ATLANTIS PRESS

Advances in Engineering, volume 87

2nd International Conference on Automation, Mechanical and Electrical Engineering (AMEE 2017)

Research on Genetic and Simulated Annealing Algorithm for Multiple Sequence Alignment Based on Hybrid Parallel Computation

Longsheng Li and Yu Liu*

College of Information Science and Engineering, Guilin University of Technology, Guilin 541004,Guangxi,China

*Corresponding author

***Abstract*—For the computational efficiency issues of multiple sequence alignment (MSA) caused by large amount of calculation. Based on the Hybrid Parallel model for MPI, OpenMP, CUDA, research on genetic and simulated annealing algorithm for multiple sequence alignment, to maximize computing capabilities is proposed.Analyzes the GSA-MSA algorithm for Implementation principle and characteristics of the serial algorithm, and excavates potential multi-level parallelism. To design and implement comprehensive multi-level parallel for the algorithm in a variety of hybrid parallel model, respectively designed the hybrid parallel algorithms MPI+OpenMP+CUDA within multiple nodes. Experimental results show that the hybrid parallel GSA-MSA algorithm has good speedup with keeping the sensitivity for serial algorithm.**

Keywords-MOC(MPI;OpenMP,CUDA); hybrid parallel; genetic annealing; sequence alignment

I. INTRODUCTION

Sequence Alignment (Sequences Alignment) is the basic method of biological information processing,and still not very good to solve combinatorial optimization problems, especially in both computing speed and matching accuracy.With the deepening of the biological sciences, the rapid growth of sequence[1] like GenBank, EMBL, DDB sequence database put higher requirements to the sequence alignment .

It is a good way to use the parallel computing technology to improve the computing speed in solving the operation speed. At present, the algorithms of parallel computing based on traditional MPI, PVM and OpenMP are mainly BLAST, Smith-Waterma and CLUSTAL W, etc. The algorithms based on CUDA are still rare. This paper discusses the mixed CPU and GPU heterogeneous parallel genetic simulated annealing model under multiple sequence alignment algorithm (GSA-MSA), combine the message passing model of MPI and OpenMP and CUDA shared memory model, realize the parallel strategy of hybrid cluster, multi-core CPU, multi-core GPU based on computing to speed up the process of multiple sequence alignment, providing support for the study of bioinformatics analysis.

II. MULTI-SEQUENCE ALIGNMENT

A. The Mathematical Description of Multi-sequence Alignment

Sequences Alignment is the basis and prerequisite for the bioinformatics,which use to the similarity of biological sequences (such as proteins, DNA or RNA) and to find the homology of sequences, and to predict the evolutionary relationships between sequences and the function of the unknown sequence [2]. The alignment process is to find the optimal match base-pair between sequences. For example, the protein sequence can be expressed by a finite character set Σ, which usually contains 20 English letters representing 20 amino acids, each of which is called a residue. In the mathematical model is to compare a number of 20 different characters in different sort order, through the alignment algorithm to find the maximum similarity of the two sequences.

Multiple Sequenee Alignment is a complication of the Pairwise Sequenee Alignment problem. It can be described mathematically by comparing the input protein sequence by matching the corresponding character or inserting "-" Similarity arrangement $SEQ'_{ij}\ (0 \le i < m, 0 \le j < n)$. Converting multiple sequences into a m*n two-dimensional matrix $SEQ'_{ij}\ (0 \le i < m, 0 \le j < n)$ the i rows in the matrix are a sequence, and j is the position of a residue.

B. Genetic Simulation Annealing Multiple Sequence Alignment Algorithm

There are many different algorithms for multi-sequence alignment in biological information analysis, and genetic annealing multi-sequence alignment algorithm is a global alignment algorithm. It is a general method based on natural selection and evolution to solve complex system optimization problem , it is a highly complex combination of optimization algorithms. The genetic simulated annealing multiple sequence alignment algorithm (GSA-MSA) can be defined as an 8-tuple [3]:

$$GSA - MSA = (\mathrm{C}, \mathrm{P}, \mathrm{F}, \mathrm{M}, \Phi, \Gamma, \Psi, \mathrm{T}) \tag{1}$$

Where C denotes the coding mode of the individual, P denotes the initial population, F denotes the fitness evaluation

function of the individual, M denotes the population size, Φ denotes the selective annealing operator, Γ denotes the cross-annealing operator, Ψ denotes the mutation annealing operator, T indicates termination of annealing conditions. The key points of the GSA-MSA algorithm are described in detail in the previous literature [4].

III. GSA-MSA ALGORITHM PARALLELIZATION STUDY

A. Test of algorithm Calculation

In order to fully prepare the parallelism of the algorithm, the running time of the serial algorithm is tested and analyzed. GSA-MSA serial algorithm program structure in the literature [4] carried out a detailed analysis. The test environment and the test data source are described in section 3.1 of this document. Randomly selected 2048 sequences, the average length of sequence is 320, serial algorithm program in the operation of the various parts of the results shown in Table 1.

TABLE I THE GSA-MSA SERIAL ALGORITHM TIME-CONSUMING

Calculation part	Program run-time(s)	Time ratio(%)
Sequence preprocessing	3.666	0.063
I nitialize the population	142.578	2.450
Sequence Alignment	4904.508	84.277
Update the saved population	763.985	13.128
Validation and output	4.190	0. 072
Other parts	0.582	0.010
Sum	5819.510	100

Table 1 shows the proportion of the calculation of each part of the time, reflecting the approximate distribution of the calculation. It can be seen that the sequence alignment process, the updated storage population process takes up most of the calculation time, indicating that this is the main part of the parallel component mining.

B. Parallelism Analysis

In the GSA-MSA algorithm, the three-layer[5] cycle of the sequence alignment process is the main body of the whole calculation process. Since the next generation annealing control conditions and the genetic population are used in the calculation process, the previous generation annealing control parameters and the population update data , So the two-tier outer loop can not achieve parallel processing. However, there is no data dependency between the population calculations of the innermost loop, and parallelization can be achieved. So the layer can be parallelize large particle size , with MPI based on the parallel computing between nodes to achieve. A population calculation process includes selecting the annealing operation and updating the population operation according to the random population parameters[6]. Using the population after operation to generate the sequence alignment matrix, the storage matrix of the population parameters is required to be operated four times, that is, the new and old population matrix and the new and old sequence ratio for the matrix, the process is quite time consuming. However, each row (each sequence) of these matrices is relatively independent when it is generated and manipulated, and can be parallelized to allocate each row of the population matrix and the matching matrix to different threads to execute, using a large number of threads Advantages, large-scale parallel processing.

C. Hybrid Parallel Model Architecture

Based on the heterogeneous hybrid parallel architecture of multi-core CPU and vertical core GPU under cluster, it is better to use message passing MPI, shared storage OpenMP mode and CUDA model on GPU to form three-level mixed programming model (MPI + OpenMP + CUDA) Choice, referred to as MOC.

In the mixed parallel model, the multi-node co-processing, the node inside the host side and the device side also collaborative parallel computing[7]. Through the parallel process between nodes, the host side and the device side of the multi-level parallel thread, the formation of MOC three parallel model. The hybrid programming model is a good combination of the advantages of three programming methods. MPI is at the top level, responsible for communication between nodes, managing compute nodes; OpenMP in the middle, opening multi-threaded management of the GPU and responsible for some computing tasks, CUDA is the bottom, open a large number of lightweight threads focused on large-scale parallel computing tasks. As shown in Figure 1.

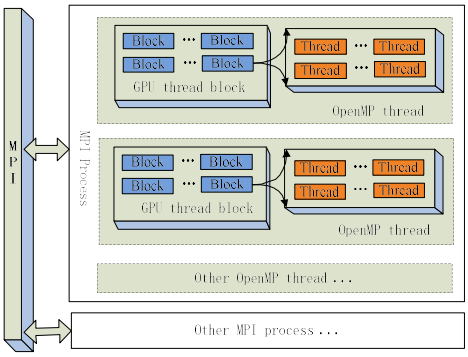

FIGURE I. HYBRID PARALLEL EXECUTION MODEL DIAGRAM

D. Implementation Method

In the overall parallel design of the algorithm, large-scale parallel tasks are calculated on the GPU. For the outer MPI parallel and middle openMP, the parallel computation of the two processes is not very complicated. The following is a detailed description of the parallel computing functions running on the GPU side of the generation sequence alignment matrix, computational fitness, and fitness conventions.

- Fitness calculation

Since MSA is a dual sequence alignment, it has natural parallelism. The fine-grained parallel approach can be determined based on the data-uncorrelated properties of the two sequence alignment versus the other two sequences. The pair of matrices, which are selected from the sequence of columns, is a pair of pairs. Respectively, each group assigned a different GPU thread parallelization to achieve processing, as shown in Figure 4. At the same time, the parallelism is also performed on the graph and the comparison group. For the first residue comparison, the parallel processing of the two sequences is carried out by using the two threads respectively.

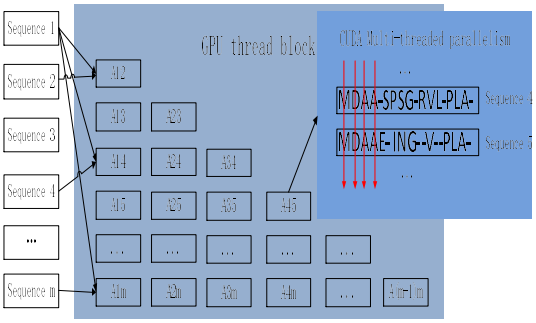

FIGURE II. GSM-MSA FINE-GRAINED PARALLEL STRATEGY DIAGRAM

- Statute processing

Parallelization of the fitness of a population will result in a large number of storage intermediate values, occupying a certain amount of storage space, and requiring parallel processing[8]. When the number of sequences is m and time-level protocol is m*(m-1)/2 data, if you do not optimize the storage structure, you have to consume a storage m*m storage unit. When the length of the matrix sequence is n, the secondary parallel storage will consume m*(m-1)/2*n storage unit, when several populations parallel computing calculate at the same time, it will have to do two levels of the value of the standard.

Protocol parallelization in the GPU can be a very good implementation, divide and rule which is a common strategy. It decomposes the original large-scale problem into a smaller sub-problem for independent solution[9], and then combines the solution of the sub-problem into the solution of the original problem, as shown in Fig5. In the figure, if there are 8 numbers to achieve the statute, then divided into two datas for one group, each protocol is reduced by half of the array. Each time the protocol can be a number of threads at the same time, each cycle to participate in the operation of the thread to reduce the half, only the protocol part, the time complexity from the down to the efficiency is improved

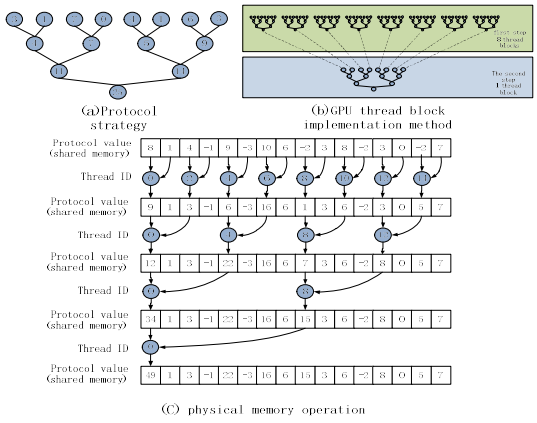

FIGURE III. FITNESS SPECIFICATION METHOD

IV. ALGORITHM PERFORMANCE TEST AND ANALYSIS

A. *Experimental Platform and Test Model*

The test is performed on a 4-node PC cluster with 1000 Mbps network connections. Each node is configured with 64-bit CentOS 6.4 operating system, Intel (R) Core (TM) i5-3470 CPU (physical 4-core), 8GB memory, GeForce GTX 680GPU (Kepler Architecture, stream processor 1536). Cluster installation PGI CDK development environment, support C / C + +, MPI / OpenMP / CUDA hybrid development.

The sequence data is taken from the gbbct1.fsa_aa and gbbct2.fsa_aa files in the GANBANK database of NCBI. Randomly selected 34 different sequence number and length of the example. Number of sequence instances: from 8 to 2048; the average length of three sequences: 128,320,640. In addition, the probability of change in the fixed point of view, it can be variation, insert and delete any of the three changes, select the probability of a change is equal. To ensure that the parallel algorithm calculation task is the same, the following parameters are used, the starting temperature is 10, the termination temperature is 1, the population size is 16, the number of iterations is 20, and the annealing coefficient is 0.5.

B. *Analysis of Test Results*

There are two criteria for evaluating the results of a multi-sequence alignment algorithm: the sensitivity and computational speed of the best alignment. The accuracy of the sensitivity of the GSA-MSA algorithm has been verified and has a certain degree of confidence [10]. So the main implementation of the algorithm and the acceleration performance of a comprehensive test evaluation.

For the execution time of the algorithm, select the instance with the average length of 128 to do a detailed analysis. Table 2 shows the execution time curves of the two algorithms (in seconds). The time of the algorithm increases with the increase of the sequence scale. The serial algorithm grows fastest, and when the number of sequences is 2048, the execution time reaches 21074.70 seconds, and the three-layer MOC hybrid algorithm The execution time is only 53.58 seconds. However, when the input data is too small, the acceleration effect is not obvious, the number of sequences is 32 and below, the execution time is greater than the serial time, which is due to process and thread communication and synchronization to take a certain time-consuming. Such as 32-bit serial algorithm only 4.83 seconds, while the CUDA algorithm to 6.81 seconds. It can be seen that the execution time of the parallel algorithm is much lower than that of the serial algorithm. The larger the sequence size is, the more obvious the execution time advantage of the parallel algorithm is.

TABLE II THE TOTAL TIME COMPARISON OF SERIAL ALGORITHM AND MOC PARALLEL ALGORITHM(THE LENGTH OF SEQUENCE IS 128)

Number of sequences	Serial time(s)	MOCtime (s)
8	0.39	5.44
32	4.83	6.81
64	20.06	6.91
128	87.68	7.18
256	348.26	7.88
384	725.46	9.35
512	1409.49	10.70
640	2160.23	11.33
768	3040.21	12.05
896	4040.06	14.85
1024	5258.53	17.17
1152	7044.90	21.56
1280	8205.81	24.72
1408	10317.20	28.81
1536	11669.20	33.03
1664	13774.50	37.71
1792	16045.10	41.97
1920	18376.00	47.64
2048	21074.70	53.58

Acceleration ratio is the most important high-performance computing metric, which is determined by the algorithm itself and the experimental environment. MOC under the GSA-MSA three-layer hybrid algorithm in theory to maximize the use of the cluster nodes of the computing resources, greatly improve the acceleration effect. The acceleration ratio curves for the three sequence lengths are shown in Fig. In the figure, the parallel algorithm achieves different acceleration effects on different sequence scales, and the acceleration ratio increases steadily with the number of sequences. When the number of sequences is 2048 and the average length is 128,320,640, the acceleration ratio is 393.42, 285.94, 256.86 respectively. And the shorter the average sequence length, the better the acceleration effect is, as in the case of a length of 128, the overall effect is greater than the length of 320 and 640. So the acceleration ratio in the larger number of sequences and the smaller the length of the sequence can get a better acceleration effect.

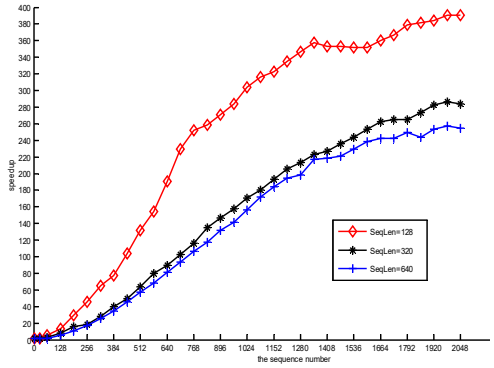

FIGURE IV. THE SPEEDUP OF SERIAL ALGORITHM AND MOC PARALLEL ALGORITHM

V. CONCLUSION

The heterogeneous hybrid parallel computing platform based on multi-machine, multi-core CPU and multi-core GPU is easy to set up, low cost, good system scalability and better use of multi-level parallel computing resources. For computationally intensive applications, parallelization based on the parallel model can achieve better acceleration.

The computational density of GSA-MSA algorithm is high, and there are many levels of potential parallelism. In particular, the genetic algorithm has the advantages of high parallelism and sequence matching process. The experimental results show that the three-layer MPI + OpenMP + The acceleration effect of parallel algorithm under CUDA model is better than that of serial algorithm. When the average length is 128, the acceleration ratio is 393.42. On the whole, the acceleration effect of hybrid algorithm is quite obvious. The reason is the number and length of the sequence, the more the number of sequences, the average length is smaller when the acceleration effect is ideal.

The algorithm has good expansibility, and the experimental results show that the acceleration effect is more obvious when the population quantity and temperature control are increased, and the acceleration effect is basically stable when the number of genetic iterations is increased.

ACKNOWLEDGEMENT

The work reported in this paper has been supported by The National Natural Science Foundation of China under research project 41264005. We would also like to thank the anonymous reviewers for their comments and valuable suggestions.

REFERENCES

[1] Attwood T K, Parry Smith D J ,The concept of bioinformatics[M],LUO Jing-chu translated. Beijing:Peking University Press,2002(:23-25).

[2] Cyntbia Gibas,Per Jambeck.Bioinformatics Computer Skills[M].SUN Chao,GUO Qing-ming,LIU Xiang-guo,WU Bin translated.Beijing:China Electric Power Press.207-230.

[3] R.Doolittle,"Similar amino acid sequences:Change or common ancestry?" Science,vol.214,no.4517,pp.149-159, oct.1981

[4] ZHANG Li.Research of multiple sequence alignment based on genetic algorithm.Hunan:Hunan University,2011.

[5] Mardis E R. The impact of next-generation sequencing technology on genetics [J]. Trends in genetics, 2008, 24(3):133-141.

[6] Kumar V, Grama A, Gupta A. Introduction to parallel computing [M].Redwood City:Benjamin/Cummings,1994:49-66.

[7] ZHOU Hong.Research on parallel algorithm of multiple DNA sequence alignment on de Bruijin graph[D].Tianjin:Tianjin University.2010

[8] ZHU Yong-zhi,XU Guo-qiang,YU Ji-guo.Optimized implementation of N-body problem based on OpenMP/MPI parallel programming model[J].Computer Engineering and Applications,2016,52(5):16-21.

[9] WANG Zhuo-wei,CHENG Liang-lun,ZHAO Wu-qing.Parallel computation performances analysis model based on GPU[J].Computer Science.2014;Vol 41,No 1.

[10] Daniel Hackenberg, Guido uckeland, Holger Brunst. Performance analysis of multiple-level parallelism: inter-node, inter-node and hardware accelerators[J]. Concurrency Computat, 2012,24:62-72.