HHS Public Access

Author manuscript

Proc Data Compress Conf. Author manuscript; available in PMC 2015 October 23.

Published in final edited form as:

Proc Data Compress Conf. 2013 ; 2013: 371–380. doi:10.1109/DCC.2013.45.

An Adaptive Difference Distribution-based Coding with Hierarchical Tree Structure for DNA Sequence Compression

Wenrui Dai,
Department of Electronic Engineering Shanghai Jiaotong University Shanghai 200240, China, daiwenrui@sjtu.edu.cn

Hongkai Xiong,
Department of Electronic Engineering Shanghai Jiaotong University Shanghai 200240, China, xionghongkai@sjtu.edu.cn

Xiaoqian Jiang, and
Division of Biomedical Informatics University of California, San Diego San Diego, CA 92093, USA, x1jiang@ucsd.edu

Lucila Ohno-Machado
Division of Biomedical Informatics University of California, San Diego San Diego, CA 92093, USA, lohnomachado@ucsd.edu

Abstract

Previous reference-based compression on DNA sequences do not fully exploit the intrinsic statistics by merely concerning the approximate matches. In this paper, an adaptive difference distribution-based coding framework is proposed by the fragments of nucleotides with a hierarchical tree structure. To keep the distribution of difference sequence from the reference and target sequences concentrated, the sub-fragment size and matching offset for predicting are flexible to the stepped size structure. The matching with approximate repeats in reference will be imposed with the Hamming-like weighted distance measure function in a local region closed to the current fragment, such that the accuracy of matching and the overhead of describing matching offset can be balanced. A well-designed coding scheme will make compact both the difference sequence and the additional parameters, e.g. sub-fragment size and matching offset. Experimental results show that the proposed scheme achieves 150% compression improvement in comparison with the best reference-based compressor GReEn.

I. Introduction

With the development of high-throughput sequencing technologies, rapid reduction of sequencing cost enables the research projects centered on individual genomics and personalized medicine. The large scale projects such as the 1000 Genomes Project (http://www.1000genomes.org/) and The Cancer Genome Atlas (http://cancergenome.nih.gov/) have been contributing to the unprecedented volume of DNA sequences. As pointed out by Kahn [1], the exponential explosion in genomic data has presented a significant challenge to the disk storage and high-performance computation. It is crucial for the development of novel efficient compression techniques to close the reality gap.

DNA sequences are characterized with repeated patterns of four different kinds of nucleotides, namely Adenine (A), Cytosine (C), Guanine (G) and Thymine (T). General purpose compression algorithms such as compress, gzip and bzip2 fail to compress DNA sequences without taking DNA structures into sufficient consideration. Consequently, a series of specialized compression methods are proposed to focus on the characteristic structures such as approximate repeats (repeats with mutations) and complementary palindromes (reversed repeats). Inspired by Ziv-Lempel data compression method [2], Grumbach and Tachi proposed the first specific DNA sequence compressor Biocompress [3], to compress the exact repeats with the specifically designed Fibonacci coding. The compression performance was improved in successive literatures by the introduction of Markov model for non-repeated regions [4], extension to the approximate repeats for further exploitation of the structures in DNA sequences [5], [6], and utilization of dynamic programming for optimal detection and matching of approximate repeats [7], [8]. Although methods based on approximate repeats show promising results, no theoretic principles on approximate matching algorithm has been established for such heuristic methods. Consequently, statistical-based methods were introduced for the intensive prediction of the generation of the nucleotides. [9]–[11] proposed the normalized maximum likelihood model to determine the best regressor for matching and substitution of variablesize approximate repeats. XM [12] estimated the probability distribution of symbols by combining a panel of ”experts” with the repeat expert concerning the approximate repeats. Finite context models are also proposed and compared to rapidly capture variable-order statistical information along the DNA sequences [13], [14]. In spite of the evolutionary development of compression techniques, reference-free methods are subjected to their low compression rate (not greater than 6:1) and prohibitive computational cost for large DNA data sets.

Since the significant part of the genome is shared among individuals of the same species, reference-based compression methods are proposed to utilize such redundancy for more efficient compression. The idea for storing and reducing redundant genomic data was firstly based on additional information, e.g. single nucleotide polymorphism (SNP) databases [15] or insert and delete operations [16]. To eliminate the additional information, the RLZ algorithm proposed by Kuruppu *et al.* [17] performed relative Lempel-Ziv compression of DNA sequences with the collection of related sequences but could not handle the sequences with characters outside the alphabet {*A, T, G, C, N*}. However, resequencing techniques inevitably introduce additional characters into the alphabet, e.g. the lower case character {*a, t, g, c, n*}, to represent the uncertainty at a certain position in DNA sequences. Wang *et al.* [18] proposed the general Genome ReSequencing (GRS) tool for compressing and storing the sequencing data with the reference by considering the chromosome varied sequence percentage. For the efficient compressive performance and robust support for arbitrary alphabets, GReEn [19] applied the copy model into the matching of exact repeats in reference sequences and established probabilistic model for such matching. GReEn achieved better coding gain when compared to [17] and [18]. The recent trend of reference-based methods implies that matching and representing repeated patterns with the reference in a probabilistic manner significantly improves the performance of genome compression techniques. However, these methods cannot fully exploit the redundancies in the reference-

based compression, since the variable sizes and offsets of repeats and the exception of insertion, deletion and substitution in matching degrade its efficiency.

In this paper, we propose a novel framework on the fragments of nucleotides with a hierarchical tree structure for the reference-based genome sequence compression. In each fragment, the sub-fragment size and matching offset for predicting are flexible to the stepped size structure. The matching with approximate repeats in reference would be imposed with the Hamming-like weighted distance measure function in a local region closed to the current fragment, such that the accuracy of matching and the overhead of representing matching offset can be balanced. Specifically, the distribution of the difference sequence from the reference and target sequences is kept concentrated and consequently suitable for compression. Finally, a well-designed coding scheme will make compact both the difference sequence and the additional parameters, e.g. sub-fragment size and matching offset. The proposed method is robust in dealing with arbitrary alphabets for the case in which the alphabet is not constrained to $\{A, T, G, C\}$ due to a low resequencing quality. Experimental results show 150% compression improvement in comparison with the best reference-based compressor GReEn.

The rest of this paper is organized as follows. Section II presents the proposed framework, which includes the construction of Hamming-like distance measure function as well as the well-designed coding scheme. The reference-based experimental results on two assemblies of human genome are evaluated in Section III. Section IV draws the conclusion and makes the discussion.

II. The Proposed Framework

A. Adaptive Difference Distribution-based Coding Framework

This section presents the proposed framework for the adaptive compression of difference between reference sequence and the encoding sequence. The introduction of difference sequence is due to the fact that DNA sequences are characterized with approximately repeated patterns with exception of single insertion, deletion and substitution. The distribution of the obtained difference sequences is not uniform, and only several symbols appear in a high frequency, as witnessed in Fig. 2-4.

The generic genome compression framework based on the difference sequence is depicted in Fig. 1. The sequence for compression is segmented into fragments of nucleotides with size MAX_FRAG_SIZE, such that the sequence is predicted individually based on each fragment. A hierarchical tree structure with MAX_TREE_DEPTH is constructed for each fragment. The fragment can be divided into sub-parts by iteratively halving its size according to the hierarchical tree. The introduction of hierarchical structure of halving sub-parts rather than fragments with arbitrary sizes is to maintain a compact alphabet of fragment sizes for coding. For example, if MAX_FRAG_SIZE is 256 and MAX_TREE_DEPTH is 6, the sub-fragment size SF_SIZE $\in \{256, 128, 64, 32, 16, 8\}$. Under such settings, the approximate repeats in the genome sequence can be flexibly predicted by adaptively switching to the proper sub-fragment size.

The prediction of each fragment is obtained by subtracting the most similar subsequences of nucleotides in the reference. Differences can be obtained directly by comparing the ASCII values of corresponding symbols in the reference and target sequences. Fig. 2 gives an example, where a sub-fragment of 16 nucleotides in target sequence is predicted by subtracting the corresponding one in reference. It is obvious that the target sequence can be reconstructed from the difference sequence with the additional parameter SF_OFFSET = 0 and SF_SIZE = 16. These two parameters are also required in the decoder.

The combination of sub-fragments in reference sequence that differ from the current fragment in shortest distance are sought in prediction. Fig. 3 shows an example for selecting two sub-fragments of 8 nucleotides as the reference for the fragments of 16 nucleotides. Commonly, such searching is constrained in the local region around the position of current sub-fragment, since emerging long offset will consume large amount of bits in coding even though it might obtain better matching. When given the MAX_OFFSET for searching, the matching offset SF_OFFSET could be {0, ±1, ⋯ , ±MAX_OFFSET}. Denote F_n the current sub-fragment for predicting and $\hat{F}_n$ the one matching F_n in the reference sequence, the coding cost $J(F_n)$ can be formulated as

$$J_{F_n}\left(F_n, \hat{F}_n, PARAM\right) = J\left(F_n - \hat{F}_n\right) + J\left(PARAM\right), \quad (1)$$

where PARAM = {SF_SIZE(F_n), SF_OFFSET(F_n)} is the parameter set indicating current sub-fragment size and matching offset. Consequently, the reference-based prediction is to find the best matching of current sub-fragment that achieves

$$\left\{\hat{F}_n^*, PARAM^*\right\} = arg \min_{\hat{F}_n, PARAM} J_{F_n}\left(F_n, \hat{F}_n, PARAM\right). \quad (2)$$

The best matching can be found by traversing all possible settings of the parameter set PARAM. Theoretically, the cost function indicating the empirical entropy, $J(\cdot) = -\log_2 P(\cdot)$, is expected to achieve the least code length. However, under such cost function, it is hard to find the concurrent optimized solution for all the sub-fragments in iterative hierarchical tree structure. For the efficient estimate of coding cost, the Hamming-like distance measurement is introduced.

B. Distance Measurement

In this subsection, the Hamming-like distance measurement is introduced. As mentioned above, since the main part of the genome is shared among individuals of the same species, the difference between the encoding sequence and reference sequence tends to be long uniform string with the emergence of unexpected symbols. These unexpected symbols are hard to be predicted accurately because of their low probabilities of appearance. Consequently, it needs much more bits to represent these unexpected symbols in the compressed files, which may be greater than their raw lengths. As a result, the Hamming-like distance can approximately estimate the coding cost for the obtained difference sequence.

Denote $F_n=\{x_i\}_{i=1}^m$ and $\hat{F}_n=\{\hat{x}\}_{i=1}^m$ the sub-fragment for encoding and its corresponding reference respectively. When x_i equals its corresponding nucleotide $\hat{x}_i$ in reference, the Hamming distance Hamm $(x_i, \hat{x}_i)$ is set to zero. If x_i does not equal $\hat{x}_i$, the Hamming distance is increased to represent the difference. However, the difference in cases of the nucleotides (e.g. 'a' and 'A', 'g' and 'G', and etc.) contributes to the majority of obtained differences. As shown in Fig. 2, the difference between 16 nucleotides is 10 '00' and 6 'E0' (0xE0 indicates the difference between lower case and upper case of the same character in ASCII). The results in Fig. 4 demonstrate the fact that zero difference and the difference between the upper and the lower case of same nucleotides commit almost all the distribution of difference symbols. Thus, a set of weights of Hamming distance are assigned to the various difference by approximately comparing their frequencies shown in Fig. 4.

$$Hamm \quad (x_i, \hat{x}_i) = \begin{cases} 0 & x_i = \hat{x}_i \\ 1 & \|x_i - \hat{x}_i\| = 0xE0 \\ 50 & otherwise \end{cases} \quad (3)$$

The weight for all the other difference is large enough such that it will not affect the detection of exact match and difference in cases. Eq. 3 implies that the difference sequence is formulated as the long uniform string of 0 or 0xE0 with the others appearing as the unexpected symbols. Consequently, the Hamming distance between two sub-fragment is defined as

$$Hamm \quad \left(F_n, \hat{F}_n\right) = \sum_{i=1}^{m} Hamm \quad (x_i, \hat{x}_i) .$$

C. Coding of Difference Sequence

The distribution of difference sequence is suitable for the high-efficiency compression, as shown in the histograms in Fig. 4(a) and (b). A switching structure is proposed for the coding of difference sequence. The switching coding structure proposes run length coding for the fragments with same values and the textual compressor PPM [20] as the routine encoder. To be concrete, the general purpose textual compression tool PPMDj is adopted for the common coding of difference sequence, which is consistent in coding by making the code length independent of the appearance order of the context symbols. The concentrated distribution of difference sequence is suitable for the symbol-based compressor. In addition, run length coding is developed for the fragments of difference with unique values, e.g. 0 or 0xE0. Such fragments are indicated with symbol "00FF" and "E0FF" for the brief representation in the coding scheme. The switching structure will decrease the coding cost by fitting the statistics of various regions in the difference sequence.

D. Corporative Coding of Parameter Sets

The parameter sets are required for the reconstruction of the encoding DNA sequence from the difference sequence. Its parameters include the size and matching offset for each predicted sub-fragment. They are stored in the unit of sub-fragment with MIN_FRAG_SIZE. Compression of these two sets of parameters is not isolated. Each set of

parameters can be taken as the context for encoding the other. Denote $\{S_1, S_2, \cdots, S_n\}$ and $\{O_1, O_2, \cdots, O_n\}$ the predicted sub-fragment size and matching offset. The contexts for predicting current size S and offset O are constructed in Table I, where n is the maximum context order. Based on above context models, the parameter sets are compressed with the arithmetic coder.

III. Experimental Results

In this section, the proposed method is evaluated by comparing with the benchmark reference-based compressor GReEn [19] and GRS [18], among which GReEn is the best reference-based compressor for FASTA format genomic data. Two assemblies of human genome, YH and KOREF_20090224 were compressed based on the reference sequences for validation. All experimental results were obtained using an Intel Core i7-3620QM CPU laptop computer at 2.2 GHz with 8 GB of memory and VC++ 9.0 compiler.

The implementation of the proposed method can be referred to Algorithm 1. In this implementation, MAX_FRAG_SIZE and MIN_FRAG_SIZE were set to 256 and 8 respectively. The maximum depth for hierarchical tree was 6 and the maximum matching offset for approximate repeats was 32. These settings can be further tuned for optimal performance, although they are already qualified to validate our method in this paper. The proposed method was implemented iteratively, where at each depth of hierarchical tree, the sub-fragments were divided into NUM_PART = 2 subparts. The hierarchical tree can be stored in memory as the proposed method is based on fragment of nucleotides with constrained size MAX_FRAG_SIZE. Consequently, the difference sequence was obtained by subtracting the combination of variable size sub-fragments based on the optimal matching in reference sequence within the constrained local region.

Table II shows the compression results for the KOREF_20090224 human genome using the KOREF_20090131 as reference. In Table II, the proposed method gives consistently better results compared to GReEn and GRS. The proposed method achieves a 480 folds compression ratio in average, which is 1.5 times better than what GReEn achieves. Since KOREF_20090224 and KOREF_20090131 are the various versions of the same ethnic group, there are massive similar repeats between the two sequences which leads to the high efficiency compression.

Besides that, an additional investigate for compression human genome assemblies is made. Table III displays the compression results for the YH human genome using KOREF_20090224 as reference. YH and KOREF_20090224 are both the individual genome based on resequencing data from massively parallel sequencing technologies. However, they are different in some extent as they are from different ethnic groups. Table III shows that GRS fails to compress most of the sequences because of the excessive difference between the reference and target sequences. The proposed method outperforms GReEn with the exception of chromosome ChrM. An average 150% improvement in compression ratio is witnessed. The reason for the less effective performance of the proposed method in ChrM is probably because ChrM is relatively small and the overhead led by the size and mode offset of sub-fragment outrides the gain in compression.

The experiments on the assemblies of human genome demonstrate that the proposed method provides the efficient and robust support for the genome compression (with reference) at the presence of large gaps and arbitrary alphabets.

IV. Conclusion and Discussion

Recognizing the insufficient exploitation of statistics of DNA sequences in the reference-based compressor, an adaptive difference distribution-based coding framework for DNA sequence is proposed. Exploiting the characteristic structures of approximate repeats in DNA sequences, difference sequences obtained from the reference and target sequences commit a more concentrated probabilistic distribution of symbols for coding. The weighted Hamming-like distance measurement in a local region is imposed is imposed to match the approximate repeats and formulate the difference sequences. The size and matching offset of the sub-fragments for prediction are determined by a hierarchical tree structure in the fragment of nucleotides. A well-designed coding scheme compresses both the difference sequence and the additional parameters, e.g. sub-fragment size and matching offset. Experimental results shows that the proposed scheme achieve 150% compression improvement in comparison with the benchmark compressor GReEn and GRS.

The introduction of difference distribution-based coding framework in DNA sequence compression is meaningful, since it could be an alternative way to exploit the specific DNA structures. Distinguished from explicit methods that find and encipher optimal matching for approximate repeats, the proposed framework implicitly extracts difference sequences from reference and target sequences for a more concentrated probabilistic distribution of symbols for coding. This framework reduces the excess overhead led by exception of insertion, deletion and substitution in matching repeats. Such adaptive hierarchical coding framework can be further improved with sophisticated coding of difference sequences and efficient prediction of size and matching offset of the sub-fragments, e.g. sliding window with dynamic decision of size for obtaining difference sequences, suffix tree for maintaining the hierarchical coding structure, and etc.

Acknowledgement

The work has been partially supported by the NSFC, under Grants U1201255, 61271218, 61271211, and 61228101. X. Jiang and L. Ohno-Machado were funded in part by EDM Forum grant U13HS19564, AHRQ grant R01HS019913 and NIH grants K99LM011392, R01LM009520, U54HL108460, and UL1TR000100.

REFERENCES

1. Kahn S. On the future of genomic data. Science (Washington). Feb.2011 331(6018):728–729. [PubMed: 21311016]
2. Ziv J, Lempel A. Compression of individual sequences via variable-rate coding. IEEE Trans. Inf. Theory. Sep.1978 24(5):530–536.
3. Grumbach, S.; Tahi, F. Proc. Data Compression Conf. Snowbird; Utah, USA: Mar.. 1993 Compression of DNA sequences; p. 340-350.
4. Grumbach S, Tahi F. A new challenge for compression algorithms: Genetic sequences. J. Inf. Process. Manage. Nov.1994 30(6):876–887.
5. Chen X, Kwong S, Li M. A compression algorithm for DNA sequences. IEEE Eng. Med. Biol. Mag. Jul.2001 20(4):61–66. [PubMed: 11494771]

6. Chen X, Li M, Ma B, Tromp J. DNACompress: fast and effective DNA sequence compression. Bioinformatics. Dec.2002 18(12):1696–1698. [PubMed: 12490460]
7. Matsumoto T, Sadakane K, Imai H. Biological Sequence Compression Algorithms. Genome Informatics. Dec.2000 11:43–52. [PubMed: 11700586]
8. Behzadi B, Le Fessant F. DNA compression challenge revisited: A dynamic programming approach. Combinatorial Pattern Matching. Jun.2005 3537:85–96.
9. Tabus, I.; Korodi, G.; Rissanen, J. Proc. Data Compression Conf. Snowbird; Utah: Mar.. 2003 DNA sequence compression using the normalized maximum likelihood model for discrete regression; p. 253-262.
10. Korodi G, Tabus I, Rissanen J, Astola J. DNA sequence compression - Based on the normalized maximum likelihood model. IEEE Signal Process. Mag. Jan.2007 24(1):47–53.
11. Korodi, G.; Tabus, I. Proc. Data Compression Conf. Snowbird; Utah, USA: Mar.. 2007 Normalized maximum likelihood model of order-1 for the compression of DNA sequences; p. 33-42.
12. Cao, MD.; Dix, TI.; Allison, L.; Mears, C. Proc. Data Compression Conf. Snowbird; Utah, USA: Mar.. 2007 A simple statistical algorithm for biological sequence compression; p. 43-52.
13. Pinho, AJ.; Neves, A.; Bastos, C.; Ferreira, P. DNA coding using finite-context models and arithmetic coding. Proc. IEEE Int. Conf. Acoustics, Speech and Signal Process.; Taipei. Apr. 2009; p. 1693-1696.
14. Pratas D, Pinho A. Compressing the human genome using exclusively Markov models. 5th Int. Conf. on Practical Applications of Comput. Biol. Bioinformatics (PACBB 2011). Mar.2011 :213–220.
15. Christley S, Lu Y, Li C, Xie X. Human genomes as email attachments. Bioinformatics. Jan.2009 25(2):274–275. [PubMed: 18996942]
16. Brandon MC, Wallace DC, Baldi P. Data structures and compression algorithms for genomic sequence data. Bioinformatics. Jul.2009 25(14):1731–1738. [PubMed: 19447783]
17. Kuruppu S, Puglisi S, Zobel J. Relative Lempel-Ziv compression of genomes for large-scale storage and retrieval. String Process. and Inf. Retrieval. Oct.2010 6393:201–206.
18. Wang C, Zhang D. A novel compression tool for efficient storage of genome resequencing data. Nucleic Acids Res. Apr.2011 39(7):e45. [PubMed: 21266471]
19. Pinho A, Pratas D, Garcia S. GReEn: A tool for efficient compression of genome resequencing data. Nucleic Acids Res. Feb.2012 40(4):e27. [PubMed: 22139935]
20. Cleary JG, Witten IH. Data compression using adaptive coding and partial string matching. IEEE Trans. Communication. Apr.1984 32(4):396–402.

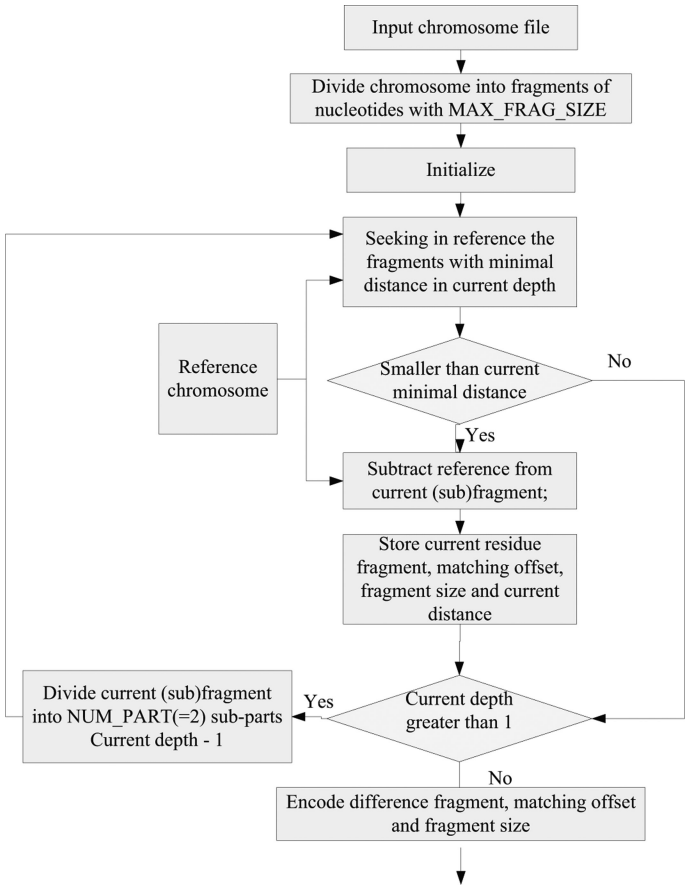


Fig. 1.
The flowing diagram for the proposed framework

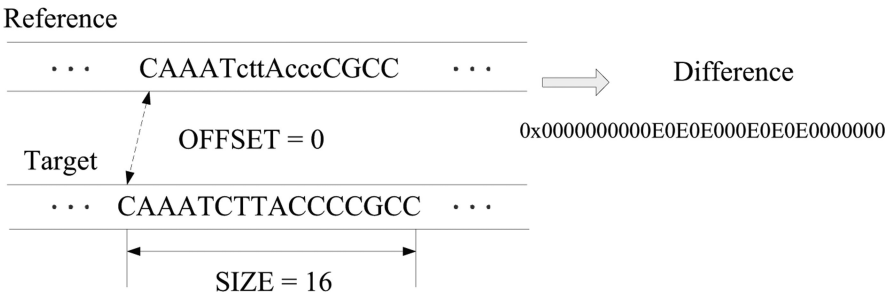


Fig. 2.

An example for the proposed framework. The fragment of 16 nucleotides is predicted based on the reference. The difference fragment is obtained by subtracting the selected reference from the target.

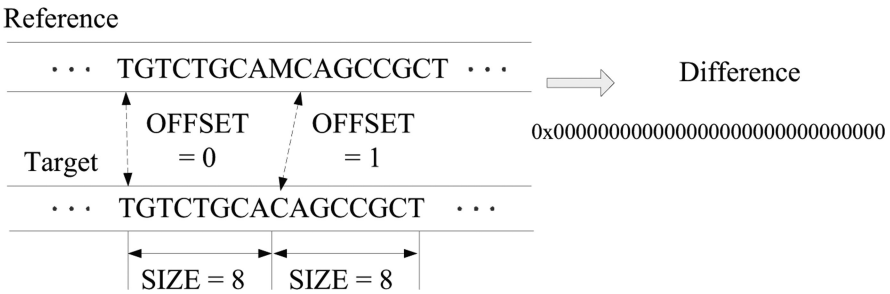
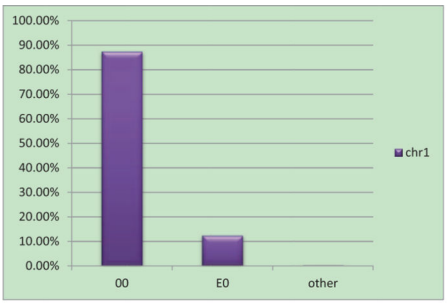
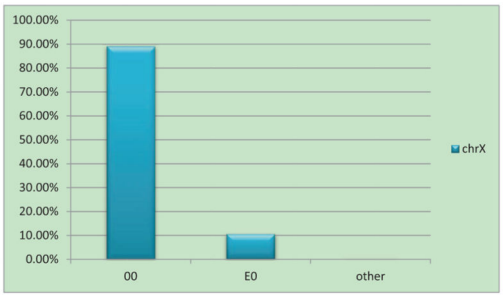


Fig. 3.

An example for the proposed framework. The fragment of 16 nucleotides is predicted based on two sub-fragment of 8 nucleotides in reference. The difference fragment is obtained by subtracting the selected reference from the target.



(a) Distribution of difference for chromosome Chr1



(b) Distribution of difference for chromosome ChrX

Fig. 4.

Distribution of difference in YH human genome with reference KOREF_20090224

TABLE I

Context construction based on predicted sub-fragment size and mode offset

$S_1 = S_2$	$S = P(\{S_1, S_2, \cdots, S_n\})$	$O = P(\{O_1, \cdots, O_n\})$
$S_1 \neq S_2, O_1 \neq O_2$	$S = S_1$	$O = P(\{O_2, \cdots, O_n\})$
$S_1 \neq S_2, O_1 = O_2$	$S = S_1$	$O = P(\{O_3, \cdots, O_n\})$

TABLE II

Performance of the proposed method in compressing the sequence KOREF_20090224 using KOREF_20090131 as the reference

Chr	Raw file (byte)	Proposed scheme		GReEn		GRS	
		Byte	Ratio	Byte	Ratio	Byte	Ratio
1	247249719	450642	548.7	1225767	201.7	1336626	185.0
2	242951149	448789	541.3	1272105	191.0	1354059	179.4
3	199501827	346616	575.6	971527	205.3	1011124	197.3
4	191273063	378619	505.2	1074357	178.0	1139225	167.9
5	180857866	328193	551.1	947378	190.9	988070	183.0
6	170899992	308719	553.6	865448	197.5	906116	188.6
7	158821424	345454	459.7	998482	159.1	1096646	144.8
8	146274826	261982	558.3	729362	200.6	764313	191.4
9	140273252	286168	490.2	773716	181.3	864222	162.3
10	135374737	257389	526.0	717305	188.7	768364	176.2
11	134452384	252522	532.4	716301	187.7	755708	177.9
12	132349534	239887	551.7	668455	198.0	702040	188.5
13	114142980	183914	620.6	490888	232.5	520598	219.3
14	106368585	171257	621.1	451018	235.8	484791	219.4
15	100338915	168867	594.2	453301	221.4	496215	202.2
16	88827254	182593	486.5	510254	174.1	567989	156.4
17	78774742	162958	483.4	464324	169.7	505979	155.7
18	76117153	137162	554.9	378420	201.1	408529	186.3
19	63811651	134458	474.6	369388	172.7	399807	159.6
20	62435964	101199	617.0	266562	234.2	282628	220.9
21	46944323	78570	597.5	203036	231.2	226549	207.2
22	49691432	88596	560.9	230049	216.0	262443	189.3
M	16571	67	247.3	127	130.5	183	90.6
X	154913754	935464	165.6	2712153	57.1	3231776	47.9
Y	57772954	165553	349.0	481037	120.0	592791	97.5
Total	3080436051	6415638	480.1	17971030	171.4	19666791	156.6

The size of compressed file (in bytes) and compression ratio of the proposed scheme, GReEn and GRS are shown respectively. The compression ratio is obtained by raw_fiie_size/compressed_file_size.

TABLE III

Performance of the proposed method in compressing the sequence YH using KOREF_20090224 as the reference

Chr	Raw file (byte)	Proposed scheme		GReEn		GRS	
		Byte	Ratio	Byte	Ratio	Byte	Ratio
1	247249719	965165	256.2	2349124	105.3	-	-
2	242951149	956853	253.9	2420007	100.4	-	-
3	199501827	781239	255.4	1730477	115.3	17410946	11.5
4	191273063	824032	232.1	1877056	101.9	-	-
5	180857866	727139	248.7	1792278	100.9	-	-
6	170899992	720526	237.2	1588739	107.6	25815446	6.6
7	158821424	714796	222.2	1820425	87.2	-	-
8	146274826	594668	246.0	1358770	107.7	-	-
9	140273252	572769	244.9	1476495	95.0	-	-
10	135374737	562035	240.9	1353193	100.0	-	-
11	134452384	564596	238.1	1274433	105.5	-	-
12	132349534	538248	245.9	1174966	112.6	16136610	8.2
13	114142980	396867	287.6	866266	131.8	11227954	10.2
14	106368585	382754	277.9	826672	128.7	-	-
15	100338915	355867	282.0	892429	112.4	-	-
16	88827254	378642	234.6	1015246	87.5	-	-
17	78774742	323710	243.3	864710	91.1	-	-
18	76117153	316497	240.5	713787	106.6	13187892	5.8
19	63811651	272346	234.3	589422	108.3	-	-
20	62435964	246879	252.9	493404	126.5	8409776	7.4
21	46944323	181559	258.6	374383	125.4	726269	64.6
22	49691432	191302	260.0	444932	111.7	-	-
M	16571	139	119.2	127	130.5	321	51.6
X	154913754	863394	179.4	3258188	47.5	-	-
Y	57772954	180713	319.7	859688	67.2	-	-
Total	3080436051	12612735	244.2	31415217	98.1	-	-

The size of compressed file (in bytes) and compression ratio of the proposed scheme, GReEn and GRS are shown respectively. The compression ratio is obtained by raw_file_size/compressed_file_size.

Algorithm 1

Proposed scheme for adaptive difference-based compression framework

```
Segment the input chromosome file into fragments with MAX_FRAG_SIZE = 256 and MAX_DEPTH = 6.
for All fragments do
    Initialize Depth = MAX_DEPTH and SF_SIZE = MAX_FRAG_SIZE.
    while Depth > 1 do
        NUM_PART = 2.
        for All possible matching offset SF_OFFSET do
            Compare current (sub)fragment with the reference with SF_OFFSET and SF_SIZE.
            Compute the Hamming-like distance as defined in II.B and store the minimal one.
                end for
        if Current distance is minimal then
            Subtract reference from current (sub)fragment with corresponding SF_OFFSET and SF_SIZE.
            Store current difference sequence, current distance, SF_OFFSET and SF_SIZE.
                end if
        Divide current (sub)fragment into NUM_PART sub-parts.
        for All NUM_PART sub-parts do
            Compare current sub-part with the reference.
            Compute the Hamming-like distance as defined in II.B and store the minimal one.
            Obtain the corresponding optimal difference sequence, SF_OFFSET and SF_SIZE.
                end for
        Obtain the total distance for the NUM_PART sub-parts.
        Compare two distances and decided the optimal difference sequence, SF_OFFSET and SF_SIZE.
        end while
    Encode difference fragment, matching offset and fragment size
end for
```