JIS Journal of Interdisciplinary Sciences
Volume 1, Issue 1; 49-60
November, 2017. © The Author(s)

A Genome Scaffold Algorithm for Extending Repetitive and Non-repetitive Contigs

Shuaibin Lian*, Gang Zheng, Ke Gong, Xiaoli Zhang and Jiantao Guo

College of Physics and Electronic Engineering, Xinyang Normal University, Xinyang, China
shuai_lian@qq.com*

Abstract: *The next generation sequencing platform produces massive short reads data. It is needed to assemble these short reads for further analysis. Unfortunately, gene assembly algorithm is still faced by a variety of complex challenges, especially for repetitive sequences. Based on De Bruijn Graph and SWA, we proposed a new genome scaffold algorithm named HashRepScaffold for extending repetitive contigs and non-repetitive contigs. By aligning original reads back to both ends of repetitive contigs and non-repetitive contigs, HashRepScaffold finds the best repetitive contig to exend the non-repetitive contig. Finally, simulation study was used to evaluate the performances of HashRepScaffold. The results indicated that HashRepScaffold have an excellent performances in extending repetitive and non-repetitive contigs.*

Keywords*:* The Next Gene Sequencing, Gene Assembly, Hash Index, Mapping Conditions, Repetitive and Non-repetitive Contigs

Introduction

Gene sequencing technology has been developed for more than 40 years, and the next gene sequencing(NGS) technology has occupied an absolute advantage in the global sequencing market (Pareek et. al. 2011). Now, NGS platforms mainly include 454 technology from Roche, Solexa, GAllx from Illumina, and SOLiD from Applied Biosystems (Reis-Filho, 2009). These sequencing platforms produces massive amounts of sequencing data, which inspired researchers assemble the data to obtain a complete DNA (Genome 10K Community of Scientists, 2009). NGS give a great impetus to species research, major disease treatment, precision medical exploration and agricultural product breeding (Treangen and Salzberg 2012).

In addition, there are a great number of repeats in the genome of eukaryotes. For instance, the repeats accounts for about 10% to 55% of the mammalian genome (Wolfe et. al. 1989), which accounts for about 45% of the human genome (International Human Genome

Consortium, 2001). The repeats play an important role in the genetic evolution of organisms as an important function-related gene for life activities. Currently, a gene splicing tool named SWA (Lian et al. 2014) can assemble short sequences into repetitive and non-repetitive contigs. However, extending repeats and non-repeats directly may lead to incorrect assembling results. Therefore, we need a new algorithm to solve the problem.

In order to extend the repeats and non-repeats accurately, we proposed a new genome scaffold algorithm named HashRepScaffold. HashRepScaffold has the following attributes: 1) the assembly process focuses on the assembly of the entire gene as a global assembly rather than partial assembly; 2) build longer scaffolds by using short pairing information and comparing the degree of correlation between the original reads. The simulation datasets are used to evaluate the accuracy of HashRepScaffold. In simulation study, we evaluate the stability of HashRepScaffold to parameters, including sequencing depth, read length, types and lengths of repeats. Results indicates that HashRepScaffold is an accurate and complete scaffold algorithm for extending repetitive and non-repetitive contigs.

Results

The highlight of HashRepScaffold is to use the paired-data of short sequences to determine the positional relationship between these contigs. The key and difficulty of HashRepScaffold is that the establishment of hash index and the optimal overlap range is set, which can increase the speed of calculation as well as the continuity of the assembly results and improves the overall performance of the assembly results.

HashRepScaffold consists of four steps: data preprocessing and generating paired-data, constructing hash index, calculate mapping relations, extending non-repetitive contigs. The detailed steps and procedures are described in detail below.

1) Data preprocessing and generating paired-data (Figure 1). HashRepScaffold firstly purifies the raw data: remove any raw reads data containing the letter “N” and delete the wrong reads. The remaining available reads can generate paired-data. The more paired information between two contigs, the stronger the association between the two contigs. And then the parameters are set in advance: fragment length, read length, sequencing depth, the left side data of the original reads br and the right side data of the original reads bl, repetitive contigs, non-repetitive contigs, read count threshold.

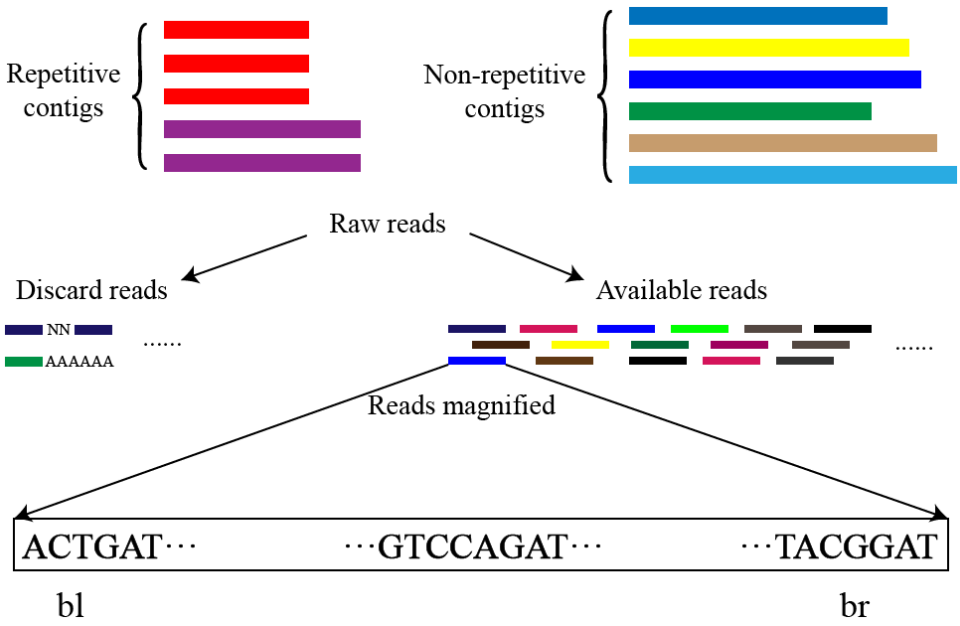


Figure 1. Data preprocessing and generating paired-data schematic. Repetitive contigs and non-repetitive contigs are generated by other sequence splicing tools. The available reads are divided into two parts: the left end is bl as well as the right end is br.

2) Constructing hash index (Figure 2). HashRepScaffold constructs a forward hash table based on the first 10 characters of the short read sequence and constructs a backward hash table based on the last 10 characters. Short reads are converted to quaternary integer and convert these quaternary integers into decimal integers to the hash table. Directly to the decimal integer to retrieve rather than search for characters, the complexity of this indexing strategy is only $L \times N / 4\hat{}10$, which L represents the length of the reference genome and N represents the length of each single short reads. This structure greatly improves the speed of calculation, especially for the preparation of short sequence of genes, which is more efficient than traditional indexing.

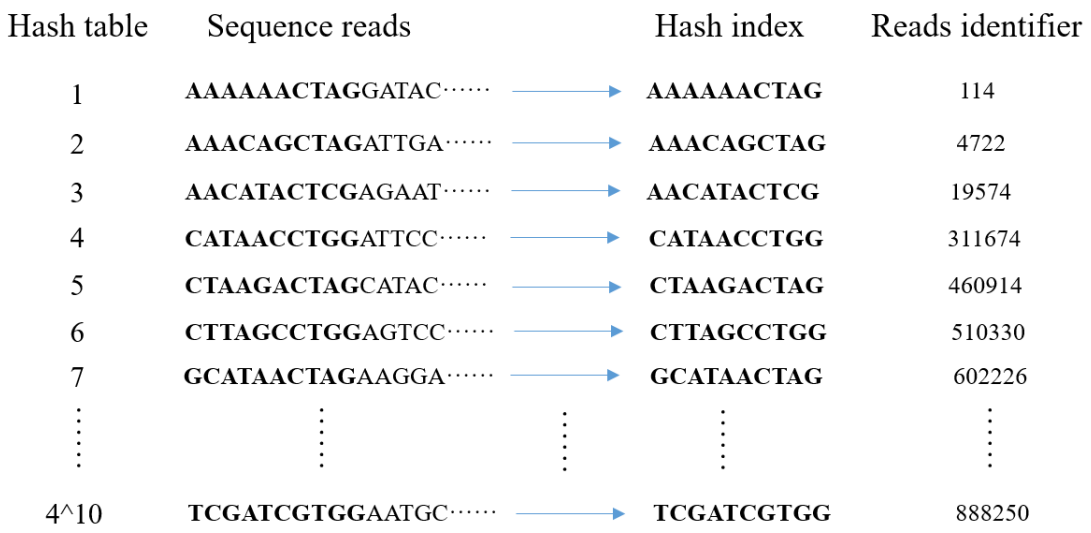
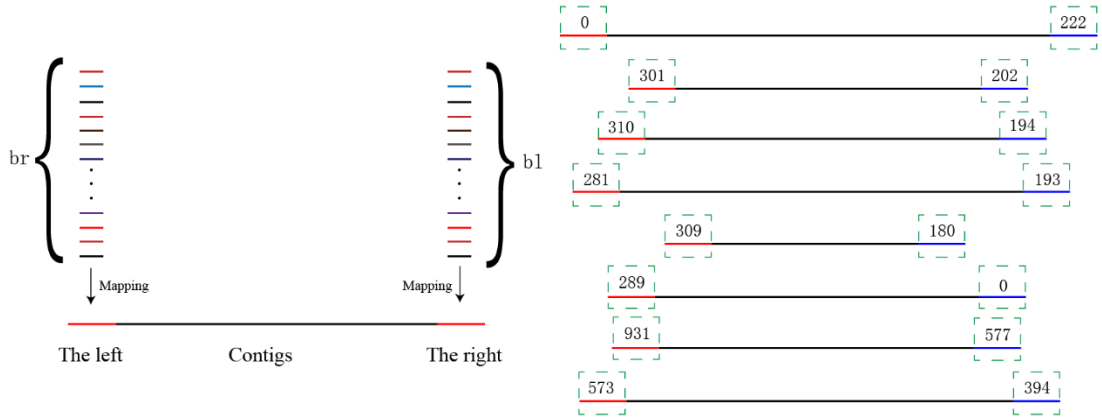


Figure 2. Schematic of a forward hash index tactic. The first 10 characters of sequence reads are converted to quaternary: A, C, G and T are represented by characters 0, 1, 2 and 3, respectively. Then convert these quaternary integers into decimal integers stored in the Hash table to retrieve, these decimal integers are reads identifier. For example, the first 10 characters of the obtained short read sequence: TACGACTAAG, the short read is converted to quaternary integer 3012013002, which is finally converted to a decimal number 811458 to the hash table. The corresponding string of the 811458 bit is the hash value of the short read.

3) Calculate mapping relations (Figure 3). After establishing the hash index, calculate the mapping relation between the left end of each contig and br, the mapping relation between the right end of each contig and bl, which can determine the number of br and bl on the left and right sides of each contig. Map all of the br and bl to each contig (Figure 3(a)). After the mapping operation is completed, the position of each contig in the reference sequence and the direction of extension are determined according to the number of br or bl matches at the left and right ends of each contig (Figure 3(b)). Record the position and number of matches for each contig respectively.



(a) (b)

Figure 3. Calculate mapping relations. (a) Map all br to the left of each contig as well as all bl map to the right of each contig according to the established hash table. (b) When the number of the contig's left end matches all of br is 0, it shows that this contig is the left end of the reference gene; When the number of the contig's right end matches all of bl is 0, it shows that this contig is the right end of the reference gene; Otherwise, the contig is in the middle of the reference gene.

4) Extending non-repetitive contigs (Figure 4). According to the calculated position and number of matches for each contig, all of the non-repetitive contigs are extended or bidirectional extended. Find the maximum intersection of each contig on br and bl to determine which repetitive contig is merged with non-repetitive contig. At last, each non-repetitive contig are extended with a repetitive contig or two repetitive contigs, while the scaffolds are generated.

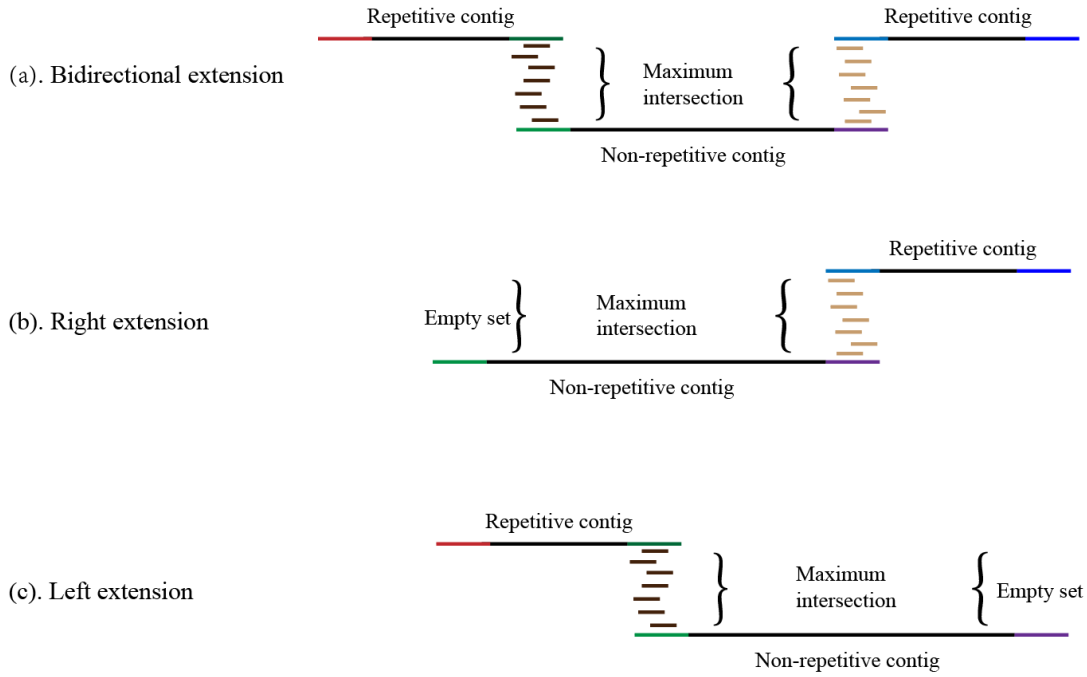


Figure 4. Extending non-repetitive contigs. (a) When all of the br that matched the left end of a non-repetitive contig as well as all of the bl that matched the right end of a non-repetitive contig is a non-empty set, the non-repetitive contig is bidirectional extended, assembled with the two corresponding repetitive contigs. (b) When all of the br that matched the left end of a non-repetitive contig is a non-empty set as well as all of the bl that matched the right end of a non-repetitive contig is an empty set, the non-repetitive contig is right extended, assembled with the corresponding repetitive contig. (c) When all of the bl that matched the right end of a non-repetitive contig is a non-empty set as well as all of the br that matched the left end of a non-repetitive contig is an empty set, the non-repetitive contig is left extended, assembled with the corresponding repetitive contig.

Assessments

Metrics

In this section, the detailed performance evaluation of HashRepScaffold proposed in this paper is performed on large, complex, and diverse simulation datasets. The simulation datasets are mainly used to verify the performance of the new algorithm and evaluate the effect of parameter settings on algorithm performance. We use some commonly used metrics including Type Number of Repetitive Contigs, Copy Number of each Repetitive Contig, Scaffold N50 (Leggett et. al 2013), Repeat-accuracy, Non-Repeat-accuracy, Total-accuracy

to assess the performance of HashRepScaffold. The specific definition of the metrics is as follows:

1) Type Number of Repetitive Contigs (TNRC): TNRC is the type numbers of repeats; Copy Number of each Repetitive Contig (CNRC): CNRC is the copy numbers of each corresponding type of repeat; Scaffold N50 is the median contig size of the genomic assembly, in general, the larger the Scaffold N50, the longer the Scaffold is, and the better the algorithm performance.

2) Rep-accuracy (R-acc): it is the correct rate of repetitive contigs on the assembled scaffolds and it is defined as follows:

$$R\text{-}acc = 1 - \frac{R_{error}}{R_{total}}$$

Where R-acc is the accuracy for correct assembly of repetitive contigs, Rerror is the number of the repetitive contigs that are not assembled or incorrectly involved in assembling. Rtotal is the number of all the repetitive contigs.

3) Non-Repeat-accuracy (NR-acc): it is the correct rate of non-repetitive contigs on the assembled scaffolds and it is defined as follows:

$$NR\text{-}acc = 1 - \frac{NR_{error}}{NR_{total}}$$

Where NR-acc is the accuracy for correct assembly of non-repetitive contigs, NRerror is the number of the non-repetitive contigs that are not assembled or incorrectly involved in assembling. NRtotal is the number of all the non-repetitive contigs.

4) Total-accuracy (T-acc): the global correct rate of repetitive contigs and non-repetitive contigs on the assembled scaffolds, which it is defined as follows:

$$T\text{-}acc = 1 - \frac{R_{error} + NR_{error}}{R_{total} + NR_{total}}$$

Where T-acc is the global accuracy for correct assembly of repetitive contigs and non-repetitive contigs. The higher the value of R-acc, NR-acc and T-acc, the better the algorithm performance.

Assessment firstly inserts a different sequence of repetitions into the reference sequence and repeats it several times to obtain a complete sequence. All of the repetitive contigs and non-repetitive contigs are generated by SWA Algorithm. In order to evaluate these parameters, the metrics, such as Scaffold N50, Repeat-accuracy, Non-Repeat-accuracy and Total-

accuracy are all computed by aligning the corresponding items back to the reference genome using swalign function in MATLAB R2015a.

Performances on different types of repeats

In this part, we evaluated the performances of HashRepScaffold on three types of repeats, including interspersed repeats, tandem repeats and compound repeats of the simulation datasets, respectively. Table 1 shows the algorithmic performance of assembling sequences with different types of repeats when the sequencing depth is 2, the length of the simulation genome is 100000bp, other parameters of the same circumstances. Repetitive contigs and non-repetitive contigs is generated by SWA Algorithm with the parameters: kmer = 10, read lenth=50, the threshold value = 50. The length of these contigs smaller than 300 is removed.

From Table 1, three types of repeats were used to validate the performances of HashRepScaffold. The accuracy of the proposed algorithm is 100% in the genome of interspersed repeats, and the assembled Scaffold N50 is longer than other types of sequences. However, the accuracy of the new algorithm is unsatisfactory in the genome of tandem repeats and compound repeats. Due to the large number of consecutive repetitive contigs, the algorithm cannot effectively extend it with non-repetitive contigs, and the Scaffold N50 is also relatively short. The result indicated that HashRepScaffold can have a very excellent performance in in the genome of interspersed repeats, it can be completely assembled correctly the scaffolds out and have a longer Scaffold N50.

Table 1. Performances on different types of repeats

Types of repeats	TNRC	CNRC	Scaffold N50 (bp)	R-acc	NR-acc	T-acc
Interspersed repeats	6	2, 3, 5, 3, 3, 4	4025	100%	100%	100%
Tandem repeats	7	5, 6, 2, 3, 4, 2, 4	2056	75.88%	100%	84.07%
Compound repeats	5	3, 5, 2, 5, 4	3599	89.95%	100%	96.18%

Performances on different coverage depth

Performances on different coverage depth is shown in Table 2. We used simulated genomic datasets with a total length of 100000bp and the repetitive type of interspersed repeats, which the TNRC is 7 and the CNRC repeat 4, 5, 3, 2, 3, 3 and 4 times respectively. Repetitive contigs and non-repetitive contigs is generated by SWA Algorithm with the parameters: kmer = 10, read lenth=50, the threshold value = 50. The length of these contigs smaller than 300 is removed.

As can be seen from Table 2, the coverage depth of HashRepScaffold has a greater impact on the performance. When the coverage depth exceeds 0.8, the proposed algorithm in this section can have a high correct rate, most of the technical indicators are excellent. However, when the coverage depth drops to 0.4, the algorithm performance begins to drop sharply and distortion occurs. The higher the coverage, the total amount of output data greater, take longer and be more costly provision gene assembly. Therefore, when the genome sequence length is around 100000bp, the algorithm is suitable for the optimal coverage depth of 0.8.

Table 2. Performances on different coverage depth

Coverage depth	TNRC	CNRC	Scaffold N50 (bp)	R-acc	NR-acc	T-acc
1	7	4, 5, 3, 2, 3, 3, 4	3156	100%	100%	100%
0.8	7	4, 5, 3, 2, 3, 3, 4	3081	98.03%	100%	99.62%
0.6	7	4, 5, 3, 2, 3, 3, 4	2673	85.34%	99.81%	93.25%
0.4	7	4, 5, 3, 2, 3, 3, 4	1263	69.2%	85.17%	78.56%

Performances on different fragment length

Performances on different fragment length is shown in Table 3. We continued to use the simulation datasets with the total length of 100000bp, the sequencing depth is 2. Performance is evaluated by using interspersed repeats, which the TNRC is 6 and the CNRC repeat 2, 3, 5, 3, 3, and 4 times respectively. To better analyze the effect of fragment length on performance, estimate the appropriate segment length and set the value from 150bp to 240bp. Repetitive contigs and non-repetitive contigs is generated by SWA Algorithm with the

parameters: kmer = 10, read lenth=50, the threshold value = 50. The length of these contigs smaller than 300 is removed.

From table 3, we can see that when the fragment length reaches 240bp, whether the Scaffold N50 or the overall accuracy of the assembly indicators are excellent. When the fragment length is less than 235bp, all the performance indicators begin to decline, especially the very short fragment length is difficult to achieve the technical indicators. HashRepScaffold have a large error in the length of the fragment less than 240bp for repeat and non-repetitive contigs assembly, which indicate that the repeat and non-repetitive contigs are not assembled or are incorrectly assembled. Therefore, HashRepScaffold is suitable for the first gene sequencing (the fragment length is about 800bp) and NGS of 454 sequencing platform (the fragment length is about 400bp).

Table 3. Performances on different fragment length

Fragment length	TNRC	CNRC	Scaffold N50 (bp)	R-acc	NR-acc	T-acc
260	6	2, 3, 5, 3, 3, 4	4534	100%	100%	100%
240bp	6	2, 3, 5, 3, 3, 4	4534	100%	100%	100%
235bp	6	2, 3, 5, 3, 3, 4	3825	98.36%	99.37%	99.06%
220bp	6	2, 3, 5, 3, 3, 4	2168	93.5%	88.32%	90..53%
200bp	6	2, 3, 5, 3, 3, 4	1263	82.94%	79.37%	80.3%
150bp	6	2, 3, 5, 3, 3, 4	864	75.12%	61.3%	67.8%

Conclusions and Discussions

All of the eukaryotic genomes contain massive repetitive sequences of different lengths and occupy an important proportion in DNA (Lander et. al. 2001). Recent studies have found that trinucleotide repeat disease has provoked a series of repetitive researches (B A de Vries, et al, 1999). Studies have also shown that the repeats can produce abnormal structures of DNA sequences, leading to the occurrence of polymorphisms and the proliferation of copies (Woodford et. al. 1997). These repetitive genes play a huge role in biological evolution, in order to better explore these repetitive genes, we developed a genome scaffold algorithm for extending repeats and non-repetitive contigs named HashRepScaffold, which can assemble

repetitive contigs and non-repetitive contigs into scaffolds independently. The results indicated that 1) HashRepScaffold can have a very excellent performance in in the genome of interspersed repeats, it can be completely assembled correctly the scaffolds out. 2) HashRepScaffold is suitable for the first gene sequencing and NGS of 454 sequencing platform. 3) HashRepScaffold is adaptable and can run on microcomputers and low coverage depth can accurately assemble Scaffolds. Overall, HashRepScaffold is an accurate repetitive contigs and non-repetitive contigs assembling tool.

Different from the whole genome de novo assembly algorithm (Salzberg et. al. 2012) based on the next generation of sequencing, HashRepScaffold independently assembles the repetitive and non-repetitive regions, preprocessing data and constructing the Hash index first, which greatly reduces the memory footprint of the computer and saves time. Nevertheless, this work is required further study to solve the following problems: 1) Similarity of repeats: repeats can be divided into the same repetition and highly similar repetition, which the same repetition is easy to be detected and assembled but for the highly similar repetition positioning becomes difficult. 2) Types of repeats: different types of repeats especially tandem repeats and compound repeats can lead to the wrong assembly. The new algorithm proposed in this paper has different advantages and applications. HashRepScaffold have an excellent performance in extending repeats and non-repeats, and it should be preferred for the scaffold algorithm.

References

de Vries BB, Mohkamsing S, van den Ouweland AM, Mol E, Gelsema K, van Rijn M, Tibben A, Halley DJ, Duivenvoorden HJ, Oostra BA, Niermeijer MF. 1999. Screening for the fragile X syndrome among the mentally retarded: a clinical study. *Medical Genetics*, 36, 467–470. http://dx.doi.org/10.1136/jmg.36.6.467

Genome 10K Community of Scientists. 2009. Genome 10K: A Proposal to Obtain Whole-Genome Sequence for 10,000 Vertebrate Species. *Journal of Heredity*, 100, 659-674. https://doi.org/10.1093/jhered/esp086

International Human Genome Consortium. 2001. Initial Sequencing and Analysis of the Human Genome. *Nature*, 409, 860-921. https://doi.org/10.1038/35057062

Jorge S Reis-Filho. 2009. Next-generation sequencing. *Breast Cancer Research*, 11, S12. https://doi.org/10.1186/bcr2431

Lander ES, Linton LM, Birren B, Nusbaum C, Zody MC, et al. 2001. Initial sequencing and analysis of the human genome. *Nature*, 409, 860-921. http://dx.doi.org/10.1038/35057062

Shuaibin Lian, Qingyan Li, Zhiming Dai, Qian Xiang, and Xianhua Dai. 2014. *A De Novo Genome Assembly Algorithm for Repeats and Nonrepeats*. Hindawi Publishing Corporation, 2014, Article ID 736473. http://dx.doi.org/10.1155/2014/736473

Richard M. Leggett, Bernardo J. Clavijo, Leah Clissold, Matthew D. Clark, and Mario Caccamo. 2013. NextClip: an analysis and read preparation tool for Nextera Long Mate Pair libraries. *Bioinformatics*, 30, 566-568. https://doi.org/10.1093/bioinformatics/btt702

Pareek, C.S. Rafal Smoczynski, Andrzej Tretyn. 2011. Sequencing technologies and genome sequencing. *Journal of Applied Genetics*, 52, 413-435. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3189340

Salzberg SL, Phillippy AM, Zimin A, Puiu D, Magoc T, Koren S, Treangen TJ, Schatz MC, Delcher AL, Roberts M, Marçais G, Pop M, Yorke JA. 2012. GAGE: A critical evaluation of genome assemblies and assembly. *Genome Research*, 22, 557-567. http://genome.cshlp.org/content/genome/22/3/557

Shuaibin Lian, Qingyan Li, Zhiming Dai, Qian Xiang, and Xianhua Dai. 2014. *A De Novo Genome Assembly Algorithm for Repeats and Nonrepeats*. Hindawi Publishing Corporation, 2014, Article ID 736473. http://dx.doi.org/10.1155/2014/736473

Treangen, T.J. and Salzberg, S.L. 2012. Repetitive DNA and Next Generation Sequencing: Computational Challenges and Solutions. *Nature Reviews Genetics*, 13, 36-46. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3324860

Wolfe K.H., Sharp P.M., Li W.H. 1989. Mutation rates differ among regions of the mammalian genome. *Nature*, 337, 283-285. https://www.nature.com/nature/journal/v337/n6204/abs/337283a0

Woodford K.J, Usdin K, Weitzmann M.N. 1997. DNA Secondary Structures and the Evolution of Hypervariable Tandem Arrays. *Journal of Biological Chemistry*, 272, 9517-9523. http://www.jbc.org/content/272/14/9517.short

Paper Received September 24th, Accepted October 8th, Published 2nd November 2017