Towards speed up of Whole Genome Alignment using Distributed Suffix Tree

Julio Cesar Garcia Vizcaino*[1], and Antonio Espinosa[2]

[1]Computer Architecture & Operating Systems Department (CAOS), Universitat Autònoma de Barcelona,Bellaterra (Barcelona), Spain.
[2]Computer Architecture & Operating Systems Department (CAOS), Universitat Autònoma de Barcelona,Bellaterra (Barcelona), Spain.

Email: Julio Cesar Garcia Vizcaino*- juliocesar.garcia@e-campus.uab.cat; Antonio Espinosa - antonio.espinosa@caos.uab.es;

*Corresponding author

Abstract

Background: Text for this section of the abstract.

Results: Text for this section of the abstract ...

Conclusions: Text for this section of the abstract ...

Background

Today, we know that species are described by DNA, a complex molecule comprised of many smaller molecules called nucleotides. The data describing a single specie,commonly called a genome, can be millions or billions of nucleotides long.

One of the most basic computational tasks that we perform on genomic data is identifying the evolutionary relationships between DNA from two or more species. On a smaller scale, we wish to identify which individual nucleotides are unique to species, and which nucleotides share ancestry. On a larger scale, we look to **find entire subsequences that are a common between them**.

Currently the availability of a huge amount of Bioinformatics data (often in the public domain), and on the other hand the need for new and efficient methods and algorithms capable of compute the information contained in the data requires the use of HPC to manipulate it. As a matter of fact, the emphasis of research in Bioinformatics and HPC is shifting from the development of efficient data storing and handling methods, to the one of methods able to extract useful information from data.

Consequently, the computational demands needed to explore and analyze the data contained in the genome databases is quickly becoming a great concern. To meet these demands, we must use high performance computing, such as parallel computers and distributed networks of workstations.

This paper focuses in the whole genome alignment problem. A whole genome alignment is the process of identifying a mapping from each position in query genome to its corresponding position in the reference genome.

We now describe some notation, give a more detailed introduction to whole genome alignment, and describe the mathematical framework on which we base most of techniques used to perform whole genome alignment. First we define some basic notation. We use R (Reference) and Q (Query) to denote sequences. For sequence R, we refer to position i as R_i and let the first position be R_0.

Sequence alignment is the primary tool for finding evolutionary relationships between DNA sequences. A DNA sequence is a string over four symbols: A, T , C, and G. These symbols represent the nucleotides adenine, thymine, cytosine, and guanine, respectively. As time passes, DNA sequences incur mutations from a variety of physical processes. Thus, DNA sequences from individuals of a specie contain many differences. When aligning DNA sequences from different species, large scale changes, such as long insertions and deletions, duplications, reversals and translocations, are common, see Figure 1. The goal of sequence alignment is to infer which changes occurred with a mathematical model that abstracts the physical mutation processes.

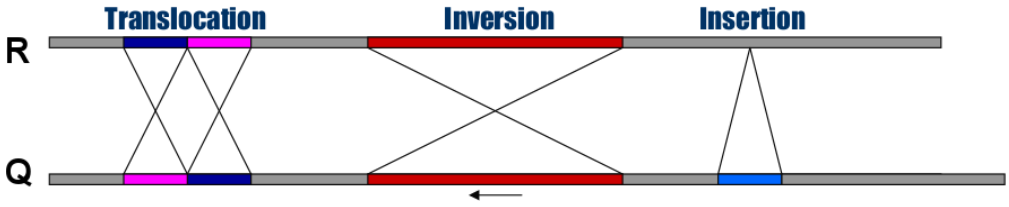

Figure 1: Operations in whole genome alignment

Given two sequences, we can interpret an alignment. We stack the aligned sequences on top of one another

and look at the content of the every position, we can see the following schema:

Original sequence:	AGGCCTC
Mutations:	AGGACTC
Insertions:	AGGGCCTC
Deletions:	AGG.CTC

Finding all these changes is a time-intensive operation in whole genome alignment. Moreover, the size of the genome can be an issue when there is not enough memory to store the reference and query genome. The standard algorithms for sequence alignment rely on either dynamic programming or hashing techniques. Naïve versions of dynamic programming use $O(n^2)$ space and time (where n is the length of the shorter of the two sequences being compared), which makes computation simply unfeasible for sequences of size ≥ 4 Mb. Hashing techniques operate faster on average, but they involve a 'match and extend' strategy, where the 'extend' part also takes $O(n^2)$ time. For dynamic programming, it is possible to reduce the required space to $O(n)$ by taking more time; this solves the memory problem but still leaves one with an unacceptably slow algorithm. Faster algorithms can be developed for specialized purposes. More complex dynamic programming methods can be used for alignment when the alignment error is expected to be low. For example, one can align two similar sequences with at most E differences (or errors) in time proportional to E times the length of the longer sequence.

0.1 MUMmer

MUMmer, a widely-used bioinformatic application, is the tool that we use to

Results and Discussion

Results sub-heading

This is a sub-sub-heading

Sub-sub-sub-headings are made with the

subsubsection command.

pb at end of lines ensures correct paragraph spacing.

Text for this sub-sub-section ...

Another sub-sub-sub-heading

Text for this sub-sub-section ...

Another results sub-heading

Text for this sub-section ...

Yet another results sub-heading

Text for this sub-section. More results ...

Conclusions

Text for this section ...

Methods

Methods sub-heading for this section

Text for this sub-section ...

Another methods sub-heading for this section

Text for this sub-section ...

Yet another sub-heading for this section

Text for this sub-section ...

Authors contributions

Text for this section ...

Acknowledgements

Text for this section ...

Figures

Figure 1 - Sample figure title

A short description of the figure content should go here.

Figure 2 - Sample figure title

Figure legend text.

Tables

Table 1 - Sample table title

Here is an example of a *small* table in LaTeX using `\tabular{...}`. This is where the description of the table should go.

My Table		
A1	B2	C3
A2	...	..
A3	..	.

Table 2 - Sample table title

Large tables are attached as separate files but should still be described here.

Additional Files

Additional file 1 — Sample additional file title

Additional file descriptions text (including details of how to view the file, if it is in a non-standard format or the file extension). This might refer to a multi-page table or a figure.

Additional file 2 — Sample additional file title

Additional file descriptions text.