HCLBLAST for Genome Sequence Matching

Monika Yadav
All Saints' of technology,
Bhopal

Sonal Chaudhary
All Saints' of technology,
Bhopal

ABSTRACT
Genome sequence matching is used to reveal biological information hidden in the DNA sequences and genome sequences. The main objective is to find whether the given sequence is like other sequence or not. To find the similarity between the diseases and intensity of the disease DNA sequences are matched. There is large number of sequences and the database is still growing. Given a genome sequence and to find matching sequences from the complete database is a big challenge. The genome sequence matching algorithms are also computation intensive like BLAST; which performs large number of string matching operations. So to handle this genome sequence matching algorithms and to store data which is Big data; Hadoop is used. Hadoop is a parallel processing Big data framework. The genome sequence database can be stored on Hadoop distributed filesystem. And then can be efficient;y processed using Map/Reduce. The data is distributed in the form of blocks and for every block an instance of mapper is mapped to process the block and then output of all the mappers is combined by reducer. This Map/Reduce process has inter-node parallelism. To further speedup the process and to efficiently utilize the resources like Central processing unit and Graphical processing unit, a parallel processing framework called OpenCL is used. In this work OpenCL is integrated with Hadoop using a API called APARAPI. In addition to inter-node parallelism, intra-node parallelism is also provided and Map/reduce is accelerated for BLAST algorithm which is termed as HCLBLAST. The HCLBLAST is compared with HBLAST and BLAST algorithm for different datasets. It is found that HCLBLAST outperforms in all cases.

Keywords
DNA, HCLBLAST, BLAST, NGS

1. INTRODUCTION
Big data is outlined as great quantity of data that has want of recent technologies and design to create potential to extort worth from it by capturing and analysis method. New sources of massive data embody location specific knowledge that has arrived from traffic management and from the trailing of private devices like Smartphone's. Huge data has acquired read as a result of we have a tendency to live within the world that makes mounting use of data intensive technologies. Because of such giant size of knowledge it becomes terribly tough to realize effective analysis mistreatment existing ancient techniques. Since huge data is new approaching technology within the market which might bring the massive advantages to the business organizations, it becomes necessary varied challenges and problems associated in delivery and adopting to the current technology are have to be compelled to be perceive. Huge data thought means that a dataset that continues grew such a lot that it becomes tough to manage it mistreatment existing info models and tools. Therefore finally huge data is data that exceeds the process capability of typical info systems. The data is big sized, moves too quick, or doesn't work the structures of our info architectures. To realize gain from this data you should have a method to process it.

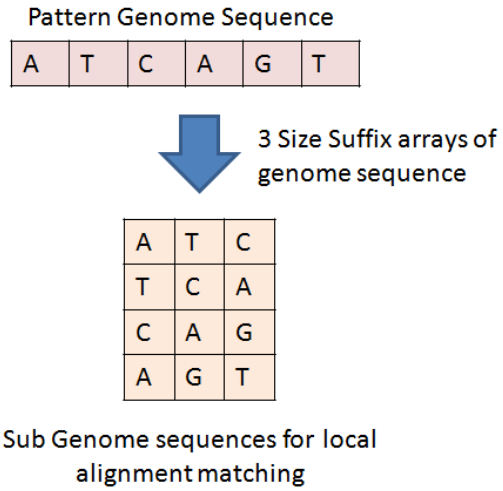
2. RELATED WORK
Alignment Search Tool (BLAST) ,a heuristic version of the pairwise native alignment Smith boatman rule, remains the foremost wide used machine procedure for alignment interrogating biological databases supported a heuristic version of the pairwise native alignment Smith boatman rule. It compares the similarity of a reference super molecule or deoxyribonucleic acid sequence against data of sequences, higher than a nominative threshold, and returns similar, statistically important, matches. In spite of its heuristic approach, it still faces important measurability challenges associated primarily with the need to go looking new and ever increasing knowledgebase; like UniMES for met genomic data sets that still expand exponentially as Next Generation Sequencing (NGS) prices still decline. BLAST, together with most different bioinformatics algorithms, is meant to execute domestically i.e. consecutive. However, the augmented turnout of ordering sequencing has light-emitting diode to large knowledge generation requiring a big increase within the speed of execution of those algorithms. the appearance of cloud computing and massive knowledge ''scale out'' technologies like Hadoop give value effective process of T sized knowledge sets therefore it's currently potential to analyse these immense datasets apace; a very important demand within the rapidly increasing field of molecular medicine. Thus, because the size of genomic knowledge sets increase earlier than native process power and disk scan speed, it's intuitive to port these naturally parallel bioinformatics tasks to use the Hadoop Map Reduce framework. Standard approaches to parallelizing BLAST mistreatment Hadoop area unit 3 fold: the primary and commonest approach distributes the input question sequences amongst a cluster of nodes, the second approach partitions the data amongst nodes and at last a hybrid approach partitions each the input sequences and therefore the data. The downside of the primary approach is that it exhibits restricted measurability and cargo equalisation doesn't occur with a little range of input sequences. The second approach needs a complicated rule to partition the data so as to make sure measurability and optimum performance. moreover, it ends up in high disk I/O. the ultimate hybrid approach is desirable because it handles giant databases yet as an oversized range of input question sequences, but it's the foremost difficult to implement and deploy whereas minimising inter-node communications and optimising the partitioning strategies. BLAST parallelization mistreatment MPI and CUDA a standard parallelised approach includes the Message Passing Interface (MPI) a parallel programming paradigm wherever a root device spawns programs on all machines in its (Ghemwat 2010)''MPI World''. Additional recently, CUDA, NVIDIA's parallel programming model for contemporary graphic processors (GPUs) addresses extremely parallel computations on one node. Darling et al. planned mpiBLAST, partitioning a sequence data supported a changed version of NCBI BLAST

with Sul et al. proposing MR-MPI-BLAST that utilises the NCBI BLAST program with AN MPI wrapper. The downside of MPI primarily based systems is that they provide restricted measurability, notably once operating with giant knowledge sets. Knowledge neck of the woods i.e. processes the info wherever its keep isn't thought-about, with knowledge instead emotional over the network to be computed on a special physical node. In distinction, fashionable paradigms like Hadoop use native storage and process to avoid network bottlenecks. Moreover, Hadoop expressly considers fault tolerance that isn't supported by default in MPI distributions. GPU primarily based approaches like GPU-BLAST, SWCUDA [12], CUDASW++ and GPU Smith-Waterman have conjointly been planned. However, such approaches exhibit inherent drawbacks like high power consumption and lower performance over multicore servers. Moreover, the utilization of GPUs doesn't carry memory constraints like MPI or Hadoop, exhibits restricted measurability and programming quality.

3. PROPOSED APPROACH

Algorithm SeqBLAST
{
1. Take pattern genome sequence from the user.
2. Divide the pattern genome sequence into suffix arrays of length 3.
3. For every suffix array generated do
   Slide suffix array until it reaches end of genome sequence in database
   If pattern sequence matches with database genome
   Sequence
   Increment match count
   End if
   End slide
   End for
4. Return match count

}

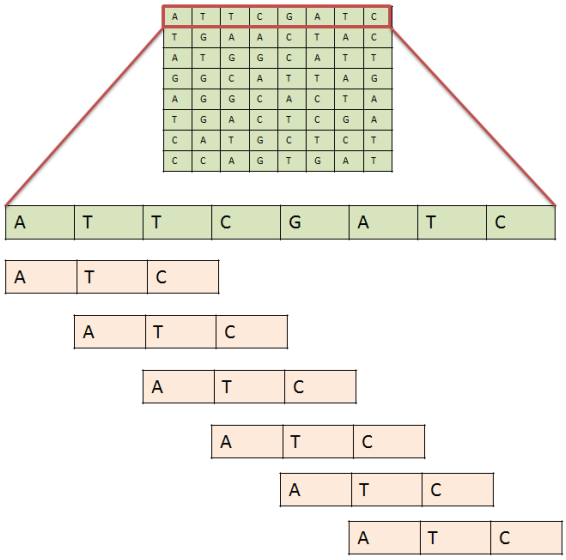


A	T	T	C	G	A	T	C
T	G	A	A	C	T	A	C
A	T	G	G	C	A	T	T
G	G	C	A	T	T	A	G
A	G	G	C	A	C	T	A
T	G	A	C	T	C	G	A
C	A	T	G	C	T	C	T
C	C	A	G	T	G	A	T

Fig 1: Database for genome sequence

Algorithm ParallelBLAST
{
1. Take pattern genome sequence from the user.
2. Divide the pattern genome sequence into suffix arrays of length 3.
3. For every suffix array generated in parallel do
   Slide suffix array until it reaches end of genome sequence in database
   If pattern sequence matches with database genome
   Sequence
   Increment match count
   End if
   End slide
   End for
4. Return match count

}



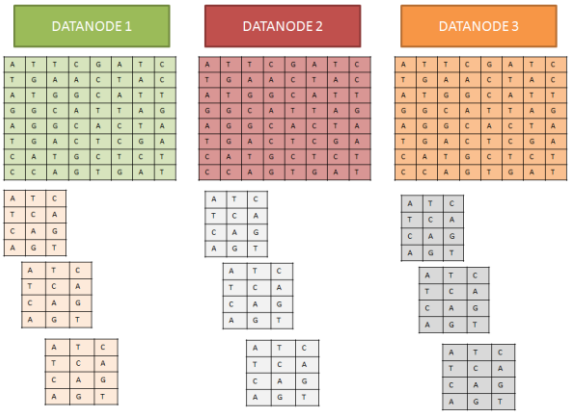


Fig 2: Database for Genome Sequence

4. RESULTS

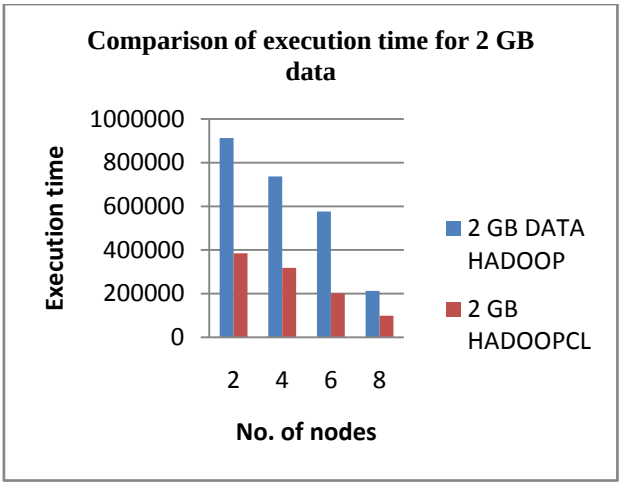
In this work Basic local alignment search algorithm is implemented on Hadoop platform and execution time for different datasets is calculated on 2 nodes cluster, 4 nodes cluster, 6 nodes cluster and 8 nodes cluster. It is found that execution on 8 nodes cluster took least time for execution.

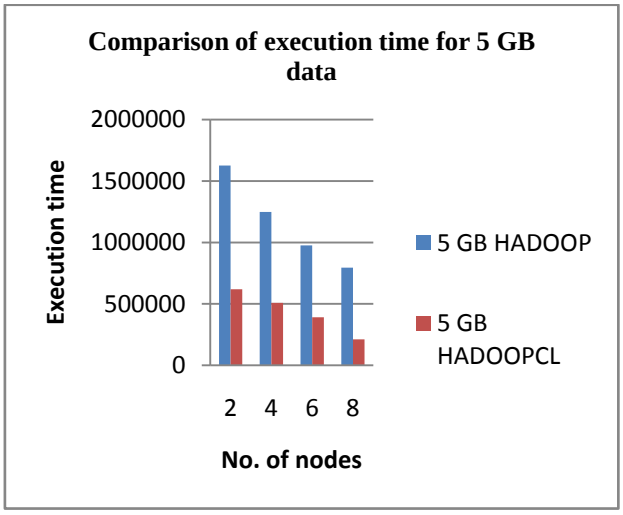
Table I Execution Time For Hadoop Blast

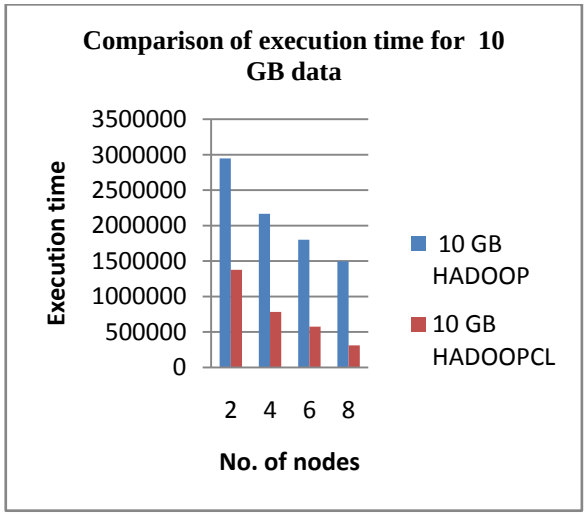
EXECUTION TIME ON HADOOP IN MILLISECONDS				
Data (In GB)	**Time on 2 nodes**	**Time on 4 nodes**	**Time on 6 nodes**	**Time on 8 nodes**
2	912751	736946	576148	212751
5	1625487	1247952	976425	794562
10	2947541	2167190	1801307	1497038

Table II Execution Time for Hadoopcl Blast

EXECUTION TIME ON HADOOPCLIN MILLISECONDS				
Data (In GB)	**Time on 2 nodes**	**Time on 4 nodes**	**Time on 6 nodes**	**Time on 8 nodes**
2	384657	317945	201907	98706
5	619037	507640	390450	210721
10	1376420	783170	576103	310640







5. CONCLUSION

Basic local alignment search is a genome sequence matching algorithm. Lot of data is involved in genome sequence matching and this database is increasing at a very fast speed. So genome sequence matching can be termed as Big data problem. To handle big data single system is not sufficient. Cluster of machines is used. In this work cluster is formed using a parallel big data processing framework Hadoop. To process large amount of genome sequences BLAST has been

implemented on Hadoop by some authors. Still improvement in terms of speedup can be done. In this work:

- Genome sequence databases are stored on Hadoop cluster.
- BLAST has been implemented using Java.
- Same algorithm is executed using Map/reduce on distributed dataset of 2 GB, 5 GB and 10 GB.
- In this parallel to increase intra-node parallelism OpenCL is integrated over Hadoop using APARAPI.
- Parallel version of BLAST on hadoop termed as HCLBLAST performs well for all datasets as compared to HBLAST.

6. REFERENCES

[1] Matsunaga A, Tsugawa M, Fortes J. CloudBLAST: Combining MapReduce and virtualization on distributed resources for bioinformatics applications. IEEE International Conference on eScience, Indiana, USA, December 2008.

[2] M. Wang, S. B. Handurukande, and M. Nassar, "2012 IEEE 4th International Conference on Cloud Computing Technology and Science RPig : A Scalable Framework for Machine Learning and Advanced Statistical Functionalities," pp. 3–10, 2012.

[3] L. P. Thompson and D. P. Miranker, "Fast Scalable Selection Algorithms for Large Scale Data," pp. 412–420, 2013.

[4] [D. Chung, X. Rui, D. Min, and H. Yeo, "Road traffic big data collision analysis processing framework," 2013 7th Int. Conf. Appl. Inf. Commun. Technol., pp. 1–4, Oct. 2013.

[5] S. H. Park and Y. G. Ha, "Large Imbalance Data Classification Based on MapReduce for Traffic Accident Prediction," 2014 Eighth Int. Conf. Innov. Mob. Internet Serv. Ubiquitous Comput., pp. 45–49, Jul. 2014.

[6] S. G. Manikandan and S. Ravi, "Big Data Analysis Using Apache Hadoop," 2014 Int. Conf. IT Converg.Secur., pp. 1–4, Oct. 2014.

[7] S. Maitrey and C. K. Jha, "Handling Big Data Efficiently by Using Map Reduce Technique," 2015 IEEE Int. Conf. Comput. Intell.Commun. Technol., pp. 703–708, Feb. 2015.

[8] J. Nandimath, "Big Data Analysis Using Apache Hadoop," pp. 700–703, 2013.

[9] J. Shafer, S. Rixner, and A. L. Cox, "The Hadoop Distributed Filesystem : Balancing Portability and Performance" in IEEE 2010.

[10] J. Conejero, P. Burnap, O. Rana, and J. Morgan, "Scaling Archived Social Media Data Analysis using a Hadoop Cloud," 2013.

[11] F. Berman, "Got data?: a guide to data preservation in the information age," Commun. ACM, vol. 51, pp. 50–56, December 2008. [Online]. Available: http://doi.acm.org/10.1145/1409360.1409376