A Computational Approach to Detect CNVs Using High-throughput Sequencing

Myungjin Moon, Jaegyoon Ahn, Chihyun Park, Sanghyun Park
Department of Computer Science, Yonsei University
Seoul, South Korea
{psiwind, ajk , tianell, sanghyun}@cs.yonsei.ac.kr

Youngmi Yoon
Department of Information Technology, Gachon University of Medicine and Science
Seoul, South Korea
amyyoon@ cs.yonsei.ac.kr

Jeehee Yoon
Division of Information and Communication Engineering, Hallym University
Seoul, South Korea
jhyoon@hallym.ac.kr

***Abstract*—Copy-Number Variations (CNVs) can be defined as gains or losses that are greater than 1kbs of genomic DNA among phenotypically normal individuals. CNVs detected by microarray based approach are limited to medium or large sized ones because of its low resolution. Here we propose a novel approach to detect CNVs by aligning the short reads obtained by high-throughput sequencer to the previously assembled human genome sequence, and analyzing the distribution of the aligned reads. Application of our algorithm demonstrates the feasibility of detecting CNVs of arbitrary length, which include short ones that microarray based algorithms cannot detect. Also, false positive and false negative rates of the results were relatively low compared to those of microarray based algorithms.**

Keywords-Copy Number Variations; CNVs; High-throughput sequencing; Genomic variants generator;

I. Introduction

Copy-Number Variations (CNVs) can be defined as gains or losses that are greater than 1kbs of genomic DNA among phenotypically normal individuals [1, 2]. It is known that CNVs account for a significant proportion of normal phenotypic variation, including disease susceptibility [3-5]. Therefore, identifying and cataloging of CNVs are essential for the genetic and functional analysis of human genome variation.

Many algorithms were proposed to assess CNV regions of human genome using microarray, which includes Whole Genome TilePath (WGTP) array [6, 7] and SNP genotyping array [8, 9]. The popularity of these methods mainly accounts for the relatively low cost of WGTP array and SNP array. However, the resolution of these platforms limits the size of CNVs found. Generally, these methods are known to be useful to detect only medium or large sized CNVs. Moreover, high noise level of these platforms tends to result in relatively high false positive and false negative rates.

The comparison of two or more human genome sequences can detect CNVs, regardless of CNVs' sizes, more precisely [10-12]. The weakest point of the sequence comparison methods is the high cost of the human genome sequence. There has been much effort to lower the cost to get the whole human genome sequence, and high-throughput sequencing is believed to take a prime role [13]. High-throughput sequencing machine can generate enormous short reads in a short time with relatively low cost. Table 1 [13] shows the details of the reads and throughput of various high-throughput sequencing platforms.

TABLE 1. HIGH-THROUGHPUT SEQUENCING PLATFORMS [13]

Company	Format	Read Length (bases)	Expected Throughput MB(million bases) / day
454 Life Sciences	Parallel bead array	100	96
Agencourt Bioscience	Sequencing by ligation	50	200
Applied Biosystems	Capillary electrophoresis	1000	3-4
Microchip Biotechnologies	Parallel bead array	850-1000	7
NimbleGen Systems	Map and survey microarray	30	1000
Solexa (Illumina)	Parallel microchip	35	500
LI-COR	Electronic microchip	20000	14000
Network Biosystems	Biochip	800+	5
VisiGen Biotechnologies	Single molecule array	NA	1000

There have been many algorithms proposed for de novo assembly of the human genome [14-16]. What makes assembly difficult is the short length of the reads – as reads get shorter, the number of necessary reads increases exponentially, which means high coverage is required. Until now, the number of coverage is limited due to high cost, therefore it might not be practical to detect CNVs by comparison of whole genome sequences. Here we propose a novel approach to detect CNVs by aligning the short reads obtained by high-throughput sequencer to the previously assembled human genome sequence, and analyzing the distribution of the aligned reads. The idea starts from the observation as follows: Though aligned reads are not sufficient to be formed into a human genome sequence due to the limited coverage, it is sufficient to detect CNVs through examining their distribution. Our algorithm can be applied even though the coverage is much less than 10, which is definitely insufficient for de novo assembly of the human genome. Roughly, a base pair in a reference sequence to which relatively more reads are aligned has higher probability to be a gain and a base pair in a reference sequence to which relatively less reads are aligned has higher probability to be a loss. The consecutive 1000 or more base pairs, which have statistically significant *score,* calculated by our algorithm, can be judged to form a gain or a loss according to their *score*. The details of this process are given in the chapter 2.3.

We used the synthetic genome sequences to determine the optimal parameters for our CNV detection algorithm. The synthetic genome sequences were generated by the synthetic genomic-variants generator, which uses a previously built human genome sequence (here, we used Build 36.3) as an input and outputs the synthetic genome sequence by implanting CNVs as well as various genetic variations, including SNPs, insertions, deletions and inversions into the input sequence. After getting the synthetic genome sequence, we get short reads through simulation of the shot gun sequencing by Illumina's high-throughput sequencer. The high-throughput sequencing simulator receives the length of the read and the coverage as an input and generates short reads as an output. After aligning the generated short reads to the reference genome sequence (Build 36.3) using Blast, we applied the CNV detection algorithm varying the input parameters and measured the false positive and false negative rates of the results. The overall process is shown in Figure 1, and details about the synthetic genomic-variants generator and high-throughput sequencing simulator will be described in the chapter 2.1 and 2.2, respectively.

Application of our algorithm demonstrates the feasibility of detecting CNVs of arbitrary length, which include short ones that microarray based algorithms cannot detect. Also, false positive and false negative rates of the results were relatively low compared to those of microarray based algorithms.

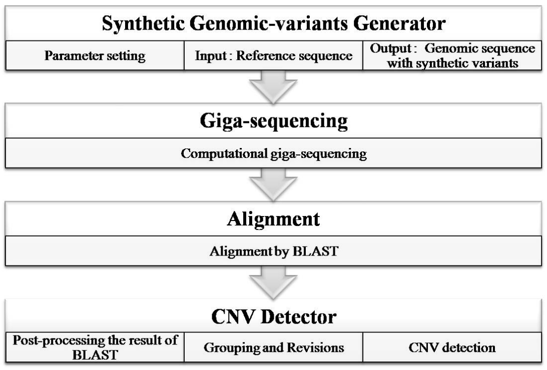
Synthetic Genomic-variants Generator
Parameter setting
Input : Reference sequence
Output : Genomic sequence with synthetic variants
Giga-sequencing
Computational giga-sequencing
Alignment
Alignment by BLAST
CNV Detector
Post-processing the result of BLAST
Grouping and Revisions
CNV detection

Figure 1. Overall process of detecting CNVs using synthetic human genome sequence

II. METHODS

A. Synthetic Genomic-variants Generator

The synthetic genome sequences can be generated by synthetic genomic-variants generator, which uses a previously built human genome sequence as an input and outputs the synthetic genome sequence by implanting CNVs as well as various genetic variations, including SNPs, insertions, deletions and inversions into the input sequence. Those synthetic genome sequences are applicable to several types of test to verify genomic variants. We call the input sequence *reference sequence*, and call the output sequence *test sequence* which is a genomic sequence with synthetic variants. The variable n means the length of reference genome sequence. User can set the values of parameters manually, or can use default parameters. We referred to Database of Genomic Variants [17] and dbSNP [19] for default values and ranges of parameters.

TABLE 2. PARAMETERS AND RANGES OF SYNTHETIC GENOMIC-VARIANTS GENERATOR

Genomic variants	Parameters	Range(Default value)
Gain	*Occurrence frequency*	0~n(303152)
	Length	1000~8031373 bps
	Copy numbers	1~10
Insertion	*Occurrence frequency*	0~n(529287)
	Length	0~999 bps
	Copy numbers	1~10
Deletion	*Occurrence frequency*	0~n(529287)
	Length	0~999 bps
Inversion	*Occurrence frequency*	0~n(6263)
	Length	0~5081342 bps
SNP	*Occurrence frequency*	0~n(145)
	Length	1 bp

Occurrence frequency is a frequency of a genomic variation. The probability that a genomic variation occurs at each position in P is (1/*occurrence frequency*), where P is the set of all position behind the end position of the previous variation (If there is no previous variation, P contains all the positions in the *reference sequence*). Namely, *occurrence frequency* means that a genomic variant may occur every *occurrence frequency* bps. Exceptionally, If *occurrence frequency* is 0, it means that there are no genomic variations. For example, if occurrence frequency is 2,000 and the end position of previous variation is 10,000, then the probability that a genomic variation occurs in position 10,001, 10,002, and so on, is 1/2,000.

TABLE 3. PROCESSES FOR IMPLANTING GENOMIC VARIANTS

Genomic variations	Description
Gain	1. Each position in P, where P is the set of all positions behind the end position of the previous gain, is selected as beginning position of a gain with probability of 1/*occurrence frequency*. 2. Length of a gain is decided by the parameter *length*. If a gain occurred at the position *s*, sequence from *s* to *s+length-1* is copied. If *s+length*-1 > *n*, *length* will be *n-s+1*. 3. Paste copied sequence to the random position *x* of *test sequence*. If other variants have already occurred at position *x*, position *x* is randomly selected again. 4. Repeat 3 as *copy numbers* times. 5. Repeat 1~4 unless P = *Φ*.
Insertion	Same as gain except that *length* is limited up to 999bps.
Deletion	1. Each position in P, where P is the set of all positions behind the end position of the previous deletion, is selected as beginning position of a deletion with probability of 1/*occurrence frequency*. 2. Length of a deletion is decided by the parameter *length*. If a deletion occurred at the position *s*, sequence from *s* to *s+length-1* is deleted. If *s+length*-1 > *n*, *length* will be *n-s+1*. 3. Repeat 1~2 unless P = *Φ*.
Inversion	1. Each position in P, where P is the set of all position behind the end position of the previous inversion, is selected as beginning position of an inversion with probability of 1/*occurrence frequency*. 2. Length of an inversion is decided by the parameter *length*. If an inversion occurred at the position *s*, the sequence from *s* to *s+length-1* is arranged in reverse order. If *s+length*-1 > *n*, *length* will be *n-s+1*. 3. Change each nucleotide of arranged sequence to its complement. 4. Repeat 1~3 unless P = *Φ*.
SNP	1. All the positions of the *test sequence* are selected as position of a SNP with probability of 1/ occurrence *frequency*. 2. If a SNP occurred at the position *s*, nucleotide at *s* will be replaced with another one. 3. Repeat 1~2 unless P = *Φ*.

Parameter *length* and *copy numbers* means the length of the variation and copy numbers of gains and insertions, respectively. The number of genomic variants varies with their size. We divide *length* into several groups according to size distribution of genomic variants, based on the statistical data from Database of Genomic Variants [17], and apply different probability of occurrence, respectively.

For example, the gain whose size is 1~10 kbps occurs with higher probability than the gain whose size is more than 1 mbps. The size group is chosen with the probability based on data from Database of Genomic Variants, then the *length* is randomly given within the group range. The value of *copy numbers* is selected in the range of 1 to 10. Lower *copy number* is selected with higher probability than higher *copy number*. Algorithm about each variant is described in Table 3.

Note that event 'loss' is not implemented. Since loss of the *test sequence* is relatively a gain of the *reference sequence*, inserting loss into the *test sequence* is equivalent to inserting gain into the *reference sequence*.

B. *High-throughput sequencing and Alignment*

The *test sequence* of the previous step, synthetic genomic-variants generator, is used for an input sequence for this stage. We simulated Illumina's high-throughput sequencing method in a computational way. Currently in Illumina's high-throughput sequencing, 5 million bases of 36bp reads can be produced per day by shot gun sequencing. The simulator repeatedly selects random, consecutive 36 base pairs from input genome sequence, and each of the selected 36 base pairs forms a read. Note that reads can overlap with other reads. Each read is called as *test query*. Test queries form 1 coverage if the sum of the length of all test queries equals the total size of input genome sequence. Thus, the total length of the *test query* is same as the length of input genome sequence × coverage. Generally, more precise data can be acquired with larger coverage, but additional time and cost are required. The length of *test queries* and coverage can be adjusted, so that various experiments can be performed with those data. Default *coverage* of our method is 3, which can be regarded as practical in cost-wide.

Generated *test queries* are aligned to the *reference sequence* by BLAST 2.2.18 [18], widely used tool for searching for sequence similarities. We allowed 2 miss matches, otherwise, sequencing errors and SNPs may interfere perfect matches.

C. *CNV Detector*

The number of times that each *test query* is aligned, as well as the location to which each *test query* is aligned is derived by post-processing the result file of BLAST run. We call the number of alignment as *score*, and location that is aligned as *position*.

A single *test query* can be aligned to the several regions of the *reference sequence*. In this case, the *score* of the region is divided by the number of the aligned regions. The formula for calculating *score* is shown as:

$$s_p = \sum_{i=1}^{n} x_{pi} \begin{cases} x_{pi} = \dfrac{1}{r},\ if\, r \neq 0, \\ \\ x_{pi} = 0,\ otherwise \end{cases}$$

s_p = total *score* at *position p*
n = total number of *test queries*
r = the number of regions where a *test query* is aligned
x_{pi} = *score* at *position p* attributed by *i*th *test query*

For example, if a single *test query* is aligned to just one region of *reference sequence*, that region will get *score* 1. If a single *test query* is aligned to two regions of *reference sequence*, each region will get *score* of 0.5. Figure 2 shows that example.

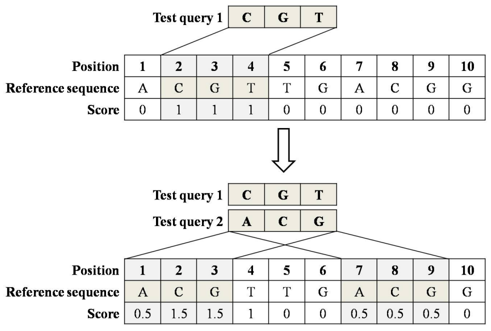

Figure 2. Scoring example

Since *test queries* are produced from random position of the *test sequence* with overlapping, they cannot cover the whole region of *test sequence* evenly. Even in a gain region, there could be some positions whose *score* is noticeably low. Therefore it is not comprehensive to identify a region as gain only by considering continuous positions having high *score*. So we make each 1kbps of nucleotides into a *group*, by the sliding window method, such as 1 to 1000, 2 to 1001, 3 to 1002, and so on. Then, the sum of *score* of each *group* is calculated. These summed *scores* are called a *group score*. *Group1 score* means the sum of *scores* from *position* 1 to 1000. *Group size* can be changed by user.

We assumed the distribution of *group scores* follows normal distribution. In the normal distribution, *group scores* which exceed $\mu + k\sigma$ (μ = average of whole *group scores*, σ = standard deviation of whole *group scores*, k = constraint parameter) can be statistically significant. In other words, the regions whose *group score* exceeds $\mu + k\sigma$ can be detected as gain. If the value of k increases, false negative rate will increase and false positive rate will decrease. On the other hand, if the value of k decreases, false positive rate will increase and false negative rate will decrease. Thus it is important to find optimal k that minimizes both false positives and false negatives.

However, it's not reasonable to determine whether the *group* is a CNV or not simply by the *score* previously mentioned, since the *group score* may be affected excessively by *score*s of both ends. We will describe how to revise the *score* with Figure 3.

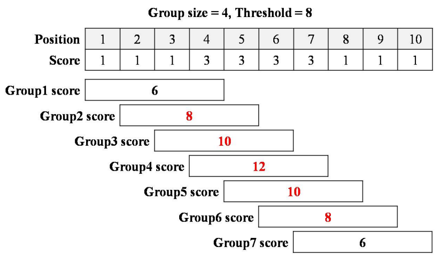

Figure 3. Revised CNV

In this example, the region which is likely to be a CNV is *position* 4~7. However *position* 2~9 are identified as a CNV according to the scoring algorithm mentioned previously. For example, *group2 score* is affected by position 4~5, which have relatively high *score*, and *group6 score* is affected by position 6~7 in the same way. Even though *group2* and *group6* have some positions whose *score* is low, like position 2~3 and position 8~9, they can be selected as gain. We do want to exclude these positions, 2~3 and 8~9 from the CNV. This problem can be avoided by setting threshold higher, however that may increase false negative rate. We reduced the CNV by adding the half of the group size to the beginning position of the CNV (positions 2~9 become 4~9), and by subtracting the half of the group size from the ending position of the CNV (positions 4~9 become 4~7). We reduced the CNV by 500bps, which is the half of the *group size*.

In addition, sometimes a *group score* may exceed threshold because of a few positions whose scores are very high, though the region is not a gain, actually. Several copies of short insertions can cause this problem. To prevent this problem, we enforced that the number of *positions* of which score is over ($\mu + k\sigma$) / *group size* should be more than a certain number, to be identified as gain. We call this parameter *minimum threshold*. We used 500, which is obtained experimentally.

If a region satisfies all the conditions previously mentioned, which are

1. The *group score* acquired by our method is higher than $\mu + k\sigma$.
2. Both ends of the continuous groups acquired by 1 are revised by 500bps.
3. The number of *positions* whose score is over ($\mu + k\sigma$) / *group size* is more than *minimum threshold*.
4. Length is more than 1kbps.

, then the region is identified as a gain.

III. Experimental Results

NT_011255.14, which is one of the contigs in chromosome 19 is used as the *reference sequence*. The total length of the *reference sequence* is 7286004bps. Then *test sequence* is generated from the *reference sequence* by synthetic genomic-variants generator.

As mentioned previously, it is important to find the optimal value of *k*. We performed iterative experiments varying *k*, using 20 genome sequences generated by synthetic genomic-variants generator. We selected *k* which minimizes the sum of the false positive and false negative rates.

The result of experiments is displayed in Figure 4. A pair of dotted line and solid line with same color represents one experiment. Dotted line indicates false positive rate, and solid line indicates false negative rate. As we can see in Figure 4, the intersecting points of each graph pair minimize the sum of false positive and false negative rates, concurrently. Therefore value of *k* at intersecting point can be said to be optimal. We can find the range of optimal *k* to be in between 1.7 and 1.85. The average value of *k* in 20 experiments is 1.75, so we used k=1.75 for further experiments.

Next, we performed experiment to detect CNVs in *test sequence* produced by synthetic genomic-variants generator, using default parameters. We set *coverage* as 3, *group size* as 1000, *minimum threshold* as 500, and *k* as 1.75.

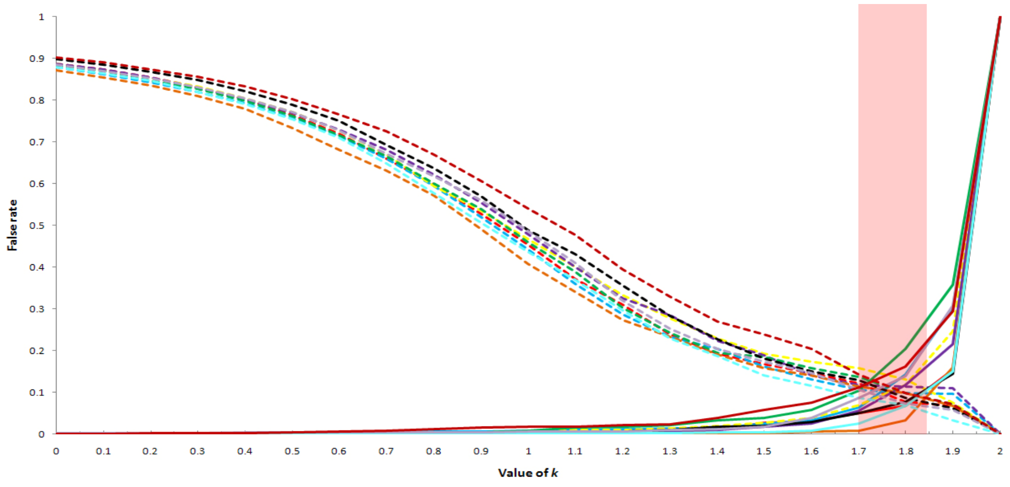

Figure 4. Experiments to detect optimal k

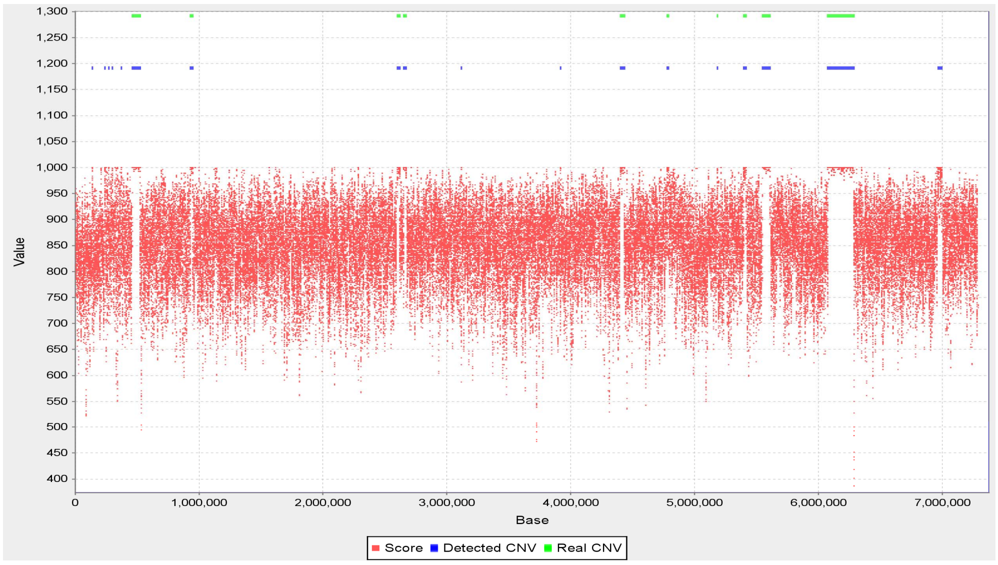

Figure 5. Result of validation

Figure 5 shows the result. X axis is the *position* of the *reference sequence*, and Y axis means the *score*. The *score* of each *group* is marked with a red dot. The green and red lines indicate the region where

CNVs actually exist and the region detected as CNVs by our method. The summary of our experiments are shown in Table 4.

TABLE 4. THE SUMMARY OF THE EXPERIMENT

Category	Value
Total CNV regions	484724 bps
Detected CNV regions	450286 bps
False positives	24824 bps
False negatives	59262 bps
False positive rate	0.0551294
False negative rate	0.122259

IV. CONCLUSION

We described the approach to detect CNVs of human genomes using the reads produced by high-throughput sequencing. Firstly, we developed synthetic genomic-variants generator to produce synthetic genom]u=i90-e sequences with various types of variants including CNVs. To our knowledge, this synthetic genomic-variants generator is the first formulation for genomic-variants problems. It can be applied to similar experiments for detecting genomic variants including CNVs in a computational way.

Then we simulated high-throughput sequencing using this synthetic genome sequence to produce test queries and align them using BLAST. Finally, we detected CNVs using our own detecting algorithm.

By comprehensive experiments, we demonstrated that our method can detect CNVs of arbitrary length, including very short ones, with relatively low coverage. Moreover, our method has low false positive and negative rates compared to those of microarray based methods.

In the future, we plan to improve our synthetic genomic-variants generator to produce sequences with genomic variants much more similar to the real human genome sequences, and apply our method to real short reads.

ACKNOWLEDGMENT

This work was supported by the Korea Science and Engineering Foundation(KOSEF) grant funded by the Korea government(MOST) (No. 2008-2004103).

REFERENCES

[1] A. J. Iafrate, L. Feuk, M. N. Rivera et al., “Detection of large-scale variation in the human genome”, Nature Genetics, Vol. 36, pp.949-851, 2004.

[2] R. Redon, S. Ishikawa, K. R. Fitch et al., “Global variation in copy number in the human genome”, Nature, Vol. 444, pp. 444–454, 2006.

[3] J. L. Freeman, G. H. Perry, L. Feuk et al., “Copy number variation: New insights in genome diversity”, Genome Research, Vol.16, pp.949-961, 2006.

[4] L. Feuk, A. R. Carson, S. W. Scherer, “Structural variation in the human genome”, Nature Reviews Genetics, Vol.7, pp. 85–97, 2006.

[5] S. A. McCarroll, D. Altshuler, “Copy-number variation and association studies of human disease”, Nature Genetics, Vol. 39, pp. S37–S42, 2007.

[6] T.S. Price, R. Regan, R. Mott, et al., “SW-ARRAY: a dynamic programming solution for the identification of copy-number changes in genomic DNA using array comparative genome hybridization data”, Nucleic Acids Research, Vol. 33, No. 11, pp. 3455− 3464, 2005

[7] H. Fiegler, R. Redon, D. Andrews et al., “Accurate and reliable high-throughput detection of copy number variation in the human genome”, Genome Research, Vol. 16, pp. 1566− 1574, 2006.

[8] D. Komura, F. Shen, S. Ishikawa et al., “Genome-wide detection of human copy number variations using high-density DNA oligonucleotide arrays”, Genome Research., Vol. 16, pp. 1575-1584, 2006.

[9] K. Wang, M. Li, D. Hadley et al., “PennCNV: An integrated hidden Markov model designed for high-resolution copy number variation detection in whole-genome SNP genotyping data”, Genome Research, Vol.17, pp.1665-1674, 2007.

[10] E. Tuzun, A. J. Sharp, J. A. Bailey et al., “Fine-scale structural variation of the human genome”, Nature Genetics, Vol. 37, No. 7, pp. 727-732, 2005.

[11] R. E. Mills, C. T. Luttig, C. E. Larkins et al., “An initial map of insertion and deletion (INDEL) variation in the human genome”, Genome Research, Vol. 16, pp. 1182-1190, 2006.

[12] R. Khaja, J. Zhang, J. R. MacDonald et al., “Genome assembly comparison identifies structural variants in the human genome”, Nature Genetics, Vol. 38, No. 12, pp. 1413-1418, 2006.

[13] F. S. Robert, “The Race for the $1000 Genome”, Science, Vol. 311, pp. 1544-1546, 2006.

[14] R. L. Warren, G. G. Sutton, S. J. Jones, R. A. Holt, “Assembling millions of short DNA sequences using SSAKE”, Bioinformatics, Vol. 23, No. 4, pp. 500-501, 2007.

[15] W. R. Jeck, J. A. Reinhardt, D. A. Baltrus et al., “Extending assembly of short DNA sequence to handle error”, Bioinformatics, Vol. 23, No. 21, pp. 2942-2944, 2007.

[16] J. C. Dohm, C. Lottaz, T. Borodina, H. Himmelbauer, “SHARCGS, a fast and highly accurate short-read assembly algorithm for de novo genome sequencing”, Genome Research, Vol. 17, No. 11, pp. 1697-1706, 2007.

[17] A. J. Iafrate, L. Feuk, M. N. Rivera et al., “Detection of large-scale variation in the human genome”, Nature Genetics, Vol. 36, No. 9, pp. 949-951, 2004.

[18] S. F. Altschul, T. L. Madden, A. A. Schäffer et al., “Gapped BLAST and PSI-BLAST: a new generation of protein database search programs”, Nucleic Acids Research, Vol. 25, No. 17, pp. 3389-3402, 1997.

[19] http://www.ncbi.nlm.nih.gov/projects/SNP/