Survey of gene splicing algorithms based on reads

Xiuhua Si a, Qian Wangb, Lei Zhang a, Ruo Wua, and Jiquan Maa

Abstract
Gene splicing is the process of assembling a large number of unordered short sequence fragments to the original genome sequence as accurately as possible. Several popular splicing algorithms based on reads are reviewed in this article, including reference genome algorithms and de novo splicing algorithms (Greedy-extension, Overlap-Layout-Consensus graph, De Bruijn graph). We also discuss a new splicing method based on the MapReduce strategy and Hadoop. By comparing these algorithms, some conclusions are drawn and some suggestions on gene splicing research are made.

Introduction
Rapidly and accurately obtaining genetic information is a highly significant task in life science research. The genome of each organism contains its entire genetic information, which is usually determined by the order of base pairs in the DNA or RNA molecules. Gene splicing is an important research area, as it can be used to reveal the complexity and diversity of the genome within a species.

Gene splicing can be viewed as the process of searching for an optimal mapping from read to reference sequence. There are two kinds of algorithms: reference genome-based and de novo splicing algorithms. Until 2005, gene splicing was mainly based on Sanger sequencing technology. With the development of large-scale parallel sequencing, next-generation sequencing technology has replaced the traditional sequencing method. Compared with Sanger sequencing, next-generation sequencing technology has many advantages, such as high speed, high throughput and low cost. The sequencing technologies commonly used in industry include Illumina’s Solexa sequencing technology,1 ABI’s SOLiD sequencing technology,2 and Roche’s 454 sequencing technology.3

In this paper, challenges in gene splicing research are first explained. Then, common gene splicing algorithms are summarized, including splicing algorithms based on the reference genome, greedy algorithms and algorithms based on OLC graphs. Finally, algorithms based on De Bruijn graphs are presented.

Challenges for next-generation sequencing technologies
In contrast to the traditional method, next-generation sequencing has many advantages, such as high sequencing throughput and short read lengths. However, low sequencing accuracy and short sequencing read lengths are challenges to gene splicing.

Sequencing read length
Due to the limitations of sequencing, the sequencer needs to break molecules for sequencing, and consequently, the lengths of the read sequences are relatively short. The read sequences obtained by first-generation sequencing techniques are 500–1000 bp. The read lengths measured by next-generation sequencing technology are shorter than those in first-generation sequencing technology, at only 25–35 bp. Because the read sequences are shorter, the number of reads obtained by sequencing is much higher than that in traditional Sanger sequencing data. For shorter read sequences, it is more difficult to judge the overlap between reads because there are many repetitions between reads, and many reads have only one base difference or no base differences.
Sequencing accuracy

The accuracy of the sequencing fragments in next-generation sequencing technology is 10 times lower than that of the Sanger technique. A large number of sequencing errors is contained in the large amount of sequencing data, which affects the results of splicing. Currently, two methods have been proposed to address this problem: a) Each sequencing segment has a sequencing quality score. A lower score indicates that the sequencing error is more likely to occur at that location. In a sequencing fragment, the rate of error varies in different locations. In general, the rate of error is higher at both ends of read, and it is lower in the middle. Therefore, sections with a higher rate of error in a read are ignored in the pre-processing step. b) Another method of sequencing error processing is to correct the sequencing errors using the frequency of k-mers. The same transcript is sequenced many times, and the position of the sequencing error is often different; thus, k-mers with no errors will occur many times. If the frequency of a k-mer is very low, this k-mer is most likely caused by sequencing errors. We usually remove these fragments with lower k-mer frequencies.

Repetitive fragments

There are many identical fragments in the DNA sequence. Errors due to repeated sequences are inevitable in gene splicing, and they will lead to some discontinuous contigs sequences, as shown in Fig. 1. The common method for solving this problem is to use longer read splicing or to consider mate-pair auxiliary information.

Gene splicing algorithm

Presently, there are two types of splicing algorithms: splicing algorithms based on the reference genome and de novo splicing algorithms. The splicing algorithms based on the reference genome require a genome sequence for reference, whereas de novo splicing algorithms do not use any reference information, relying on coincidence information between gene fragments to complete the genome reconstruction.

Splicing algorithms based on the reference genome

Generally, there are three steps in gene splicing, namely, read mapping, read clustering and splicing. First, the sequencing fragments from different genes are mapped to different positions of the genome. Then, the mapping results form fragment clusters. Finally, splicing is carried out using these fragment clusters.

Two popular algorithms based on this idea were proposed by Cufflinks and Scripture.

For the sequence of fragments mapped to the genome, Cufflinks defines a partial ordering relation – the compatibility relationship. If two sequencing fragments are possibly from the same transcript, the two sequencing segments are compatible. Based on this compatibility relationship, a directed graph, called an overlay graph, is created. Each sequencing fragment is a point on the graph. In addition, if the two sequencing segments are compatible, they are given an edge. The path of this overlay graph can represent the individual transcripts of the current gene. To find transcripts in the diagram, Cufflinks applies a minimal path model to find a collection that can reasonably interpret the current sequencing data, and when this number is as small as possible, the final set of transcripts is established.

Scripture constructs another graph – the connectivity graph. It treats every base in the genome as a point. If a sequencing fragment is found to be linked from one base to another, an edge is added between the two bases. Then, from the graph of the structure, Scripture finds the road that contains all the sequencing depths above a certain threshold, which is defined as the final set of transcripts.

Methods based on a reference genome can achieve good performance in efficiency and precision. First, mapping is the process of converting a large splicing problem into a small fragment splicing problem,
which greatly improves the computational efficiency by parallel processing. Second, when the expression of a transcription is low, reference genome splicing can effectively solve this problem. Furthermore, the reference genome is known. Gaps in the transcript can be populated by genomic sequences. Even if some of the transcripts are not fully covered by the sequencing fragments, they are likely to be correctly spliced out.

There are several disadvantages. First, the algorithm relies on the reference genome. Second, the splicing results are related to the quality of the reference genome. Splicing results are closely related to the mapping process. In the mapping process, many of the unmapped genomic sequences will be discarded, which is a major issue in gene information loss. Moreover, mapping errors will be brought into the next splicing process, which will directly affect the final result. Third, splicing algorithms based on a reference genome cannot find fusion genes or fusion transcripts.

**De novo algorithms**

There are three main classes of de novo splicing: Greedy-extension, Overlap-Layout-Consensus (OLC) graphs and De Bruijn graphs.8

Algorithms based on OLC and Greedy-extension were proposed for the first generation of sequencing data. In first-generation sequencing, the reads are longer, and the number of reads is relatively small. It is easy to find the overlap between reads. To transform the DNA splicing problem into a graph model, an overlapping graph based on the ideal scenario of overlap-layout-consensus is established. In contrast to the traditional method, next-generation of sequencing has some distinct characteristics, such as shorter read lengths and larger read numbers. It is difficult to find the overlap between reads. Therefore, the first two methods are no longer applicable. To overcome the above shortcomings, algorithms based on De Bruijn graphs were proposed. In these methods, the reads are translated into fixed-length k-mers. Then, a De Bruijn map is established by looking for the overlap between k-mers. Finally, the problem of DNA splicing graph theory can be solved.

**Greedy-extension**

Algorithms based on a greedy-graph strategy were one of the earliest proposed splicing algorithms for the new generation of gene sequencing. The greedy strategy chooses a preliminary solution and gradually deduces the target solution according to given rules. The goal is to get a better solution with a lower cost.

Splicing algorithms based on the greedy strategy select an initial read sequence according to certain rules, then expand it. Specifically, a heuristic search technique is used to find sequence fragments that have the highest degree of matching to the initial read. Reads are merged into a contig until the two ends of the read cannot be extended. To ensure the quality of splicing, the longest matching reads with the best splicing quality are selected. When a conflict occurs, the search is terminated. Algorithms based on such strategies include SSAKE,9 VCAKE,10 and SHARCGS.11 Due to the locality of the greedy strategy, the sequence segment of greedy-extension only contains a local maximum value. The splicing effect is not ideal, as shown in Fig. 2.

**Overlap graph**

The basic idea of OLC is that the original read sequence is used as the vertex of the graph, and reads with overlapping information are searched to construct the adjacent edges, which are used to construct the whole sequence graph. Splicing software based on this strategy includes Newbler,12 Edena,13 and Shorty.14 The steps based on the Hamilton path method are shown below:

1. **Overlap:** Locate the overlapping information. Reads are compared to locate their possible overlapping positions. If the number of the overlapping bases exceeds a given threshold, the second step is started.
2. **Layout:** According to the relationship between overlapping reads, an overlapping graph is constructed. The overlapping reads are combined to form contigs.

**Figure 2.** Gene splicing based on a greedy algorithm (sequence A selects the local optimal sequence D, resulting in genetic splicing errors).
(3) Consensus: To generate longer contigs, a Hamilton path from the start node to the end node must be found.

**De Bruijn graph**

In the splicing algorithms based on De Bruijn graphs, overlapping reads are mapped together. This approach reduces the complexity of computing and memory consumption. Presently, splicing software using the De Bruijn graph method includes ABySS, PASHA, Velvet, (http://www.ebi.ac.uk/~zerbino/velvet/velvet_1.2.10.tgz), and Soapdenovo. The general flow of splicing algorithms based on the De Bruijn diagram strategy is as follows:

1. De Bruijn graph is constructed: Reads are divided into a series of consecutive k-mers. The k-mers serve as edges of the graph. Two adjacent k-mers have (k-1) base overlap. The construction of a De Bruijn graph is shown in Fig. 3.

2. De Bruijn graph is simplified: Nodes with only one in-degree and out-degree are combined. In addition, substructures with tips and bubbles are removed according to a specified rule.

3. Contigs are constructed: The next task is to find an optimal Euler path (through each side once and only once) in the De Bruijn graph or its subgraph. The corresponding base sequence in this path represents the contig.

4. Scaffolds are generated: Matching data are used to determine the relative direction and position relationship between the contigs. Then, the contigs are assembled and the gaps between the contigs are filled. Eventually, the scaffold sequences are obtained.

In the process of constructing De Bruijn graphs, errors are inevitable. These errors include tip structures, the bubble structures, and repeat structures.

**Comparison of de novo splicing algorithms**

Different methods have their own characteristics, as shown in Table 2. The greedy strategy is suitable for small-scale genetic data. When dealing with large-scale genetic data, the correctness of splicing cannot be guaranteed because there are many regional repetitive problems. The method based on OLC graphs does not overcome the problem of repeat sequences.
However, it directly uses reads to splice, and it retains the integrity of fragment information. Compared with the OLC graph, the splicing method based on De Bruijn graphs has a better splicing effect due to the further splitting of reads, which overcomes the repetitive sequence splicing problem to a certain extent. However, the splitting of reads also loses some information on the fragment itself to some extent.

**Improved gene splicing algorithm based on De Bruijn graph**

After introducing the challenges of next-generation sequencing technologies, we summarize the optimization of de novo splicing algorithms and describe two new splicing algorithms that shorten sequence splicing times, reduce wasted memory space, and splice longer base sequences.

To effectively assemble next-generation sequencing data, a genome sequence mosaic system GSnake based on a Markov model was proposed. First, the system builds a Markov model that satisfies the characteristics of the genome using vast amounts of sequence data. Then, the system constructs a state transition probability matrix using short reads that are stored in a hash table. A fragment of bases is seen as a state of the model. Based on the model, a de novo assembly method is proposed. At the same time, due to the high error rate of next-generation sequencing data and the existence of a large number of repetitive fragments in the genome, a series of heuristic algorithms are proposed to optimize the splicing process in order to ensure the accuracy and splice length of the splicing results.

For construction of the De Bruijn graph, dynamic hash techniques can be used. Based on the design of a well-behaved hash function and an effective solution to the conflict, an appropriate loading factor is chosen such that the time complexity of the insertion and look-up operation of the hash table is low. In the process of generating contigs and scaffolds, distributed cluster resources are fully utilized through multi-line technology based on shared memory and multi-processes based on distributed memory technology.

For the splicing algorithms based on De Bruijn graphs, in the process of constructing figure, the storage mode of bidirectional De Bruijn patterning in the YAGA algorithm can be improved so that less storage space is used. In the simplification of a graph, a depth-first-search algorithm is used to traverse the graph, which greatly reduces the communication between processors. It can calculate the movement of nodes and reduces time and space consumption in the

| Platform | Roche 454 | Illumina | SOLID |
|----------|-----------|----------|-------|
| Fragment length | 500~800 bp | 35 bp, 50 bp, 75 bp | 35 bp, 50 bp |
| Accuracy | 99% | 99% | 99% |
| Data volume | 0.7 G | 600 G | 120 G |
| Time | 24 hours | 3~10 days | 7~14 days |
| Advantages | Sequencing fragments are long | High throughput | Higher accuracy |
| Disadvantages | Low throughput | Sequencing fragments are short | Sequencing fragments are short |

| Algorithm principle | Greedy-extension | OLC | De Bruijn |
|---------------------|------------------|-----|----------|
| Algorithm principle | Greedy strategy | Overlap/Layout/Consensus | Euler path |
| Algorithm evaluation | Advantages: The principle is simple. It is easy to operate. | Advantages: The sequence splicing problem is converted to a graph theory problem. | Advantages: It is an effective solution to the repeat sequence problem. It also uses hash table storage for fast and convenient searches. |
|                     | Disadvantages: The algorithm can easily obtain a local optimal solution. It is difficult to obtain a global optimal solution. | Disadvantages: The problem of repeated sequences cannot be solved. For next-generation gene sequencing data, the algorithm must perform a large number of computations. | Disadvantages: For next-generation sequencing data, the algorithm builds complex De Bruijn diagrams and consumes computing resources. |
| Sequencing length   | A shorter sequence, comparable to gene lengths in bacteria and fungi. | The gene length is approximately 100~800 bp. | The gene length is approximately 25~100 bp. |
| Hardware requirements | As a result of multiple sequence alignment reads, the algorithm occupies a large amount of memory. | Due to the large amount of data to be read, graph construction is complicated. The algorithm occupies a large amount of memory. | The algorithm does not need sequence alignment, and it uses a hash table to optimize storage. Hardware requirements are not high. |
splicing process. In vertex data processing, vertices below a threshold are removed according to the principle of frequency. The number of vertices in the graph is reduced. The problem of wrong spelling caused by sequencing errors is reduced.21

Erroneous bases in sequence fragments should be deleted or modified before the sequence is spliced. This kind of data preprocessing can not only reduce required computing resources but also reduce the running time and improve the splicing effect. For a review, see ref. 22. It presented a data processing method that can effectively correct the wrong base in sequence segments.

For a review, see ref. 23. An improved method based on the De Bruijn graph algorithm was proposed. A collision-free hash table structure was used to store the De Bruijn graph. The concept of a decision table was introduced on the basis of the above De Bruijn graph algorithm. The selection of the optimal path in the De Bruijn graph was optimized by updating the information in the decision table. The selection range of the subsequent k-mer was reduced, shortening the sequence splicing time and improving the accuracy of contigs.

For a review, see ref. 24. A new algorithm named Bridger was designed using the knowledge and method of combinatorial optimization. A large number of tests showed that Bridger spliced out more full-length transcripts than previous de novo splicing algorithms. It not only increased the sensitivity but also reduced the false positive rate of the predicted results. Its time and memory usage were also much lower than those in other splicing algorithms.

For a review, see ref. 25. The idea of sequence alignment based on reads was proposed for the first time. The scoring method based on a combination of information accumulation and data features was proposed. From the perspective of whole reads, the algorithm not only considered the accumulated information in the splicing process but also referred to the data characteristics of the reads. It provided comprehensive and objective scoring criteria for contigs splicing. Priority was given to matching read data, which can be directly for navigate contig splicing. Compared to the traditional splicing calculation model based on k-mers, this method avoids the problems of a non-scientific scoring method the loss of data feature information.

**Current research based on Hadoop and MapReduce**

Algorithms based on a Hamilton path and Euler’s two algorithms are the most representative of de novo splicing modes. The two algorithms use more complex graph structures to build information between fragments, which carries a great deal of overhead in storage and time requirements and generates temporary data in the stitching process. Euler’s super path algorithm, which splits the read first, forms a large number of k-mers, and then constructs a De Bruijn graph, is more expensive in terms of storage and time requirements. Using the Hadoop open source framework to build the cluster, a MapReduce programming model for parallel algorithm development overcomes the run time and space issues. In the article, “The Research of DNA Fragment Assembly Algorithm based on MapReduce”, ideal results are obtained in terms of the correctness of parallel algorithm and the efficiency of implementation of the experiment.

**Hadoop and MapReduce**

Hadoop is a sub project of Apache Lucene. The development language is Java. It was developed by the Apache software foundation. Hadoop is a distributed computing platform that is separate from the Nutch project.26 It is responsible for distributing storage and computing data. More specifically, Hadoop is easier to develop and can be used to handle massive data.
Hadoop has become a collection of projects to provide better solutions to the issues associated with large data, and it includes a number of sub items, as shown in Fig. 7. The core of Hadoop includes three parts, which are applied to the distributed storage HDFS (Hadoop Distributed File System), MapReduce calculation framework for distributed computing and HBase for large-data oriented columns.

MapReduce is one of the core components of Hadoop. It is also a simple software framework. The program, which was implemented by the MapReduce architecture, can be implemented in a computer cluster. The MapReduce programming model is designed to address massive data in a distributed cluster. It is composed of map and reduce. Data are distributed to each node before processing. Each node takes the locally stored data for the mapping process. The input data are converted to the key-value by logic operations. They are merged and sorted, then distributed to the reduce nodes. The MapReduce data processing process is shown in Fig. 8.

Gene splicing based on MapReduce

In 2008, Michael C. Schatz developed a high efficiency comparison software of short sequences. The software was called BlastReduce.\(^{27}\) It uses the MapReduce calculation model.\(^{28}\) In 2009, Michael C. Schatz used a MapReduce parallel computing model to develop the high-sensitivity sequence alignment software CloudBurst.\(^{29}\) To find SNP mutations, in 2009, Ben Langmead also developed a software called Crossbow based on a MapReduce parallel computing model.\(^{30}\) In 2013, Pireddu et al published the Seal tool. The Seal analysis process used Pydoop and BWA tools for sequence alignment and de-emphasis. In 2014, the MetaSeq process was developed by Puckelwartz et al.\(^{31}\)

It used a class MapReduce script to analyze the genome, which uses the GATK recommended best-of-breed process. However, the MetaSeq core uses a high-throughput, high-performance computer to analyze the genome.\(^{31}\)

In China, Huada gene has been a leader in sequencing data research. On April 20, 2015, Huada Genesis announced a cloud computing platform service product called BGI Online.\(^{32}\)

In recent years, there have been many tools and processes to address bioinformatics data using the MapReduce framework.\(^{33}\) Among them, there is a sequence alignment tool that is used to sort the sequencing data and correct the quality value. According to different features, a Hadoop MapReduce parallel computing model can be used, and the class MapReduce model can also be used.

Presently, there are more and more discussions on parallel sequence splicing algorithms. CloudBurst is a highly-sensitive parallel splicing algorithm based on the MapReduce model that uses a seed-and-extend splicing technique.\(^{34}\) This splicing technique differs from OLC and De Bruijn. It uses reads and a reference sequence for splicing.

DNA sequence assembly and gene homology comparisons are the most basic and most frequently used methods in bioinformatics, and they can be used to estimate genetic integrity and gene structure and function. In recent years, many scholars have worked on sequence assembly algorithms. They have also put forward some effective methods. However, an accurate and efficient assembly algorithm is very rare. Thus, an accurate DNA sequence assembly algorithm based on MapReduce has emerged.\(^{35}\)

By studying the design idea and realization method of Euler’s super-parallel algorithm, a parallel algorithm of sequence splicing based on Euler’s super-path was developed with MapReduce. Because of the data storage characteristics of the Hadoop framework, the adverse effects of graph partition on the splicing results were avoided.\(^{36}\)

Current research mainly focuses on de novo splicing, which mainly includes greedy algorithms, Hamilton path algorithms and Euler super-path algorithms. Hamilton path algorithms do not overcome the problem of repetitive sequences. Euler super-path algorithms are mainly single-machine operations with limited operation speeds, whereas a stand-alone mode is not conducive to the completion of a series of
follow-up information processing. After analyzing the current assembly algorithms, an assembly algorithm based on MapReduce is proposed. The error data in the assembly process are removed by statistics. The repeated data in the assembly process are eliminated by increasing the length of the k-mers. Finally, the parallel assembly algorithm is implemented in the MapReduce platform.37

Conclusion

Presently, most of the algorithms in gene splicing are based on the De Bruijn graph. If we can consider the influence of the number of stagger bits between k-mer vertices on the structure of De Bruijn graphs, we can improve the existing stitching algorithm and obtain better splicing results.

The difficulties faced by sequence splicing include the overhead of the algorithm in time and space requirements. The use of a Hadoop open source framework to build a cluster and the use of a MapReduce programming model for parallel algorithm development can improve the efficiency of the algorithm. In the future, we will further study splicing algorithms and MapReduce to further improve the efficiency of the algorithms and solve the more comprehensive problem of splicing through further expansion at the cluster scale.

Disclosure of potential conflicts of interest

No potential conflicts of interest were disclosed.

ORCID

Xiuhua Si http://orcid.org/0000-0002-5528-2813
Lei Zhang http://orcid.org/0000-0003-4088-2426

References

[1] Whiteford N, Skelly T, Curtis C, Ritchie ME, Löhr A, Wait Zaranek A, Abnizova I, Brown C. Swift: primary data analysis for the Illumina Solexa sequencing platform. Bioinformatics. 2009;25(17):2194-2199. doi:10.1093/bioinformatics/btp383.

[2] Luo C, Tsementzi D, Kyrpides N, Read T, Konstantinidis KT. Direct Comparisons of Illumina vs. Roche 454 Sequencing Technologies on the Same Microbial Community DNA Sample. Plos One. 2012;7(2):e30087. doi:10.1371/journal.pone.0030087.

[3] Hedges DJ, Guettouche T, Yang S, Bademci G, Diaz A, Andersen A, Hulme WF, Linker S, Mehta A, Edwards YJK. Comparison of Three Targeted Enrichment Strategies on the SOLiD, Sequencing Platform. Plos One. 2011;6(4):e18595. doi:10.1371/journal.pone.0018595.

[4] Yang X, Chockalingam SP, Aluru S. A survey of error-correction methods for next-generation sequencing.[J]. Briefings Bioinformatics. 2013;14(1):56. doi:10.1093/bib/bbs015.

[5] Narzisi G, Schatz MC. The challenge of small-scale repeats for indel discovery. Frontiers Bioengineering Biotechnol. 2015;3:8. doi:10.3389/fbioe.2015.00008.

[6] Trapnell C, Williams BA, Pertea G, Mortazavi A, Kwan G, van Baren MJ, Salzberg SL, Wold BJ, Pachter L. Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation. Nat Biotechnol. 2010;28(5):511-5. doi:10.1038/nbt.1621. PMID:20436464

[7] Guttman M, Garber M, Levin JZ, Donaghey J, Robinson J, Adiconis X, Fan L, Koziol MJ, G nirke A, Nusbaum C, et al. Ab initio reconstruction of cell type-specific transcriptomes in mouse reveals the conserved multi-exonic structure of lincRNAs. Nat Biotechnol. 2010;28(5):503-10. doi:10.1038/nbt.1633. PMID:20436462

[8] Lam KK. Data-efficient De novo genome assembly algorithm: theory and practice. (PhD Dissertation). UC Berkeley: Electrical Engineering Computer Sci. 2016.

[9] Warren RL, Sutton GG, Jones SJ, Holt RA. Assembling millions of short DNA sequences using SSAKE. Bioinformatics. 2007;23(4):500-01. doi:10.1093/bioinformatics/btl629. PMID:17158514

[10] Jeck WR, Reinhardt JA, Baltrus DA, Hickenbotham MT, Magrini V, Mardis ER, Dangl JL, Jones CD. Extending assembly of short DNA sequences to handle error. Bioinformatics. 2007;23(21):2942-944. doi:10.1093/bioinformatics/btm451. PMID:17893086

[11] Dohm JC, Lottaz C, Borodina T, Himmelbauer H. SHARCGS, a fast and highly accurate short-read assembly algorithm for de novo genomic sequencing. Genome Res. 2007;17(17):1697-706. doi:10.1101/gr.072033.107. PMID:17908823

[12] Margulies M, Egholm M, Altman WE, Attiya S, Bader JS, Bemben L A, Berka J, Braverman MS, Chen YJ, Chen Z, et al. Genome sequencing in microfabricated high-density picolitre reactors. Nature. 2005;437(7057):376-80. doi:10.1038/nbt.1621. PMID:16056220

[13] Hernandez D, Frangis P, Farinell L, Osteras M, Schrenzel J. De novo bacterial genome sequencing: millions of very short reads assembled on a desktop computer. Genome Res. 2008;18(5):802-809. doi:10.1101/gr.072033.107. PMID:18332092

[14] Margulies M, Egholm M, Altmann WE, Attiya S, Bader JS, Bemben LA, Berka J, Braverman MS, Chen YJ, Chen Z, et al. Genome sequencing in microfabricated high-density picolitre reactors. Nature. 2005;437(7057):376-80. PMID:16056220
