Species Identification through DNA String Analysis: Summary

Mark Vorster
Department of Computer Science
Rhodes University
Grahamstown, South Africa 6139
Email: g07v3343@campus.ru.ac.za

***Abstract*—This projects aim is to understand how to best to utilise assumptions made by the Rhodes University Department of Biochemistry, Microbiology & Biotechnology in their research and apply string matching techniques to bioinformatic sequence analysis in order to aid the bioinformaticians in their research. In particular the objective is to create a tool for their specific problem that is able to process the large datasets in a timely manner.**

I. Introduction

The Rhodes University Department of Biochemistry, Microbiology and Biotechnology has, in their research involving bacterial DNA sequences, found need to identify and group the sequences into distinct species. However, combining the fact that the datasets are very large — upward of 10000 samples per set — and the lack of existing tools to solve the problem, they have found it takes an inordinate amount of time. Based on assumptions that they are able to make given, their specific problem, an approximate string matching algorithm can be applied to speed up this process and aid them in their research. By implementing this algorithm we will show that significant speedup can indeed be attained.

There are two specific areas that the approximate string matching algorithm can take into account to solve the problem more efficiently. Firstly, as similarities that the bioinformaticians are searching for require similarity within a small percentage difference, branches can be pruned from the search tree once the difference exceeds the threshold. Secondly, as the sequences are assumed to begin at the same location, and thus has no need for a global alignment, this reduces the complexity saving processing time.

After examining the broad area of bioinformatics focusing on framing the problem, this paper looks at how progression in the fundamentals of genetics along with the rise of computers began bioinformatics. It then focuses more specifically on sequence analysis including mentioning the problems of sequence alignment and phylogenetic analysis, then continuing with a discussion of the approximate string matching algorithm applied.

II. Background

In the last 50 years the field of genetics, the study of biological hereditary through genes, has been able to take advantage of the ever increasing power of computer processing, this is most evident in the emergence of bioinformatics, a field of study focused on the applications of computer science to biology [1][1] [8]. While only referred to as 'bioinformatics'

[1]The Biomedical Information Science and Technology Initiative's Definition Committee's definition of bioinformatics:

> 'Research, development, or application of computational tools and approaches for expanding the use of biological, medical, behavioural or health data, including those to acquire, store, organize, archive, analyse, or visualise such data.' [8]

in the 1990s, work from many years before can be grouped into this field. This is a result of the understanding of the fundamentals of genetics as quantitative units as discovered by Nobel prize winners Watson, Crick, Wilkins and Franklin in 1953 [6] [7, p. 163]. Before their discovery living organisms were classified through qualitative observation, animals that looked the same were grouped into the same species, or more generally the grouping of all animals mammary glads as mammals. It was known that traits were passed from parents to children throughout living organisms but the process was unknown. The study of cells was made possible by the development of the microscope allowing biologists to move beyond the reach of the naked eye [9]. There they found that cells seemed to contain complete instruction and mechanisms to completely copy itself. Further technological advances in the 19th century in genetics and cellular biology and 20th century advances in molecular biology as explained by Xu et al. [16] have made it possible to observe the fundamental units of genetics, RNA and DNA. These DNA macromolecules form a ladder-like helix from two strands, the runs of which are formed by pairs of a nitrogenous bases either adenine and thymine or guanine with cytosine. The ability to represent these digitally, each base as a single letter — A, T, G or C — is an important feature for quantifying the hereditary information, as it allows the use of computers to aid in the analysis of the data [3] [14, p. 18].

A. Sequence Analysis

The main reason for DNA sequence analysis is that often knowledge can be inferred about newly sequenced samples through knowledge of the functions of other sequences, as for example homologous sequences are expected to be similar [15] [14, p. 77]. The analytical study of DNA sequences falls into 5 general types:

- Knowledge-based single sequence analysis for sequence characteristics.
- Pairwise sequence comparison and sequence-based searching.
- Multiple sequence alignment.
- Sequence motif discovery in multiple alignments.
- Phylogenetic inference.

[7, pp. 159,160]

Sequence alignment has been recognised as the most successful applications of Computer Science in Bioinformatics [4] it is the process of finding a best fit match between sequences by finding their most similar locations. The reason sequences may need alignment is that in many situations, and due to the workings of the genes themselves, similar sequences can have bases inserted, deleted or changed, the alignment ensures that the related parts can be examined with greater value [13, p. 771] [14, p. 55] [12]. While the sequence analysis of this project falls under phylogenetic analysis and not the alignment of sequences, it is still central to the nature of genes and therefore must be taken into account in many aspects of sequence analysis. Phylogenetics is the study of homogeneity of organisms including determining their evolutionary relationships, for example their species, that deals with numerical data such as DNA sequences. Another application of phylogenetics, apart from that of grouping sequences by species, is building phylogenetic trees based on similarities between organisms [10].

B. String Matching

Pattern matching is found in many areas of computer science particularly text analysis, approximate matching is used when strings are expected to have minor differences as apposed to exact matches, as is the case with our sequence data. Baase [2, pp. 504 - 508] discusses an algorithm for building a difference table between two strings taking into account insertions and deletions, similar to how they may occur in DNA sequences. The approach uses a dynamic programing technique to speed up the process of building a two dimensional matrix where each point D[i][j] has the minimum number of differences between two string segments P and T, each ending at p_i and t_j respectively.

The matrix is built up column by column where D[i][j] is

calculated as the minimum of three possible numbers either a 'matchCost' (if $p_i = t_j$) or a 'reviseCost'(if $p_i \neq t_j$), a 'insertCost' and a 'deleteCost' where each is defined as the following:

$$\begin{aligned}
\text{matchCost} &= D[i-1][j-1] && \text{, if } p_i = t_j \text{ or} \\
\text{reviseCost} &= D[i-1][j-1] + 1 && \text{, if } p_i \neq t_j \\
\text{insertCost} &= D[i-1][j] + 1 \\
\text{deleteCost} &= D[i][j-1] + 1
\end{aligned}$$

III. Design and Implementation

Working with the bioinformaticians they have defined some requirements and placed some limitations on the development of the tool. The most important assumption is that the data is preprocessed such that it is pre-aligned globally, this means that the beginning of the sequences are in the same place of the gene. This, along with the fact that the processed alignment is not required as a result of our processing, allows us to take into account branch pruning much faster than was possibly before. The bioinformaticians have requested that the resulting tool be limited to a single thread such that given a PC with multiple cores they can run multiple sample sets concurrently instead of finishing the processing of each sample sequentially but as fast as possible. There are a couple of additional research limitations based on decisions made from their side: firstly, when it is possible working with DNA the sequence should be converted to its related protein sequence as these are shorter and more diverse [11], secondly, while similarity does not imply homogeneity, the bioinformaticians have chosen to correlate these traits due to the lines being blurred when dealing with bacterial DNA [5].

A. Implementation

The algorithm at the core of the sequence analysis of this work is an adaptation of Baase's [2, pp. 504 - 508] approximate string matching algorithm along with a grouping algorithm to place each sequence in its inferred species group. Baase's algorithm uses a dynamic programing approach which will allow branch pruning, but there are additional changes that need to be made to the algorithm to take into account the difference when dealing with sequences as apposed to string patterns. The main difference is the initializations which are simplified due to the fact that firstly we do not need to account for the null string as found in Baase's examples, and can further assume either a insert or delete in both the first row and column of the table, as the sequences are assumed to be globally aligned. This assumption too means that we do not need to compare more bases than occur in the shortest sequence as anything past this point would require an extra delete to be found somewhere else within the shortest sequence and therefore result in a worse match. Along with the initialisations the stop cases need to be examined as now we are dealing with sequences, instead of a pattern and a text, a match being found within the threshold in either the last row or column would indicate a match. The last case that can added is the main branch pruning case which occurs in the situation where every value in the currently evaluated row is greater than the threshold, this can be taken into account as a stop case, as with our dynamic programming approach, at no point later in processing can we get a smaller result. It should be noted that as we are dealing with small percentage difference this case can confirm lack of a match very quickly.

1) Grouping Algorithm: Once all matches between the sequence have been calculated they need to be placed into their inferred species groups. The initial algorithm used for the grouping of the samples was a greedy algorithm that first takes sequence with the most matches within the threshold and grouped all those sequences with it. This however tended to yield less than optimal results as the sequence with the most matches tended to be amongst the shortest sequences in the set. This meant that longer sequences that were grouped with this one often where not themselves matches with each other. So instead of using the heuristic of sorting by the sequence with the most matches it was found that sorting by the length of the sequence worked significantly better. It follows that the resulting groups are no longer displayed with the largest

group first running down to the outliers, but the sequences are grouped in better matching sets.

IV. Results

A. Output

The system's output is a list of the species groups containing the names of all the sequences within each group as requested by the bioinformaticians. The format begins with the longest sequence heading the first row, followed by all sequences that fall within the difference threshold with it and finally a count thereof. Each following line undergoes the same process on the remaining sequences.

B. Performance Results

Testing was done on a Intel Core i7 870 with 4GB of RAM on Ubuntu 11.10. In order to obtain statistically accurate results the program was run multiple times on a range of sample sets of sequences: 25 to 1250 sequences in increments of 25. Four of these tests were run concurrently on separate data files. The program was modified to limit the number of sequences to this number rather than considering the entire file, even when the largest file was used the overhead was still negligible — only a fraction of a second compared to the hours to take to run a sample of that size. In addition, testing was done on the largest data set available, which contained 11764 sequences the results of which were:

```
Sequences: 11764
Comparisons: 69189966
Elapsed Time: 38201.69922 (10hrs 36min)
Comparisons per second: 1811.175089
```

This test was run separate from other processing and as a result only serves as a ballpark estimate of the systems rate of processing. As the bioinformaticians wish to run multiple samples concurrently they might suffer slower performance due to low level memory swaps, therefore four larger tests were conducted concurrently.

Below figure 1 shows that the number of comparisons achieved per second is independent of the scale of the problem, and that the overhead is indeed negligible small [2], thus the majority of the processing time is required by the sequence by sequence comparisons.

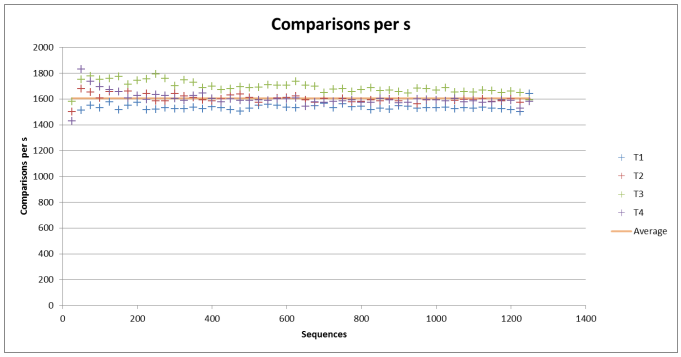

Fig. 1. Comparisons per Second

It can be seen in the next graph, figure 2, that there is a linear relationship between the number of comparisons and processing time. And can therefore conclude that, as expected, the processing time for a sample is order n squared as observed in figure 3. We can also see that these tests managed an average of over 1600 comparisons per second, this is a more reliable result as it has taken into account many different sequences run concurrently.

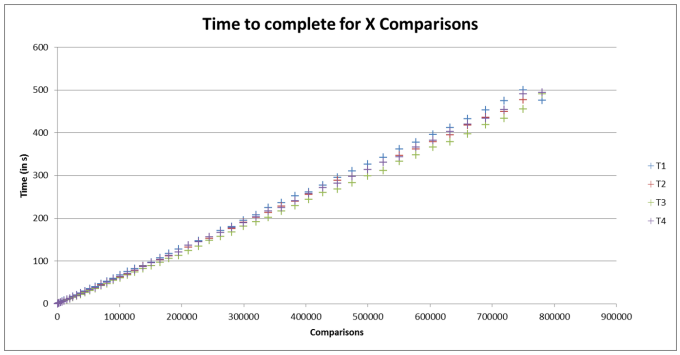

Fig. 2. Overal Time to Complete for X Comparisons

V. Conclusion

Sequence analysis generally begins with a full sequence alignment, when comparing two or a small number of sequences it does not matter that this process is computationally

[2]Further analysis shows that the overhead falls to significantly less than one percent of processing time for samples of decent size.

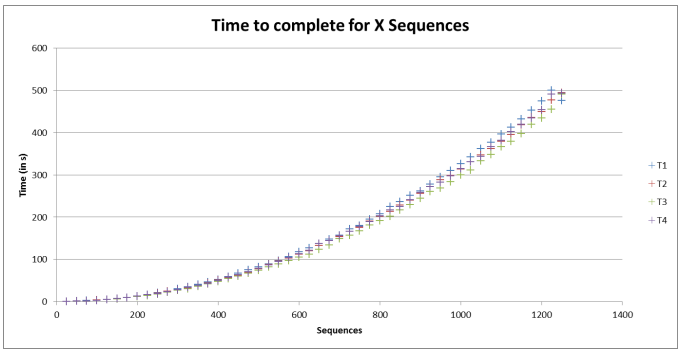

Fig. 3. Overall Time to Execute for X Sequences

expensive. Cutting out the need for the compute alignment fully allows this project to significantly decrease processing time for the large number of sequences that the bioinformaticians are dealing with. The goal of this project is to create a tool to aid bioinformaticians with the task of analysing large sets of bacterial DNA sequences grouping them by species, through an understanding of sequence analysis and string matching theory. The tool needs to be efficient as current techniques take the bioinformaticians upwards of ten days of processing. Results showed a stable rate of 1600 comparisons per second was achieved, which is a huge improvement resulting in similar data sets to those taking ten days to processing completing in approximately eight hours.

REFERENCES

[1] Oxford English Dictionary: Bioinformatics. [online]. Accessed on 2 April 2012. Available from: http://www.oed.com/view/Entry/255935.

[2] BAASE, S., AND VAN GELDER, A. *Computer Algorithms: Introduction to Design and Analysis*. Addison-Wesley, 2000.

[3] BALDI, P., AND BRUNAK, S. *Bioinformatics: The Machine Learning Approach*. Adaptive Computation and Machine Learning. Mit Press, 2001.

[4] BRUDNO, M., DO, C. B., COOPER, G. M., KIM, M. F., DAVYDOV, E., PROGRAM, N. C. S., GREEN, E. D., SIDOW, A., AND BATZOGLOU, S. Lagan and multi-lagan: Efficient tools for large-scale multiple alignment of genomic dna. *Genome Research 13*, 4 (2003), 721–731.

[5] BUCKLEY, M., AND ROBERTS, R. Reconciling microbial systematics and genomics, 2006. American Academy of Microbiology.

[6] CRICK, F., AND WATSON, J. Molecular structure of nucleic acids: A structure for dna. *Nature 171* (April 1953), 737 – 738.

[7] GIBAS, C., AND JAMBECK, P. *Developing Bioinformatics Computer Skills*. O'Reilly Series. O'Reilly, 2001.

[8] HUERTA, M., DOWNING, G., HASELTINE, F., SETO, B., AND LIE, Y. NIH Working Definition of Bioinformatics and Computational Biology. [online], July 2000. Accessed on 2 April 2012. Available from: http://bisti.nih.gov/docs/CompuBioDef.pdf.

[9] JONES, N., AND PEVZNER, P. *An Introduction To Bioinformatics Algorithms*. Computational Molecular Biology. Mit Press, 2004.

[10] KANEHISA, M. *Post-Genome Informatics*. Post-genome Informatics. Oxford University Press, 2000.

[11] KRAWETZ, S., AND WOMBLE, D. *Introduction to Bioinformatics: A Theoretical And Practical Approach*. Humana Press, 2003.

[12] LI, H., AND HOMER, N. A survey of sequence alignment algorithms for next-generation sequencing. *Briefings in Bioinformatics 11*, 5 (2010), 473–483.

[13] NARDONE, J., LEE, D. U., ANSEL, K. M., AND RAO, A. Bioinformatics for the 'bench biologist': how to find regulatory regions in genomic dna. *Nature Immunology 5*, 8 (Aug. 2004), 768–774.

[14] TRAMONTANO, A. *Introduction to Bioinformatics*. Chapman and Hall mathematics series. Chapman & Hall/CRC, 2007.

[15] WON, J.-I., PARK, S., YOON, J.-H., AND KIM, S.-W. An efficient approach for sequence matching in large dna databases. *Journal of Information Science 32*, 1 (2006), 88–104.

[16] XU, D., KELLER, J., POPESCU, M., AND BONDUGULA, R. *Applications of Fuzzy Logic in Bioinformatics*. Series on Advances in Bioinformatics and Computational Biology. Imperial College Press, 2008.