Computational Methods for Large Scale DNA Data Analysis

Xiaohong Qiu[1], Jaliya Ekanayake[1,2], Geoffrey Fox[1,2], Thilina Gunarathne[1,2], Scott Beason[1]
[1]*Pervasive Technology Institute,* [2]*School of Informatics and Computing,*
Indiana University
{xqiu, jekanaya, gcf, tgunarat, smbeason}@indiana.edu

Abstract

We take two large scale data intensive problems from biology. One is a study of EST (Expressed Sequence Tag) Assembly with half a million mRNA sequences. The other one is the analysis of gene sequence data (35339 Alu sequences). These test cases can scale to state of the art problems such as clustering of a million sequences. We look at initial processing (calculation of Smith Waterman dissimilarities and CAP3 assembly), clustering and Multi Dimensional Scaling. We present performance results on multicore clusters and note that currently different technologies are optimized for different steps.

1. Introduction

We abstract many approaches as a mixture of pipelined and parallel (good MPI performance) systems, linked by a pervasive storage system. We believe that much data analysis can be performed in a computing style where data is read from one file system, analyzed by one or more tools and written back to a database or file system. An important feature of the MapReduce style approaches is explicit support for data parallelism which is needed in our applications. .

2 CAP3 Analysis

We have applied three cloud technologies, namely Hadoop, DryadLINQ [1], and CGL-MapReduce [2] to implement a sequence assembly program CAP3 [3] which is dominant part of analysis of mRNA sequences into DNA and performs several major assembly steps such as computation of overlaps, construction of contigs, construction of multiple sequence alignments and generation of consensus sequences, to a given set of gene sequences. The program reads a collection of gene sequences from an input file writing out the results. The “pleasingly parallel” nature of the application makes it extremely easy to implement using the technologies such as Hadoop, CGL-MapReduce and Dryad. In the two MapReduce implementations, we used a “map-only” operation to perform the entire data analysis, where as in DryadLINQ we use a single “Select” query on the set of input data files.

Figs 1 and 2 show comparisons of performance and the

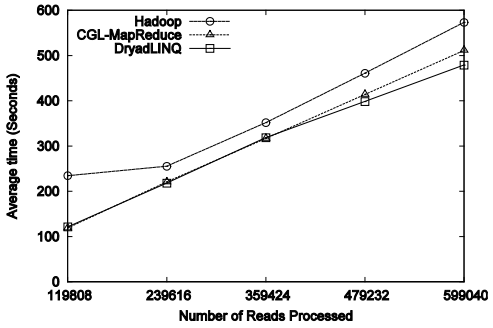

Fig. 1: Performance of different implementations of CAP3

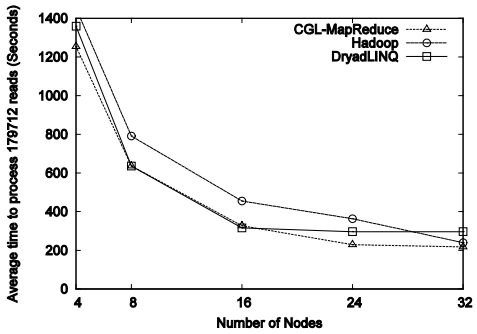

Fig. 2: Scalability of different implementations of CAP3

scalability of the three cloud technologies under CAP3 program. The performance and the scalability graphs shows that all three runtimes work almost equally well for the CAP3 program, and we would expect them to behave in the same way for similar applications with simple parallel topologies. With the support for handling large data sets, the concept of moving computation to data, and the better quality of services provided by the cloud technologies such as Hadoop, DryadLINQ, and CGL-MapReduce make them favorable choice of technologies to solve such problems.

3. Alu Sequencing Applications

Alus represent the largest repeat families in human genome with about 1 million copies of Alu sequences in human genome. Alu clustering can be viewed as a test for the capacity of computational infrastructures because it is of great biological interests, and of a scale for other large applications such as the automated protein family classification for a few millions of proteins predicted from large metagenomics projects.

3.1. Smith Waterman Dissimilarities

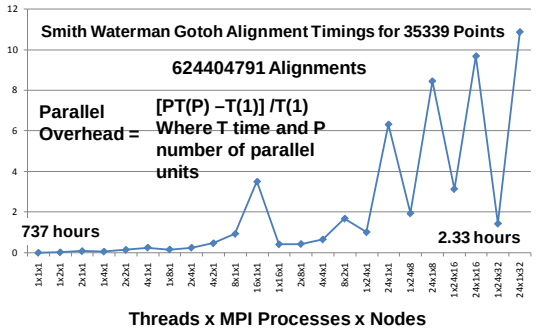

Fig 3. Performance of Alu Gene Alignments versus parallel pattern

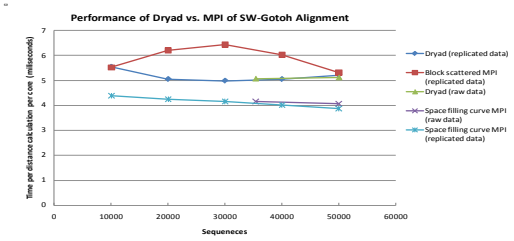

Fig. 4 Comparison of MPI with Dryad on Alu Alignment

In initial pairwise alignment of Alu sequences, we used open source version of the Smith Waterman – Gotoh algorithm SW-G modified to ensure low start up effects. We compare threading and MPI on a 32 node (768 core) Windows HPCS cluster on Fig. 3 which shows MPI easily outperforming the equivalent

threaded version. We note that threaded version has about a factor of 100 more context switches than in MPI. Fig. 4 shows Dryad getting good performance in this case lying between two MPI implementations.

We must calculate in parallel Distance D(i,j) in a way that avoids calculating both D(i,j) and the identical D(j,i). The implicit file transfer step needs optimization and is termed gather or scatter in MPI.

3.2 Pairwise Clustering

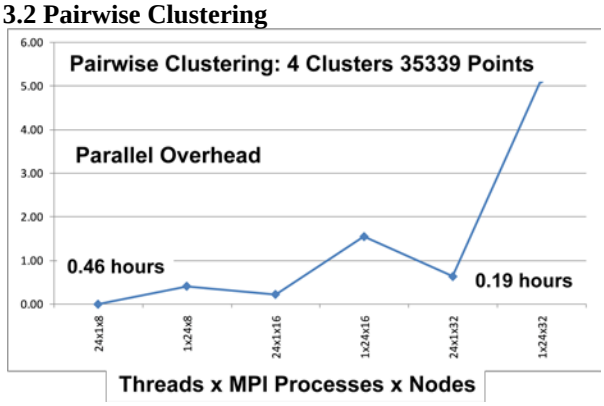

Fig 5: Performance of Pairwise Clustering for 4 clusters on 768 core Tempest. 10 Clusters take about 3.5 times longer

We have implemented a robust parallel clustering algorithm using deterministic annealing that finds clear clusters in the 35339 Alu sample with performance given in fig. 5. This uses an approach that uses no vectors but just pairwise dissimilarities [4].

3.3 Multidimensional Scaling MDS

Given dissimilarities D(i,j), MDS finds the best set of vectors $\underline{\mathbf{x}}_i$ in any chosen dimension d minimizing

$$\Sigma_{i,j}\, weight(i,j)\, (D(i,j) - |\underline{\mathbf{x}}_i - \mathbf{x}_j|^n)^2 \qquad (1)$$

The weight is chosen to reflect importance of point or to fit smaller distance more precisely than larger ones. We have previously reported results using Expectation Maximization but here we use a different technique exploiting that (1) is "just" χ^2 and one can use very reliable nonlinear optimizers to solve it. We support general choices for the *weight(i,j)* and power *n* and is fully parallel over unknowns $\underline{\mathbf{x}}_i$. All our MDS services feed their results directly to powerful Point Visualizer. The excellent parallel performance of MDS will be reported elsewhere. Note that total time for all 3 steps on the full Tempest system is about 6 hours and clearly getting to a million sequences is not unrealistic and would take around a week on a 1024 node cluster. All capabilities discussed in this paper will be made available as cloud or TeraGrid services over next year [5].

6. References

[1] Y.Yu et al. "DryadLINQ: A System for General-Purpose Distributed Data-Parallel Computing Using a High-Level Language," OSDI Symposium CA, December 8-10, 2008.

[2] J. Ekanayake and S. Pallickara, "MapReduce for Data Intensive Scientific Analysis," Fourth IEEE International Conference on eScience, 2008, pp.277-284.

[3]X. Huang and A. Madan, "CAP3: A DNA Sequence Assembly Program," Genome Research, 9,. 868-877, 1999.

[4] T Hofmann, JM Buhmann, "Pairwise data clustering by deterministic annealing", *IEEE Transactions on Pattern Analysis and Machine Intelligence 19*, pp1-13 1997.

[5] Geoffrey Fox et al., "Parallel Data Mining from Multicore to Cloudy Grids", *Proceedings of HPC 2008 Workshop*, Cetraro Italy, July 3 2008.