Vol.4, No.2, 47-51 (2013)
http://dx.doi.org/10.4236/jbpc.2013.42006

Journal of Biophysical Chemistry

# Improvements in the score matrix calculation method using parallel score estimating algorithm

**Geraldo F. D. Zafalon[1,2*], Evandro A. Marucci[2], Julio C. Momente[2], José R. A. Amazonas[1], Liria M. Sato[1], José M. Machado[2]**

[1]Escola Politécnica University of São Paulo Av. Prof. Luciano Gualberto, São Paulo, Brazil;
[*]Corresponding Author: zafalon@gmail.com
[2]Departamento de Ciências de Computação e Estatística, São Paulo State University Rua Cristóvão Colombo, São José do Rio Preto, São Paulo, Brazil

Received 21 February 2013; revised 25 March 2013; accepted 10 April 2013

## ABSTRACT

**The increasing amount of sequences stored in genomic databases has become unfeasible to the sequential analysis. Then, the parallel computing brought its power to the Bioinformatics through parallel algorithms to align and analyze the sequences, providing improvements mainly in the running time of these algorithms. In many situations, the parallel strategy contributes to reducing the computational complexity of the big problems. This work shows some results obtained by an implementation of a parallel score estimating technique for the score matrix calculation stage, which is the first stage of a progressive multiple sequence alignment. The performance and quality of the parallel score estimating are compared with the results of a dynamic programming approach also implemented in parallel. This comparison shows a significant reduction of running time. Moreover, the quality of the final alignment, using the new strategy, is analyzed and compared with the quality of the approach with dynamic programming.**

**Keywords:** Algorithms; Scoring Matrix; Parallel Programming; Alignment Quality

## 1. INTRODUCTION

The biologists need the help of Bioinformatics because everyday they generate a huge amount of data from their experimental results and they need to analyze these results aligning the sequences, searching for some patterns on them and identifying some hot spots [1,2]. However, they can not perform this action in time without computers [3].

When considering multiple sequence alignments (MSA), the dynamic programming algorithms are efficient only for a limited number of multiple lengthy sequences [4]. The MSA algorithms, using stochastic methods, have become a feasible alternative for the situations in which the dynamic programming approach is not convenient [5].

Nevertheless, with the increase of problems' complexity, the number of sequences to be analyzed has grown from several dozens to several thousands [6], demanding more computing power to perform this analysis [3].

With the demand for more computing power, parallel and distributed computing were used to improve the performance of task execution in Bioinformatics, specially stochastic algorithms, thus putting together high performance computing and Bioinformatics.

Stochastic approaches can not arrive at the exact solution, but they try to obtain the best optimality degree of the solution. The progressive MSA is one of the most used stochastic algorithms for aligning sequences with a good performance and a reasonable quality.

The progressive MSA algorithm is divided into three stages: the first stage is the score matrix calculation, the second is the phylogenetic tree construction and the last is the multiple alignment. The score matrix calculation is the most computationally complex of the three, because it performs the pairwise comparisons among sequences, generally using dynamic programming. However, some strategies can be used to reduce the computational complexity of first stage and score estimating is one of them [7,8].

This work shows some results obtained by an implementation of a parallel score estimating technique (which we call the new approach) in the score matrix calculation stage and the comparison of this method with traditional dynamic programming also implemented in parallel (which we call the standard approach). The execution time and the quality of the obtained alignments were compared between the approaches. This new approach can reduce the computational complexity of this step from $O(mn)$ to $O(m + n)$, considering two sequences $X$ and $Y$ of lengths $m$ and $n$, respectively [8].

This paper is organized as follows: Section II reviews the main concepts of the multiple sequence alignments and discusses some related works; in Section III the score estimating algorithm is described; in Section IV, the obtained results are presented and discussed, and in Section V, the conclusions are presented.

## 2. SEQUENCE ALIGNMENT

The sequence alignment is not an easy task and the biologists constantly need evaluations over genes and the characteristics of proteins of species.

The alignment among sequences of DNA/RNA and proteins of different species is a hypothesis of homology among the components of genes and proteins. The alignments can be used as models to propose and test evolutionary hypothesis which are also important to the studies of phylogeny [1,3].

The use of MSA algorithms, the parallelization of them and the optimization techniques for these algorithms have been improved in the last years.

Some methods to improve the execution time of MSA algorithm based on pairwise comparison, where the goal is to find an optimal alignment with some restrictions, were proposed [9]. However, they did not achieve the efficiency of parallel solutions [10-12].

Otherwise, with the growing size of the sequences, the parallel solutions began to lose their high efficiency, and the addition of optimization techniques in the algorithms of parallel solutions became necessary [13,14].

Our work explores all the parallelism power in the solution of multiple sequence alignment problems, developing a parallel version over the sequential score estimating optimization technique proposed by Chen *et al.* [8], and applying it in the first stage (the score matrix calculation) of a progressive multiple sequence alignment (MSA). We performed comparisons in the execution time of this method with traditional dynamic programming implemented in parallel and also applied in this first stage.

## 3. SCORE ESTIMANTING

Usually, the standard progressive MSA algorithm uses in its first step the dynamic programming algorithm. However, from our research experience we realized that there are different strategies to calculate this score matrix (a matrix containing the final score of aligned pairs) which work better. More specifically, the standard approach is based on the progressive algorithm of Clustal [6,15], which is a well known, largely used and consescrated strategy in Bioinformatics. The source code of the tool can be obtained in the web page of the European Bioinformatics Institute[1]. Basically, we have taken the first stage of this algorithm, the pairwise alignment, and parallelized it (standard approach) as can be seen in **Figure 1**. The sequence pairs are distributed among the processors of the parallel machine and then each processor calculates the score of each pairwise alignment. When the processor finishes the task, it returns the score value to be written in the score matrix.

In the new approach, we used the estimating score technique to obtain the score matrix results. However we did not use it in a sequential way, as it is reported in the literature [8]: we parallelized the stages of this technique to improve the performance of the algorithm. To the extent of our knowledge, this approach was not previously reported in the literature.

Each one of the four stages of the estimating score algorithm performs a scan in the sequences which are placed in pairs. Each pair of sequences has to perform the four stages, necessarily. The stages are classified as: Right-Upper, Right-Lower, Left-Upper and Left-Lower. The classifications Upper and Lower are related to the position of the sequences in the analysis. The Right and Left movements are related to the scanning directions. The maximum score among four stages is used to the matrix score.

**Figure 2** shows an illustration of the developed parallel score estimating algorithm. To illustrate, it will be explained the Right-Upper execution, which is executed in one node. The other steps (Left-Upper, Left-Lower and Right-Lower) work likewise, the differences are the sequences scanning directions and the starter character.

In the Right-Upper execution, the last character on the right side of the upper sequence is chosen and set as the

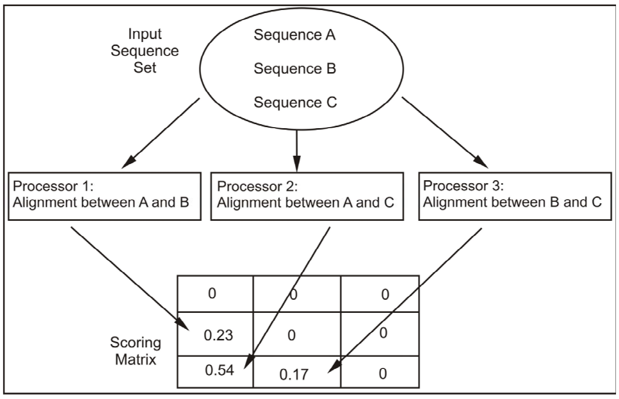
Input Sequence Set
Sequence A
Sequence B
Sequence C
Processor 1: Alignment between A and B
Processor 2: Alignment between A and C
Processor 3: Alignment between B and C
Scoring Matrix
0
0
0
0.23
0
0
0.54
0.17
0

**Figure 1.** Illustration of parallel pairwise alignment algorithm.

[1] http://www.ebi.ac.uk/Tools/clustalw/

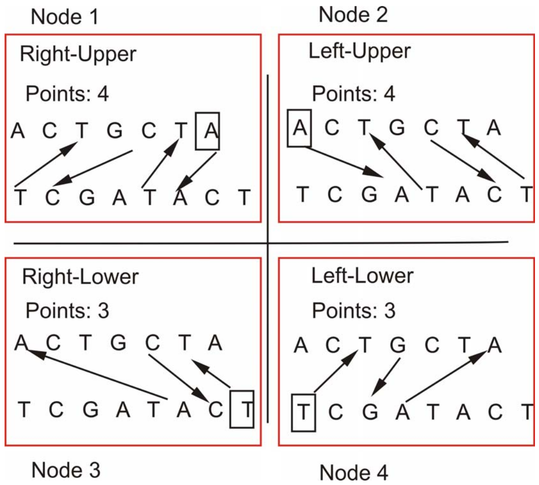

**Figure 2.** Illustration of the parallel score estimating al gorithm.

starter character. In this case, it is the character *A*. Departing from it, a series of comparisons with the characters of lower sequence are performed, going from right to left (this is the direction of scanning). If there is no equal character (a match) in the lower sequence, the algorithm moves to the next left character in the upper sequence, (*i.e.* considering it the new starter character) and repeats the complete scanning process again. Otherwise, if an equal character is found in the lower sequence (this is the case in our example, when we find a match), one point is scored and the scanning starts again, now taking the new starter character in the lower sequence. The match is with the character *A* (third from right to left in the lower sequence), as it can be seen in the **Figure 2**. The new starter character, now in the lower sequence, is the next left character from where is the match. Then, taking character *T* as the new starter character (fourth character from right to left in the lower sequence), we perform comparisons with upper sequence, starting from its second character, which is also character *T*, from right to left in this sequence. This process is repeated until the two sequences are covered.

The square around the illustration of sequence pairs indicates that each task is performed in a different processor unit. The distribution of the stages is done as soon as the processor is or becomes available. This approach is possible, because the stages are totally independent.

The algorithm might be executed in any amount of processors, because the distribution of the stages is done through an order queue, where the four stages of the pair of sequences in time are distributed for the processors and, if there are more processors available, the stages of the next pair of sequences in the queue are distributed too, until all the processors are working (busy). This process is repeated until the end of the pairs of sequences and their stages. Below, we present an algorithm for this control:

```
While (pair_of_sequences_has_not_yet_executed > 0)
{
    waiting(); //waiting message of processor's availability
        check_the_order_queue();
        allocate_the_correct_stage_of_time();
    decrement_counter_pair();
}
```

## 4. RESULTS

In this section we report the results that demonstrate the improvement in the performance of the new approach, implemented by the score estimating algorithm, when compared to the standard multiple progressive alignment, implemented with parallel dynamic programming. It is important to emphasize that the performance results showed here are related only to the execution time of the first stage of the algorithm.

The tests were performed with 550 residues on average for nucleotides and with 180 residues on average for amino acids, with different amounts of sequences for both approaches. They were run under a Linux Debian Beowulf cluster of Athlon XP 2100 + with 9 operational nodes. The front-end node has 2 GB of memory and 2 disks of 80 GB each, and the other 8 nodes have 1 GB of memory and 1 disk of 80 GB for each node. The communication interface is based on Fast Ethernet 10/100 and uses MPICH as a communication library.

**Figure 3** shows the execution time results for tests with nucleotides, for the standard and new approaches.

It can be seen in **Figure 3** that the new approach, using the score estimating, has better performance than the standard approach. The improvement in the execution time is around 15%.

**Figure 4** shows the execution time results for tests

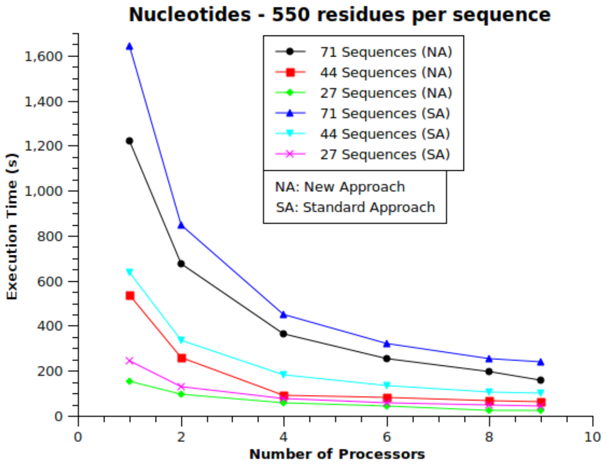

**Figure 3.** Performance evaluation of standard approach (SA) and new approach (NA) algorithms for sequences with nucleotides.

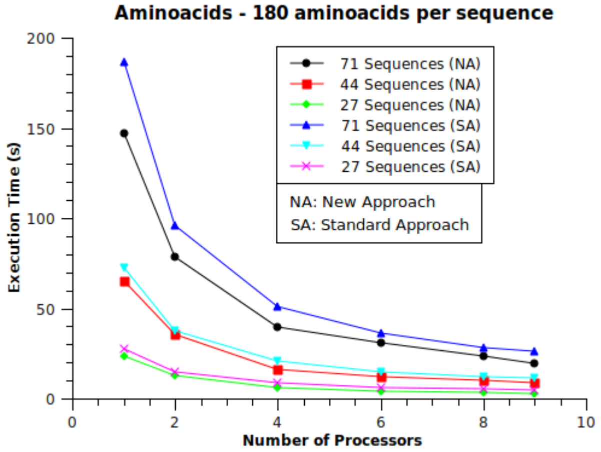

**Figure 4.** Performance evaluation of standard approach (SA) and new approach (NA) algorithms for sequences with amino acids.

with amino acids for the standard and new approaches.

Analyzing **Figure 4**, it can be seen an improvement in the execution time of the new approach when both approaches are compared. The reduction of the execution time is around 12%.

From the results showed previously, the execution time reduction was significant and, consequently, the speedup of the new approach is improved too.

Besides the performance results showed earlier, the quality of the final alignments produced were analyzed, to show that the results produced in score matrix of new approach are correct to be used in the other stages, when compared with the results of the standard approach, and produces results with biological significance. Since as both approaches are the first stage of a MSA algorithm, the quality tests of the final alignments were performed by completing the other two stages of the MSA algorithm, that are the phylogenetic tree construction and the multiple alignment, with the same algorithm and rules.

On average, the new approach showed very close results (slightly better, some times) when compared with the standard approach, as presented in the **Table 1**. It shows some results about the quality of the final alignments for some amino acids sequences. The two first columns are, respectively, the name of the analyzed gene (*Gene*) and its number of sequences (*N. Seq*). The other two columns (*New Approach and Standard Approach*) are the score values of the tools ranging from 0 (worst) to 1 (best). These score values were obtained by the *bali_score* software[2] the choice of sequences was completely randomic, and it was made on the *BaliBase* website[3] [16]. The *BaliBase* is an aligned protein database, and each set of amino acids sequence alignments is made manually through a structural study of the protein. This ensures that alignments of the *BaliBase* are 100% reliable. The bali_score software compares the *BaliBase* alignment with the alignment obtained by different tools.

**Table 1.** Quality comparison between the new approach and the standard approach.

| Gene | N. Seq. | New Approach | Standard Approach |
|---|---|---|---|
| 1ad3_ref1 | 4 | 0.929 | 0.899 |
| 1cpt_ref1 | 9 | 0.787 | 0.845 |
| 1fmb_ref4 | 9 | 0.882 | 0.852 |
| 1tis_ref5 | 18 | 0.898 | 0.867 |
| 1ppn_ref5 | 26 | 0.906 | 0.893 |
| 1gdoA_ref5 | 42 | 0.877 | 0.823 |
| 1ubi_ref3 | 48 | 0.797 | 0.786 |
| 1thm_ref5 | 49 | 0.794 | 0.775 |
| kinase_ref4 | 62 | 0.802 | 0.804 |

## 5. CONCLUSIONS

The present work reports some improvements in progressive MSA through the application of the score estimating as an optimization technique. The results show that the new approach when compared with the standard one can achieve better results in performance evaluation when we compare the execution times. Besides, the new approach assured the quality results of the final alignments when compared with the standard approach.

The obtained results are relevant because the improvement of the performance with the assurance of the alignments quality makes this approach interesting and very useful for computer scientists and biologists. Moreover, it might be portable and implementable in a Grid or GPU environments to work with even larger data sets.

## REFERENCES

[1] Chou, K.C., Zhou, D., Fan, X., Tan, D., Xu, Y., Tavis, J.E. and Bisceglie, A.M.D. (2007) Separation of near full-length hepatitis c virus quasispecies variants from a complex population. *Journal of Virological Methods*, **141**, 220-224. doi:10.1016/j.jviromet.2006.12.002

[2] Edgar, R.C. and Batzoglou, S. (2006) Multiple sequence alignment. *Current Opinion in Structural Biology*, **16**, 368-373. doi:10.1016/j.sbi.2006.04.004

[3] Arcuri, H.A., Zafalon, G.F.D., Marucci, E.A., Bonalumi, C.E., Da Silveira, N.J.F., Machado, J.M., De Azevedo, W.F. and Palma, M.S. (2010) SKPDB: A structural database of shikimate pathway enzymes. *BMC Bioinformatics*, **11**, 1-7. doi:10.1186/1471-2105-11-12

[4] Needleman, S.B. and Wunsch, C.D. (1970) A general method applicable to the search for similarities in the amino

[2] http://www.bio3d-igbmc.u-strasbg.fr/balibase/BalibaseDownload/
[3] http://www.bio3d-igbmc.u-strasbg.fr/balibase/

acid sequence of two proteins. *Journal of Molecular Biology*, **48**, 443-453. doi:10.1016/0022-2836(70)90057-4

[5] Wallace, I.M., Blackshields, G. and Higgins, D.G. (2005) Multiple sequence alignments. *Current Opinion in Structural Biology*, **15**, 261-266. doi:10.1016/j.sbi.2005.04.002

[6] Larkin, M., Blackshields, G., Brown, N., Chenna, R., McGettigan, P., McWilliam, H., Valentin, F., Wallace, I., Wilm, A., Lopez, R., Thampson, J., Gibson, T. and Higgins, D. (2007) Clustal w and clustal x version 2.0. *Bioinformatics*, **23**, 2947-2948. doi:10.1093/bioinformatics/btm404

[7] Zomaya, A.Y., Ercal, F. and Olariu, S. (2001) Solutions to parallel and distributed computing problems—Lessons from biological sciences. John Wiley & Sons, Chichester.

[8] Chen, Y., Pan, Y., Chen, J., Liu, W. and Chen, L. (2006) Partitioned optimization algorithms for multiple sequence alignment. *Proceedings of the* 20*th International Conference on Advanced Information Networking and Applications* (*AINA*'06), 18-20 April 2006, **2**. doi:10.1109/AINA.2006.260

[9] Bilu, Y., Agarwal, P.K. and Kolodny, R. (2006) Faster algorithms for optimal multiple sequence alignment based on pairwise comparisons. *IEEE/ACM Transactions on Com-Putational Biology and Bioinformatics*, **3**, 408-422. doi:10.1109/TCBB.2006.53

[10] Thorsen, O., Smith, B., Sosa, C.P., Jiang, K., Lin, H., Peters, A. and Chung F.W. (2007) Parallel genomic sequence-search on a massively parallel system. *Proceedings of the* 4*th International Conference on Computing Frontiers*, Ischia, 7-9 May 2007, 59-68. doi:10.1145/1242531.1242542

[11] Altschul, S.F., Gish, W., Miller, W., Myers, E.W. and Lipman, D.J. (1990) A basic local alignment search tool. *Journal of Molecular Biology*, **215**, 403-410. doi:10.1016/S0022-2836(05)80360-2

[12] Gardner, M.K., Chung F.W., Archuleta, J., Lin, H. and Mal, X. (2006) Parallel genomic sequence-searching on an ad-hoc grid: Experiences, lessons learned, and implications. *Proceedings of the* 2006 *ACM/IEEE Conference on Supercomputing*, Tampa, 11-17 November 2006, 22. doi:10.1109/SC.2006.46

[13] Moss, J. and Johnson, C.G. (2003) An ant colony algorithm for multiple sequence alignment in bioinformatics. *Artificial Neural Networks and Genetic Algorithms*, 182-186. doi:10.1007/978-3-7091-0646-4_33

[14] Lee, Z.-J., Su, S.-F., Chuang, C.-C. and Liu, K.-H. (2008) Genetic algorithm with ant colony optimization (ga-aco) for multiple sequence alignment. *Applied Soft Computing*, **8**, 55-78. doi:10.1016/j.asoc.2006.10.012

[15] Ebedes, J. and Datta, A. (2004) Multiple sequence alignment in parallel on a workstation cluster. *Bioinformatics*, **20**, 1193-1195. doi:10.1093/bioinformatics/bth055

[16] Thompson, J.D., Koehl, P., Ripp, R. and Poch, O. (2005) Balibase 3.0: Latest developments of the multiple sequence alignment benchmark. *Proteins*: *Structure*, *Function*, *and Bioinformatics*, 61, 127-136. doi:10.1002/prot.20527