picture(0,0)(-35,0)(1,0)30 (0,35)(0,-1)30 picture

picture(0,0)(35,0)(-1,0)30 (0,35)(0,-1)30 picture

Bioinformatics
doi.10.1093/bioinformatics/xxxxxx
Advance Access Publication Date: Day Month Year
Sequence Analysis

OXFORD

Sequence Analysis:

Burrows Wheeler Transform for Alignment in Protein and Nucleotide Space.

Eliezer Zimble [1]*.

[1]Department of Computer Science, Columbia University, New York, New York 10027 , USA.

*ez2313@columbia.edu

Abstract

Motivation: The Burrows Wheeler transformation (BWT) offers efficient indexing through use of cyclic shifts. BWT has been used to allow for efficient alignment of DNA reads by transforming the nucleotide reference; however, for the large datasets involved even such an approach is time consuming. Performing alignment in the protein space allows for faster alignment and a greater emphasis on functional profiling; however, this inherently comes at the cost of a loss of information Our approach aims to use the BWT to perform the indexing and alignment in parallel in the protein and nucleotide space to allow for fast and sensitive alignment while minimizing the loss of information.
Results: Our work has demonstrated that using BWT to first align in the protein space and then switch to the nucleotide space is more efficient and faster when compared to dynamic programming global alignment when tested on randomly generated reads and references of different sizes and quantity. Further work is needed to test the model with respect to sensitivity in alignment with remote references as would be expected from the focus on functional profiling in the protein space. Likewise, further work is needed to incorporate an interface for common input and output formats.
Availability: Please see the repository at https://github.com/ez2313/CBMF4761-BWT-Final-Project for source code and further information.
Contact: ez2313@columbia.edu

1 Introduction

Alignment of DNA reads against a reference genome is a widespread procedure used by many studies involving sequencing. The computational costs and utility of different alignment methods can therefore have a broad practical effect.

The Burrows Wheeler transformation utilizes cyclic shifts of texts to achieve efficient indexing with lower memory costs and was initially developed for the purpose of efficient compression (Burrows and Wheeler, 1994). In the context of DNA sequencing, the transformation has been applied to aid alignment of reads against a reference genome. The transformation has been further extended to alignment in the protein space which allows for greater computational speed and a focus on functional profiling (Westbrook 2017). These benefits; however, come at a cost of a loss of information. Performance of alignment fully in the protein space may miss out on information preceding the start of the open reading frame. Additionally, it is possible for multiple differing nucleotide sequences to share the same protein sequence, obscuring the underlying taxonomic differences.

My work aims to preserve the benefits of the speed and functional sensitivity of alignment in the protein space, while also reducing the loss of information by performing the alignment in both the protein and nucleotide spaces in parallel.

2 Methods

The project focused on developing an aligner tool written in Python. The aligner tool uses the Burrows Wheeler transform in the form of an FM index (Ferragina P and Manzini G. 2000), an approach that uses suffix arrays to perform the transformation in a manner that minimizes memory cost. The project builds off of an open source FM index implementation (Langmead), and extends its use to alignment in protein space by converting given nucleotide reads into protein sequences for all 6 open reading frames (ORFs). The aligner tool attempts to perform an exact match for each

© The Author 2015. Published by Oxford University Press. All rights reserved. For permissions, please e-mail: journals.permissions@oup.com

picture(0,0)(-35,0)(1,0)30 (0,-35)(0,1)30 picture

picture(0,0)(35,0)(-1,0)30 (0,-35)(0,1)30 picture

picture(0,0)(-35,0)(1,0)30 (0,35)(0,-1)30 picture

picture(0,0)(35,0)(-1,0)30 (0,35)(0,-1)30 picture

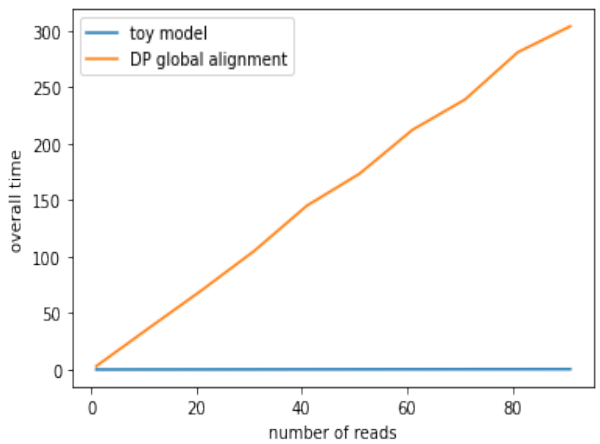

Fig. 1. Runtime vs. Number of reads

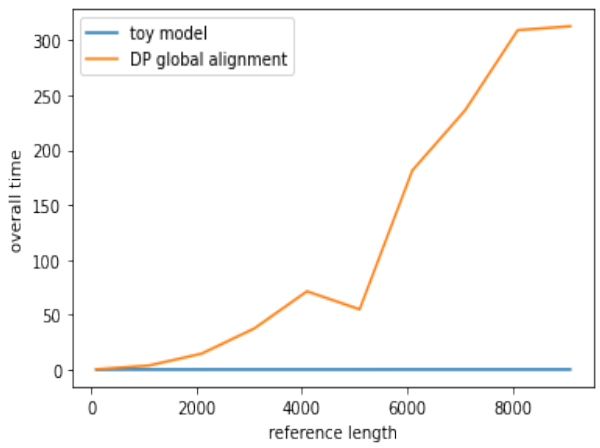

Fig. 2. Runtime vs. Reference length

ORF and determines the best ORF. If all information is not exhausted, the aligner tool then continues gapped alignment in the nucleotide space. For each read, the aligner tool returns an overall score based on the number of matches in the nucleotide space and the location of all matches in the protein space.

2.1 Results

The aligner tool was tested on randomly generated reads and references to compare its speed performance with standard dynamic programming global alignment. For both varying number of reads (fig. 1) and varying reference length (fig. 2), the aligner tool outperformed standard dynamic programming. The results indicate that the aligner tool offers improved speed.

3 Conclusion

The aligner tool employs the Burrows Wheeler transform to use efficient indexing for performance of alignment first in the protein space, and then transitions into alignment in the nucleotide space. In this way, the aligner aims for the benefits of alignment in the protein space while minimizing the loss of information involved. The aligner tool outperforms dynamic programming global alignment in terms of its speed performance.

The project faces several limitations. At this stage of the project, the nucleotide alignment primarily addresses "leftover" nucleotides that were not translated into the protein sequence: depending on the ORF and length of the read, there will be a remainder modulo 3. The nucleotide alignment also allows for a safety in cases where no exact match is found through the FM index. Further work is needed to allow the transition between protein and nucleotide alignment to occur at adaptable strategic points specific to the sequence.

Further testing is required to check the aligner tool's sensitivity. Ideally, a study would be performed that tests whether the aligner can correctly match reads from bacteria to a reference database in which the bacteria sampled are removed. The alignment would then be based off of functional matching of remote species and not the exact sequence match, and would indicate whether the aligner achieves the desired sensitivity to functional profiling. Future work would hope to establish the aligner tool as able to detect remote matches through its reliance on alignment in the protein space reflecting preservation of certain functions across distant species.

The project also needs to develop an interface that allows for input and output to be provided in standard formats for sequencing (such as FASTA and SAM), and to allow for direct interaction with outside services (such as UNIPROT). Ideally the project would merge into a plugin for a platform such as PALADIN (Westbrook 2017) to enable such an interface.

Use of the BWT for alignment in parallel in protein and nucleotide spaces offers improved speed of alignment, and the potential for higher sensitivity as well. Future work is needed to test the aligner tool's capabilities and develop a more robust interface.

References

[1]Burrows, M., Wheeler, D.J, (1994) A Block-sorting Lossless Data Compression Algorithm, *SRC Research Report*.

[2]Bunchfunk,B. et al. (2015) Fast and sensitive protein alignment using DIAMOND. *Nature Methods*, 12, 59–60.

[3]Ferragina P, Manzini G. 2000. Opportunistic data structures with applications. *In Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, 12–14 November 2000, Redondo Beach, California, pp. 390–98. Los Alamitos, CA: IEEE Comput. Soc.

[4]Langmead, Benjamin. Notebook:FM Index, https://langmead-lab.org/teaching-materials/.

[5]Saier Jr, Milton H. (2019) "Understanding the Genetic Code", *American Society for Microbiology, Journal of Bacteriology* Volume 201, Issue 15.

[6]Westbrook et al (2017) "PALADIN: protein alignment for functional profiling whole metagenome shotgun data", *Bioinformatics*, 33(10), 2017, 1473–1478

picture(0,0)(-35,0)(1,0)30 (0,-35)(0,1)30 picture

picture(0,0)(35,0)(-1,0)30 (0,-35)(0,1)30 picture