Chromatin 3D Reconstruction from Chromosomal Contacts Using a Genetic Algorithm

Viacheslav Kapilevich, Shigeto Seno, Hideo Matsuda, and Yoichi Takenaka

Abstract—Recent epigenetics research has demonstrated that chromatin conformation plays an important role in various aspects of gene regulation. *Chromosome Conformation Capture (3C)* technology makes it possible to analyze the spatial organization of chromatin in a cell. Several algorithms for three-dimensional reconstruction of chromatin structure from 3C experimental data have been proposed. Compared to other algorithms, *ShRec3D*, one of the most advanced algorithms, can reconstruct a chromatin model in the shortest time for high-resolution whole-genome experimental data. However, *ShRec3D* employs a graph shortest path algorithm, which introduces errors in the resulting model. We propose an improved algorithm that optimizes shortest path distances using a genetic algorithm approach. The proposed algorithm and *ShRec3D* were compared using in silico 3C experimental data. Compared to *ShRec3D*, the proposed algorithm demonstrated significant improvement relative to the similarity between the algorithm's output and the original model with a reasonable increase to calculation time.

Index Terms—Chromosome conformation capture, 3D reconstruction, genetic algorithm, ShRec3D

1 Introduction

Until recently, research has focused on DNA nucleotide sequences and very little attention has been paid to the physical conformation of DNA in a cell's nucleolus. It has been determined that the complex macromolecular structure of chromatin plays an important role in various aspects of gene regulation [1], such as gene expression and DNA replication and repair. It has been suggested that changes in chromatin structure are related to more complex processes, such as aging [2]. Recent studies have demonstrated that abnormal changes in chromatin structure due to exposure to environmental influences could lead to changes in gene transcription and subsequently to the emergence of particular types of cancer [3]. In addition, chromatin structure has attracted attention in stem cell research [4] because changes in chromatin structure can influence cell differentiation processes.

Chromatin organization is similar across cells of the same type, age, and life cycle stage if environmental conditions are the same [5]. This, as well as a variety of possible research applications, makes capturing chromatin structure an important task.

In the first decade of the 21st century, technology to retrieve information about chromatin structure from a sample cell population became available. The original Chromosome Conformation Capture (3C) technology described by Job Dekker in 2002 [6] allows researchers to determine the relative contact frequency between particular loci of a given cell population. An improved technology, i.e., Hi-C [7], was presented in 2012. Hi-C has a highly parallel nature and can produce genome-wide interaction maps.

Since the original 3C technology was introduced, several approaches to reconstructing three-dimensional (3D) models using the obtained data have been proposed. Some approaches are based on the same principles as methods designed to reconstruct protein structures, such as targeted growth or iterative structure optimization. Such algorithms, e.g., BATCH [8] and ChromSDE [9], obtain comparatively good quality reconstructed models; however, they require significant computational resources. Therefore, they cannot be used for large (more than a few hundred loci) contact maps.

Over the previous decade, there has been continuous improvement in the resolution of 3C-like technologies. The latest technologies are based on next-generation sequencing and produce high-resolution whole-genome contact maps. However, traditional global optimization algorithms cannot process such input in a reasonable timeframe.

A *multidimensional scaling (MDS)* based algorithm for 3D reconstruction has been proposed. MDS was originally used in psychometrics and is now applied in various disciplines. MDS utilizes similarity (or dissimilarity) values between objects to find a set of vectors in N-dimensional space such that the distance matrix corresponds as closely as possible to some function of the input matrix. The main advantage of this approach is execution speed.

ShRec3D is an MDS-based algorithm proposed by Lesne et al. in 2014 [10] and later improved in 2016 [11]. *ShRec3D*

- *V. Kapilevich, S. Seno, and H. Matsuda are with the Graduate School of Information Science and Technology, Osaka University, Suita 565-0871, Japan. E-mail: {skkap, senoo, matsuda}@ist.osaka-u.ac.jp.*
- *Y. Takenaka is with the Faculty of Informatics, Kansai University, Takatsuki, Osaka 569-1095, Japan. E-mail: takenaka@kansai-u.ac.jp.*

Manuscript received 19 Dec. 2017; accepted 3 Mar. 2018. Date of publication 20 Mar. 2018; date of current version 7 Oct. 2019.
(Corresponding author: Viacheslav Kapilevich.)
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TCBB.2018.2814995

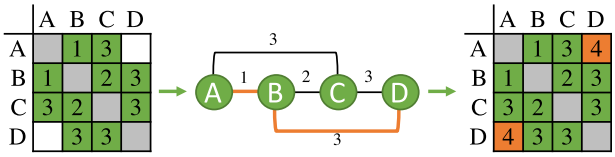

Fig. 1. Process of calculating unknown distances using graph shortest path algorithm.

combines the Floyd-Warshall graph shortest path algorithm [12] and MDS to achieve 3D model reconstruction from an input sparse distance matrix.

ShRec3D has been shown to be effective for reconstruction of 3D chromatin structure from high-resolution 3C experimental data. However, we consider that using a graph shortest path algorithm to fill in unknown distances in a sparse input distance matrix introduces errors in the output 3D structure. *ShRec3D* and its limitations are described in Section 2.

We propose combining a graph shortest path algorithm and a genetic algorithm approach to improve the similarity between the output 3D structure and the *original model*. The proposed algorithm is described in Section 3. We used generated in silico input data to compare *ShRec3D* to the proposed algorithm in terms of the quality and calculation time of the output model. Experimental results are given in Section 4. The results are discussed in Section 5, and conclusions are presented in Section 6.

2 Conventional Method

As explained in Section 1, *ShRec3D* [10] is one of the most recent and one of only a few effective algorithms for whole-genome high-resolution 3C experimental data. In this paper, we refer to *ShRec3D* as the conventional method. In this section, we explain how 3D reconstruction is performed by *ShRec3D*.

The *ShRec3D* algorithm consists of three main steps.

(1) A sparse distance matrix is generated from raw 3C experimental data.
(2) Unknown distances are calculated using a graph shortest path algorithm.
(3) The 3D structure is reconstructed using MDS on the complete distance matrix.

In the first step, the contacts between each pair of loci are converted to a distance by inverting the frequency of contacts between each pair. Since contact frequencies are only available for closely located loci, we obtain a sparse symmetric matrix that represents the spatial proximity of loci.

In the second step, all unknown distances in the input distance matrix are calculated. As originally proposed, *ShRec3D* uses the input distance matrix as an adjacency matrix that represents a weighted undirected graph. Unknown distances of input distance matrix are represented as absent edges in the graph. Then, the constructed graph is used to find the shortest path between two nodes without a connecting edge. The sum of weights of all nodes on the found shortest path is considered the distance between these nodes.

The process to calculate unknown distances is shown in Fig. 1. As can be seen, the distance between loci A and D is unknown. Running the shortest path algorithm for nodes A and D finds the shortest path, i.e., $A-B-D$, where the sum of the weights $AB+BD=1+3=4$. *ShRec3D* uses the Floyd-Warshall algorithm to find the shortest paths between all node pairs in a single run. After calculating all unknown distances, the complete distance matrix is used in step 3.

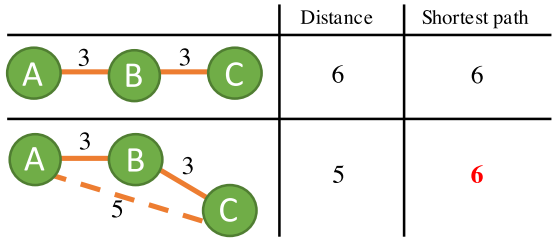

Fig. 2. Difference between actual distance and graph shortest path.

The third step determines loci positions in 3D space by applying MDS to the complete distance matrix. MDS is a distance geometry method that achieves dimensional reduction using deterministic linear algebra formulas; thus, it can obtain stable output in predictable time. MDS treats the input distance matrix as a dissimilarity matrix, which is used to place points in n-dimensional space to preserve dissimilarities as much as possible in a 3D projection. The output of MDS is a set of coordinates of points in 3D space. This is considered the final *ShRec3D* result. As all steps of the algorithm are deterministic, each *ShRec3D* run with the same input matrix will output an identical reconstructed model.

As explained above, the conventional method uses a graph shortest path algorithm to calculate unknown distances. Obviously, this calculated distance is correct only if all edges on the shortest path are aligned, i.e., in a straight line, in the *original model* (Fig. 2), which is normally not the case. Therefore, in most cases, the calculated distances are longer than the distances in the original model. Thus, distances can be considered the upper bounds for correct distance candidates. Using such overestimated distances as input for MDS will result in errors in the output 3D structure.

3 Method

As explained in the previous section, *ShRec3D* can be improved. We believe that the distances calculated in the second step of *ShRec3D* could be optimized to improve the output model. Here, we describe the proposed algorithm in detail.

In the third step of the conventional method, MDS is used to reconstruct the model in 3D space. MDS works such that the distances between each pair of points in the output model are as similar as possible to the distances given in the MDS input. If an input matrix has erroneous distances, it is impossible to reconstruct a model with perfectly matching distances. Therefore, MDS adjusts both the known and calculated distances.

This MDS property can be used to define an optimization problem as follows.

Input: M_i (distance matrix containing *known distances*).

Output: M_o (distance matrix obtained from the model reconstructed by MDS).

Object of Optimization: Calculated distances of M_i.

Optimization Criterion: Root mean square error between *known distances* before (M_i) and after (M_o) MDS.

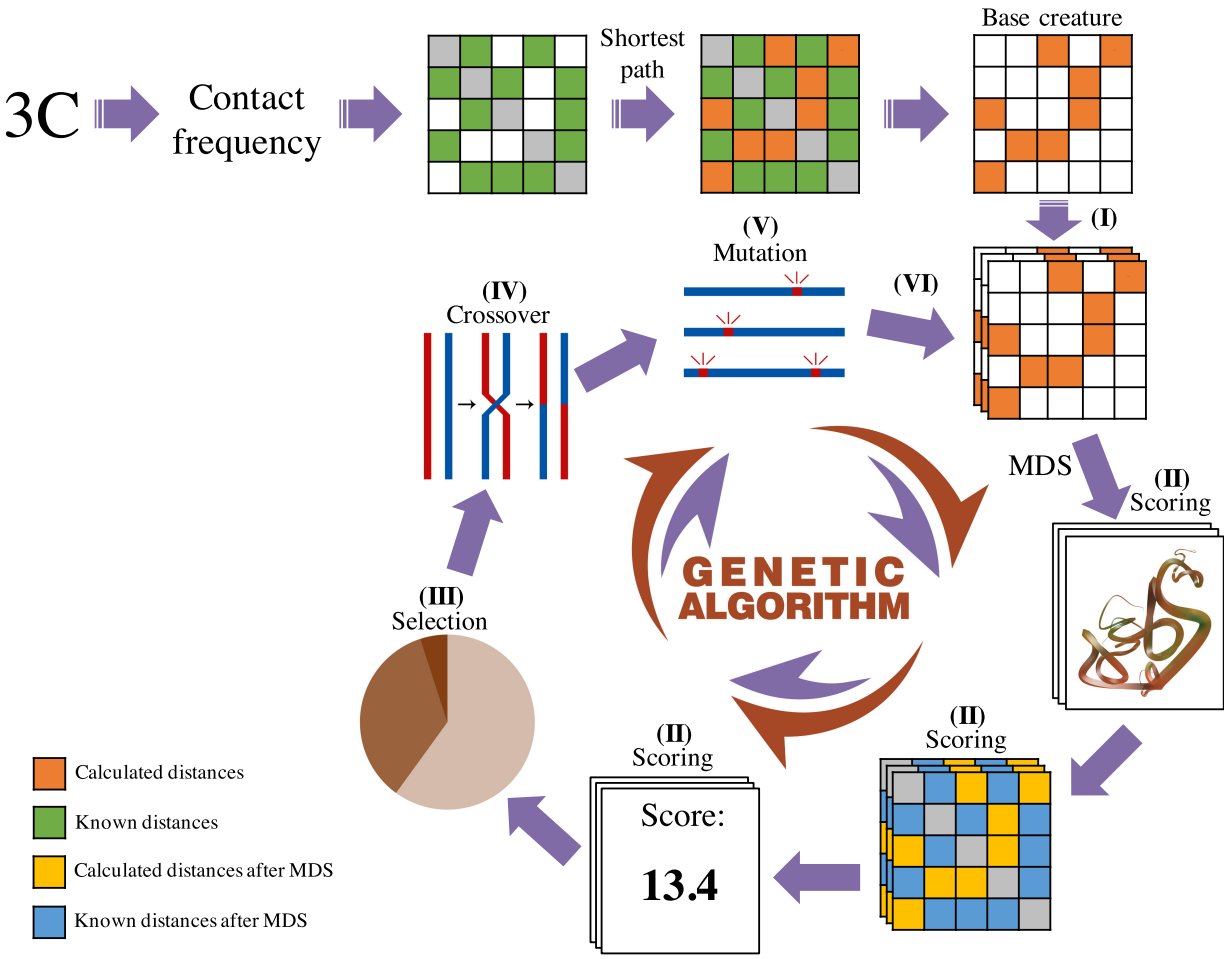

Fig. 3. Scheme of the proposed algorithm.

We propose to use a genetic algorithm to perform this optimization to improve the quality of output model. The genetic algorithm approach for this optimization problem is performed in six steps (Fig. 3).

(I) Generate the initial population, which becomes the current population for this iteration.
(II) Each creature in the current population is given a score based on its genome.
(III) Perform selection. Creatures with a high score have greater chance to be selected as parents of the next generation's creatures.
(IV) Crossover the genomes of each selected pair of creatures to form the genome of the new creature.
(V) Mutate each gene of the new creature's genome with a certain probability defined by the mutation rate parameter.
(VI) This cycle repeats for a newly formed population as the current population.

(I) First, unknown distances in the sparse distance matrix are calculated using a graph shortest path algorithm, similar to *ShRec3D* (Section 2). At this point, all distances in the complete distance matrix are divided into two groups: *known distances* and *calculated distances*. The *calculated distances* of this distance matrix compose a genome of the *base creature*, which serves as the base for the *initial population*.

Then, we generate the genomes of creatures in the initial population. To perform optimization, the initial population should have sufficient diversity to cover a wide search space. We use the *base creature's* genome as the upper limit for the creatures in the initial population because the distance values calculated by the graph shortest path algorithm cannot be less than the same distances in the *original model*. It was determined experimentally that the *original model's* distance with high probability lies within the interval with upper limit—the *base creature's* distance (d_b) and lower limit—two-thirds of the upper limit. Therefore, we draw values for the genes of each new creature in the initial generation from uniform distribution $\mathcal{U}(\frac{2d_b}{3}, d_b)$. The size of the population is considered an input parameter for the algorithm.

(II) Next, creatures in the current population are scored. In our case, we begin by generating the complete distance matrix using both the *calculated distances* from the creature's genome and the *known distances*. Then, we reconstruct the 3D model using MDS. The output model is then converted to a distance matrix by calculating the euclidean distances between each pair of points.

The score is denoted *known distances error* (E_{known}), which is defined as the root mean square error between the *known distances* and the same distances in the model reconstructed by MDS. The known distances error is calculated using

$$E_{known} = \sqrt{\frac{\sum_{\substack{i,j \\ d_0(i,j)\neq\varnothing}} \Big(d_0(i,j) - d(i,j)\Big)^2}{n}}. \quad (1)$$

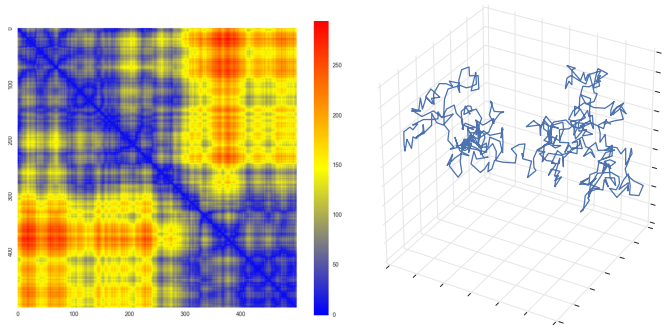

Fig. 4. Simulated 3C experimental data [size of the model is 500 loci]: (left) distance matrix and (right) 3D structure.

Here, n is the number of points (i.e., the number of rows or columns in the distance matrix), $d_0(i,j)$ is the distance from the original distance matrix, and $d(i,j)$ is the distance from the distance matrix obtained from the model after running MDS.

(III) To select creatures that will become the parents of the new generation's creatures, the calculated *known distances errors* are normalized as follows.

(1) Error values are normalized by subtracting the smallest value from each error value.
(2) Creatures with smaller error values are considered better; thus, error values are inverted by subtracting each value from the largest value.
(3) At this point, the creature with the greatest error has zero value. To avoid losing the genotype of this creature completely, we add the smallest value to each one.

The values calculated in this manner are used in the Roulette Wheel algorithm to give the fittest creatures the greatest chance to be selected.

(IV) To create each new creature, two parent creatures are selected as described. Then, single-point crossover of their genomes is performed. Here a random position in the genome is chosen. Then, the part of the genome before this point is taken from the first parent and the part after this point is taken from the second parent.

(V) Before becoming part of the new generation, each creature goes through the final stage, i.e., mutation. The mutation rate is an algorithm parameter ranging from 0 to 1. A mutation rate of 0 will eventually lead to convergence to a local minimum because diversity in the population will gradually disappear. With a mutation rate of 1, the algorithm will perform global optimization because heredity will be absent from one generation to the next. Values between 0 and 1 define the probability at which each gene will be mutated. Mutation is performed by assigning a new value to the given gene in the same interval used during generation of the *initial population*.

(VI) A newly formed population takes the place of the previous population and proceeds through steps (II) to (VI).

The genetic algorithm continues until one of the stop conditions is met. There are several tactics that can be used to determine when to stop the genetic algorithm, such as fixing the number of generations, limiting the allowed execution time, or thresholding the convergence speed. These tactics are discussed in detail in Section 5.

The creature with the smallest *known distances error* becomes the final creature, and its genome is used to form the complete distance matrix, which is used to reconstruct the final 3D structure by MDS. This model is considered the output of the proposed algorithm.

Compared to the conventional method, the proposed algorithm is stochastic. The steps in the proposed algorithm, i.e., initial population generation, crossover, and mutation, are based on randomly-generated numbers. Therefore, runs with identical input will yield different outputs.

4 Results

4.1 Generation of Input Experimental Data

To compare algorithms, we generated artificial 3C experimental data (Fig. 4), which allows us to evaluate the quality of the output of different algorithms. The left part of Fig. 4 shows a distance matrix that indicates the distances between each pair of points in the generated 3D structure. This is represented as a heat map, where short distances are shown by cool colors (blue) and long distances are shown by warm colors (red). The right part of Fig. 4 shows an isometric representation of the generated 3D structure.

First, a linear structure is generated in 3D space. DNA molecules are linear, and neighboring loci in 3C experiment are separated by approximately the same number of base pairs; therefore, similar euclidean distance separate them. First, the model generator puts the initial point at random coordinates in the space. Each following point is positioned at a random distance (determined by normal distribution $(\mu = 10, \sigma = 2.5)$) from the previous point in the direction of uniformly at random chosen Euler angle. In this manner, we can obtain a 3D structure that resembles the linear structure of DNA folded inside a nucleus. Note that we do not consider specific folding features, such as wrapping around nucleosomes, because such features are not critical for testing reconstruction algorithms.

Next, the distance matrix is constructed using the euclidean distances between each pair of points. Even though 3C experimental data consist of a number of contacts between loci, this value can also serve as a proxy for the distance between loci [13]. The distance matrix is sparse in real 3C experimental data because, for distant loci, no or very few contacts are observed. According to real 3C experimental data [13], only approximately 5 percent of the distances are known. To achieve this in simulated experimental data, 95 percent of the greatest distances are removed from the distance matrix.

The generated distance matrix is used as the input to both the conventional and proposed algorithms. The complete distance matrix (before removal of 95 percent of the distances) is used to verify the output of both algorithms.

4.2 Experimental Parameters

The proposed algorithm has two parameters (population size and mutation rate) related to the underlying genetic algorithm. Here, the population size was set to 512 creatures per generation, and the mutation rate was set to 0.02.

In the current test environment, we terminate algorithm execution at the 500th generation, which is sufficient to observe error dynamics between generations and obtain an improved solution compared to the conventional method.

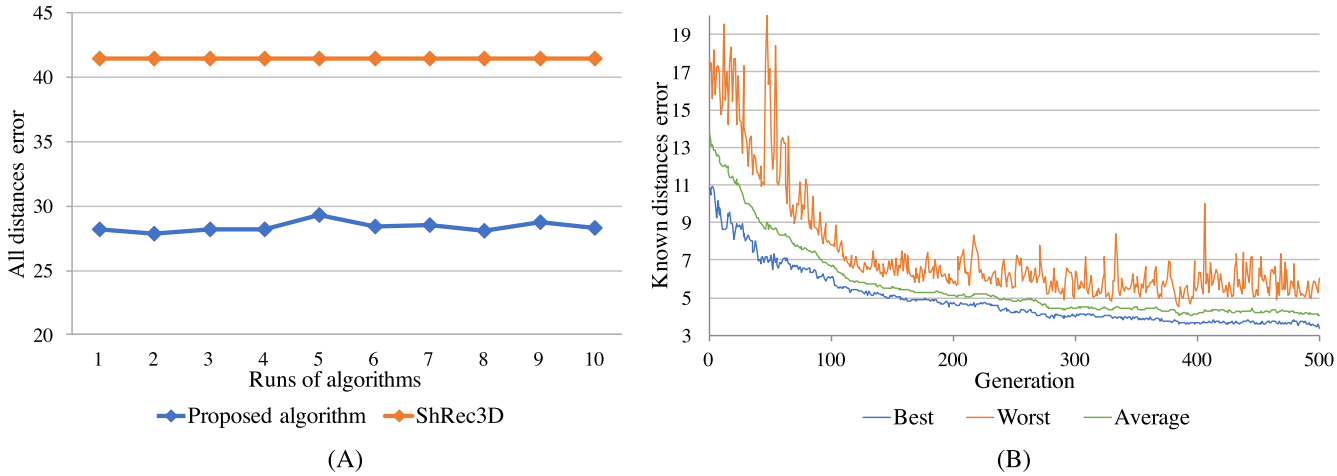

Fig. 5. (A) All distances error of models reconstructed by *ShRec3D* and the proposed algorithm. (B) Dynamics of known distances error during execution of the proposed algorithm [size of the model is 500 loci].

4.3 Experimental Results

The distribution of *all distances errors* for the models constructed using distance matrices of the best creatures across all runs is shown in Fig. 5A. The abscissa values show 10 separate runs of both algorithms with the same input. Here, the orange line represents *all distances errors* for the *ShRec3D* output for each run, which are identical due to the deterministic nature of *ShRec3D*. The blue line shows *all distances errors* for the output of the proposed algorithm. The *all distances error* for the model generated by *ShRec3D* was *41.4*. The average *all distances error* across all runs of the proposed algorithm with the same model was *28.31*, i.e., approximately 1.5 times smaller than that of the *ShRec3D* result.

The stochastic nature of optimization by the genetic algorithm raises question regarding the consistency of the obtained result. To evaluate the stability of the solution, the proposed algorithm was executed 10 times with the same input and parameters. Due to the random nature of the genetic algorithm, each result differed slightly; however, convergence was observed in each run. *All distances errors* were reduced by approximately the same ratio in each run (Fig. 5A).

The smallest error, greatest error, and average error (Fig. 5B) were calculated for each generation to evaluate convergence and population diversity dynamics.

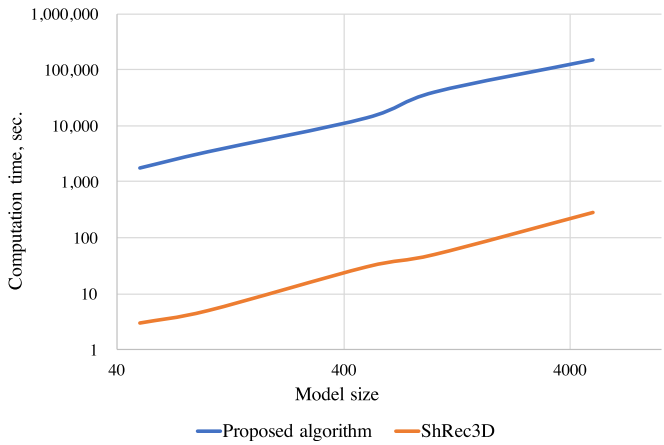

Fig. 6. Computation time for *ShRec3D* and the proposed algorithm.

4.4 Computation Time

Both algorithms were executed several times with different model sizes to evaluate computation time. Here, we used the genetic algorithm parameters discussed in Section 4.2. The average times required to reconstruct 3D models of various sizes are shown in Fig. 6. The abscissa values show the size of the simulated input model (Section 4.1). The ordinate values show the time required for 3D model reconstruction for both algorithms.

Measurements were performed on a consumer-grade computer (3.1 GHz Intel Core i5 processor, 16 GB 2,133 MHz LPDDR3 RAM). The algorithms were implemented and executed using the Python interpreted programming language.

5 Discussion

In this paper, we have proposed an algorithm for 3D reconstruction of chromatin structure from 3C experimental data. The proposed algorithm was compared to the conventional *ShRec3D* method. The experimental results demonstrate that the *all distances error* of the model reconstructed using proposed algorithm was approximately 1.8 times smaller than that of the model reconstructed by *ShRec3D*. The all distances error metric is the root mean square error between all distances in the original and reconstructed models; thus, it quantitatively represents the quality of the reconstructed model and therefore the quality of the reconstruction algorithm. Given the observed improvement to the *all distances error* metric, the proposed algorithm is considered a significant improvement in terms of the quality of the output model.

Authors of *ShRec3D* compared it to other algorithms like BATCH [8] and ChromSDE [9] which use global optimization approach and therefore can be used only for 3C experimental data with very small resolution up to hundreds of loci. It was shown that *ShRec3D* is better than global optimization algorithms in all aspects; thus, we assume that comparisons to *ShRec3D* are sufficient to determine the best available algorithm for 3D model reconstruction.

Even though the proposed algorithm objectively produces better results, it requires more computation time compared to the conventional *ShRec3D* method. The computation times of the conventional and proposed algorithms (Fig. 6) were analyzed using simulated input in the same environment. As

expected, the proposed algorithm was slower by approximately 1,000 times and was computation time linearly dependent on the genetic algorithm parameters, such as population size and the number of generations. Considering that the conventional algorithm is very fast, the computation time of the proposed algorithm is reasonable for reconstruction of chromatin 3D structure tasks, which are performed infrequently and have no strict calculation time limitations. Note that several studies have examined the parallelization of genetic algorithms [14], [15], [16], [17], [18], and many effective solutions have been proposed. Therefore, the calculation time of the proposed algorithm could be improved significantly by an effective parallel implementation.

One problem with the genetic algorithm approach is determining effective stop conditions. The proposed algorithm performs iterative optimization; thus, it can run infinitely if not terminated by predefined stop conditions. Depending on the given objective, various tactics can be used to determine effective stop conditions.

- Fixing the number of generations. The advantage of this approach is that execution time is predictable; however, there is no guaranteed method to predict the number of generations that will yield the best result.
- Limiting allowed execution time. This approach has the same advantages and disadvantages as fixing the number of generations but increases the predictability of execution time.
- Applying a threshold to the *known distances error* convergence speed. This is considered the most effective method in terms of the quality of the final result; however, in some cases, limiting convergence speed could lead to local optima results.

For any approach, when the algorithm terminates, the *best creature* in the final generation is selected as the final solution. In this study, we fixed the number of generations to make all runs as similar as possible. For real 3C experimental data, we believe it would be preferable to use a combination of convergence speed thresholding and an execution time limit. This would guarantee that some results will be obtained in predictable time if convergence is stable and slow, but in the same time will give result much faster if convergence is fast enough.

6 Conclusions

The 3D reconstruction of DNA structure leads to better understanding of DNA regulation mechanisms. Currently, existing algorithms for chromatin structure reconstruction using 3C experimental data have some limitations and are being improved continuously.

The MDS-based *ShRec3D* algorithm introduces error in the output model due to the use of a graph shortest path search algorithm. In contrast, the proposed algorithm utilizes a genetic algorithm approach to optimize the distances calculated by the graph shortest path search algorithm.

The proposed algorithm demonstrated significant improvement to the output model compared to the conventional *ShRec3D* algorithm. However, in some situations, due to the increased reconstruction time of the proposed algorithm, it may be not practical for very high-resolution whole-genome 3C experimental data. In terms of computation time, the combination of deterministic MDS and metaheuristic optimization by a genetic algorithm places the proposed algorithm between the deterministic *ShRec3D* approach and traditional global optimization algorithms like BATCH [8] and ChromSDE [9]. Global optimization algorithms are much easier to implement but are only useful for very low-resolution data.

The proposed algorithm's convergence speed and the quality of the output model could be improved by adjusting the genetic algorithm parameters, e.g., mutation rate and population size, which will be the focus of future work.

The source code for the proposed algorithm implementation and the sample 3C experimental data generator are available at *https://github.com/skkap/mdsga*.

7 Terminology

Here, we define unconventional terms used in this paper. Some terms, e.g., *creature genome*, are genetic algorithm terminology that could be confused for biological terms because this paper is related to bioinformatics.

Original model: 3D structure representing real chromatin conformation. This model is not available in real 3C experiments, but it is known when using simulated input.

Known distances: Distances directly obtained from the 3C experimental data. These distances never change during algorithm execution.

Calculated distances: Distances not initially available from the 3C experimental data that are calculated and optimized during algorithm execution.

Creature: Single candidate solution obtained during genetic algorithm execution.

Creature genome: Set of all unknown distances (*calculated distances*) required to reconstruct the full distance matrix.

Population: Set of creatures that belong to a single generation.

Base creature: Creature that serves as a base when generating the initial population.

Initial population: Pre-generated population used in the first generation of the genetic algorithm.

Known distances error: Root mean square error between the *known distances* and the same distances in the model reconstructed from the distance matrix constructed using a particular creature's genome.

All distances error: Root mean square error between all distances in the *original model* and the same distances in the model reconstructed from the distance matrix constructed using a particular creature's genome. This value cannot be calculated from real experimental data because all original distances are unknown. It is used to evaluate the final result when using simulated input.

Best creature: Creature with the smallest *known distances error* within its population.

Acknowledgments

This study was partially supported by JSPS KAKENHI Grant Numbers 15K00402, 16H03261, 262801060, and 16K125250.

REFERENCES

[1] C. L. Woodcock and S. Dimitrov, "Higher-order structure of chromatin and chromosomes," *Current Opinion Genetics Develop.*, vol. 11, no. 2, pp. 130–135, 2001.

[2] J. Feser and J. Tyler, "Chromatin structure as a mediator of aging," *FEBS Lett.*, vol. 585, no. 13, pp. 2041–2048, 2011.

[3] M. A. Morgan and A. Shilatifard, "Chromatin signatures of cancer," *Genes Develop.*, vol. 29, no. 3, pp. 238–249, 2015.

[4] J. R. Dixon, I. Jung, S. Selvaraj, Y. Shen, J. E. Antosiewicz-Bourget, A. Y. Lee, Z. Ye, A. Kim, N. Rajagopal, W. Xie, Y. Diao, J. Liang, H. Zhao, V. V. Lobanenkov, J. R. Ecker, J. A. Thomson, and B. Ren, "Chromatin architecture reorganization during stem cell differentiation," *Nature*, vol. 518, no. 7539, pp. 331–336, 2015.

[5] J. Dekker and L. Mirny, "The 3D genome as moderator of chromosomal communication," *Cell*, vol. 164, no. 6, pp. 1110–1121, 2016.

[6] J. Dekker, K. Rippe, M. Dekker, and N. Kleckner, "Capturing chromosome conformation," *Sci.*, vol. 295, no. 5558, pp. 1306–1311, 2002.

[7] J. M. Belton, R. P. McCord, J. H. Gibcus, N. Naumova, Y. Zhan, and J. Dekker, "Hi-C: A comprehensive technique to capture the conformation of genomes," *Methods*, vol. 58, no. 3, pp. 268–276, 2012.

[8] M. Hu, K. Deng, Z. Qin, J. Dixon, S. Selvaraj, J. Fang, B. Ren, and J. S. Liu, "Bayesian inference of spatial organizations of chromosomes," *PLoS Comput. Biol.*, vol. 9, no. 1, 2013, Art. no. e1002893.

[9] Z. Zhang, G. Li, K.-C. Toh, and W.-K. Sung, "3D chromosome modeling with semi-definite programming and Hi-C data," *J. Comput. Biol.*, vol. 20, no. 11, pp. 831–846, 2013.

[10] A. Lesne, J. Riposo, P. Roger, A. Cournac, and J. Mozziconacci, "3D genome reconstruction from chromosomal contacts," *Nature Methods*, vol. 11, no. 11, pp. 1141–1143, 2014.

[11] J.-B. Morlot, J. Mozziconacci, and A. Lesne, "Network concepts for analyzing 3D genome structure from chromosomal contact maps," *EPJ Nonlinear Biomed. Physics*, vol. 4, no. 1, 2016, Art. no. 2.

[12] R. W. Floyd, "Algorithm 97: Shortest path," *Commun. ACM*, vol. 5, no. 6, 1962, Art. no. 345.

[13] E. Lieberman-Aiden and N . van Berkum, "Comprehensive mapping of long range interactions reveals folding principles of the human genome," *Sci.*, vol. 326, no. 5950, pp. 289–293, 2009.

[14] V. Sumati, "Parallel compact genetic algorithm on CUDA-C platform," *Int. J. Comput. Appl.*, vol. 84, no. 5, pp. 13–16, 2013.

[15] R. Faraji and H. R. Naji, "An efficient crossover architecture for hardware parallel implementation of genetic algorithm," *Neurocomput.*, vol. 128, pp. 316–327, 2014.

[16] S. Zhang and Z. He, "Implementation of parallel genetic algorithm based on CUDA," in *Proc. Int. Symp. Intell. Comput. Appl.*, 2009, pp. 24–30.

[17] P. Pospichal, J. Jaros, and J. Schwarz, "Parallel genetic algorithm on the CUDA architecture," in *Proc. Int. Conf. Appl. Evolutionary Comput.*, 2010, pp. 442–451.

[18] Y. Y. Liu and S. Wang, "A scalable parallel genetic algorithm for the Generalized Assignment Problem," *Parallel Comput.*, vol. 46, pp. 98–119, 2015.

Viacheslav Kapilevich received the bachelor's and master's degrees in computer science from the National Research Tomsk Polytechnic University, Russia. He studied high-performance computing with the Brno University of Technology, Czech Republic. He has wide practical experience in applied software engineering in various areas. As a research student, he did various research in areas of molecular biology, genomics, and artificial evolution with Osaka University. Now, he is doing research as a bioinformatician in the field of chromatin reconstruction and epigenetics with the Graduate School of Information Science and Technology, Osaka University.

Shigeto Seno received the BE, ME, and PhD degrees from Osaka University, in 2001, 2003, and 2006, respectively. He is an associate professor with the Graduate School of Information Science and Technology, Osaka University. He is a member of IPSJ, JSBi, and MII. His research interests include bioinformatics and bioimage-informatics.

Hideo Matsuda received the BSc degree in physics from Kobe University, Japan, in 1982 and the MEng and PhD degrees in computer science from Kobe University, Japan, in 1984 and 1987, respectively. He has been a professor with the Department of Bioinformatic Engineering, Graduate School of Information Science and Technology, Osaka University, since 2002. His research interests include genomic data analysis, gene regulatory networks, and gene expression analysis.

Yoichi Takenaka received the MEng and PhD degrees in computer science from Osaka University, Japan, in 1997 and 2000, respectively. He was an assistant professor from 2000 to 2002 with Osaka University. He was an associate professor with the Department of Bioinformatic Engineering, Graduate School of Information Science and Technology, Osaka University, from 2002 to 2017. He has been a professor with the Faculty of Informatics, Kansai University Japan since 2017. His research interests include gene expression analysis, highthroughput sequence analysis, legal informatics, and data science.