A new algorithm on detecting structural variations with no reference genome

参赛学生：张瑞霖
Ruilin Zhang

广东实验中学 广州 广东 中国
Guangdong Experimental High School,
Guangzhou, Guangdong, China

指导老师：戴智明
Tutor: Zhiming Dai

中山大学
Sun Yat-Sen University,
Guangzhou, China

A new algorithm on detecting structural variations with no reference genome

Ruilin Zhang [1]

Tutor : Zhiming Dai [2]

[1] Scientific Researcher Training Class in honor of Academician Zhong Nanshan, Senior 3, Guangdong Experimental High school,Guangzhou 510375, China; [2] School of Data and Computer Science , Sun Yat-Sen University, Guangzhou 510006, China.

Abstract

Structural variations (large scale variation) in genome could be more critical than SNP (Single-Nucleotide Polymorphism) in resulting in tumor cells, immune system diseases and fatness[1~8]. It is necessary and important to detect structural variations with next generation sequencing data by means of efficient and accurate algorithms. This paper introduced a new algorithm using k-mers (short fragments of genome) and graph algorithms, which combines variation detection and sequence assembly. Our results showed that our method is more reliable than methods using reference and more efficient than brute force search.

Key Words: bioinformatics; structural variation; sequence assembly; k-mer; genome mutation; cancer;

Contents

1. Introduction

Next generation sequencing technology provides us with large quantities of genomics data[9,10]. The raw data we can obtain is a massive number of reads. Adapting to the data's characteristics, many researchers have achieved encouraging results with reference genome, such as COSMOS[11] and SV-Bay[12] . SV-Bay[12] is an algorithm which uses reference and a combination of Bayesian model and clustering to detect SVs. COSMOS is an algorithm using mapping, clustering and checking of sequencing depth to detect the SVs.

There are certainly problems that methods using reference may cause. The most serious one comes from the difference in human's genome. Currently, the references we use are from limited people. If we want to use their genome and regard theirs as the normal ones, hoping that this method can treat another person, it is hard to figure out what are SVs and what are polymorphic differences.

Methods that do not use reference are needed. SMUFIN[13] is an algorithm developed by a research group in Spain. It use both normal and abnormal cells from the patient and directly sequence and construct a “Quaternary sequence tree” to quite violently detect the difference between them. It is very time consuming and its consumption could be seen in the paper of COSMOS[11].

K-mer[14~15], is the name of short sequences that contain k bases. k-mers are widely used in feature recognition of biological information. In this paper, we use k-mers to detect variations and assemble the genome.

Sequence assembly[16~18], is a technique uses reads to construct the whole genome[18]. Currently, methods based on the next generation sequencing mainly use three strategies[18]: Greedy, Overlap-Layout-Consensus(OLC) and De Bruijn graph[19].

De Bruijn graph[19] is a kind of graph whose nodes represent sequences which include k bases of the read and edges represent their neighborhood relationship. To construct the whole genome, we do some modifications on this graph and find Euler paths on the graph[18]. Based on this theory, many softwares such as SOAP[20] were developed.

We developed a method using k-mers to detect SV, SNP and assemble the genome.

The method is:

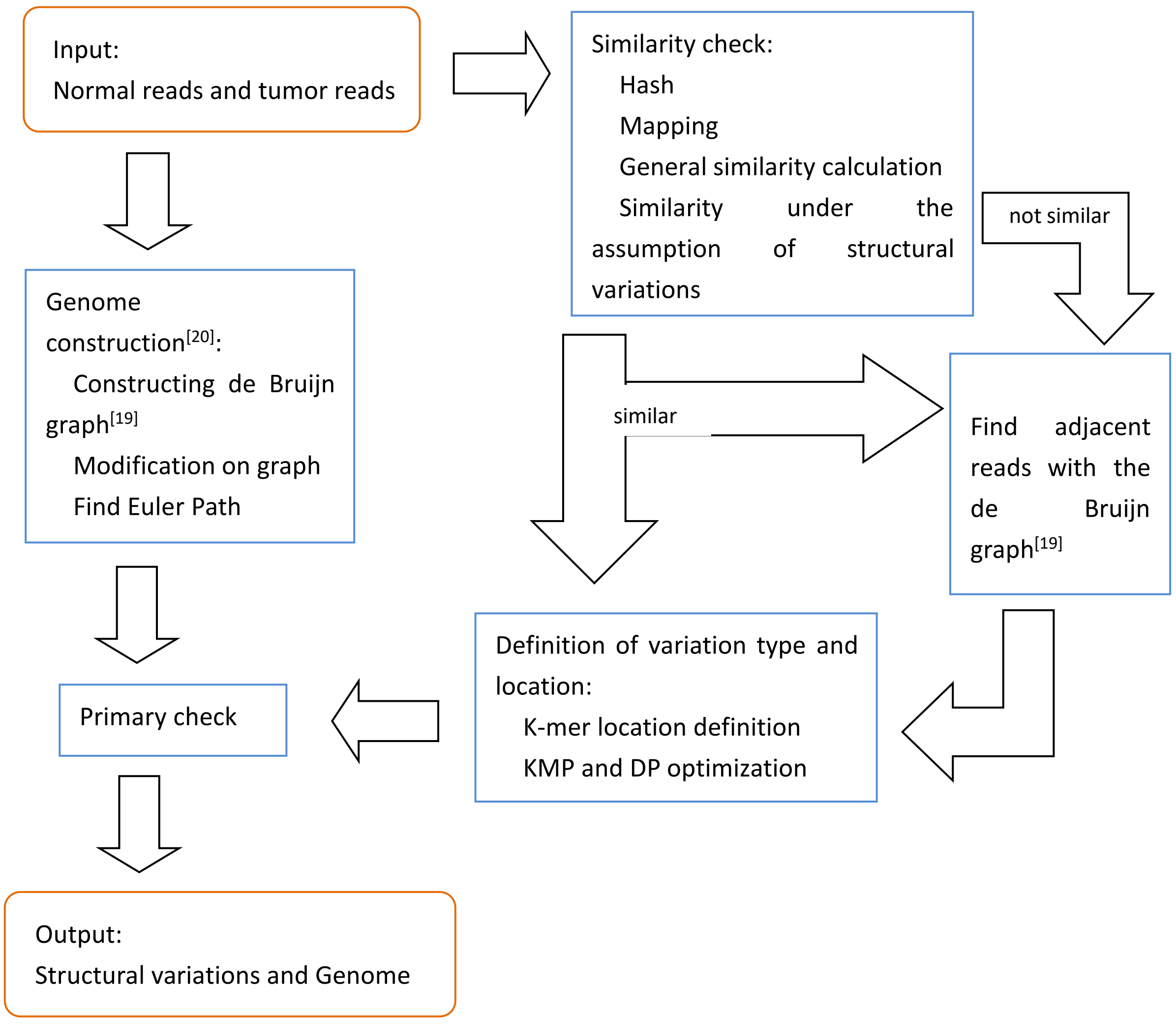
1. A reliable algorithm, for it does not use any reference genome to ensure it will not be affected by polymorphic differences.

2. An efficient algorithm, as is shown in "3.Comparison with another

method" , it can use less memory and ensure sensitivity and specificity with a higher speed.

3. An accurate algorithm, as is shown in "4. the result of test running" , it achieved a 99.926% successful rate in similarity check and a high percentage of successful identification of structural variations.

2. Method


Input:
Normal reads and tumor reads
Similarity check:
Hash
Mapping
General similarity calculation
Similarity under the assumption of structural variations
not similar
similar
Genome construction[20]:
Constructing de Bruijn graph[19]
Modification on graph
Find Euler Path
Find adjacent reads with the de Bruijn graph[19]
Definition of variation type and location:
K-mer location definition
KMP and DP optimization
Primary check
Output:
Structural variations and Genome

(Figure 1) **The process of the algorithm**

As input, we use normal reads and tumor reads from the same person in order not to be affected by polymorphic differences. We will firstly check the similarity between reads and construct de bruijn graph. Part of reads that have their similar reads will directly go to the procedure of finding structural variations. We will find the adjacent reads of other reads and identify the structural variations. As we construct the whole genome, we can mark the structural variations on the genome and output the results.

Each of the procedure that are mentioned in the previous paragraph contains many steps. These steps are shown in the figure above and will be discussed in the following chapters.

2.1 Similarity check

We will first check the similarity between tumor reads and normal reads for detection of structural variations.

2.1.1 Data processing

Information of reads from next generation sequencing platforms differ from one software to another. In our research, we use ART[21] , a simulator of next generation sequencing data to generate reads in different regulations. It can generate reads in the regulation of Illumina, SOLiD , 454,etc. Also, Bowtie[22,23], a memory-efficient short read aligner, can map reads to the reference and provide mapping results.The two softwares is of great importance to data simulation[24] and testing.

When it comes to the next procedure, we refer to different information form , extract the reads from the data and turn them into our preferred form . For example, SAM(Sequence Alignment/Map) [25] is a popular format. We would like to extract the read itself and its position , sum of flags and mapping quality from the SAM format data.

2.1.2 Hash

From next-generation sequencing platforms, we get normal reads, which are from normal cells of the patient, and tumor reads,which are from the tumor and mutated tissue of the patient.

Using hash technology, we mark several hash values of a read. They would be calculated by the reads itself and the sequences without one base (and two/three bases) at its beginning and sequences without one base (and two/three bases) at its end.

For a DNA sequence S, we use S_i represent the number corresponding to its i-th base and len(S) represent its length.

Corresponding number:

A(Adenine) 0 (00_2)	T(Thymine) 1 (01_2)
C(Cytosine) 2 (10_2)	G(Guanine) 3 (11_2)

(Table 1) **The corresponding number of base in DNA**

Its hash value:

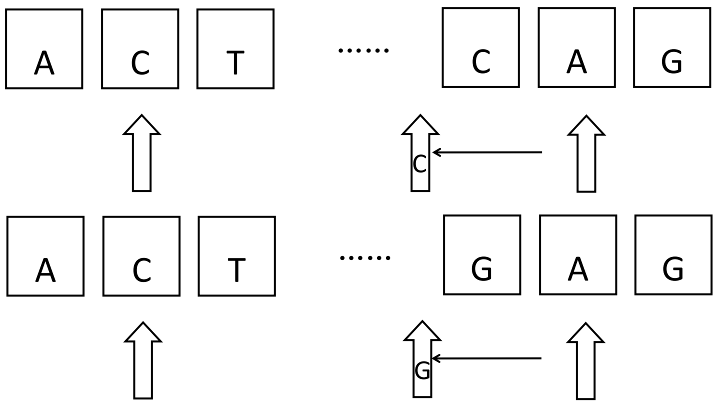
$$\mathrm{H}(S) = \sum_{i=1}^{len(S)} S_i * k^{len(s)-i} \bmod p$$

Consider the possible rare or unknown base and the U(Uracil)base, we let k=6 for the compatibility reason. P is a big prime number.

When two reads have the same hash value, we will check whether their sequence is similar. With this procedure we can quickly identify highly similar reads which have the same bases in a large scale.

2.1.3 Mapping

In order to avoid the situation that too many single-nucleotide mutations (SNPs) have negative effect , we can use mapping to directly find out the similar reads which are the same except for one or two bases.



(Figure 2) **The mechanism of the mapping procedure**

Each arrow will move from one side to the other side. If the same-side arrows of two reads reach the same base, they would check the next base. If not, they will later skip them and mark this place as a difference. Each arrow have one (or two)chance to skip. If the two arrows can finally meet, we can consider two reads similar .

Mapping is an essential procedure to find single-nuclotide mutation and simplify the later calculation.

2.1.4 General similarity calculation

2.1.4.1 k-mer cutting

In the calculation of general similarity, we first cut reads into k-mers. In my source code, the lengths of k-mers are from 1 base to φ times the length of the whole read (integer). Take a 50-base-long read and $\varphi = 0.618$ for example ,we will have twenty 31-base-long k-mers, twenty-one 30-base-long k-mers, ……,49 two-base-long k-mers and 50 single-base k-mers.

The number of k-mers would be: $$\sum_{i=len(S)-\varphi\times len(S)+1}^{len(S)} i \; .$$

In the formula above, len(S) represent the length of the read, φ represents the ratio of the length of longest k-mer to the length of the read.

2.1.4.2 Find the different k-mers

We sort the k-mers we cut in a specific order. And find the different k-mers between reads.

Definition of different k-mers: k-mers that exist in read A but not in read B, including k-mers that occur different times in the two reads.

We then compute the percentages of the number different k-mers to the total number. The percentages are computed in different length respectively.

The general similarity based on k-mers can be calculated as:

$$\sqrt[k]{\sum_{l=\varphi\times len(S)}^{len(S)} \left[(1-p_l)\times w_l\right]^k} \quad .$$

P$_l$ is the percentage of the number of different k-mers to the total number in the length l. W$_l$ is the weight of this length. W$_i$s could be all 1.0 or have an ascending pattern. K is an essential parameter, related to the sensibility of the general similarity calculation . When k become smaller, it will more sensitive when there are some extreme numbers.

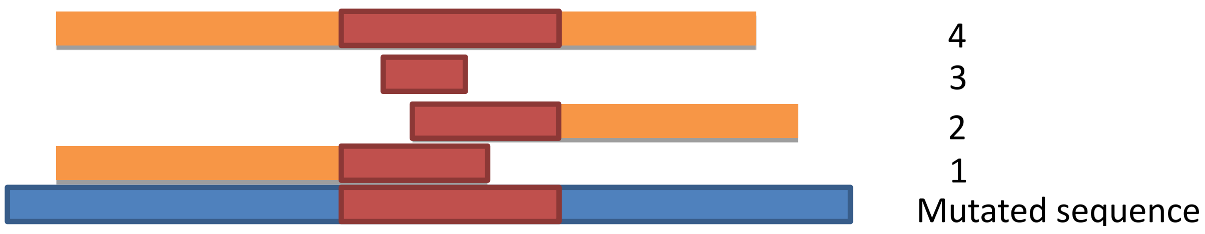
We set a threshold value θ for the general similarity , if the number we calculate is large than the threshold value θ, we would consider two reads similar.

It is worth noticing that, in order to avoid the situation where the number of k-mers are quite different between two reads due to the difference in length, if the original different k-mers explain most of the differences in the k-mers, we would consider two reads similar.

2.1.5 Similarity under the assumption of SV

Because the possibility that there is more than one SV in a single read is low, we can assume that there is only one or no structural variation in a read. Though we accept this assumption and so design our algorithm, there is still a large possibility that the algorithm can identify two or more SVs in one read.

As we have already accepted the assumption, there are different kinds of k-mers.



(Figure 3) **Mutated k-mers generated from mutated sequence**

The middle red zone represent the mutated area of the read. The k-mers lies in the non-mutated area are not drawn in the figure. There are four kinds of unusual k-mers:

1.Part of its end locates in the mutated area;

2.Part of its beginning locates in the mutated area;

3.All of it locates in the mutated area;

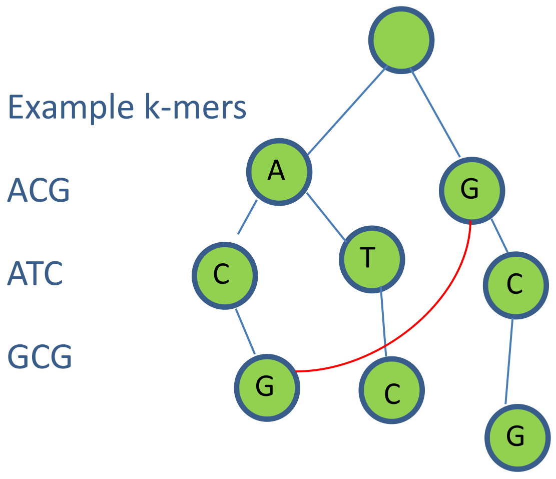
4.It overlaps all the mutated area.

2.1.5.1 positive and negative direction tree

Type 2 and 4 k-mers

We construct a trie tree[26] in the positive direction, for the goal to find and rule out the type 2 and type 4 k-mers. In this tree ,each node represent a base , each edge connects the previous base and following base in the positive direction. There are also special edges which contribute to the calculation of the estimation value.

In order to find type 2 and type 4 k-mers, for each node we have an estimation value which consider the length in the previous non-mutated zone and the number of k-mers which



(Figure 4) **Positive-direction tree**

include the base represented by the node.

Its estimation value :

$$\frac{W_l + W_n}{(\frac{W_l}{Dep} + \frac{W_n}{Cnt}) \times k} .$$

Dep is the depth of the node. W_l is the weight of the length which is particularly the depth of the node.Cnt is the number of k-mers that have the base represented by the node.W_n is the weight of the number of k-mers.

In the formula above, k is an essential parameter to prevent the situation where the depth of the node which is also the length of the chosen non-mutated area in front of the mutated area too long from happening. When the depth of the node increase, k will become larger correspondingly.

Like false link in AC-automaton(Aho-Corasick automaton)[27], we construct special edge (the red one in the figure above) to connect nodes which satisfy the following requirement : the whole sequence represented by one node is a suffix of the sequence represented by the other node.

So the estimation value could be written as:

$$\frac{W_l + W_n}{(\frac{W_l}{Dep} + \frac{W_n}{Cnt}) \times k} + r \times \sum \frac{W_l + W_n}{(\frac{W_l}{Dep'} + \frac{W_n}{Cnt'})} .$$

Dep' and Cnt' is the depth and the number of k-mers respectively of the node connected with this node by a special edge and its sequence is the suffix of the sequence represented by this node. R is the parameter to mainly weaken its contribution to the calculation.

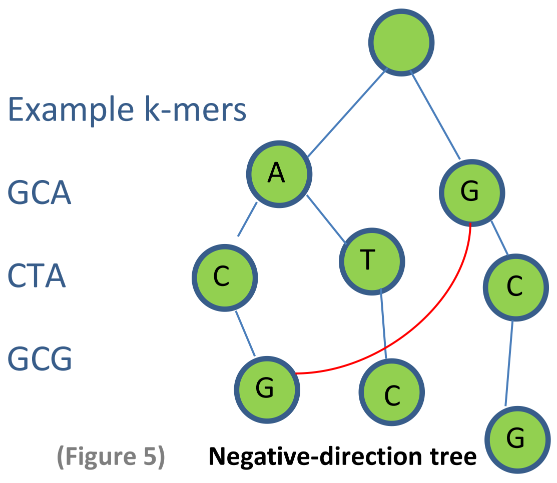
The sequence with the highest estimation value would be consider as the non-mutated area in front of the mutated area. All k-mers whose prefix is the sequence represented by the node which also are k-mers overlap the base represented by the node will be ruled out as type 4 and type 2 k-mers. These k-mers will not take part in the later computation.

Type 1 k-mers

The method we use to find type 1 k-mers is similar to the method we use to find type 2 and type 4 k-mers.

We construct a trie tree[26] in the negetive direction. In this tree ,each node represent a base , each edge connects the following base and previous base in the negative direction. There are also special edges which contribute to the calculation of the estimation value.

Like the special edges in positive-direction tire tree, we construct special edge (the red one in the figure) to connect nodes which



(Figure 5) Negative-direction tree

satisfy the following requirement : the whole sequence represented by one node is a prefix of the sequence represented by the other node.

The estimation value could be also written as:

$$\frac{W_l + W_n}{(\frac{W_l}{Dep} + \frac{W_n}{Cnt}) \times k} + r \times \sum \frac{W_l + W_n}{(\frac{W_l}{Dep'} + \frac{W_n}{Cnt'})} \quad .$$

The meanings of W , Dep , Cnt, k and r are the same as the positive direction tire tree. Dep' and Cnt' is the depth and the number of k-mers respectively of the node connected with this node by a special edge and its sequence is the prefix of the sequence represented by this node.

If the sum of the length of the chosen sequence in the positive-direction tree and the length of the current sequence represented by the node is larger than the whole read. The estimation value of this node in the negative-direction tree would be replaced by a negative number. And we will not consider this node and nodes whose sequence's length larger than it.

The sequence with the highest estimation value would be consider as the non-mutated area that follows the mutated area. All k-mers overlap the base represented by the node will be ruled out as type 1 k-mers. These k-mers will not take part in the later computation.

2.1.5.2 Inclusive relations of k-mers

Type 3 k-mers

In order to find type 3 k-mers, we adopt another approach. Since we have already found the non-mutated area in front of and behind the mutated area , and ruled out the type 1,2,4 k-mers. It means that nearly all k-mers we have is located exactly in the mutated area.

We will check the inclusive relations among the left k-mers to check whether the reads are similar.

Data processing

In the procedure that we generate k-mers from the read, with the optimization that we divided the procedure by the length of k-mer and k-mers have inclusive relations in the read, we can have a graph which shows the inclusive relations of the k-mers.

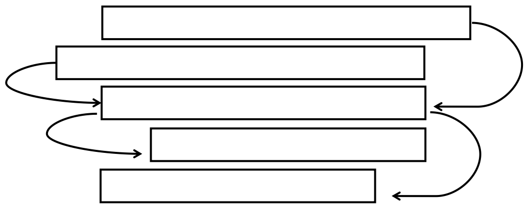
Pseudo-code:

```
For i = 1 to  φ* len (S) {       //length
For j = 0 to len(S) - i + 1 {     //location in the read

Register_k-mer (location j ,length i)
Construct_edge (current k-mer, recorded k-mer)
//construct edge represents inclusive relation

Update_record(current k-mer,location j,length i)
}           }
```

We construct a graph with the procedure above, and sort the edges in a specific order in the graph.

Ideal number of edges

In the ideal case, all k-mers we left have the ideal inclusive relations. For one k-mer(not located at the edge of the mutated area), it is included in two k-mers whose length is one base longer than it (except for the longest one) and it includes two k-mers whose length is one base shorter than it(except for the one-base-long k-mers).



(Figure 6) **Inclusive relations between k-mers**

In the figure above , rectangles represent k-mers. The middle rectangle is the k-mer we discuss in the previous paragraph. The arrows represent the inclusive relation.

If the k-mer is located at the edge of the mutated area. It will be included by only one k-mer whose length is one base longer than it.

There are two calculation method for the ideal number of arrows(inclusive relations).

Method one :

Method one is based on the length of the expected mutated area. The longest k-mer includes two k-mer, two k-mers included by the longest will have four inclusive relations with three k-mers whose length is one base shorter than them. Finally, the (n-1) two-base-long k-mer have (2n-2) inclusive relations with the single-base-long k-mers.

So the ideal number is:

$\sum_{i=2}^{2n-2} i$, and it can be simplified as $n^2 - n$.

n is the length of the expected mutated area.

Method two :

Method two is based on the number of the left k-mers.

Let n' be the length of mutated area calculated by the number of k-mers (N) .

N and n' have the following relation:

$N = \sum_{i=1}^{n'} i = \frac{n'^2 + n'}{2}$, that is $2N = n'^2 + n'$.

So we have an equation $x^2 + x - 2N = 0$, let x be the positive solution of the equation. The ideal number of inclusive relations is:

$$x^2 - x = 2N - 2x = 2(N - x) .$$

With the ideal numbers calculated by two methods , we can set an acceptable interval for the actual inclusive relations.

Assumed that two ideal numbers (a and b ($b \geq a$)) are two ends of an interval which overlap a certain length(percentage) of our final acceptable part and , then (b-a) and the length of the acceptable interval satisfy $(b-a)=\varphi \times l$. φ is the ratio of the distance between a and b to the length of the interval. Moreover, if we assume the interval between a and b is in the exact middle of the acceptable interval, then the acceptable interval can be written as

$$\left[a-\frac{1-\varphi}{2}\times\frac{b-a}{\varphi}, b+\frac{1-\varphi}{2}\times\frac{b-a}{\varphi}\right].$$

And it can be simplified as

$$\left[\frac{(\varphi+1)a+(\varphi-1)b}{2\varphi}, \frac{(\varphi-1)a+(\varphi+1)b}{2\varphi}\right].$$

If the inclusive relations between the k-mers is in the acceptable interval , it indicates that our left type 3 k-mers satisfy our expectation, two reads are similar under the assumption of structural variations. If not, two reads are probably not similar though we have assumed that there would be structural variations.

2.2 Definition of variation type and location

There are various kinds of structural variations. But most of these variations could be considered as some combinations of several basic structural variations such as insertion , deletion , translocation, inversion and CNV(copy number variation)[28]

Insertion	ACTAGCAG—>ACTAGCATGCAG
Deletion	ACTAGCATGCAG—>ACTAGCAG
Translocation	ACTAGCATGCAG—>AGCATCTAGCAG
Inversion	ACTAGCATGCAG—>ACTATACGGCAG
CNV	ACTAGCATGCAG—>AGCTAGCATGCGCATAG

(Figure 7) **Examples of simple structural variations**

Insertion, is a variation which will insert a sequence in the normal DNA sequence. Deletion is a variation which will delete a sequence from the normal DNA sequence. Translocation is a variation which will make a sequence leave its previous location and insert in another location. Inversion is a variation which will invert a sequence . CNV is a variation which will change the copy number of a certain sequence.

2.2.1 k-mer location definition

Since we have had the k-mers from reads, it is efficient and accurate to detect the variation type and location through k-mers.

We adopt a method which firstly check the location where the k-mers can be mapped in the reads , and then construct a graph to show the location relation of the k-mers.

It is unrealistic to map all the k-mers with various length on the read because of its high consumption. It is accurate and efficient to check k-mers with several lengths.

We will map 3-5 kinds of k-mers for the goal to detect structural variations. The length of k-mers are 1, 3, x, y, $\frac{1}{4}\times$ the length of the read, respectively. The value of x and y can be defined by the user, and satisfy $3 < \mathrm{x} < \mathrm{y} < \frac{1}{3} * length\_of\_the\_read$ to guarantee that the k-mers are useful.

Having known the relation between k-mers and the location in the read, we will begin to identify the variations.

CNV(copy number variation): there are distinct number variation in the locations that a certain k-mer or a few k-mers can be mapped . At the same time ,other k-mers can be quite perfectly mapped to the corresponding locations.In this case, the type of variation can be defined as CNV.

Insertion : there are distinct gap where there cannot be the location for some long k-mers to map, while k-mers can be quite perfecty mapped to other corresponding locations. In this case, the type of variation can be defined as insertion and the location is clear.

Deletion : there are k-mers cannot be mapped to any places reasonably , while all locations can be places for other k-mers to map . In this case, the type of variation can be defined as deletion and the location is clear.

Translocation : distinct changes take place in the location for certain k-mers to be mapped while other k-mers can be quite perfectly mapped to corresponding locations .In this case, the type of variation can be defined as translocation and locations of disappearance and insertion are clear.

Inversion: in order to detect inversion, we first invert the k-mers and search place for them to be mapped again. If there are corresponding places for inverted k-mers to be mapped while other k-mers can be mapped to corresponding locations, the type of variation can be defined as inversion and the location is clear.

2.2.2 KMP[29] and DP[30] optimization

In the procedure to construct a graph which shows the location relation of the k-mers and the read. It is unwise to check whether the bases are match one base by another.

In our algorithm, we use KMP algorithm to optimize. KMP algorithm is Knuth–Morris–Pratt algorithm[29] . It is an excellent algorithm in string match, especially when the letters of the string are not so many. It uses the feature of the sequence to avoid unnecessary comparison and reach an $O(m+n)$ complexity (m is the length of the k-mer and n is the length of the read) .

For each k-mer ,we construct a “fail array” for later match.

The c++ code is :

```
// kmer[i][j] the j th base in the i th kmer
// fail[i][j]    the j th fail location of the i th kmer
inline void getfail(int x){
int j,len=kmerlength[x];
fail[x][0]=-1;
fail[x][1]=0;
for(int i=2;i<=len;i++){
j=fail[x][i-1];
while(kmer[x][j+1]!=kmer[x][i]&&j!=-1)j=fail[x][j];
fail[x][i]=j+1;
}
}
void constructnextarray(){
for(int i=1;i<=kmertot;i++){
getfail(i);
}
}
```

In the procedure of matching, if the bases at the ith base’s location are not match, the whole k-mer will move forward by a distance

$(i-1)-fail[this\_kmer][i-1]$ for later matching.

In the procedure of identifying variation, the map situation on the read is quite complicated.If we use the method of brute force search, it would cause high consumption.

Here, we use dynamic programming[30] to solve it.

We construct an array to represent the situation.

$bool\_dp[i][j][k]$ represents the match situation of the i th base in the detecting read and the j th base in read which generates k-mers , it can also represent k bases are mismatch before this situation.

The Dynamic transfer equation :

$$dp[i][j][k]=(dp[i-m][j-m][k]\&\&\exists kmer(length\_m, match))\,|\,dp[i-m'][j-m'][k-m'].$$

If dp[i-m][j-m][k] is true(match) and there is a k-mer whose length is m and perfectly match this m-base-long place, then dp[i][j][k] is true.

If dp[i-m'][j-m'][k-m'] is true(match) , dp[i][j][k] can be true.

2.3. Detecting SV in adjacent reads

Though we have finished the procedure of similarity check and the definition of variation, some reads are still out of our consideration.

If reads are defined as not similar with other reads, they probably have SVs overlapping two reads or they are waste reads.

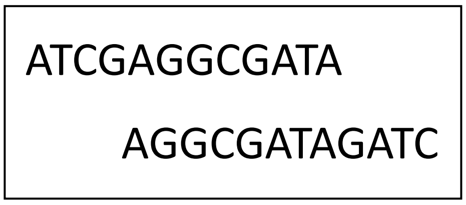
Three kinds of reads need to be connected with their adjacent reads to detect structural variations:

1.Reads that are not similar with other reads according to the standard in similarity check.

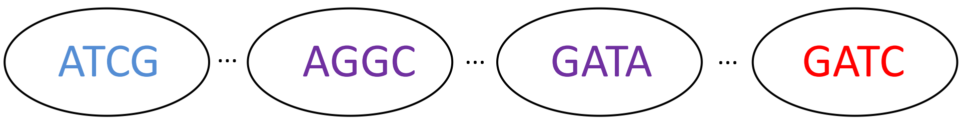
2.Reads overlap a large part on each other.

3.Reads have SV at their ends.

In the de bruijn graph[19] we mentioned in Introduction, two adjacent reads could be found with some same nodes which represent k-mers they have ,and there is a distinct link which contains nodes and edges in which they exist simultaneously.



(Figure 8) **Examples of adjacent reads**



In the figure above, k-mers extracted from two probably adjacent reads are located under two reads. The purple bases are located in the overlapping region, while the blue and red ones belong to two reads respectively.

Many k-mers between each pair of adjacent k-mers are not shown in Figure 8 , such as purple k-mers: GGCG,GCGA,CGAT, etc.

For reads that are not similar with other reads according to the standard in similarity check , we will find the most distinct overlap reads on both of its ends to connect to it and try to match a normal read and detect possible structural variations.

For reads overlap a large part on each other, after connecting them and detecting structural variations in the connected read, we will check whether the new detected SV is the same as the former one. If not , we will try to identify all possible combinations of SVs that can explain this change in the DNA sequence and obey the rule of KISS[31] (keep it simple stupid) to accept the most simple combination as its variation.

For reads that have structural variations at their ends, we will check whether its adjacent read also has structural variation, at the end which they overlap each other. If so, we will consider that there is a large structural variation.

2.4. Genome construction[20]

2.4.1 Data processing

For each read, we have already cut them into k-mers. In the procedure we generate k-mers from reads, we can simultaneously construct a de bruijn graph. Every pair of adjacent nodes which represent k-mers must have the same (k-1) bases. If the previous node is ATG, the following node must be TGX, X could be any kind of base of DNA. It guarantee each pair of nodes can represent exactly (k+1) bases and the continuous property of the graph.

The best length of k-mers , which is also the value of k is controversial. And it is consuming to construct de bruijn graphs with all kinds of k-mers. We will still choose several kinds of k-mers to construct de bruijn graphs.

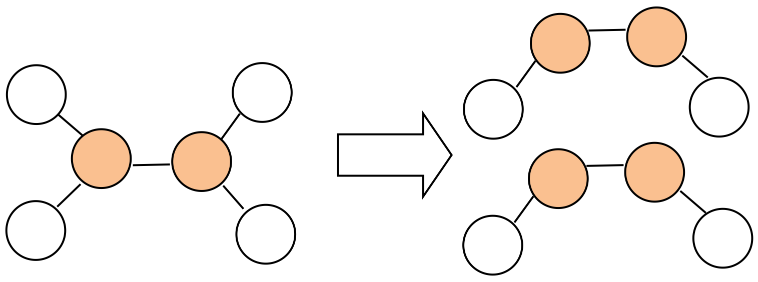
2.4.2 Graph modification

1. We will mark all the nodes whose degree is one. Because they are not likely to appear in the Euler path we are looking for, expect for the situation that they are the starting point.

2.We will remove some edges with distinctly low coverage, because they are probably accidental errors.

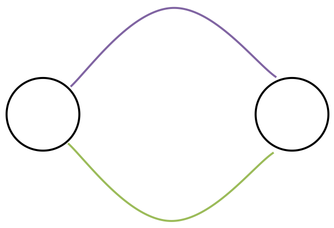
3. We will turn the structure shown in the figure below into a

structure with two dependent path , because the former structures would be obstacles for us to find an ideal Euler path.



(Figure 9) **Modification of a certain structure**

4.We will merge some circle in the graph or remove some edges from the graph in order to find Euler path. Because circles in the graph will not likely provide us with a ideal Euler path.



Some of the edges in a circle would be merged or deleted.

(Figure 10) **Modification of circles in the graph**

2.4.3 Find Euler path[32]

Fleury's algorithm[33] is an elegant but inefficient algorithm which dates to 1883 when "Deux problèmes de Géométrie de situation" was published.

Choosing start node: If all nodes on the graph have an even degree, the start node can be a random one. If there are two nodes with odd degree, the start node would be either of them.

Choosing edges: Setting off from the start node, we will delete all edges we have chosen from the original graph. In no case except for there are not any alternatives will we choose the edge whose deletion will make the graph disconnected.

The original complexity of the Fleury algorithm is $O(|E|^2)$. E is the set of edges. A dynamic bridge-finding algorithm of Thorup (2000)[34] allows this to be improved to $O\left(|E|(\log|E|)^3 \log\log|E|\right)$[35].

2.5. Primary check and output

After detecting the structural variations and constructing the genome, we will check:

1.whether the genome we constructed matches the information in the reads and the genome constructed by the normal reads.

2.whether the adjacent reads we have checked are still adjacent.

3.whether the result of SV detection is simple and quite perfectly explains the variation in the genome.

4.We will also arbitrarily choose a certain number of structural variations to check whether it can explain the current situation in the genome and the similarity between our genome and some reference genomes.

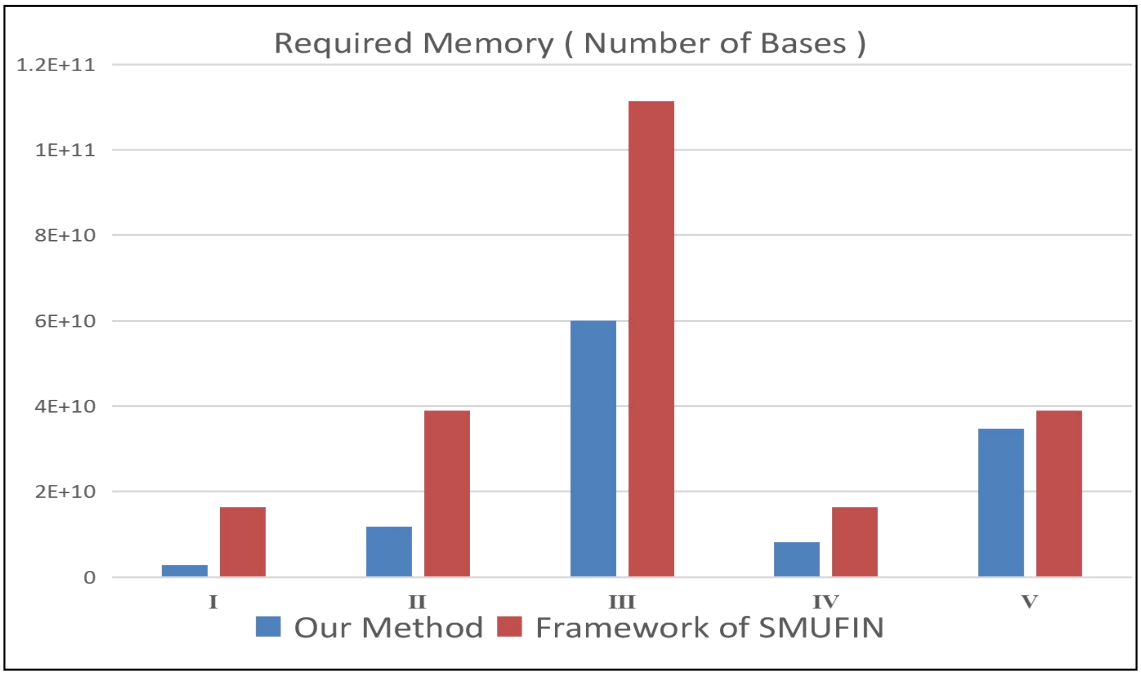
In the output file, we will provide similarity between reads, results of detection of structural variations and the genome we construct.

3.Comparison with another method

Mainly based on the idea of SUMFIN[13], we have a framework of its method which constructs a tree and compare it with our algorithm using k-mers.

It constructs a "Quaternary sequence tree" and uses it as a generalized suffix array. This method puts all reads from tumor and normal cells and their suffixes whose length is larger than a certain number into the tree. The memory this method uses to construct the tree will be huge. If we assume that all reads has the same length, the upper bound of the memory it uses would be estimated as $\frac{1}{2}(\frac{M}{n})^2 n$,(M is the total number of bases of all reads , n is the number of reads) and it can be simplified as $\frac{Average\_length\_of\_read}{2} \times Number\_of\_bases$. Because of the length restriction of the sequences to insert in the tree, the real memory can be calculated as $n \times \frac{a^2 - c^2 + a + c}{2}$ (n is the number of reads,a is the average length of the reads, c is the length of the shortest sequence) which is smaller than its upper bound. But if we consider the memory our method uses in the similar procedure, we will find that if we first register all frequently used k-mers whose length is under or equals to 12 (22,369,620 k-mers in all), the memory we need will be controlled in a low level. And our method performs distinctly well when the coverage of sequencing is high and the length of reads is low.

The figure below demonstrates the difference in required memory between two methods.



(Figure 11) **Required Memory of Two Methods**

	Number of reads(n)	Length of reads(l)	Sequencing coverage(c)	Our method	Framework of SMUFIN
Ⅰ	10million	60	30x	$\frac{n \times l}{c} \times \sum_{i=13}^{20} i + p$	$n \times \frac{60^2 - 20^2 + 80}{2}$
Ⅱ	10million	90	30x	$\frac{n \times l}{c} \times \sum_{i=13}^{30} i + p$	$n \times \frac{90^2 - 20^2 + 110}{2}$
Ⅲ	10million	150	30x	$\frac{n \times l}{c} \times \sum_{i=13}^{50} i + p$	$n \times \frac{150^2 - 20^2 + 170}{2}$
Ⅳ	10million	60	10x	$\frac{n \times l}{c} \times \sum_{i=13}^{30} i + p$	$n \times \frac{60^2 - 20^2 + 80}{2}$
Ⅴ	10million	90	10x	$\frac{n \times l}{c} \times \sum_{i=14}^{30} i + p'$	$n \times \frac{90^2 - 20^2 + 110}{2}$

P is the number of bases which have been processed

$p \approx 261{,}000{,}000$ and $p' \approx 1{,}130{,}000{,}000$

(Table 2) **Required Memory of Two Methods in Different Circumstances**

This method depends on the tree to cluster the tumor-specific reads which are form by the same variation. Since the method inserts a lot of sequences including reads and their suffixes into the tree, the number of this kind of comparisons and sequence extractions will be large. Our method with k-mer can control the number of comparisons with an upper bound $\frac{number\_of\_reads^2 + number\_of\_reads}{2}$. In fact, if we classified the reads by k-mers first and optimize the procedure of similarity check, the real number of comparison will be smaller than the upper bound and the complexity of a single comparison is shown as the above , which is not too high .

Due to the clustering requires that beginnings of a certain length of two sequences must be exactly the same, the certain length must not be too big. If two sequences must be the same in a large number of bases, the method will lose much of its sensitivity. When the certain length is short, it leaves the task of ruling out the false-positive to the following procedures. As is said in the paper of SMUFIN[13], a large fraction of false-positive structural variations are rules out when constructing “break point block”.

After comparing and clustering the mutated reads, the method using the tree will do several interrogations on the tree to find “all detectable reads that are overlapping and complementary to construct break point block”. This procedure consumes a lot while our method has already found the related normal reads in the procedure of similarity check.

In the procedure of identification of SV and the detection of large SVs, two method don't have too much difference.

In the procedure of showing the location in the genome, SMUFIN doesn't use any method like sequence assembly. Instead, it maps the reads onto a reference. Its results would be affected by the polymorphic differences between the patient's genome and the reference genome. If we adopt this procedure of SMUFIN, we can also accelerate our algorithm but the results will also be affected.

	Our method	Framework of SMUFIN
Comparison	Upper bound $\frac{n^2+n}{2}$	$\frac{n^2*(a-c+1)^2+n*(a-c+1)}{2}\times\varphi$ φ(percentage of similar reads)
Finding overlapping and complementary reads	×	√(time-consuming)
Showing results	Sequence assembly OR Mapping to reference	Mapping to reference

(Table 3) **Difference in procedures of two methods**

In summary, our algorithm can use less memory and ensure sensitivity and specificity with a distinctly higher speed than another method.

	Our method	SUMFIN[13]
Insertion	√	√
Deletion	√	√
Inversion	√	√
SNV(SNP)	√	√
Translocation	√	x
CNV	√	x

(Table 4) **Difference in detectable variations of two methods**

The method using k-mers can detect the variations shown in the table above. SMUFIN's competence in detecting translocations and CNV is weak. There are also many complex variation types that were defined by researchers of other method using reference, such as mirror duplication and co-amplification. But the simple ones can also explain these situations.

4. Result of test running

4.1 Similarity check

With a simulation program , we generate an adequate number of reads to test the performance of the algorithm.

The original data and programs for generating reads can be seen in the supplement material.

(Table 5) **The result of test running of similarity check**

Type of variation	length	Length of variation	Rate of being recognized as similar
SNP(Single-Nucleotide Polymorphism)	30 bases	1	100/100(100%)
	50 bases	1	1000/1000(100%)
TNP(Three-Nucleotide Polymorphism)	30 bases	3	100/100(100%)
	50 bases	3	100/100(100%)
	50 bases	3	999/1000(99.9%)
Insertion	30 bases	3	100/100(100%)
	30 bases	5	100/100(100%)
	50 bases	5	100/100(100%)
	50 bases	10	100/100(100%)
	50 bases	30	100/100(100%)
Deletion	30 bases	3	100/100(100%)

(Table 5) continued			
		5	100/100(100%)
	50 bases	5	100/100(100%)
		10	100/100(100%)
		30	100/100(100%)
Translocation	50 bases	3	100/100(100%)
		5	100/100(100%)
		12	100/100(100%)
		20	100/100(100%)
		30	99/100(99%)
Inversion	50 bases	3	100/100(100%)
		5	100/100(100%)
		10	100/100(100%)
		20	100/100(100%)
CNV(Copy Number Variation)	50 bases	3	100/100(100%)
		5	100/100(100%)
		10	99/100(99%)

We generated 9000 reads and did 27 test running. The variation types are SNP, TNP, insertion, deletion, translocation, inversion,and CNV. We achieved an average successful rate 99.92%(calculated by the results of each test) or 99.93% (calculated by the number of pairs being recognized as similar) . Most of the reads with the specific structural variations according to the table above can pass the similarity check under the assumption of SV.

4.2 Identification of structural variations

As is shown in the discussion in the paper of the algorithm COSMOS[11], it is difficult to identify how complicated structural variations were formed. The programme to identify the type and location of structural variations can have high rate of success to identify simple insertion or deletion. For the complex structural variations, it can also provide a certain number of possible explanations of the variation which are corresponding to the actual variation .

When we find and identify the structural variations, we can easily mark the SVs in the genome by a low complexity process according to the position of the reads.

5. Discussions

There are also many possible and achievable improvements of the algorithm.

According to the feature of the calculation in the algorithm, its computation would be accelerated if we can develop a parallel-computing[36] or distributed-computing[37] version of the algorithm.

Many features of variations caused by some diseases are being found by researchers in the biological field. Moreover, features of variations can be obtained by the techniques of machine learning. If we can use these features correctly, it will enhance the ability of the algorithm to identify certain variations.

6. Conclusions

Detecting structural variations is essential to the cure of illness[1~8]. Methods without using reference avoid a lot of problems caused by the difference between the genome of the patient himself and the reference. Computer science can certainly help to detect SVs and provide reliable, accurate and efficient algorithm.

In this work, we proposed a method without using reference. The process of the algorithm can be divided as similarity check, identification of structural variations , detecting SVs in adjacent reads and genome construction. By using k-mers, we combined the process of SV detection with sequence assembly with only the normal reads and tumor reads of the patient. With several procedures in similarity check, we can rule out the reads that are not similar though under the assumption of SV, and recognize reads that have SV as a similar one with the non-mutated one. The algorithm can use less memory and have a higher speed than the framework of SUMFIN[13], another method which dose not use reference. The successful rate of similarity check of reads with SV is about 99.926% . At the same time, the result of identification of SV can explain the change in the genome at a high percentage of success.

Acknowledgement

I sincerely thank the organizers and sponsors of this competetion. I sincerely thank the guidance by Dai Zhiming. I sincerely thank the organizers of Science Talent Program and China Association of Science and Technology. I sincerely thank the support of Sun Yat-sen University. I sincerely thank the support of Hu Zhengyong, Zhuo Peigong, Xie chun and Shi Leyi in Guangdong Experimental High School. I also want to thank my parents and classmates for their support .

References

[1]SACHIDANANDAM R, WEISSMAN D,SCHMIDT S C, et al. A map of human genome sequence variation containing 1. 42 million single nucleotide polymorphisms[J]. *Nature*,2001, 409 (6822) : 928933.

[2] IAFRATE A J, FEUK L,RIVERA M N, et al. Detection of large scale variation in the human genome[J] . *Nature Genetics*, 2004, 36(8) : 949-951.

[3] SEBAT J， LAKSHMI B,TROGE J, et al. Largescale copy number polymorphism in the human genome[J]. *Science*， 2004， 305 (5683) : 525-528.

[4] REDONR, ISHIKAWA S, FITCH K R,et al. Global variation in copy number in the human genome[J]. *Nature*, 2006, 444(7118) : 444-454.

[5]BOCHUKOVA E G,HUANG N, KEOGH J, et al. Large, rare chromosomal deletions associated with severe early-onset obesity[J]. *Nature*, 2010, 463(7281) : 666-67.

[6]DISKIN S J, HOU Cui-ping, GLESSNER J T,et al. Copy number variation at 1q21. 1 associated with neuroblastoma [J].*Nature*, 2009, 459(7249) : 987-991.

[7]FANCIULLI M, NORSWORTHY P J, PETRETTO E, et al. FCG3B copy number variation is associated with susceptibility to systemic,but not organspecific， autoimmunity[J].*Nature Genetics,* 2007, 39 (6) : 721-723 .

[8]STEFANSSON H,RUJESCU D,CICHON S,et al. Large recurrent microdeletions associated with schizophrenia[J]. *Nature*, 2008, 455 (7210) : 232-236.

[9]高敬阳,齐飞,管瑞,基于高通量测序技术的基因组结构变异检测算法[J] 生 物 信 息 学, 1672-5565(2014)-01-005-05 .GAO Jing yang, QI Fei,GUAN Rui, Highthroughput based algorithm of detecting genome structural variation[J],*Chinese Journal of Bioinformatics* ,1672-5565(2014)-01-005-05

[10] SCHUSTER S C. Next-generation sequencing transforms today's biology[J] .*Nature Methods*,2008， 5(1) :16 -18.

[11] Yamagata *et al*. COSMOS: accurate detection of somatic structural variations through asymmetric comparison between tumor and normal samples[J] . *Nucleic Acids Research*, 2016 ,1, doi: 10.1093

[12]Iakovishina *et al*. SV-Bay: structural variant detection in cancer genomes using a Bayesian approach with correction for GC-content and read mappability. *Bioinformatics Advance Access* ,January 6, 2016

[13]Valentí Moncunill *et al* .Comprehensive characterization of complex structural variations in cancer by directly comparing genome sequence reads , 2014 ,*Nature ,* doi:10.1038/nbt.3027

[14] Yi Zhang, Xianhui Wang, Le Kang.A k-mer scheme to predict piRNAs and characterize locust piRNAs [J]. *Bioinformatics*,2011,27:771-776.

[15] 张程 基于朴素贝叶斯的 piRNA 识别问题研究 [学位论文]硕士，2013.

[16] MILLER J R, KOREN S, SUTTON G. Assembly algorithms for next-generation sequencing data[J] . *Genomics*, 2010, 95(6):315-327.

[17] LI R, ZHU H, RUAN J, et al. De Novo assembly of human genomes with massively parallel short Read sequencing[J]. *Genome Research*, 2009, 20（2）:265－272 .

[18]曾培龙, 王亚东. 全基因组序列拼接研究进展[J] .*智能计算机与应用*.2095-2163（2012）04-0004-05 ZENG Peilong, WANG Yadong, Research Progress of Whole Genome Assembly[J] .(2012) *INTELLIGENT COMPUTER AND APPLICATIONS*

[19] MILLER J R, KOREN S, SUTTON G. Assembly algorithms for next-generation sequencing data[J].*Genomics*, 2010, 95（6）:315－327 .

[20]Ruiqiang Li, Hongmei Zhu ,Jue Ruan,*et al*.De novo assembly of human genomes with massively parallel short read sequencing [J]. *Genome Research*.(2010),20:265–272

[21]Huang *et al* .ART: a next-generation sequencing read simulator [J]. *BIOINFORMATICS APPLICATIONS NOTE*. Vol. 28 no. 4 2012, pages 593–594 doi:10.1093

[22] The web of Bowtie. http://bowtie-bio.sourceforge.net/index.shtml

[23]Bowtie:Manual http://bowtie-bio.sourceforge.net/manual.shtml

[24]Escalona,*et al.* A comparison of tools for the simulation of genomic next-generation sequencing data[J].*Nature Reviews Genetics* doi:10.1038/nrg2016.57

[25]Sequence Alignment/Map Format Specification. The SAM/BAM Format Specification Working Group.18 Nov 2015

[26]de la Briandais, René (1959). File searching using variable length keys. Proc. Western J. Computer Conf. pp. 295–298.

[27] Aho, Alfred V.; Corasick, Margaret J. (June 1975). "Efficient string matching: An aid to bibliographic search". *Communications of the ACM*. 18 (6): 333–340. MR 0371172. doi:10.1145/360825.360855.

[28]林勇.面向下一代测序技术的结构变异检测算法综述[J].*计 算 机 应 用 研 究*. LIN Yong. Survey on structural variants detection algorithms for next generation sequencing technology [J]. *Application Research of Computers*.1001-3695(2014) 02-328-05 doi:10.3969/j.issn.1001-3695.2014.02.002

[29]Knuth, Donald; Morris, James H.; Pratt, Vaughan (1977). "Fast pattern matching in strings". *SIAM Journal on Computing*. 6 (2): 323–350. doi:10.1137/0206024

[30]https://en.wikipedia.org/wiki/Dynamic_programming. Dynamic Programming in wikipedia.

[31]The Routledge Dictionary of Modern American Slang and Unconventional English, Tom Dalzell, 2009, 1104 pages, p.595, webpage: BGoogle-5F: notes U.S. Navy "Project KISS" of 1960, headed by Rear Admiral Paul D. Stroop, Chicago Daily Tribune, p.43, 4 December 1960.

[32]N. L. Biggs, E. K. Lloyd and R. J. Wilson, Graph Theory 1736–1936, Clarendon Press, Oxford, 1976, 8–9, ISBN 0-19-853901-0.

[33]Fleury, M. (1883), "Deux problèmes de Géométrie de situation", *Journal de mathématiques élémentaires*, 2nd ser. (in French), 2: 257–261.

[34]Thorup, Mikkel (2000), "Near-optimal fully-dynamic graph connectivity", Proc. 32nd ACM Symposium on Theory of Computing, pp. 343–350, doi:10.1145/335305.335345.

[35]https://en.wikipedia.org/wiki/Eulerian_path. Eulerian path in wikipedia.

[36]https://en.wikipedia.org/wiki/Parallel_computing . Parallel computing in wikipedia.

[37]https://en.wikipedia.org/wiki/Distributed_computing. Distributed computing in wikipedia.