35th Annual International Conference of the IEEE EMBS
Osaka, Japan, 3 - 7 July, 2013

Nucleotide sequence alignment using sparse coding and belief propagation

Aminmohammad Roozgard*, Nafise Barzigar*, Shuang Wang°, Xiaoqian Jiang°, Lucila Ohno-Machado° and Samuel Cheng*

Abstract— **Advances in DNA information extraction techniques have led to huge sequenced genomes from organisms spanning the tree of life. This increasing amount of genomic information requires tools for comparison of the nucleotide sequences. In this paper, we propose a novel nucleotide sequence alignment method based on sparse coding and belief propagation to compare the similarity of the nucleotide sequences. We used the neighbors of each nucleotide as features, and then we employed sparse coding to find a set of candidate nucleotides. To select optimum matches, belief propagation was subsequently applied to these candidate nucleotides. Experimental results show that the proposed approach is able to robustly align nucleotide sequences and is competitive to SOAPaligner [1] and BWA [2].**

I. Introduction

The latest sequencing technologies have generated numerous sequenced genomes for various species. This increasing volume of data requires tools that can accurately compare multiple genome sequences to aid in the study of populations, pan-genomes, and genome evolution [1], [2]. For a particular study, many individual genomes may be sequenced to investigate genetic diversity. For example, the Cancer Genome Atlas [3] and 1000 Genomes Project [4] will generate genome sequences from several thousand people. The complete bacterial genomes in public databases are already over one thousand. To better utilize this huge amount of sequenced genome information, many tools have been developed that are capable of efficiently finding similar sequences from whole genomes. Whole genome sequence alignment are used for studying genome evolution and genetic diversity [5], [6]. For example, Blanchette *et al.*, defined a Threaded Blockset Aligner (TBA) and built a threaded blockset under the assumption that all matching segments occur in the same order and orientation in a given sequence [7]. TBA was designed for aligning megabase-sized regions of multiple mammalian genomes. Darling *et al.* [8] implemented a method for identification and alignment of conserved genomic DNA in the presence of rearrangements and horizontal transfer called Mauve. Mauve has been applied to align nine enterobacterial genomes and to determine global rearrangement structure in three mammalian genomes. There are other whole-genome alignment tools that can align multiple whole genomes such as [9]–[11].

Whole-genome alignment tools are classified from collinear multiple sequence alignment tools, such as tools in [12]–[14] where they can align very long sequences and detect the presence of rearrangements, duplications, and large-scale sequence gains and losses. For example, Bradley *et al.*, in [12] proposed a program for the alignment of multiple biological sequences that is statistically motivated and fast for practical size problems. It was based on pair hidden Markov models which approximate an insertion/deletion process on a tree and used a sequence annealing algorithm to combine the posterior probabilities estimated from these models into a multiple alignment. Edgar *et al.* proposed another alignment tool named MUSCLE which is a program for creating multiple alignments of protein sequences [13]. Elements of the algorithm include fast distance estimation using *k-mer* counting, progressive alignment using a log-expectation score, and refinement using tree-dependent restricted partitioning. In spite of collinear alignment technologies, non-collinear alignment such as [15], [16] contains the elements that are arranged in some non-linear order (see Fig. I.1).

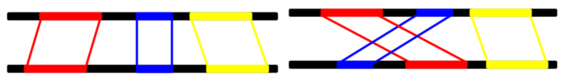

(a) Collinear (b) Non-collinear

Fig. I.1: Collinear vs. Non–collinear nucleotide sequence alignment.

In this paper, we propose a novel alignment technique using sparse coding and belief propagation. First, we build an overcomplete dictionary out of extracted nucleotide features of a reference sequence. We then find a set of candidate nucleotide for each nucleotide of the test sequence using sparse coding from the constructed dictionary. The match score of each candidate nucleotide will be evaluated taking both local and neighboring information into account using belief propagation. The best match will be selected as the candidate with the highest score. The rest of this paper is structured as follows. Section II introduces our proposed method. In Section III, we show and discuss our simulation results and compare them with SOAPaligner [1] and BWA [2], followed by a brief conclusion in Section IV.

II. Proposed Method

The proposed method described here is inspired by our recent work, SCoBeP [17]. First, we map the reference nucleotide sequence $\mathcal{X}$ of size N and the test nucleotide sequence $\mathcal{Y}$ of size M into the two integer sequences,

*School of Electrical and Computer Engineering, Oklahoma University, Tulsa, OK, USA.
°Division of Biomedical Informatics, University of California, San Diego, CA, USA.
Shuang Wang, Xiaoqian Jiang and Lucila Ohno-Machado were funded in part by the NIH grants K99LM011392, UH2HL108785, U54HL108460, UL1TR00010003 and the AHRQ grant R01HS019913

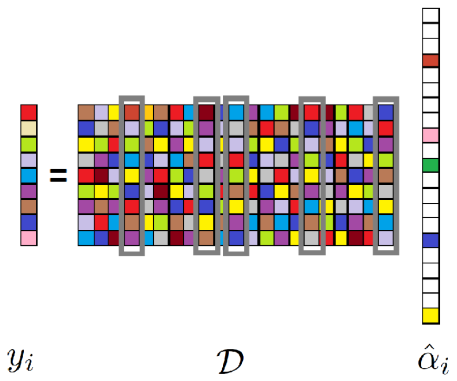


Fig. II.1: Sparse representation of a feature vector y_i with a dictionary $\mathcal{D}$: $\hat{\alpha}_i$ as a sparse vector constructs the feature vector y_i using a few columns (highlighted in gray) of dictionary $\mathcal{D}$.

$\mathbf{X}$ and $\mathbf{Y}$, respectively. We then extract the features from the reference sequence $\mathbf{X} \in \mathbb{Z}^{N\times 2}$ and the test sequence $\mathbf{Y} \in \mathbb{Z}^{M\times 2}$.

Second, we create a dictionary $\mathcal{D}$ containing all the extracted feature vectors of the reference sequence $\mathbf{X}$ to match to the corresponding extracted features of the test sequence $\mathbf{Y}$. The dictionary includes all vectorized one dimensional features as its columns where all of them have been normalized. We then apply sparse coding to each extracted feature of the test sequence. Sparse coding will reconstruct a nucleotide vector at each nucleotide g_i as a linear combination of the reference sequences. Note that the obtained representation coefficients α_i should be sparse, i.e., it should be 0 for most coefficients. The non-zero coefficients of α_i indicate the corresponding nucleotides on the reference sequence (see Fig. II.1).

To select n candidate nucleotides, we simply pick those corresponding to n largest coefficients in the sparse coefficient vector. We store the locations of these candidate nucleotides in a length-n vector $\mathcal{L}_i$ and a probability vector ρ_i as a length-n vector storing the corresponding probabilities of the sparse coefficient vector. Each coefficient in the probability vector ρ_i serves as a prior probability of matching the nucleotide at i to a nucleotide of reference sequence. This probability vector takes only local characteristics into account but ignores neighborhood characteristics of the matches.

Finally, we expect that nearby nucleotides in the test sequence should also match the nucleotides that are close to each other in the reference sequence. To incorporate these neighborhood characteristics, we model the problem by a factor graph and apply the Belief Propagation (BP) algorithm to identify the best matches. We consider a one dimensional factor graph as follows: for each nucleotide in the test sequence, one variable node was assigned and each variable node was connected to its two neighbors by a factor node. Also, we consider one extra factor node for each variable node to impose the restriction of prior probabilities for the candidate nucleotides (see Fig. III.1).

BP [18] is an efficient inference method used on graphical models such as factor graphs. It was performed by passing

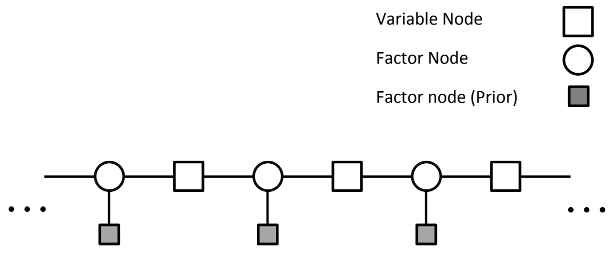


Fig. II.2: Nucleotides model: One dimensional factor graph used in Belief Propagation.

messages through the factor graph of our problem. We apply BP on the factor graph of test sequence with n candidate nucleotides as prior knowledge. BP updates the probability of candidate nucleotides based on the probabilities of their neighbors.

Define $N(i)$ and $N(a)$ as two sets of neighbors of a variable node i and a factor node a, respectively, and denote $m_{i\to a}$ and $m_{a\to i}$ as the forward and backward messages from node i to node a, respectively. Message updates for $m_{i\to a}$ and $m_{a\to i}$ are based on the messages received by the incoming messages towards nodes i and a, respectively. More precisely, in our factor graph, the message update rules are given by [18]

$$m_{i\to a}(x_i) = \prod_{b\in N(i)\backslash a} m_{b\to i}(x_i), \tag{II.1}$$

$$m_{a\to i}(x_i) = \sum_{x_a\backslash x_i} f(x_a) m_{j\to a}(x_j), \tag{II.2}$$

where $N(a)\backslash i$ means all neighbors of node a excluding node i; the factor node x_a is located between variable nodes x_i and x_j. Also, we model $f(x_a)$ as follows:

$$f(x_a) = \tilde{f}(x_i, x_j) = e^{-\frac{||\mathcal{L}_i - \mathcal{L}_j||_2}{\sigma^2}} \tag{II.3}$$

where σ^2 is a parameter to control the relative strength of the geometric constraint imposed by a neighboring node. If we increase the value of σ^2, the belief of each variable node will have less effect on its neighbors.

To align the test nucleotide sequence, we select the nucleotide candidate with highest probability and then calculate the displacement β between the current nucleotide and the selected candidate nucleotide. Therefore, for each nucleotide in the test sequence, we have $\mathcal{Z}_i$, β_i and ρ_i for each nucleotide g_i which are the most probable match nucleotide, the most probable displacement and the probability of the current match, respectively.

The main procedure for our proposed alignment method is summarized in Algorithm 1.

A. Implementation details:

- $\mathbf{X}$ = $ConvertData(\mathcal{X})$ maps the input string of the nucleotide sequence to a sequence of integer values for the further processing. The mapping function of four nucleotides $\{A, C, G, T\}$ can be defined in a two-dimension space where $A = (1, 1)$, $C = (1, -1)$, $G =$

Algorithm 1 Proposed nucleotide sequence alignment algorithm - estimate the location of the input sequence

```
Inputs: a reference sequence $\mathcal{X} \in \mathbb{R}^M$, a test sequence $\mathcal{Y} \in \mathbb{R}^N$, a threshold $\theta$, the number of the candidate points $n$
Convert a string sequence to numeric sequence:
    • $\mathbf{X} = ConvertData(\mathcal{X})$
    • $\mathbf{Y} = ConvertData(\mathcal{Y})$
Extract feature and construct dictionary:
    • $\hat{Y} = ExtractFeature(\mathbf{Y})$
    • $\hat{X} = ExtractFeature(\mathbf{X})$
    • $\mathcal{D} = MakeDic(\hat{X})$
Find the initial estimate of the match location: For each vector $y_i \in \hat{Y}$ perform:
    • $\alpha_i = FindSparseVector(\mathcal{D}, y_i)$
    • $[\mathcal{L}_i, \hat{\rho}_i] = FindTopScoreMatch(n, \alpha_i)$
Refine the candidate match location:
    • $\rho = BP(\mathcal{L}, \hat{\rho})$
Find the correspond nucleotides:
    • $[\mathcal{Z}, \beta] = Warp(\hat{X}, \rho, \mathcal{L})$
Output: the estimated version of aligned sequence $\mathcal{Z}$
```

$(-1, 1)$, and $T = (-1, -1)$, respectively. Therefore, $\{A, C, G, T\} \mapsto \{(1, 1), (1, -1), (-1, 1), (-1, -1)\}$.

- $\hat{Y} = ExtractFeature(\mathbf{Y})$ presents a vector extractor algorithm using $\mathbf{Y}$ as a source sequence, where the result is a two dimensional matrix containing the vectorized one dimensional sequences. To this end, we consider a vector of size $S = 2 \times (2a + 1)$ containing neighboring nucleotides on two sides of a nucleotide, where a is a positive integer and the first "2" is corresponding to the dimension of the mapping space). For each nucleotide g_i in the test sequence $\mathbf{Y}$, we vectorized a sequence centered around the nucleotide g_i to a feature vector $y_i \in \mathbb{Z}^{S \times 1}$. A two dimensional test feature data $\hat{Y} \in \mathbb{Z}^{N \times S}$ is then constructed from y_i as follows:

$$\hat{Y} = \{y_i \mid 1 \leq i \leq M\}. \tag{II.4}$$

  Note that $\hat{X}$ is created in the same manner as $\hat{Y}$ but from the reference sequence $\mathbf{X}$ instead.
- $\mathcal{D} = MakeDic(\hat{X})$ creates a dictionary $\mathcal{D}$ using the vectors of $\hat{X}$. Later, the dictionary $\mathcal{D}$ is used to match the extracted features of the source sequence to corresponding the extracted features of the reference sequence. We can write $\mathcal{D} = [x_1\ x_2 ...\ x_N]$ where x_i is a feature vector of $\hat{X}$. Note that we normalize dictionary $\mathcal{D}$ to guarantee the norm of each feature vector to be 1.
- $\alpha_i = FindSparseVector(\mathcal{D}, y_i)$ finds the candidate match nucleotide using the sparse coding algorithm, where α_i is a sparse vector. Mathematically, we try to solve the following sparse coding problem to find the most sparse coefficient vector α_i (see Fig. II.1) such that

$$y_i = \mathcal{D}\hat{\alpha}_i. \tag{II.5}$$

  Although there are several methods to solve (II.5) [19]–[21], in our work, we employ Subspace Pursuit (SP) [20] because of its computational efficiency.
- $[\mathcal{L}_i, \hat{\rho}_i] = FindTopScoreMatch(n, \alpha_i)$ picks up the n largest coefficients of α_i as n candidates. $\mathcal{L}_i$ is a $n \times 1$ vector that stores the locations of these candidate nucleotide and $\hat{\rho}_i$ is the length-n vector that stores the corresponding probabilities of $\mathcal{L}_i$. Each coefficient in $\hat{\rho}_i$ serves as a prior probability of matching the source sequence at i to a sub-sequence centered around the nucleotide g_i. After finding the candidates and their initial probabilities, we concatenate the result of each nucleotide and construct following matrices:

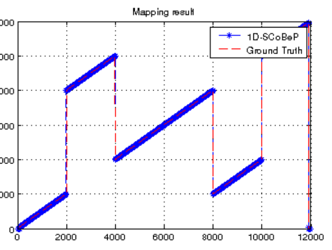
$$\mathcal{L} = [\mathcal{L}_1\ \mathcal{L}_2 ...\ \mathcal{L}_N]\,, \hat{\rho} = [\hat{\rho}_1\ \hat{\rho}_2 ...\ \hat{\rho}_N]\,, \tag{II.6}$$

  which we will use to apply belief propagation at the next step.
- $\rho = BP(\mathcal{L}, \hat{\rho})$ models the problem by a factor graph (see Fig. III.1) and applies belief propagation [18] to update probability ρ. The updated probability ρ can be used to align the reference sequence onto the test sequence. In our case, we assign a variable node for each nucleotide on the source sequence and connect each pair of neighboring nucleotide with a factor node. Also, we introduce one extra factor node to take care of the prior knowledge obtained in the sparse coding step for each nucleotide of the source sequence (for more details, see [17]).
- $[\mathcal{Z}, \beta] = Warp(\mathbf{X}, \rho, \mathcal{L}, \theta)$ returns the aligned sequence $\mathcal{Z}$ and a displacement vector β which contains the movement of each nucleotide in the reference sequence. Note that in step $BP(\cdot)$, we calculated the refined probability of each candidate nucleotide match.

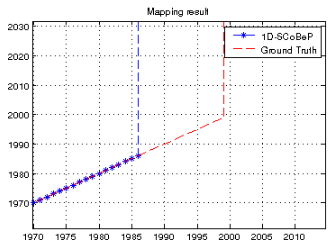
III. Experimental Results

In this section, we present our experiments to evaluate the proposed method for aligning the nucleotide sequences and compare it with SOAPaligner [1] and BWA [2]. We considered the problem of aligning a sequence of human nucleotides from the National Center for Biotechnology Information (NCBI) [22]. To evaluate the performance of our approach, we conducted two sets of tests on the nucleotide sequences. In the first set, we selected twenty short sub-sequences of human genomes and then used SOAPaligner, BWA and the proposed method to find the location of selected sub-sequence nucleotide in the human chromosome. All of three algorithms successfully passed this test. We created five shuffled sub-sequences of the reference sequence. We chose six indexes j_s,j_1, j_2, j_3, j_4 and j_e where $j_s < j_1 < j_2 < j_3 < j_4 < j_e$ and we considered j_s and j_e as the index of the first and the last nucleotide in the original sequence, respectively. Then we shuffled the reference sequence by swapping the sub-sequence $[g_{j_1} ... g_{j_2 - 1}]$ with $[g_{j_3} ... g_{j_4 - 1}]$. Therefore, the test nucleotide sequence can be presented as

$$\{g_{j_s}, ..., g_{j_1 - 1}, g_{j_3} ... g_{j_4 - 1}, g_{j_2} ... g_{j_3 - 1}, g_{j_1} ... g_{j_2 - 1}, g_{j_4} ... g_{j_e}\} \tag{III.1}$$



(a)



(b)

Fig. III.1: The result of proposed method for non-collinear nucleotide sequence alignment. a) alignment result of proposed method in compare to the ground truth. b) zoomed plot to show the gap between the proposed method and ground truth on the jump point.

where g_j is the nucleotide for index j. In this set of the test sequences, SOAPaligner and BWA could not find proper alignments. In contrast, Fig. III.1 shows the proposed method alignment output agrees with the ground truth by a gap of 10 to 15 nucleotides. Throughout the experiments, the following parameters were used: the number of candidate points n is set to be 3, $k = 3$ and $a = 20$.

The computational complexity of proposed is mainly determined by the following three steps: 1) extracting subsequence nucleotides as features and constructing the dictionary, 2) finding candidate nucleotides via sparse coding, and 3) applying BP. Assume the size of the test and reference sequences are N and M nucleotides, respectively. The required time of feature extraction will be $O\big(h(M + N)\big)$, where h is the size of the vector of extracted features for each nucleotide. As for dictionary construction, the only time needed is for the normalization of each column, which requires $O(hM)$ amount of time. Thus the total time complexity of the first step is $O\big(h(M + N)\big)$. In the second step of the SCoBeP, the time complexity of SP is $O\big(\log(f)hM\big)$ [23], where f is the number of iterations for finding the sparse vector. Since we have to repeat the process of finding candidate points for all N feature vectors, the time complexity of finding candidate points by SP is $O\big(\log(f)hMN\big)$. In the third step, the time complexity of BP in our factor graph is $O(vn^2M)$, where v is the number iterations before converging. Consequently, if the SCoBeP uses SP, its time complexity will be $O\Big(vn^2M + h(M + N) + \log(f)hMN\Big)$.

IV. Conclusion

In conclusion, we have proposed a nucleotide sequence alignment method based on the sparse coding and belief propagation. Our technique performs alignment by first running sparse coding over an over-complete dictionary constructed from the nucleotides of a reference sequence to gather possible candidate nucleotides. Belief propagation is then applied to eliminate bad candidates and to select the best matches. The experimental results illustrate that our proposed algorithm shows comparable performance to those of SOAPaligner [1] and BWA [2]. We believe that the proposed method can be used for various collinear and non-collinear nucleotide sequence alignment applications.

References

[1] R. Li, C. Yu, Y. Li, T. Lam, S. Yiu, K. Kristiansen, and J. Wang, "Soap2: an improved ultrafast tool for short read alignment," *Bioinformatics*, vol. 25, no. 15, pp. 1966–1967, 2009.

[2] H. Li and R. Durbin, "Fast and accurate short read alignment with burrows–wheeler transform," *Bioinformatics*, vol. 25, no. 14, pp. 1754–1760, 2009.

[3] T. Hampton, "Cancer genome atlas," *JAMA: The Journal of the American Medical Association*, vol. 296, no. 16, pp. 1958–1958, 2006.

[4] N. Siva, "1000 genomes project," *Nature biotechnology*, vol. 26, no. 3, pp. 256–256, 2008.

[5] S. Batzoglou, "The many faces of sequence alignment," *Briefings in bioinformatics*, vol. 6, no. 1, pp. 6–22, 2005.

[6] C. Dewey and L. Pachter, "Evolution at the nucleotide level: the problem of multiple whole-genome alignment," *Human Molecular Genetics*, vol. 15, no. suppl 1, pp. R51–R56, 2006.

[7] M. Blanchette, W. Kent, C. Riemer, L. Elnitski, A. Smit, K. Roskin, R. Baertsch, K. Rosenbloom, H. Clawson, E. Green, *et al.*, "Aligning multiple genomic sequences with the threaded blockset aligner," *Genome research*, vol. 14, no. 4, pp. 708–715, 2004.

[8] A. Darling, B. Mau, F. Blattner, and N. Perna, "Mauve: multiple alignment of conserved genomic sequence with rearrangements," *Genome research*, vol. 14, no. 7, pp. 1394–1403, 2004.

[9] I. Dubchak, A. Poliakov, A. Kislyuk, and M. Brudno, "Multiple whole-genome alignments without a reference organism," *Genome research*, vol. 19, no. 4, pp. 682–689, 2009.

[10] M. Höhl, S. Kurtz, and E. Ohlebusch, "Efficient multiple genome alignment," *Bioinformatics*, vol. 18, no. suppl 1, pp. S312–S320, 2002.

[11] B. Paten, J. Herrero, K. Beal, S. Fitzgerald, and E. Birney, "Enredo and pecan: genome-wide mammalian consistency-based multiple alignment with paralogs," *Genome research*, vol. 18, no. 11, pp. 1814–1828, 2008.

[12] R. Bradley, A. Roberts, M. Smoot, S. Juvekar, J. Do, C. Dewey, I. Holmes, and L. Pachter, "Fast statistical alignment," *PLoS computational biology*, vol. 5, no. 5, p. e1000392, 2009.

[13] R. Edgar, "Muscle: multiple sequence alignment with high accuracy and high throughput," *Nucleic acids research*, vol. 32, no. 5, pp. 1792–1797, 2004.

[14] J. Thompson, D. Higgins, and T. Gibson, "Clustal w: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice," *Nucleic acids research*, vol. 22, no. 22, pp. 4673–4680, 1994.

[15] J. Bartoš, Č. Vlček, F. Choulet, M. Džunková, K. Cviková, J. Šafář, H. Šimková, J. Pačes, H. Strnad, P. Sourdille, *et al.*, "Intraspecific sequence comparisons reveal similar rates of non-collinear gene insertion in the b and d genomes of bread wheat," *BMC Plant Biology*, vol. 12, no. 1, p. 155, 2012.

[16] R. Song and J. Messing, "Gene expression of a gene family in maize based on noncollinear haplotypes," *Proceedings of the National Academy of Sciences*, vol. 100, no. 15, pp. 9055–9060, 2003.

[17] N. Barzigar, A. Roozgard, S. Cheng, and P. Verma, "Scobep: Dense image registration using sparse coding and belief propagation," *Journal of Visual Communication and Image Representation*, 2012.

[18] F. Kschischang, B. Frey, and H. Loeliger, "Factor graphs and the sum-product algorithm," *Information Theory, IEEE Transactions on*, vol. 47, no. 2, pp. 498–519, 2001.

[19] A. Yang, S. Sastry, A. Ganesh, and Y. Ma, "Fast 1-minimization algorithms and an application in robust face recognition: A review," in *Image Processing (ICIP), 2010 17th IEEE International Conference on*. IEEE, 2010, pp. 1849–1852.

[20] W. Dai and O. Milenkovic, "Subspace pursuit for compressive sensing signal reconstruction," *Information Theory, IEEE Transactions on*, vol. 55, no. 5, pp. 2230–2249, 2009.

[21] R. Maleh, A. Gilbert, and M. Strauss, "Sparse gradient image reconstruction done faster," in *Image Processing, 2007. ICIP 2007. IEEE International Conference on*, vol. 2. IEEE, 2007, pp. II–77.

[22] N. C. Institute, "The national center for biotechnology information," http://www.ncbi.nlm.nih.gov/genome?db=genome, January 2013. [Online]. Available: http://www.ncbi.nlm.nih.gov/genome?db=genome

[23] W. Dai and O. Milenkovic, "Subspace pursuit for compressive sensing: Closing the gap between performance and complexity," DTIC Document, Tech. Rep., 2008.