

Received 17 November 2004
Accepted 10 March 2005

Short Communication

PARALLELISATION OF THE BLAST ALGORITHM

YUTAO QI^{1*} and FENG LIN²

^{1,2}BioInformatics Research Center, School of Computer Engineering
Nanyang Technological University, Nanyang Avenue, Singapore 639798

Abstract: Retrieving homologous DNA and protein sequences from existing databases is a fundamental routine in bioinformatics research. Programs of the NCBI BLAST family are widely used for this purpose. We evaluated paraBLAST, a parallelised version of the NCBI BLAST algorithm, using a Message Passing Interface (MPI) on a multi-node compute cluster. Here, we propose static and dynamic database-partitioning schemes based on the availability of the cluster. We evaluated the application of the algorithm in querying nucleotide sequences against a large-scale sequence database with different numbers of database partitions, and hence, different numbers of CPUs. Since the program's tasks are performed independently of each other, each available CPU can run its own copy of BLAST queries, resulting in reduced interference between processes and leading to a highly scalable solution.

Key Words: Computational Biology, BLAST, Sequence Searching, Parallel Computing, Compute Cluster, MPI

INTRODUCTION

There has been an explosive growth in the sizes of sequence databases over the past few years. This has effectively overwhelmed the standard version of NCBI BLAST [1], thus creating an urgent demand for the development of a more powerful version. Naturally, researchers have turned to highly scalable parallel and distributed computing [2]. At the Bioinformatics Research Centre (BIRC), Nanyang Technology University, we set up a multi-node compute cluster for high-performance computing, and installed a variety of software of industry standard, based on MPI for parallel applications. MPI allows processes in a parallel program to exchange information with each other.

* Corresponding author, e-mail: antonioqi@pmail.ntu.edu.sg (Yutao Qi),
tel.: +65 67906609, fax: +65 63162780

BLAST is a set of similarity search programs designed to explore all of the available sequence databases, and can be used when the query is protein or DNA [1]. BLAST uses a heuristic algorithm that seeks local as opposed to global alignments, and is therefore able to detect relationships between sequences that share only isolated regions of similarity. Since a typical BLAST uses serial query on a large-scale database, it is relatively very slow. Several parallel BLAST programs have been developed, e.g. mpiBLAST [3]. Based on the above, we also implemented our own version of the parallelisation of NCBI BLAST, called paraBLAST. paraBLAST segments the BLAST database and distributes it into cluster nodes, and accordingly, it executes BLAST queries on multiple nodes simultaneously, resulting in significantly faster large-scale database query resolutions.

The abstract to this paper was also published in the booklet from the International Conference on Bioinformatics 2003 [4].

IMPLEMENTATION

Our analyses indicate that some BLAST processes are highly scalable, for example, generating seeds and finding the better HSP. To speed up computation, the proposed paraBLAST software processes all the tasks in parallel and integrates the results in a unified output. paraBLAST includes one Master process and several Slave process. The Master oversees the entire program execution including that of the File Provider, an application which manages the storage, versioning, and distribution of the sequence databases. The Slaves perform all of the computations for paraBLAST. All of the components of paraBLAST make use of an MPI which manages processor scheduling and inter-processor communication.

```

if (User_Defined_No_of_Fragments) {
    Size_Of_Fragments = Database_Size / Number_Of_Fragments + 1;
    /* Here add 1 to make sure that the database can be divided into
    Number_Of_Fragments partitions. */
    Number_Of_Fragments = ceil (Database_Size / Size_Of_Fragments);
} else if (Nucleotide_Sequence_Database) {
    if (Database_Size <= 700.0Mb) { //700MB is the experienced experimental value
        Number_Of_Fragments = 2;
        /* In order to use paraBLAST */
        Size_Of_Fragments = Database_Size / Number_Of_Fragments + 1;
    } else {
        Size_Of_Fragments = 700;
        Number_Of_Fragments = ceil (Database_Size / Size_Of_Fragments);
    }
} else {
    ...
    /* Only Size_Of_Fragments is different from the Nucleotide case, identical
    implementation. */
}

```

Algorithm 1. Dynamic database fragmentation

As mentioned in the previous section, paraBLAST segments the BLAST database, distributing it into cluster nodes so queries can be executed on multiple nodes simultaneously. Two different database fragmentation schemes are allowed. The simpler way requires the user to segment the database into the desired number of cluster nodes. The more sophisticated method, controlled by Algorithm 1, is based on the available resources of the used computing cluster. If the database to be queried can be fitted into the main memory, the searching time will be greatly reduced. The AlphaServer SC45 which we used gets 1 GB of memory for each CPU; after a series of experiments, we found that splitting the database into 700-MB fragments yields the best system performance, since it can be fitted into the main memory without affecting the program's execution.

EXPERIMENTAL RESULTS

To evaluate the performance of paraBLAST, we implemented the programs on the AlphaServer SC45 and queried a 560-bp nucleotide sequence named *Escherichia coli* K12 MG1655 from a large nucleotide sequence database nt [5] downloaded from the NCBI FTP server. This sequence was selected to prove that paraBLAST can output an identical result to the original NCBI BLAST result, and because it is well suited to this purpose as it is of fair length and easy to illustrate. Using paraBLAST for the search, the output with the highest hit rating was found to be the same sequence as obtained with the previous version of the software; the other output consisted of homologous sequences with high hit ratings.

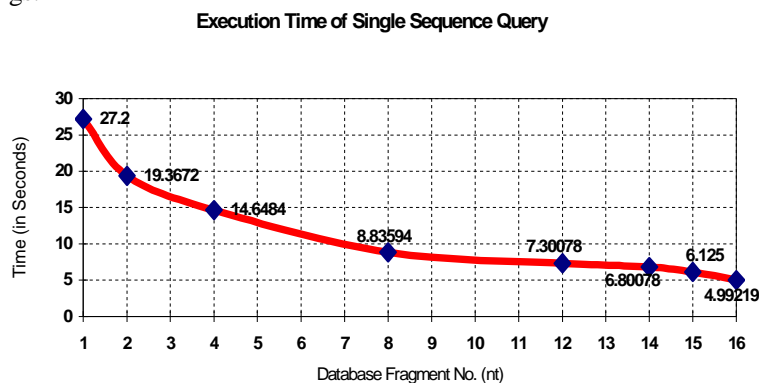


Fig. 1. Analysis of execution time.

We used parablasm, which is one of the available searching programs in our project, to perform a DNA sequence search, and found that it can also produce the same results as the original NCBI blastn. The testing datasets were randomly selected from the abovementioned database, which contains 1, 100 and 500 sequences. The execution time was evaluated with different numbers of fragments of the database; thus, different number of CPUs were utilized each

time. For comparison, the original NCBI BLAST program running on a single CPU was also executed. In order to find the fragment number with the best performance, the database was first split into an even number of fragments and then the number was adjusted. The experimental result is shown in Fig. 1.

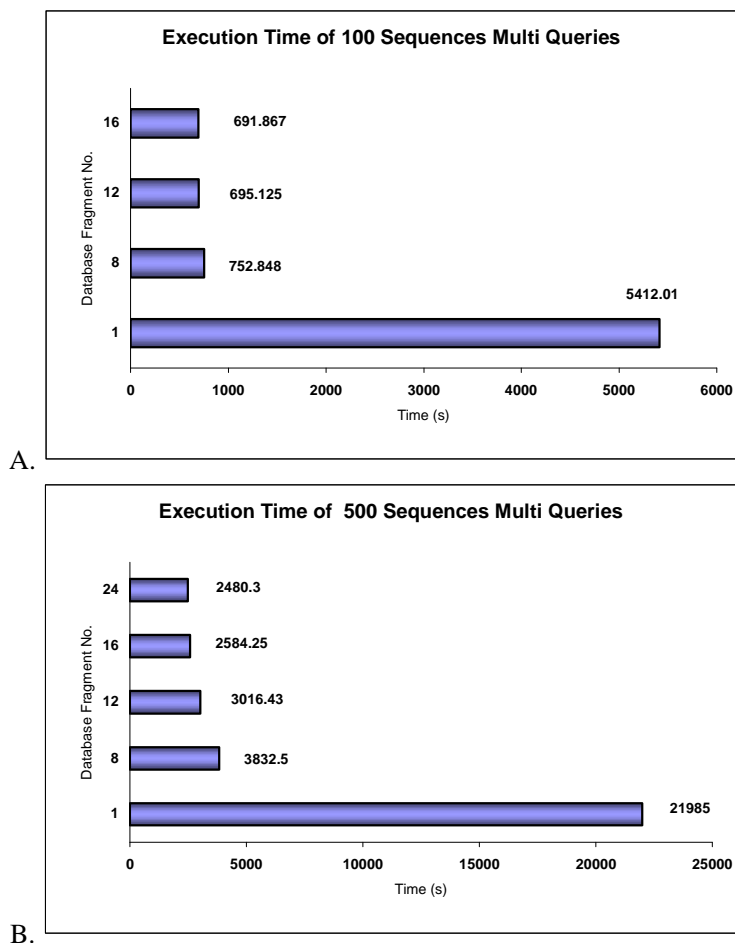


Fig. 2. Batch queries of DNA sequences by paraBLAST.

As shown in Fig. 1, the serial BLAST program takes 27.2 seconds to complete the search, while paraBLAST is considerably faster. The execution time is consistently reduced when the database fragment number is increased from 2 to 16, with an up to 444.7% speedup. To evaluate the proposed fragmentation scheme, we further used 100 and 500 DNA sequences, each with an average length of 2021 bps, to query the same database. The performance test results are shown in Fig. 2. As expected, the scalability of paraBLAST on multiple

sequence queries is even better, up to 682.23% for 100 sequences (a) and 786.4% for 500 sequences (b), compared with 444.7% in the single sequence query. The focus here is on the sensitivity of the database fragmentation. In Fig. 2a, when the database was segmented into 8 fragments, the execution time was sharply reduced. With 16 processors available in the cluster, the paraBLAST system worked out a fragmentation with 12 (a system-desired number calculated through the proposed algorithm) fragments, each of 700 MB. The execution time (695.125 s) was very close to that for 8 fragments (752.848 s) and 16 fragments (691.867 s), when the number was manually set for comparison. This experimental series also proved that our proposed dynamic database-fragmentation scheme is practical.

Similarly, the sharp reduction in the execution time is achieved in Fig. 2b. The difference is that, in this case, with 24 processors available in the cluster, the paraBLAST system worked out a full fragmentation with 24 fragments for more query sequences. We observed a consistent reduction of execution time when the number of fragments increased from 8 to 24. In other words, the fragmentation scheme proved to be robust for a variety of sequence queries.

CONCLUSIONS

In conclusion, paraBLAST is designed to speed up database searches through parallelization. It was created by adapting the serial BLAST algorithm and applying MPI communication tools on a multi-node compute cluster. paraBLAST faithfully implements the BLAST searching algorithm, but works at a much faster speed. Furthermore, paraBLAST is especially efficient for massive queries and large-scale databases, and has better scalability. It will help bioinformatics researchers save much time in their routine work on sequence similarity searches.

REFERENCES

1. Altschul, S.F, Gish, W., Miller, W., Myers, E.W. and Lipman, D.J. Basic local alignment search tool **J. Mol. Biol.** 215 (1990) 403-410.
2. Pacheco, P.S. **Parallel programming with MPI**, Morgan Kaufmann, San Francisco, 1997.
3. Aaron Darling, mpiBLAST open source project, <http://MPIBLAST.lanl.gov/>
4. Qi, Y.T. and Lin, F. Parallelisation of BLAST Algorithm, International Conference on Bioinformatics 2003 (InCoB2003), September 2003, Malaysia.
5. The NCBI Amino Acid Database – nt, <ftp://ftp.ncbi.nlm.nih.gov/>.