NP-hard problem that is computationally challenging. Therefore, the current methods that rely on heuristic and approximation algorithms to assemble genomes prevent them from arriving at the most accurate solution. This paper presents a novel approach by gamifying whole-genome shotgun assembly from next-generation sequencing data; we present "Geno", a human-computing game designed with the aim of improving the accuracy of whole-genome shotgun assembly. We evaluate the feasibility of crowdsourcing the problem of whole-genome shotgun assembly by breaking the problem into small subtasks. The evaluation results, for single-cell Escherichia coli K-12 substr. MG1655 with a read length of 25 bp that produced 144,867 game instances of mean 25 sequences per instance at 40x coverage indicate the feasibility of sub-tasking the problem of genome assembly to be solved using crowdsourcing.

Keywords—Genome assembly; Gamification; Human Computing Games; Next Generation Sequencing.

1 Introduction

Reconstruction of whole-genome sequences undergoes two main steps, namely genome fragmentation and genome assembly. Genome fragmentation is the process of generating fragments of DNA called reads where genome sequencing technologies are used. These reads from sequencing methods are the inputs for genome assemblers to reconstruct the original whole-genome of the organism. Genome assembling is an essential step as sequencing methods cannot read the whole genome with a single iteration. Once reads are generated, these reads should be carefully aligned and merged.

There are two main genome sequencing technologies:

1. Sanger sequencing technology introduced in 1977 [1], which produces longer DNA reads of around 2000 base pairs (bp) in length
2. Next Generation Sequencing (NGS) technology introduced in 2005 [2], which can read relatively shorter but more voluminous DNA reads starting around 30 base pairs and can grow up to a few hundred base pairs in length.

With the use of Sanger sequencing that produces relatively long sequence reads, assembly tools such as Phrap [3], GigAssembler [4] and Celera [5], assemble genomes into contigs using overlap or string graph-like data structures. However, these tools are inefficient in handling the large volumes of data produced by NGS platforms. Since 2005, genome assemblers were introduced for the assembly of NGS data. NGS data assemblers include different approaches such as Illumina [6], Roche 454 [7] and SOLiD [8].

There are two major differences between NGS and Sanger methods: the sequencing volume and the sequencing length. As NGS approach is based on short DNA pieces, it can bind millions of reads at the same time where Sanger sequencing is limited to determine the order of one fragment of DNA per reaction. This high-throughput feature makes NGS sequencing highly cost-effective when sequencing a large amount of DNA. Producing short reads is also straightforward as they are easy to manage [9]. Most of the assemblers that use these reads are based on overlap/layout/consensus (OLC), de Bruijn graph (DBG) and greedy approaches. These approaches rely on heuristics and approximation algorithms to assemble genomes and hence may settle for sub-optimal solutions. Thus, investigating alternative approaches such as crowdsourcing to increase the accuracy of the results produced by current computational methods is justified.

Crowdsourcing and human computing games are emerging paradigms that leverage human knowledge and expertise to solve hard computing problems. Crowdsourcing is a distributed problem solving and production model, where the intelligent contributions of online communities are collected [10]; it requires incentives to increase public participation in the process. Crowdsourcing is useful as NGS method has many short reads of the divided sequence to reproduce aggregated sequence by the public.

Gamification provide entertainment for the player while getting complex problems solved through crowdsourcing. It is estimated that people spend over three billion hours per week playing computer games [11]; human computing games can effectively utilize this effort to solve complex computational problems. On the other hand, the use of interactive tools in an educational context can also be identified as a new tendency that increases student motivation and engagement [12]. Understanding complex subject areas using interactive games, boost up the knowledge retention than traditional teaching approaches. Rewards and leader boards can also be used to enhance student engagement in gamified learning. Complex concepts such as genome assembly in bioinformatics can be taught using these methods more effectively.

A novel approach for genome assembly using short reads is presented here. The approach is supported by both crowdsourcing and gamification. For that we introduce the novel human computing game "Geno". This paper explains the step by step approach from task division to assemble the complete genome through the proposed method. The paper evaluates the feasibility of crowdsourcing the genome assembly process using

"Geno", which is designed to use crowdsourcing to whole-genome assembly from NGS data. Section II presents the relevant literature. Section III and IV describe the methodology used and implementation of Geno. Section V evaluates the validity of the proposed method. Section VI and VII conclude the paper.

2 Literature Review

2.1 Current methods in genome assembly

The three standard methods to assemble genomes from short reads, produced either by Sanger sequencing or NGS technologies, are

1. Overlap/layout/consensus (OLC) approach using an overlap graph.
2. De Bruijn graph (DBG) approach using a k-mer graph.
3. Greedy approach using a lookup table to construct non-overlapping contiguous DNA sequences or contigs from short reads.

The overlap-layout-consensus (OLC) is an assembling algorithm supported by a directed graph based on the overlapping of reads. These overlappings are generated by considering all pairwise comparisons of sequences. In the directed graphs, nodes represent sequences where edges between them are responsible for representing the overlaps. Once the graph is constructed algorithm tries to find a Hamiltonian traversal path that contains all the nodes exactly once. Afterwards, it combines the nodes of the path using overlappings to get the final assembled genome. This approach is most applicable to the longer reads, such as greater than 200bp. It becomes inefficient when there are many reads as initially, it needs to get all pairwise overlappings [13].

However, most of the next-gen sequencing methods are generating a large number of shorter reads. Hence, using a method like OLC for assembling might not be a useful choice when considering the time consumption. The de Bruijn–graph-based approach can be used as a solution for this limitation of scalability as it fits for both shorter and longer reads. Here in the de Bruijn graph, it does not consume the actual sequence reads for the assembling process. Instead of that, it uses smaller sequence fragments called k-mers. Then these k-mers are aligned from (k-1) sequence overlaps. Ultimately with this single sequence read produces many overlapping k-mers that are well enough to represent all k-mers from the genome by itself. Although it is not a fool-proof process, it can solve some significant problems in de novo assembly. However, in certain cases where errors in reads and repeat structures are present still it can make the graphs more complex, inefficient and high memory consumable [14].

Greedy methods can be considered as a rapid approach of genome assembly. This method first focuses on the sequence reads that are most likely to be similar to each other based on the overlapping. The greedy approach makes a comparison to detect such sequences and then they are merged using the overlappings. Here some reads might not be used for the assembling as they are not well overlapped with the other reads. Thus, those reads will be shown as gaps. Alternate mechanisms such as pair-end

sequencing can be used to fill the gaps in those circumstances. Most of the greedy assemblers are supported by heuristics to speed up the process of eliminating incorrect assembling of recurring sequences. A limitation in this method is that it can be stuck in a local-maxima in the process of generating the assembled genome [15].

Table 1 and Table 2 give an overview of genome assembly literature. OLC and DBG genome assembly are graph-based solutions. Specifically, these graph-based approaches have a major limitation when sequencing errors are present. Such errors in the reads lead those methods to tangles that are hard to resolve.

Table 1. Summary of existing genome assembling techniques

Approach	Technique	Advantages	Limitations	Examples
OLC	Overlap graph	Easy to implement.	Cannot assemble large sets of reads from NGS. High Computation cost (overlaps are pre-computed by sets of pairwise seq. alignments).	Celera [5], phrap [3], GigAssembler [4], CAP
DBG	k-mer graph	Computationally inexpensive, sensitive to repeats and sequencing errors than the OLC.	Difficult to get the correct sequence, as the k-mer graph gives many sequences. Graph size increases by false-positive nodes & repeat structures.	Euler [16], AllPaths [17], Velvet [18], ABySS [19]
Greedy method	Word Lookup table	Computationally efficient than OLC and DBG	Can get stuck in local maxima.	SSAKE [20], SHARCGS [21] VCAKE [22],

Table 2. Summary of existing genome assembling tools

Work	Assembling Method			Sequence Method			Scaffolding	Features
	OLC	*Greedy*	*DBG*	*Sanger*	*454*	*Illumina*		
Newbler [23]	Y	N	N	Y	Y	N	Y	Good for SSF data; cDNA assembly. Incremental assembly. reads<50bp auto removal.
Phrap [3]	Y	N	N	Y	Y	N	Y	Quality assignment of each individual base. Fast assembly. Uses the Smith-Waterman-Gotoh algorithm to search.
Velvet [18]	N	N	Y	Y	Y	Y	Y	Good for short reads. Supports single-end and pair-end read. Strong error correction
MIRA [24]	Greedy and OLC Mix		N	Y	Y	Y	N	Iterative multiple-pass strategies Searches patterns on a symbolic level. Reconstruction of pristine mRNA transcripts.
ALLPATHS [17]	N	N	Y	N	N	Y	Y	de novo assembly, no Map assembly. Handle massively parallel seq. data. Offer short-range contiguity.
SSAKE [20]	N	Y	N	N	N	Y	Y	Detect the longest overlaps by Prefix tree. Supports single-end and pair-end read. de novo assembly, no Map assembly

Nevertheless, genome assembly itself can be considered as a graph reduction problem [25], which is NP-hard, and an efficient solution is yet to be found. Because of the efficiency requirements, genome assemblers such as greedy assemblers rely on suboptimal heuristic and approximation algorithms to remove redundancy and repair errors. This research explores a novel perspective for genome assembly problem, by reducing the limitations and obtain quality genome assembling. The presented method using crowdsourcing and gamification addresses the challenges with high accuracy and efficacy of genome assembling.

2.2 Computing challenge in bioinformatics

When processing whole genome sequences, computational capability is highly significant for maintaining efficiency and throughput. In a 2017 survey, over 700 biologists stated they need high-performance computing for executing tasks [26]. Grid and cloud computing, parallel computing and GPU processing are used as contemporary solutions to address these different techniques such as scaling up server resources. The main motivation for this is the recent improvements in genome sequencing and biological data-generating technologies that result in immediate increments in biological data. Major challenges exist the difficulty of handling large amounts of data and producing the results in a reasonable amount of time. The bioinformaticians are encouraged to use GPUs in their systems to cater to computational limitations [27].

The usage of cloud computing also seems practical in several scenarios as most of the biological data sets are also distributed through the internet. With the traditional methods, major bottlenecks in such cases include storage and pre-processing difficulties [28]. In addition to that, several applications use these methods in combination to meet requirements. Recent bioinformatics literature stated the use of GPUs with cloud computing, e.g. BioCloud, which enables higher computational power [29]. Similarly, [27] has used GPUs for bioinformatics workflow management. Generally, it seems most of these methods are used by only a limited scientific community. The main reason that can be identified as the major architectural alterations that are required for traditional computing approaches appeared to be challenging. However, to meet increasing computational requirements and to address computationally hard problems in bioinformatics, human computing and crowdsourcing are also explored as an alternative as human intervention is present. Further, workflow modelling tools are available for bioinformatics learner support and using GPU accelerated web services [30, 31].

2.3 Crowdsourcing in bioinformatics

Crowdsourcing is a highly effective technique to solve complex problems. Today crowdsourcing is applied for numerous fields including bioinformatics in solving computationally challenging tasks with the help of human contribution [10, 32]. Thus, the effective utilization of human-powered systems can be used to solve computationally hard problems in scientific domains. A vital factor in crowdsourcing is user participation in large numbers to produce high-quality outputs. Foldit [33], has emphasized the role of "gamified" crowdsourcing systems by rewarding users with entertainment.

Foldit is a multi-player game to deduce the structure of protein molecules. Protein structures are given as puzzles, where the players can move puzzle blocks to create the highest-scoring solution structures. Results have shown that Foldit can solve complex structural biochemistry problems that are hard to be solved by standard computations.

Kawrykow et al. have introduced a crowdsourcing approach to improve the accuracy of multiple sequence alignment (MSA) of genome sequences [34]. They have developed a human computing game, Phylo, that converts an MSA problem into a series of puzzles, which can be played by non-expert users. High scoring alignments produced by the players are evaluated at the server to get a high-quality MSA. Hence, the accuracy of MSA can be improved by combining classical algorithms with crowd computing methods. Colony B [35] is another type of puzzle game developed for microbiome research. This game aims to grow a colony of microbes by circling clusters of microbes that seem to be close to each other. The cluster patterns produced by the players are used to analyze microbiome data from the American gut project [36], to identify people with similar microbial profiles and derive correlations between different types of microbes, lifestyles and health conditions.

Ribo [11] is a human computing game for improving the accuracy of RNA sequence alignments. This is similar to Phylo but contains additional puzzle blocks to represent base pairs of RNA secondary structure. The existing RNA sequence alignments are broken into a series of sub alignments and given to players to be re-aligned through the Ribo platform. The crowdsourced solutions are later stitched together and stored in public repositories. They have shown that the public RNA alignment repositories can be maintained and improved by crowdsourcing. Recent scientific games such as Dizeez [37], The Cure [38] and EteRNA [39] are used to solve complex molecular biology problems, highlights the use of gamified crowdsourcing in biology.

Gamified crowdsourcing in bioinformatics show improvements in the results accuracy and the ability to cater to computational challenges of bioinformaticians. Computing challenges in current genome assembling techniques for NGS data and evidence of successful attempts of using games and crowdsourcing in solving biological problems motivates us to use crowdsourcing for genome assembly. We investigate the technical feasibility of using crowdsourcing for whole-genome assembly from NGS data.

2.4 Use of gaming to enhance the knowledge of students

Gamification has the capability of making difficult subject areas more approachable to the students. Hurse et al. have discussed how gamification can improve the quality of learning some science concepts wherein traditional learning students only get an abstraction [40]. By converting boring lessons to joyful gaming experiences, students are provided with a different perspective to look at subject matters. Learner engagement is high in a gamified approach. Challenges exist however such as insufficient evidence for supporting the long-term benefits of gamification in educational contexts, the practice of gamified learning outpaces researchers' understanding of its mechanisms and methods, the limited knowledge on how to gamify an activity following the specifics of the educational context, etc. [41]. Thus, better design of the game with intended learning outcome mapping is essential to get maximum benefit in gamified education.

Games such as Phylo, Colony B, Ribo have the capability of giving users learning experience along with the completion of biological tasks. From Geno, the game presented from this paper also provides users with an understanding of genome assembly in a brief manner. Ultimately with these approaches, complex bioinformatics concepts can be familiarized to the students.

3 Methodology

3.1 Overview

The process of crowdsourcing the whole genome assembly from NGS data is shown in Fig. 1(left). Fig. 1(right) shows the workflow in clustering and assembling short reads to a draft genome assembly with crowdsourcing. Initially, the problem is divided into subtasks that can be solved by individual crowd workers. For this purpose, the short reads produced by NGS technology are mapped to the most closely related genome assembly and cluster the short reads based on the local regions that they have mapped to. This helps to divide the whole genome assembly problem into subtasks and produce genome sequences of high sequence coverage with a few gaps. The crowd workers must align these short reads via the mobile game. Here the length of reads is picked as 25bp making the game usable with the screen size. Even though reads are short, it can be tolerated when there are many crowd workers engaged in. Clustering all reads to a local region or chromosome based on mapping information has been used in assembling the 2.91-billion base pair consensus sequence of the euchromatic portion of the human genome as well [42].

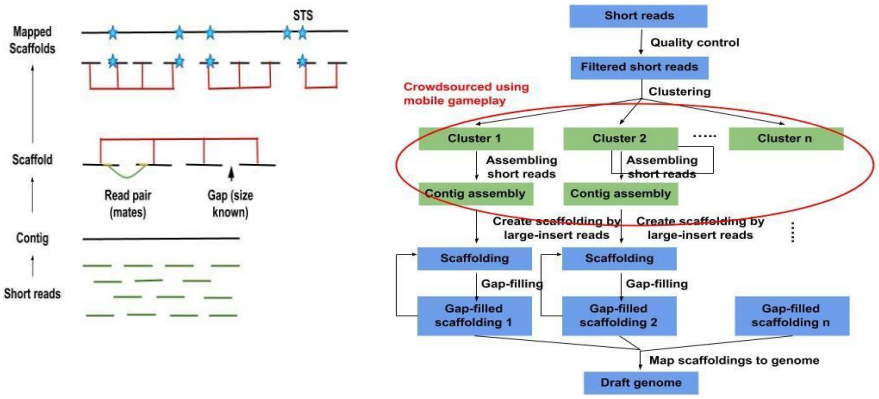

Fig. 1. Genome assembly (left) Process [27] and (right) workflow

After clustering, the overlapping short reads are aligned and merged to produce non-overlapping contiguous DNA sequences called contigs. Many such non-overlapping contigs are produced, and they are connected into scaffolds by using mate-pair infor-

mation. In a scaffold, the order and orientation of contigs, and the size of the gaps between consecutive contigs are known with reasonable precision. Finally, the scaffolds are mapped to the genome that we wish to reconstruct using physical markers known as sequence-tagged sites (STSs) [43].

3.2 Game design

The developed human computing game "Geno" aims at using crowdsourcing to improve genome assembly. The game maps the short reads in to puzzle blocks having differently coloured tiles indicating the ACGT nucleotides. The game engine gives a pool of such puzzle blocks from a cluster to each player. The players can intuitively interact with blocks by dragging, dropping and align them such that a maximum number of overlapping tiles (nucleotides) get aligned with each other. After each movement, a score representing the quality of the alignment is shown on the screen. An efficient score calculation mechanism is used, which is sensitive to the block movements. At the end of each game level, these alignments are merged to form a final contiguous sequence or contig with a score and stored in the server. If the Internet connectivity is low, the offline gameplay is allowed using local storage, which syncs the data with the server once become online. Further, HCI requirements such as real time-performance, ability to run in multiple types of devices and operating systems, have considered.

3.3 Formulation of task levels

The number of short reads in a cluster and the average length of short reads are used to predict the level of difficulty of a puzzle and categorize them into different levels in the game. The players can progress through levels by completing tasks. In each level, a leader board of players is maintained such that a maximum alignment accuracy can be reached by the players. A game level is completed by exceeding the minimum alignment score produced by the machine-computed assembly. To get to the top of the leader board, the player should try to reach the maximum alignment score in each level.

3.4 Aggregation of results

After each level of the game, contigs are produced with different scores as a result of gameplays. The best scoring contigs produced by players are selected and aggregated to produce scaffolds. When aggregating contigs, there are gaps based on the nature of the assembly. The pair mate information can be used to aid such gaps and complete the scaffolding task. The scaffolds are mapped to the original genome that wish to reconstruct using STS physical mapping data. A sequence-tagged site (or STS) is a short DNA segment whose location and base sequences are known, with a single occurrence in the genome. A separate algorithm can periodically perform this. The reconstructed genomes can be exposed through a public web interface for further research.

4 Implementation

4.1 Technologies used

A prototype of "Geno" is implemented using the Unity 3D game engine. Unity editor version 2017.3 is used for scripting. Gradle in JDK 1.8.0 101 and Android SDK 3.1.1 are used to build the application for the Android mobile platform. Adobe Photoshop CS6 is used to create the graphics-rich UI of "Geno". Game scores and contigs created by a given player are stored in real-time in Firebase, a cloud-hosted real-time NoSQL database. The advantage of using Firebase is its ability to persist data locally even at times when not connected to the Internet. ART NGS Simulator [44] is used to generate synthetic NGS reads. It mimics the real sequencing process with empirical error models or quality profiles summarized from large recalibrated sequencing data. Burrows-Wheeler Aligner (BWA) software [45] is used to map the generated short reads to the most closely related reference sequence. The BWA-backtrack algorithm that maps Illumina sequence reads up to 100bp is used for this. R (v3.5.1) is used to generate clusters using the location data from BWA.

4.2 Graphical user interface

Fig. 2 shows some user interfaces of "Geno". This is an extension to our previous work on “Genenigma” tool [32]. After the login, players are directed to the puzzle block selection where each puzzle block consists of four different blocks representing the ACGT nucleotides in the DNA sequence reads. After the selection, the player is directed to the gameplay interface shown in Fig. 2 left, with a pool of DNA reads to be aligned and merged that form a single contig. All the DNA reads shown as puzzle blocks are draggable for better interaction. Longer reads in the pool can be viewed by sliding the screen. The login UI, a user manual UI and the leaderboard UI with the top-scoring players are also designed with the same theme. The designed UIs are similar to existing scientific puzzle games such as “Phylo"; but they are more attractive with high-quality graphics, animations, and sounds. A novel feature in "Geno" compared to other state-of-the-art human computing puzzle games is prompting pop-up scores giving hints on how likely the player move can increase the score. Hence players can produce high-quality alignments with a minimum error.

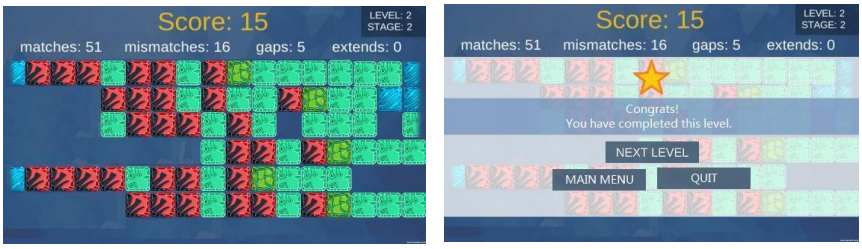

Fig. 2. Process of genome Graphical user interface of” Geno”. (left) Snapshot of the gameplay. (right) Interface prompted at level completion.

4.3 Score calculation algorithm

A score that measures the alignment quality of a contig produced is displayed after each player's movement. The algorithm for contig score calculation is given in Fig. 3. It considers the number of matches, mismatches, and indels in the overlapping regions of the short reads. Gaps and extends are allowed with a penalty to minimize them. When comparing the two sequences, the scoring algorithm checks for Matches, Mismatches, Gaps, and Extends to calculate the score. To do that first, it takes a pair of sequences and an ancestor sequence to facilitate the score calculation (line 1 in the algorithm). Then it initiates a variable to store the total score as done in line 2. Afterwards, the comparison starts. A match occurs when both sequences have similar puzzle pieces in the same position in the horizontal axis. For a match, the player earns a score of +1. When there are different puzzle pieces in the same position, it's a Mismatch, and the player will be penalized with -1 or -2. The lines 5 and 6 in the algorithm calculates corresponding scores of matches and mismatches. Score for both gaps and extends are calculated in line 7. Gaps occur if there is a gap between two pieces in the same sequence, and the player will get a score of -4 irrespective of the gap size. Extend is additional free space in a gap larger than two pieces. The player will earn -1 for each of the extend. For instance, if there is a gap of 4 pieces, the player will get -3 points for 3 extends and -4 for the gap itself, and players will be penalized with -7 points.

Algorithm: Calculate the score for a given alignment

Require: Position of the puzzle block, attached tags

Ensure: When to stop further optimizing the assembly

1. **Input:** pair of sequence (*firstSeq, secondSeq*), *ancestorSequence*
2. ***Integer*** *totalScore* ← 0
3. ***Integer*** *matches, mismatches, gapsAndExtends*
4. ***foreach*** (Change of a position of a puzzle block)
5. *matches* ← *CalMatches(firstSeq, secondSeq, ancestorSequence)*
6. *mismatches* ← *CalMismatches(firstSeq, secondSeq)*
7. *gapsAndExtends* ← *CalGapsAndExtends (firstSeq, secondSeq)*
8. *totalScore* ← *totalScore + matches – mismatches – gapsAndExtends[0] – gapsAndExtends[1]*
9. ***End foreach***
10. **Output:** Total score for the comparison

Fig. 3. The algorithm for contig score calculation

This score is calculated each time a player changes the position of a puzzle block, using its position and attached tags. A separate algorithm selects the best scoring contig from a pool of contigs produced by the players and combines them to form scaffolds using mate-pair data. Then the scaffolds are mapped to the original genome using STS data. The quality of the whole genome assembly is measured through a set of metrics such as the total number of contigs in the assembly, the length of the largest contig, the total length of the assembly, genome fraction and the number of mis-assemblies. These

metrics are computed using QUAST [46], a quality assessment tool for genome assemblies, that determines the optimum stop point of the assembly.

5 Evaluation and Results

5.1 Feasibility of sub tasking the genome assembly problem

To crowdsource the problem of genome assembly, the feasibility of dividing the entire problem into subtasks that can be solved by individual crowd workers is essential. Here, we used the single-cell *Escherichia coli* K-12 substr. MG1655 as a reference and simulated NGS reads using ART NGS simulator [44], which were mapped to the reference genome using the Burrows-Wheeler Aligner (BWA) software [45]; duplicate removal and sorting the reads using the location mappings given by BWA. R version 3.5.1 is used to generate clusters using the DBSCAN clustering algorithm such that the reads that are likely to overlap belong to the same cluster. Each of these clusters is considered as levels in the game or subtasks the solo game players can complete.

Reads of length 25bp are simulated with four coverage levels: 10x, 20x, 30x and 40x until the optimal clustering is obtained. Read length 25bp was chosen, as this length can be fully rendered on the screen. However, lower read lengths require a large number of reads with high coverage to obtain clusters of sufficiently overlapping reads. The eps and mints parameters used to cluster reads using the DBSCAN algorithm are 8% of the read length and 2, respectively. Fig. 4 shows the clusters generated for the first 100 reads of the single-cell *E. coli* bacterial genome. Clustering results for the whole genome are excluded from the paper as the clusters are not visible under small image size. The points shown in black are noise reads which do not belong to any cluster; they do not have overlaps with any of the neighbouring read sequences. Here, when the coverage is increased, the number of noise reads gradually decreased while the average number of reads per cluster gradually increased. When the coverage is increased up to 40x, the number of noise reads without any overlaps could be minimized and clusters having a feasible number of reading sequences that can be aligned by an individual player could be obtained.

It produced 144,867 game instances having an average of 25 sequences per instance. This experiment conducted with single-cell *E. coli* K-12 substr. MG1655 provides sufficient proof that the problem of genome assembly can be divided into subtasks and is feasible to be solved using crowdsourced gameplay. The summary of NGS simulation and clustering results are given in Table III.

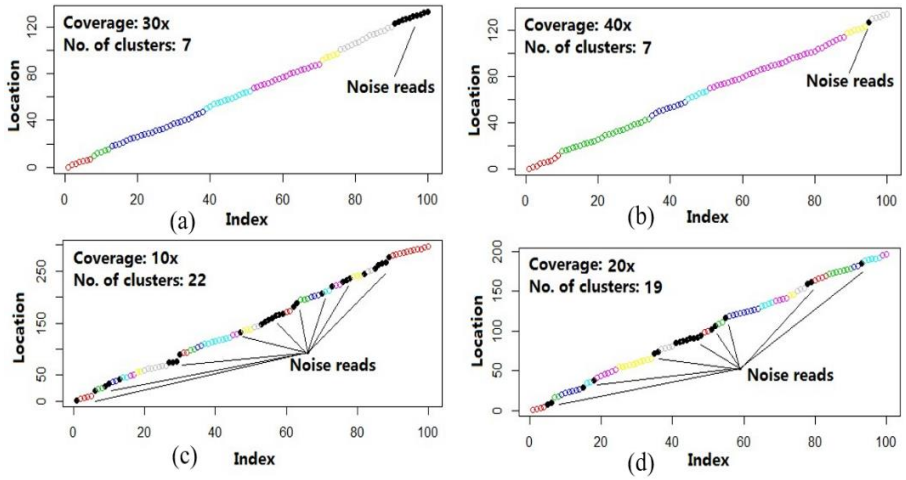

Fig. 4. Clustering results for the first 100 reads (sorted) of single-cell E.coli bacterial genome.

Table 3. Summary of NGS simulation and clustering results

Read length	Coverage	Reads	Reads without duplicates	Clusters	Noise reads	Avg. reads per cluster
25bp	10x	1,856,660	1,529,979	378,363	309,163	3
25bp	20x	3,713,320	2,556,727	411,064	104,405	6
25bp	30x	5,569,982	3,243,346	267,666	26,758	12
25bp	40x	7,426,643	3,705,308	144,867	6,140	25

5.2 Usability evaluation

An expert review was conducted with five experts in both bioinformatics and mobile game development domains to evaluate the playability of the "Geno" prototype. This was conducted to identify the user experience issues in the design, interfaces, and interactivity affecting the game playability. The experts were asked to report the observed usability issues and their level of severity based on playability heuristics.

Table IV gives the usability issues identified by the reviewers, their corresponding mean severity rating and practical suggestions to overcome these issues. The mean severity rating is calculated based on the ratings from all five experts. These can be considered for possible improvements in the game.

Table 4. Selected set of usability issues and suggestion received from evaluators

Usability issue	Suggestions
(Critical) Level of difficulty and pace of the game is challenging for a novice player. (Moderate) Inefficient screen layout as the player has to scroll horizontally to align puzzle blocks. (Moderate) No proper mechanism to increase the difficulty with the progress; it only increases the number and the size of the puzzle blocks that should be aligned. (Low) Lack of game rewards apart from the game score and the position in the leader-board.	Provide tips and guide a novice player to understand and progress through the game. Reduce the size of the puzzle blocks and make the puzzle appear more compact. Randomize the order of rendering the sequence reads so that the player cannot guess the order the alignment of the puzzle blocks. Give reward points to purchase more tiles when the game score reaches certain limits.

6 Discussion

The divide and conquer method is a strategy that is used to solve complex problems by splitting them into simple and straightforward sub-tasks. Following the divide and conquer approach, from this research, we have attempted to solve the complex genome assembly problem in a gamified and crowdsourcing environment using the alignment of inexpensive short reads. Other than representing an NGS assembling technique, this research reflects the feasibility of solving computational challenges in other NGS related methods which are based on short and ultra-short reads. Additionally, with Geno as a learning tool, it is expected to increase student motivation and engagement in the genome assembly area and make it easily approachable.

Usage of short or ultra-short reads brings computational challenges in next-generation sequencing methods. More particularly, it is required to enhance the number of parallel tasks when read length is decreasing. On the other hand, the usage of short reads is highly cost-effective and simple to complete. These reasons motivate the implementation of Geno, where crowdsourcing facilitates task parallelism and simple game instances fit the low computational mobile environment. Several successful pieces of research done in the medical context with the aid of gaming and crowdsourcing justify the approach used in Geno. From this paper too, we have evaluated the feasibility of the usage of crowdsourcing with Geno in detail.

Although there are several existing kinds of research done in genome processing with the help of gamification and crowdsourcing, this can be identified as one of the first attempts of using crowdsourced gameplay on genome assembly. Moreover, Geno brings the complex assembly problem to the mobile environment where it can approach users more than traditional crowdsourcing based methods. Usage of ultra-short reads also differs this research from other genome assembly methods. Computational feasibility is also enhanced with this approach. Geno has weighted more on game design and HCI aspects to provide more usability and entertainment to the crowd workers. Geno can be used as a study tool for genome assembly where students are expected to.

As further improvements, providing the support material and guidance to make Geno a self-learning tool can be mentioned. In addition to that usage of 3-D graphics and alternation of different resolution, levels can be used to enhance the read length where

the number of game instances can be reduced. Adding more difficulty levels with many alignments also can be used to speed up the assembling process.

7 Conclusion

This paper evaluates the feasibility of crowdsourcing the problem of genome assembly through interactive gameplay. The feasibility study conducted using single-cell Escherichia coli bacterial genome proves that the problem of genome assembly can be divided into subtasks that can be solved through crowdsourced gameplay. The paper presents the design and implementation of "Geno", a human-computing mobile game designed to improve the accuracy of whole-genome shotgun assembly using crowdsourcing. The expert review, evaluating the playability of "Geno", identified potential usability issues with the game design and suggested improvements. Not only the existing genomes that are already assembled but also the unassembled new genomes can be assembled using this approach, by mapping NGS reads to the most closely related genome assembly available. Similar to the NGS workflow tool as a learning aid, "Geno" can be used as an educational tool for learning genome assembly in bioinformatics curricula. The game can be added with 3D support for enhanced player experience and to incorporate complex assembling tasks. Another addition is to analyze the level of player competence and assign challenging tasks to expert players; this improves system effectiveness while enhancing player entertainment.

8 Acknowledgement

We acknowledge the support from the Senate Research Committee, Conference & Publication Grant, University of Moratuwa, Sri Lanka.

9 References

[1] Sanger, S. Nicklen and A. Coulson, "DNA sequencing with chain-terminating inhibitors", Proceedings of the national academy of sciences, vol. 74 no. 12, pp. 5463-5467, 1974. https://doi.org/10.1073/pnas.74.12.5463

[2] C. Schuster," Next-generation sequencing transforms today's biology", Nature Methods, vol. 5, no. 1, pp. 16, 2007.

[3] Green Group", Phrap.org, 2018. [Online]. Available: http://www.phrap.org. [Accessed: 10-Aug- 2018]

[4] W. J. Kent, and D. Haussler," Assembly of the working draft of the human genome with GigAssembler", Genome Research, vol. 11, no. 9, pp. 1541-1548, 2001.https://doi.org/10.1101/gr.183201

[5] G. Denisov et al.," Consensus generation and variant detection by Celera Assembler", Bioinformatics, vol. 24, no. 8, pp.1035-1040, 2008. https://doi.org/10.1093/bioinformatics/btn074

[6] Illumina — Sequencing and array-based solutions for genetic research", Illumina.com, 2018. [Online]. Available: https://www.illumina.com/[Accessed: 10- Aug- 2018]

[7] Home", Sequencing.roche.com, 2018. [Online]. Available: https://sequencing.roche.com. [Accessed: 10- Aug- 2018]
[8] "SOLiD (ThermoFisher) - AllSeq", AllSeq, 2018. [Online]. Available: http://all-seq.com/knowledge-bank/sequencing-platforms/solid/. [Accessed: 10- Aug- 2018]
[9] J. Kong, S. Huh, J. I. Won, J. Yoon, B. Kim and K. Kim, "GAAP: A Genome Assembly + Annotation Pipeline", BioMed Research International, pp. 1-12, 2019.https://doi.org/10.1155/2019/4767354
[10] S. Hewawalpita, S. Herath, I.Perera, and D. Meedeniya, "Effective Learning Content Offering in MOOCs with Virtual Reality An Exploratory Study on Learner Experience", Journal of Universal Computer Science, vol 24, no. 2, pp. 129-148, 2018,
[11] O. Goethe, Gamified Learning Experiences, Springer, 2019.
[12] R. Mavrevski, M. Traykov and I. Trenchev, "Interactive Approach to Learning of Sorting Algorithms", International Journal of Online and Biomedical Engineering (iJOE), vol. 15, no. 8, pp. 120-134, 2019. https://doi.org/10.3991/ijoe.v15i08.10530
[13] B. Foxman, "A Primer of Molecular Biology," in Molecular Tools and Infectious Disease Epidemiology, Elsevier, 2012, pp. 53–78.https://doi.org/10.1016/b978-0-12-374133-2.00005-8
[14] Y. He, Z. Zhang, X. Peng, F. Wu, and J. Wang, "De novo assembly methods for next-generation sequencing data," Tsinghua Science and Technology, vol. 18, no. 5, pp. 500–514, 2013. https://doi.org/10.1109/tst.2013.6616523
[15] X. Si, Q. Wang, L. Zhang, R. Wu, and J. Ma, "Survey of gene-splicing algorithms based on reads", Bioengineered, vol, 8, no. 6, pp. 750–758, 2017. https://doi.org/10.1080/21655979.2017.1373538
[16] P. A. Pevzner, et al., "A Eulerian path approach to DNA fragment assembly", Proceedings of the National Academy of Sciences, vol. 98, no. 17, pp. 9748-9753, 2001. https://doi.org/10.1073/pnas.171285098
[17] J. Butler et al.," ALLPATHS: de novo assembly of whole-genome shotgun microreads", Genome Research, vol. 18, no. 5, pp. 810-820, 2008. https://doi.org/10.1101/gr.7337908
[18] D. Zerbino, and E. Birney," Velvet: algorithms for de novo short read assembly using de Bruijn graphs", Genome Research, vol. 18, no.5, pp. 821829, 2008. https://doi.org/10.1101/gr.074492.107
[19] J. T. Simpson et al.," ABySS: a parallel assembler for short read sequence data", Genome Research, vol. 19, no. 6, pp. 1117-1123, 2009.https://doi.org/10.1101/gr.089532.108
[20] R. L. Warren, G. G. Sutton, S. J. Jones, and R. A. Holt," Assembling millions of short DNA sequences using SSAKE", Bioinformatics, vol. 23, no. 4, pp. 500-501, 2006. https://doi.org/10.1093/bioinformatics/btl629
[21] J. C. Dohm, C. Lottaz, T. Borodina, and H. Himmelbauer," SHARCGS, a fast and highly accurate short-read assembly algorithm for de novo genomic sequencing", Genome Research, vol. 17, no. 11, pp. 1697-1706, 2007. https://doi.org/10.1101/gr.6435207
[22] VCAKE: a bioinformatics tool for Genome assembly — GBS analysis", omictools, 2018. [Online]. Available: https://omictools.com/vcake-tool. [Accessed: 10- Aug- 2018]
[23] G. G. Silva et al., "Combining de novo and reference-guided assembly with scaffold_builder," Source Code for Biology and Medicine, vol. 8, no. 1, p. 23, 2013.
[24] B. Chevreux, "Using the mira EST Assembler for Reliable and Automated mRNA Transcript Assembly and SNP Detection in Sequenced ESTs", Genome Research, vol. 14, no. 6, pp. 1147-1159, 2004. https://doi.org/10.1101/gr.1917404

[25] E. Polevikov and M. Kolmogorov, "Synteny Paths for Assembly Graphs Comparison," 19th International Workshop on Algorithms in Bioinformatics (WABI), LIPIcs, vol. 143, 2019.
[26] L. Barone, J. Williams, Unmet needs for analyzing biological big data: A survey of 704 NSF principal investigators, PLoS Computational Biology, vol. 13, no. 10, 2017. https://doi.org/10.1371/journal.pcbi.1005755
[27] A. Welivita et al.," Managing Complex Workflows in Bioinformatics: An Interactive Toolkit with GPU Acceleration," IEEE Transactions on NanoBioscience, vol. 17, no. 3, pp. 199-208, 2018. https://doi.org/10.1109/tnb.2018.2837122
[28] B. Calabrese and M. Cannataro, "Cloud Computing in Bioinformatics: current solutions and challenges." PeerJ, 06-Jul-2016, doi: 10.7287/peerj.preprints.2261v1. https://doi.org/10.7287/peerj.preprints.2261
[29] H. Jo, J. Jeong, M. Lee, and D. Choi, Exploiting GPUs in Virtual Machine for BioCloud, BioMed Research International, pp. 1-11, 2013. https://doi.org/10.1155/2013/939460
[30] V. Mallawaarachchi, A. Wickramarachchi, A. Welivita, I. Perera, and D. Meedeniya, "Efficient Bioinformatics Computations through GPU Accelerated Web Services", 2nd International Conference on Algorithms, Computing and Systems (ICACS), ACM, pp. 94-98, 2018. https://doi.org/10.1145/3242840.3242848
[31] V. Mallawaarachchi, A. Wickaramarachchi, A. Weliwita, I. Perera, D. Meedeniya, "Experiential Learning in Bioinformatics – Learner Support for Complex Workflow Modelling and Analysis", International Journal of Emerging Technologies in Learning (iJET), vol 13, no 12, pp. 19-34, 2018. https://doi.org/10.3991/ijet.v13i12.8608
[32] D. A. Meedeniya, S. A. P. A. Rukshan and A. Welivita, "An Interactive Gameplay to Crowdsource Multiple Sequence Alignment of Genome Sequences: Genenigma", 9th International Conference on Bioscience, Biochemistry and Bioinformatics, pp. 28-35, 2019. https://doi.org/10.1145/3314367.3314374
[33] B. Koepnick et al., "De novo protein design by citizen scientists," Nature, vol. 570, no. 7761, pp. 390–394, Jun. 2019, doi: 10.1038/s41586-019-1274-4.
[34] A. Kawrykow et al.," Phylo: a citizen science approach for improving multiple sequence alignment", PloS one, vol. 7, no. 3, pp. e31362, 2013. https://doi.org/10.1371/journal.pone.0031362
[35] Play.google.com, 2018. [Online]. Available: https://play.google.com/store/apps/details?id=pontax.cs.mcgill.ca. [Accessed: 10- Aug- 2018]
[36] D. McDonald et al.," American Gut: an Open Platform for Citizen Science Microbiome Research", mSystems, vol. 3, no. 3, pp. e00031-18, 2018.
[37] S. Loguercio, B. M. Good, and A. I. Su," Dizeez: an online game for human gene-disease annotation", PLoS One, vol. 8, no. 8, pp. e71171, 2013. https://doi.org/10.1371/journal.pone.0071171
[38] B. Good et al," The cure: design and evaluation of a crowdsourcing game for gene selection for breast cancer survival prediction", JMIR Serious Games, vol. 2, no. 2, pp. e7, 2014. https://doi.org/10.2196/games.3350
[39] J. Lee et al.," RNA design rules from a massive open laboratory", Proceedings of the National Academy of Sciences, vol. 111, no. 6, pp. 2122-2127, 2014.
[40] C. Hursen, and B. Cizem, "Use of Gamification Applications in Science Education", International Journal of Emerging Technologies in Learning (iJET), vol. 14, no. 1, pp. 4-23, 2019.
[41] C. Dichev and D. Dicheva, "Gamifying education: what is known, what is believed and what remains uncertain: a critical review," International Journal of Educational Technology in Higher Education, vol. 14, no. 1, Feb. 2017. https://doi.org/10.1186/s41239-017-0042-5

[42] J. C. Venter et al," The sequence of the human genome", science, vol. 291, no. 5507, pp. 1304-1351, 2001.
[43] Sequence-Tagged Sites (STS)", Ncbi.nlm.nih.gov, 2018. [Online]. Available: https://www.ncbi.nlm.nih.gov/probe/docs/techsts/. [Accessed: 11- Aug- 2018]
[44] W. Huang, L. Li, J. R. Myers, and G. T. Marth," ART: a next-generation sequencing read simulator", Bioinformatics, vol. 28, no. 4, pp. 593-594, 2011. https://doi.org/10.1093/bioinformatics/btr708
[45] H. Li, and R. Durbin," Fast and accurate short read alignment with Burrows Wheeler transform", Bioinformatics, vol. 25, no. 14, pp. 17541760, 2009. https://doi.org/10.1093/bioinformatics/btp324
[46] A. Gurevich, V. Saveliev, N. Vyahhi, and G. Tesler," QUAST: quality assessment tool for genome assemblies", Bioinformatics, vol. 29, no. 8, pp. 1072-1075, 2013. https://doi.org/10.1093/bioinformatics/btt086

10 Authors

G. Gamage is a Junior Consultant in Dept. of Computer Science & Engineering (CSE), University of Moratuwa, Sri Lanka. Former Research Assistant at LiveLabs, School of Information Systems, Singapore Management University. His research areas are Bioinformatics, Computer Vision, Data Science and Machine Learning.

I. Perera is a Senior Lecturer at the Dept. of CSE, University of Moratuwa, Sri Lanka. He holds a PhD (St Andrews, UK) MBS (Colombo), MSc (Moratuwa), PGDBM (Colombo) and B.Sc. Eng. (Hons) (Moratuwa). He is a Fellow of HEA(UK), MIET, SMIEEE and a Chartered Engineer registered at EC (UK) and IE(SL).

D. Meedeniya is a Senior Lecturer at the Dept. of CSE, University of Moratuwa, Sri Lanka. She holds a PhD (St Andrews, UK), MSc (Moratuwa), BSc (CS) (Peradeniya). She is a Fellow of HEA(UK), MIET, MIEEE and a C.Eng. registered at EC (UK). Email: dulanim@cse.mrt.ac.lk

A. Welivita, is a PhD student at EPFL (École polytechnique fédérale de Lausanne), Lausanne, Switzerland. She graduated from the Dept. of CSE, University of Moratuwa.

Article submitted 2020-04-14. Resubmitted 2020-05-05. Final acceptance 2020-05-07. Final version published as submitted by the authors.