Original Research Paper

Performance Improvement of SNP Search Using Multithread Programming

[1,2,3]Geraldo Francisco Donegá Zafalon, [1]Álvaro Magri Nogueira da Cruz,
[1,3]Anderson Rici Amorim, [1]Matheus Carreira Andrade,
[1,3]Allan de Godoi Contessoto, [1]Laendro Alves Neves,
[1]Rogéria Cristiane Gratãode Souza, [1]Carlos Roberto Valêncio and [3]Liria Matsumoto Sato

[1]Department of Computer Science and Statistics - DCCE, São Paulo State University (Unesp),
Institute of Biosciences, Humanities and Exact Sciences (Ibilce), Campus São José do Rio Preto, São Paulo, Brazil
[2]Universidade Paulista – ICET – Campus de São José do Rio Preto, São Paulo, Brazil
[3]Escola Politécnica – University of São Paulo, São Paulo, Brazil

Article history
Received: 30-07-2018
Revised: 12-10-2018
Accepted: 02-11-2018

Corresponding Author:
Geraldo Francisco Donegá Zafalon
Department of Computer Science and Statistics - DCCE, São Paulo State University (Unesp), Institute of Biosciences, Humanities and Exact Sciences (Ibilce), Campus São José do Rio Preto, São Paulo, Brazil
Email: geraldo.zafalon@unesp.br

Abstract: Pattern recognition is an important field in Bioinformatics and a well-known task is the search for Single Nucleotide Polymorphism (SNP). It is possible to search for a known SNP position and analyze it using patterns of DNA bases, called masks. Nonetheless, this process becomes computationally expensive as the amount of available genomic data increases. Thus, in this study, we have developed a parallelization scheme, based on multithread programming, to SNP analysis using masks. In our tests, we noticed that the proposed scheme improved the execution time in 98.05 times when compared with the sequential approach.

Keywords: Multithreaded Approach, SNP Analysis, Bioinformatics

Introduction

Due to the huge amount of genomic data available, performing manual biological analysis becomes unfeasible. Thus, it is necessary the development of computational methods to support biologists in their analysis and inferences. This fact has originated Bioinformatics (Amorim *et al.*, 2016).

Bioinformatics is a Computer Science branch that aims to provide computational solutions to biological problems, as sequence alignment (Marucci *et al.*, 2014; Zafalon *et al.*, 2015) and pattern recognition (Hemalatha and Vivekanandan, 2008; Khuri, 2008). Concerning the pattern recognition problems, the search problem is one of the most important (Wang *et al.*, 2016) and a well-known one is the search and analysis for Single Nucleotide Polymorphism (SNP) (Trick *et al.*, 2009). The detection of a SNP can be performed through the amplification of the target sequence and its identification using hybridization probe (Real-Time PCR) or through DNA sequencing using capillary electrophoresis (Sanger Sequencing) (Sanger and Coulson, 1975) or Next Generation Sequencing (NGS) technology (Margulies *et al.*, 2005). After that, the analysis of the SNPs is performed manually through chromatogram of the sequence. The visualization a chromatogram is assisted by software, as can be seen in Fig. 1.

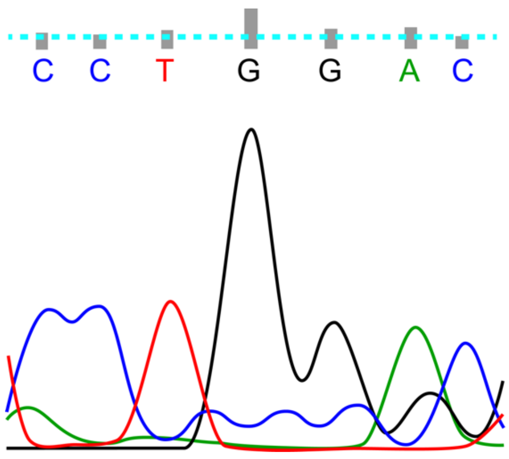


Fig. 1: Chromatogram view of a DNA sequence

This is an empirical process, since SNP identification is performed through the search of known patterns of DNA bases, here defined as masks. When a mask is found on a DNA sequence, the subsequent base is a SNP position. The problem is to analyze a large number of sequences and multiple masks manually. In this way, human error tends to be maximized. Thus, the development of computational methods to assist SNP analysis is essential. In other recent work, we have developed a method to perform search and analysis of known SNP positions. Nonetheless, this process becomes time expensive when the amount of data increases, because many iterations are required to perform the task.

Thus, in this study we propose a parallelization scheme for SNP analysis, using multithread programming, in order to improve its performance in the process of SNP verification.

This way, this work is organized as follows: In section *SNP Concepts*, we define SNP and show its relevance indifferent areas of biology. In section *Materials and Methods*, it can be seen the materials and methods used to develop this work. In section *Tests and Results* we show the tests performed with the tool and the obtained results. Finally, in section *Conclusion*, the conclusions are presented.

SNP Concepts

A SNP is defined as a single base change in a DNA sequence. The DNA sequence is a linear combination off our nucleotides: Adenine (A), Thymine (T), Cytosine (C) or Guanine (G). When comparing two DNA sequences, position by position, a SNP is defined as a presence of different nucleotides in the same position (Trick *et al*., 2009; Sachidanandam *et al*., 2001), as can be seen in Fig. 2. However, to be considered as a SNP position, this occurrence must be at least in 1% of analyzed population. Otherwise, this occurrence is defined as a mutation position (Rallón *et al*., 2010; Li and Sadler, 1991).

When a SNP position is detected, the subsequence immediately before of this position is observed. This subsequence can be classified as a pattern which always precedes a SNP occurrence. This specific pattern is called mask. Some specific ones are previously defined by research interests.

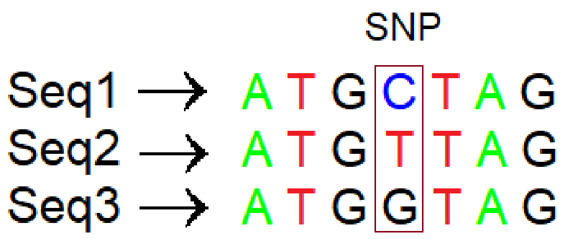


Fig. 2: SNP position

Moreover, the real importance of SNPs is their occurrence in large amount in the human genome. On average, every 1000 to 2000 bases throughout human DNA, approximately one nucleotide position differs between two genomes (Li and Sadler, 1991; Kwok *et al*., 1996). Considering that there are 3.2 billion nucleotides in the human genome, a complete genome can reach the magnitude of 1.6 million to 3.2 million SNPs.

In case of humans, many SNPs do not have any known effect on health or evolution yet. Nevertheless, some of these genetic variations are important in the study of human health. It has been shown that some SNPs may contribute to the individual predisposition to the response to certain drugs, in addition to susceptibility to environmental factors such as toxins and the risk of developing diseases (Harrow *et al*., 2012).

Many studies have focused their efforts on the SNPs research, as well as the search for their relationship with different types of diseases and their respective treatments, such as HIV (AIDS) (Westrop *et al*., 2017), HCV (Hepatitis C) (Elsedawy *et al*., 2016), obesity (Usher *et al*., 2015) and cancer (Kocarnik *et al*., 2015). However, it is important the aid of computational methods to perform these studies more efficiently, which simplifies the analysis and reduces the time to obtain the results.

There are methods that have different approaches to perform SNP detection, such as SNP detector (Zhang *et al*., 2005) and SNP Server (Savage *et al*., 2005). SNP detector uses sequence segments for nucleotide base calls and determines the quality of the files that were sequenced and in a later stage, it performs the alignment of these sequences using the Smith-Waterman algorithm (Smith and Waterman, 1981). Finally, it compares the alignment produced with a base alignment looking for positions of divergence, in this case, SNPs. However, SNP detector uses an external method, Phred (http://www.phrap.org/consed/consed.html\#howToGet), which has a license you must pay for use.

The SNP Server (Savage *et al*., 2005) is a real-time implementation of the auto SNP method (Barker *et al*., 2003), which implies the use of a web server to operate it. It is a web interface and a wrapper to three other programs: BLAST (Altschul *et al*., 1990; Rajendrakumar, 2015), CAP3 (Huang and Madan, 1999) and auto SNP. These ones are a SNP discovery pipeline as can be seen in Fig. 3. The pipeline accepts an input sequence and compares it to a database of previously specified nucleotide sequences, using BLAST algorithm to identify related sequences. The obtained sequences can be selected to an assembly (grouping) process using CAP3 and after performing the SNP discovery using auto SNP. The users must use input sequences in the FASTA format to proceed the assembly process, or sequences in the ACE format to skip this step.

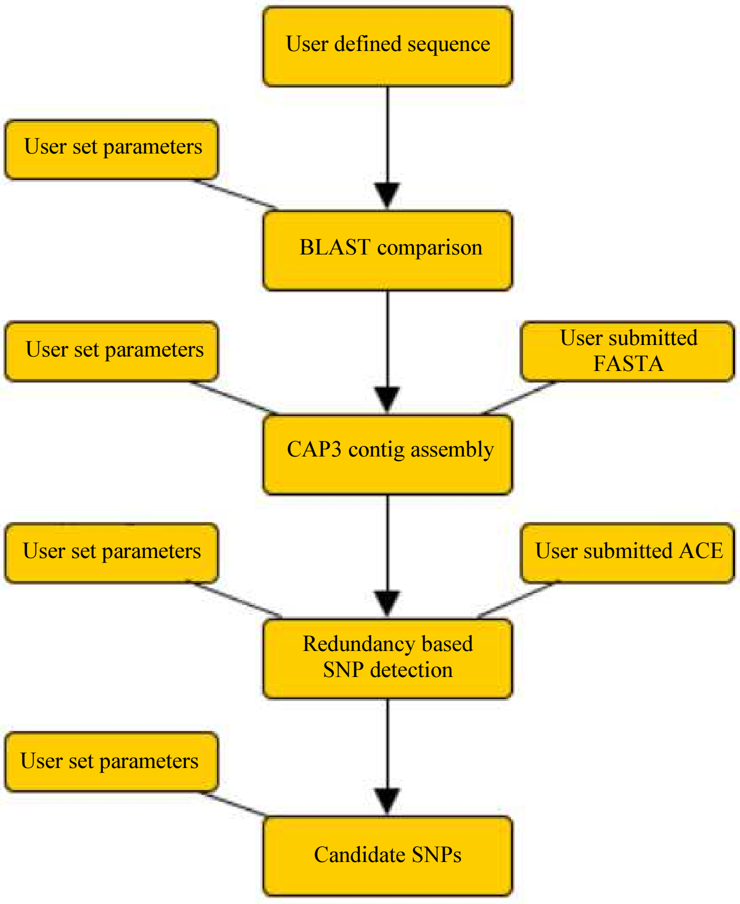


Fig. 3: An overview of components of the real-time auto SNP web server, the SNP Server (Savage *et al*., 2005)

There are also approaches that use parallel computing in the SNP analysis. As an example, the method ParDMETMiner (Agapito *et al*., 2016), which is based on a multithreaded Master-Slave architecture, where the Master is responsible for partitioning and distributing the data to each slave and collecting the results. The method is a parallelization of the DMET-Miner tool (Agapito *et al*., 2015), developed by Affymetrix, which allows the investigation of 1936 different nucleotides. These are candidates for possible SNPs in 255 genes that are related to the absorption, distribution, metabolism and excretion of the drugs.

In a similar way, SPRITE (Rengasamy and Madduri, 2016) is an open-source method that provides a parallel implementation of the genomic data analysis workflow for detection of SNPs. This software is divided into three parts: PRUNE for reading and sequence alignment, SAMPA for intermediate file processing and PARSNIP for parallel SNP lookup queries. In the parallelization of PARSNIP, the Message Passing Interface (MPI) library is used, dividing the number of sequences to be analyzed into the number of existing threads (Rengasamy and Madduri, 2015).

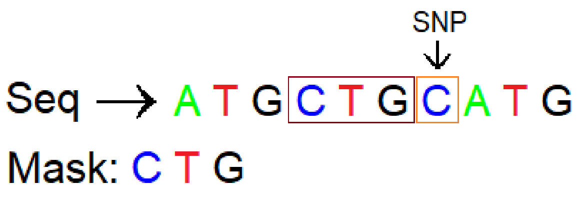


Fig. 4: Representation of a SNP search with mask

The methods previously presented do not use the mask strategy for SNP identification and analysis. Thus, our parallel method is capable to analyze multiple SNPs, unlike the other methods. It is important to stand out that our method does not search for unknown SNP positions, but it analyzes known SNP positions instead.

An efficient way to perform the analysis for SNPs is through masks, as shown in Fig. 4. This strategy consists in a definition of a subsequence to be found into target sequences and, after finding it, a subsequent position analysis will be performed where the subsequence was found.

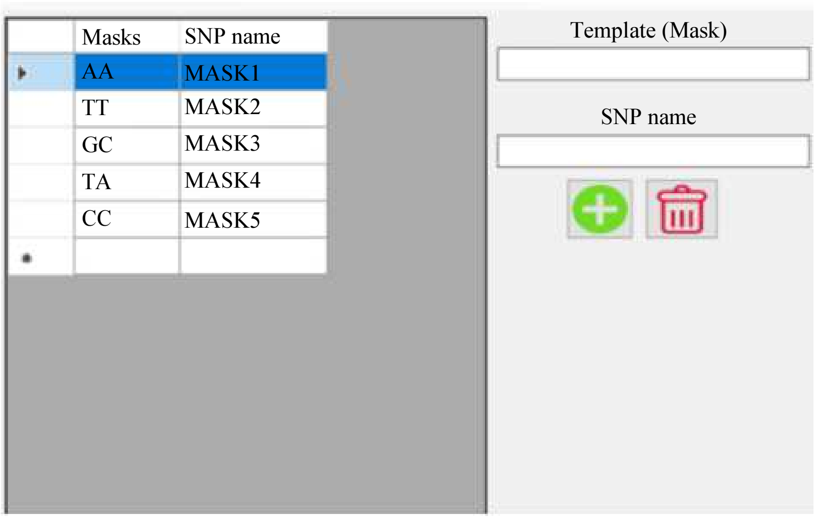
The enhancement of SNP search and analysis algorithms is desirable, considering the development and application of new sequencing technologies. Next Generation Sequencing produces huge amount of genomic data due its parallel sequencing approach, which generates hundreds of thousands of biological sequences (Liu *et al*., 2012). Thus, sequential approaches to SNP analysisbe come unfeasible due to the increasing of genomic data. This strategy can be performed in parallel given the nondependency of the data structures involved in the operation. Thus, it is not necessary blocking operations to access the sequences in a shared way.

Materials and Methods

SNP Verifier Method

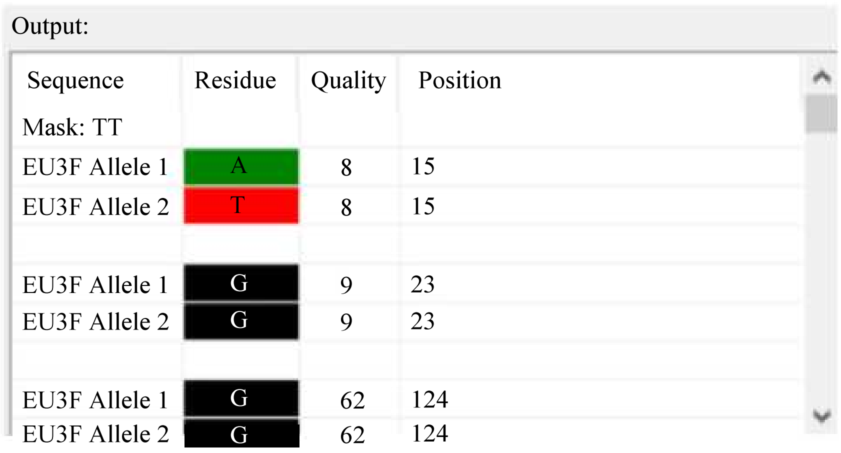
The SNP Verifier method, which we have implemented in other recent work, allows multi-mask sequential analysis of known SNPs. It receives *ab1* files as input data and convert them to *XML* files for analysis by means of tags, since the converted file provides the primary sequence, secondary sequence, sequencing quality and the peak values of the primary and secondary sequences. After the conversion process, the SNP identification and analysis are performed, using the masks previously inserted. When a SNP position is found, the occurrence of homozygous or heterozygous is evaluated, which will be better described in the subsection *Parallelization of SNP position analysis* and the result is finally stored.

As can be seen in the Fig. 5, the masks and a name for the SNP to be analyzed are given to the method. In the Fig. 6, we can observe the obtained results using the multi-mask identification and analysis. Nonetheless, as the number of sequences and masks to be analyzed grows, the sequential approach becomes unfeasible. Thus, it is necessary the use of some parallel approach to reduce the computational cost of the method, in order to reduce its execution time.



Masks	SNP name
AA	MASK1
TT	MASK2
GC	MASK3
TA	MASK4
CC	MASK5

Fig. 5: Insertion of masks on SNP verifier tool



Output:

Sequence	Residue	Quality	Position
Mask: TT			
EU3F Allele 1	A	8	15
EU3F Allele 2	T	8	15
EU3F Allele 1	G	9	23
EU3F Allele 2	G	9	23
EU3F Allele 1	G	62	124
EU3F Allele 2	G	62	124

Fig. 6: Results table on SNP verifier tool

Parallelization of the Masks Search

With our proposed parallel approach, we are able to verify many masks simultaneously. Thus, it allows performing multiple analysis simultaneously. The parallelization of this stage consists of assigning to each mask a different thread, so given N masks, it is assigned to them N threads. Finally, considering that a processor has K cores, it is called a $\frac{N}{K}$ threads per core, as can be seen in Fig. 7. As each task is terminated, the available core performs the next thread assigned to that core. This is done until all threads assigned to that core are performed. This way, the simultaneous analysis by means of masks is efficiently parallelized without impairing the analysis operation.

To implement our multithreaded method, we have used the *C#* language, since it is the same language used to develop the SNP Verifier method. The used library was *System.Threading.Task*, which contains the *Parallel* class that supports regions and loops that are executed in parallel (Okur and Dig, 2012).

Parallelization of SNP Position Analysis

After dividing the masks into the processor cores, we have the stage of analyzing the SNP positions. This analysis can be performed by extracting from the input file the values of the primary and secondary peaks of the sequences.

In order to differentiate a homozygous SNP (refers to an individual having identical alleles for a single trait) from a heterozygous (refers to an individual having two different alleles for a specific trait), the primary and secondary peak values from the *ab1* files, converted to *XML*, are evaluated. When the secondary peak has a value greater than or equal to 20% of the primary peak value, the SNP is considered to be heterozygous, as can be seen in the Fig. 8. If the value of the secondary peak value is less than 20% of the value of the primary peak, this SNP is classified as homozygous.

The parameter of 20% was based on that used by SNP detector (Zhang *et al*., 2005), since it is efficient in the detection of homozygous and heterozygous by attenuating the occurrence of false positives. However, it is important to notice that this value is a user-defined parameter.

To model this process in our parallel approach, we defined the steps that must be performed, as we can see in Fig. 9.

In this stage, we divide each SNP position found for a different thread. Thus, each thread is scheduled to a processor core, in order to parallelize the SNP analysis. Each thread, firstly, find the SNP position through its respective mask. Thus, the SNP position is analyzed based on its peak values, to check the homozygous or heterozygous occurrence. After this process, it is verified if the sequence under analysis has been completely investigated. If the end of the sequence is found, the process ends. Otherwise, the analysis of other positions of the same SNP will continue. It is important to notice that the higher the number of cores in the processor, the lower is the number of threads per core, so the analysis becomes faster.

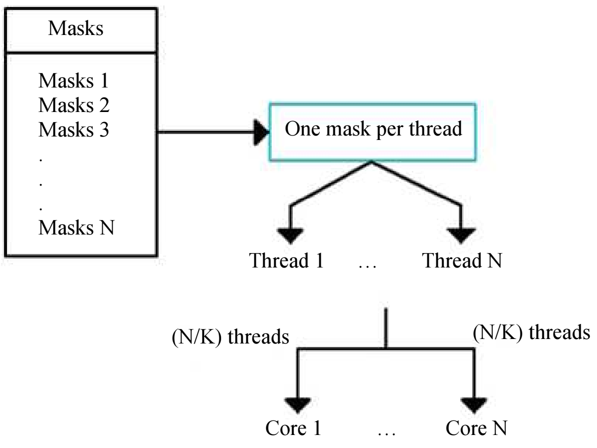


Fig. 7: Representation of our parallel SNP search with masks

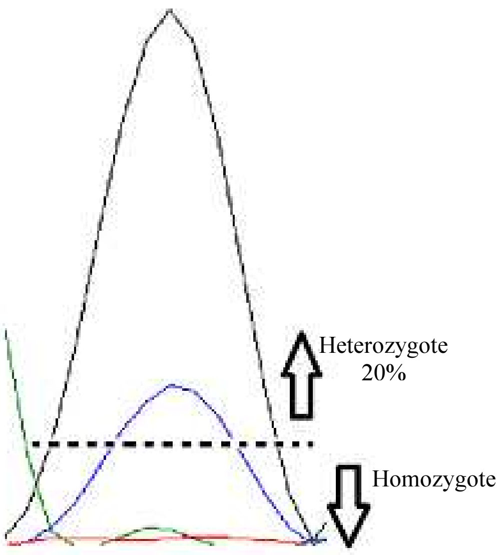


Fig. 8: Secondary peak value is greater than 20% of the primary peak value, thus it is heterozygote

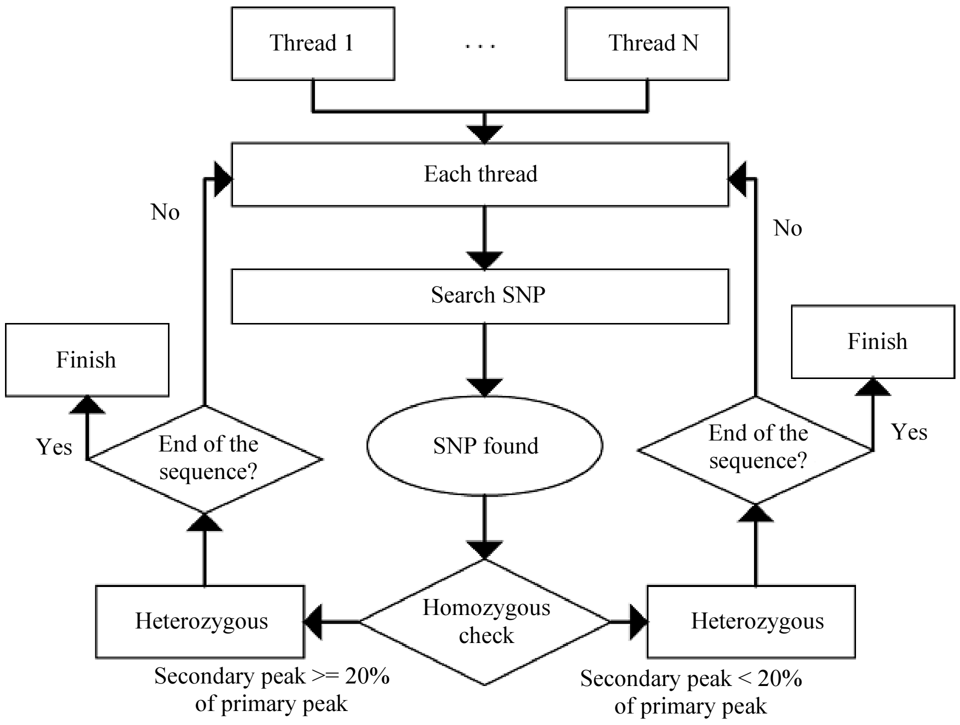


Fig. 9: Flowchart describing the analysis stage

Tests and Results

Test Platform

The tests were performed using a computer with Intel(R) Core (TM) i7-6700HQ CPU @ 2.60 GHz processor, with 32GB of RAM memory and Windows10 Home operating system.

Test Cases

The tests were run for the sequential SNP Verifier method and our parallel approach, in order to observe the obtained improvement. This method cannot be compared with the other methods presented in section *SNP Concepts*, since they perform detection of unknown SNP positions, while our method performs the analysis of known SNP positions, using the mask search approach. Hence, any comparison with other methods that do not use the mask approach is inappropriate. To the extent of our knowledge, there is not any other method similar to ours. Thus, the tests performed have the following structure:

- Test 1: 5 sequences
- Test 2: 10 sequences
- Test 3: 15 sequences
- Test 4: 20 sequences
- Test 5: 25 sequences
- Test 6: 30 sequences

Moreover, we used five different masks to perform the tests:

- AA
- TT
- GC
- TA
- CC

The sequences, which were used to perform the tests, have the length between 380 and 390 nucleotides and they were provided by Genomic Studies Laboratory (LEGO) at São Paulo State University (Unesp). The masks were selected considering their biological significance, based on previous studies. Finally, we have executed six test sets four times each to ensure statistical robustness and fidelity in the results. Considering a test set, each execution of it we have called round, so we have round 1 (R1), round 2 (R2), round 3 (R3) and round 4 (R4). Basically, the difference among the test sets is the number of sequences in each one, considering 5, 10, 15, 20, 25 and 30 sequences, respectively. Finally, another relevant issue to be explained here concerns the mask length, where we have chosen all masks with length two. This decision was related to the length of the masks, because it implies directly in the processing time. As it is more common to find masks with small lengths than masks with high lengths, with this kind of test, the algorithm can amplify the stress on the processors, where we can improve the performance measure.

Results

In the Table 1, we show the execution times of all rounds, in seconds, for sequential and parallel approaches. To show the statistical consistency of these results, in the Table 2 can be seen the execution time averages and standard deviations of executed tests. Moreover, still in the Table 2, we present the sequential and parallel standard deviation averages.

It can be noticed in Table 1 the considerable difference between the obtained execution times for sequential and parallel approaches. Considering the execution time average of all test cases, both for sequential and parallel, we can verify that the parallel version wastes, on average, just 0.18% of the time of sequential one. From these results, it is possible to verify the huge performance improvement of the parallel approach for all test cases from 5 to 30 nucleotide sequences. Another important consideration about the results is concerning the standard deviations presented in Table 2, where the standard deviations of parallel approach are lower than sequential one, which indicates higher robustness and regularity of the parallel one. When a comparative analysis is performed between the approaches, it can be noticed that the average of standard deviations is 0.0219 and 0.59, for parallel and sequential one, respectively.

Still concerning the analysis of the execution time, it is important to show as the data volume increases the execution time presents a linear growth for both sequential and parallel approaches. This can be verified in the Fig. 10 and 11, respectively.

Table 1: Sequential and parallel approach tests

	N of seqs.	R1	R2	R3	R4
Sequential	5	55.39	54.06	54.15	53.89
	10	136.70	134.50	135.30	134.30
	15	199.39	199.07	198.67	198.74
	20	365.10	363.90	363.40	363.80
	25	406.10	407.30	407.17	406.38
	30	563.57	562.05	562.57	563.34
Parallel	5	0.20	0.22	0.21	0.21
	10	0.28	0.30	0.31	0.31
	15	0.36	0.39	0.39	0.41
	20	0.45	0.52	0.50	0.49
	25	0.51	0.59	0.60	0.58
	30	0.61	0.68	0.70	0.68

Table 2: Sequential and parallel approach statistics

	N of seqs.	Average	Standard deviation
Sequential	5	54.37	0.60
	10	135.20	0.94
	15	198.96	0.28
	20	364.05	0.63
	25	406.73	0.50
	30	562.88	0.60
Average			0.59
Parallel	5	0.21	0.0069
	10	0.30	0.0119
	15	0.38	0.0161
	20	0.49	0.0258
	25	0.57	0.0352
	30	0.67	0.0357
Average			0.0219

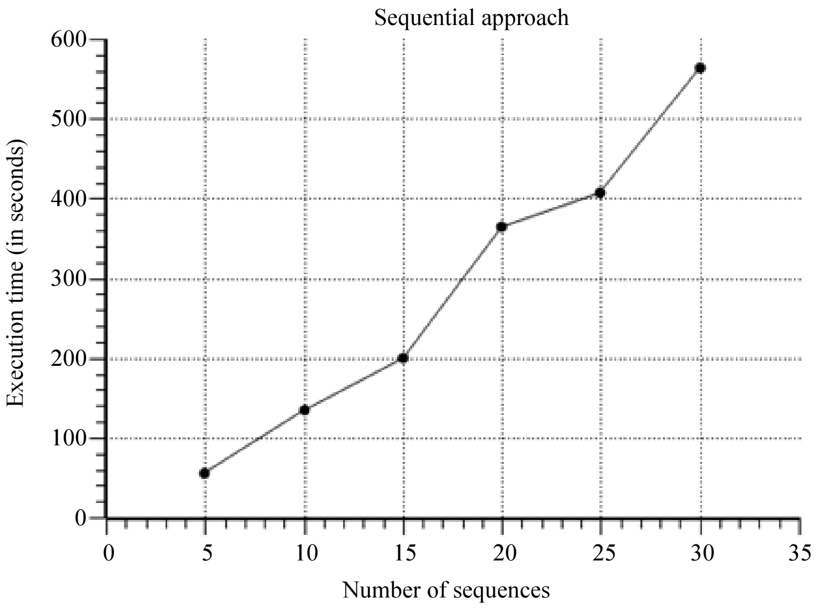


Fig. 10: Average execution times of sequential approach

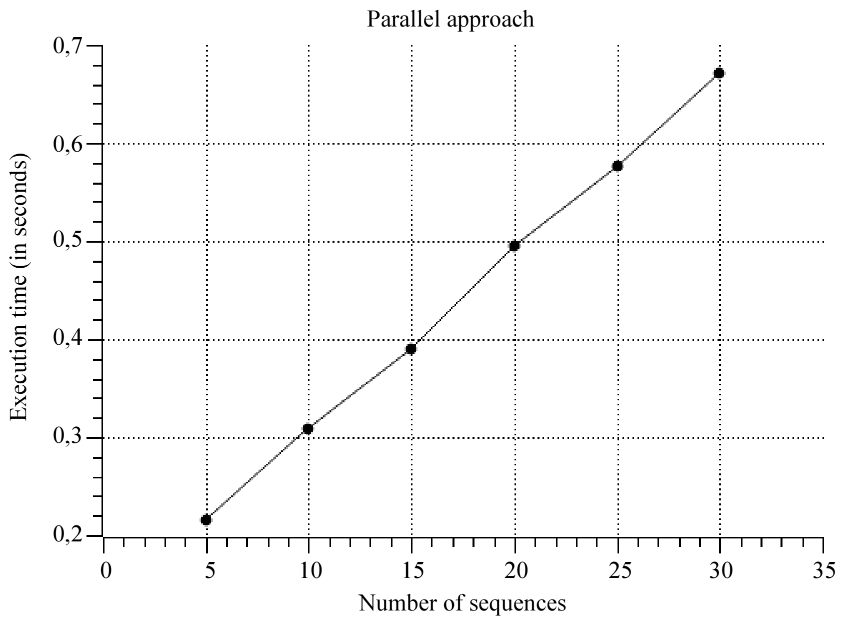


Fig. 11: Average execution times of Parallel Approach

In the Fig. 10, with the analysis of linear regression coefficient R^2, we have obtained the Equation (1):

$$y = 100.64x - 65.19 \tag{1}$$

andR^2 = 0.9776 for sequential approach.

In the Fig. 11, with the same analysis of linear regression coefficient R^2, we have obtained the Equation (2):

$$y = 0.0909x + 0.1242 \tag{2}$$

And R^2 = 0.9992 for parallel one. The linear regression coefficient demonstrates how the proposed model fits to the linear model. As closer value 1 the coefficient is, the proposed model is more adequate to the ideal one. Thus, we can verify that the parallel approach fits better to the linear model than the sequential one, because its coefficient reaches the value of 0.9992, while the sequential is 0.9776. Thus, it is possible to notice the stability of the parallel approach with this linear growth.

Finally, another issue of performance analysis of parallel programs is the speedup (Eager *et al*., 1989). Speedup is defined as the portion of time to execute a program with one processor (sequential execution) to time to execute when p processors are available. Thus, the speedup is considered as Equation 3:

$$S(p) = \frac{T(1)}{T(p)} \tag{3}$$

Where:
$T(1)$: Execution time with one processor
$T(p)$: Execution time with p processors

In our tests, we have used five processing cores. As we considered five masks and their division into five threads, we have allocated five processing cores (Fig. 7). Thus, the average speedup obtained is:

$$S(5) = \left(\begin{array}{l} \frac{54.37}{0.21} + \frac{135.20}{0.30} + \frac{198.96}{0.38} \\ + \frac{364.05}{0.49} + \frac{406.73}{0.57} + \frac{562.88}{0.67} \end{array} \right) / 6 = 98.05 \tag{4}$$

Therefore, it is possible to notice that, on the average, the parallel approach improved 98.05 times in relation to the sequential one.

Conclusion

In this study, we have proposed a parallelization scheme to the SNP Verifier method using multithread programming. We can conclude that our work improved the performance of the analysis of known SNP positions, in terms of execution time. This fact allows the users to perform SNP analysis in a feasible time, even with huge amount of genomic data.

With the tests, we can observe great performance improvement of our parallel method, for all test cases from 5 to 30 nucleotides. The speedup shows that the performance of our multithreaded approach is 98.05 times faster than the sequential one. Moreover, it can be seen that our multithreaded method is more statistically robust because the calculated average standard deviation is considerably smaller than the value obtained by the sequential approach.

Acknowledgement

The authors would like to thank Dr. Livia Rossi and Dr. Gilberto Vaughan from Centers for Disease Control and Prevention (CDC – USA) for the collaboration in the research development and Universidade Paulista (Unip/ICET) to partially support this research under grant number 7-02-1053/2017. The authors would like to thank FAPERP (Fundação de Apoio à Pesquisa e Extensão deSão José do Rio Preto).

Author's Contributions

Geraldo Francisco Donegá Zafalon: Developed and implemented the strategies, analyzed the results and wrote the paper.

Álvaro Magri Nogueira da Cruz: Developed and implemented the strategies and wrote the paper.

Anderson Rici Amorim: Developed and implemented the strategies.

Matheus Carreira Andrade: Helped in the improvement of the model and analyzed the results.

Allan de Godoi Contessoto: Helped in the improvement of the interface and analyzed the results.

Leandro Alves Neves: Helped in the improvement of the interface and analyzed the results.

Rogéria Cristiane Gratão de Souza: Helped in the improvement of the model and analyzed the results.

Carlos Roberto Valêncio: Helped in the improvement of the model, evaluated the data and analyzed the results.

Liria Matsumoto Sato: Helped in the improvement of the parallel model, evaluated the data and analyzed the results.

References

Agapito, G., P.H. Guzzi and M. Cannataro, 2016.Parallel processing of genomics data. Proceedings of the 2nd International Conference on Numerical Computations: Theory and Algorithms, (CTA' 16). DOI: 10.1063/1.4965364

Agapito, G., P.H. Guzzi and M. Cannatro, 2015. Dmetminer: Efficient discovery of association rules from pharmacogenomic data. J. Biomed. Inform., 56: 273-283. DOI:10.1016/j.jbi.2015.06.005

Altschul, S.F., W. Gish, W. Miller, E.W., Myers and D.J. Lipman, 1990. Basic local alignment search tool. J. Molecular Biol., 215: 403-410. DOI: 10.1016/S0022-2836(05)80360-2

Amorim, A.R., J.M.V. Visotaky, A.D.G. Contessoto, L.A. Neves and R.C.G. De Souza *et al*., 2016. Performance improvement of genetic algorithm for multiple sequence alignment. Proceedings of the 17th International Conference on Parallel and Distributed Computing, Applications and Technologies, IEEE Xplore Press, pp: 69-72. DOI: 10.1109/PDCAT.2016.029

Barker, G., J. Batley, H. OSullivan, K.J. Edwards and D. Edwards, 2003. Redundancy based detection of sequence polymorphisms in expressed sequence tag data using autosnp. Bioinformatics, 19: 421-422. DOI: 10.1093/bioinformatics/btf881

Eager, D.L., J. Zahorjan and E.D. Lazowska, 1989. Speedup versus efficiency in parallel systems. IEEE Trans. Comput., 38: 408-423. DOI: 10.1109/12.21127

Elsedawy, Y.S., M.A. Khattab, S.A. El Hady, A.A. El Sayed and A.M. Albreed *et al*., 2016. Single nucleotide polymorphisms of toll-like receptor 7 in hepatitis c virus infection and hepatocellular carcinoma patients. Egypt. J. Med. Microbiol., 25: 428-34. DOI: 10.3349/ymj.2014.55.2.428

Hemalatha, M. and K. Vivekanandan, 2008. Genetic algorithm based probabilistic motif discovery in unaligned biological sequences. J. Comput. Sci., 4: 625-630. DOI: 10.3844/jcssp.2008.625.630

Harrow, J., A. Frankish, J.M. Gonzalez, E. Tapanari and M. Diekhans *et al*., 2012. Gencode: The reference human genome annotation for the encode project. Genome Res., 22: 1760-1774. DOI: 10.1101/gr.135350.111

Huang, X. and A. Madan, 1999. Cap3: A DNA sequence assembly program. Genome Res., 9: 868-877. DOI: 10.1101/gr.9.9.868

Khuri, S., 2008. A bioinformatics track in computer science. ACM SIGCSE Bull., 40: 508-512. DOI: 10.1145/1352322.1352305

Kocarnik, J.M., S.L. Park, J. Han, L. Dumitrescu and I. Cheng *et al*., 2015. Pleiotropic and sex-specific effects of cancer was SNPS on melanoma risk in the population architecture using genomics and epidemiology (page) study. PloS One, 10: e0120491-e0120491. DOI: 10.1371/journal.pone.0120491

Kwok, P.Y., Q. Deng, H. Zakeri, S.L. Taylor and D.A. Nickerson, 1996. Increasing the information content of STS-based genome maps: Identifying polymorphisms in mapped STSS. Genomics, 31: 123-126. DOI: 10.1006/geno.1996.0019

Li, W.H. and L.A. Sadler, 1991. Low nucleotide diversity in man. Genetics, 129: 513-523.

Liu, L., Y. Li, S. Li, N. Hu and Y. He *et al.*, 2012. Comparison of next generation sequencing systems. BioMed Res. Int., 2012: 251364-251364. DOI: 10.1155/2012/251364

Margulies, M., M. Egholm, W.E. Altman, S. Attiya and J.S. Bader *et al.*, 2005. Genome sequencing in microfabricated high-density picolitrereactors. Nature, 437: 376-380. DOI: 10.1038/nature04774

Marucci, E.A., G.F. Zafalon, J.C. Momente, L.A. Neves and C.R. Valêncio *et al.*, 2014.An efficient parallel algorithm for multiple sequence similarities calculation using a low complexity method. BioMed Res. Int., 2014: 563016-563016. DOI: 10.1155/2014/563016

Okur, S. and D. Dig, 2012. How do developers use parallel libraries? Proceedings of the ACM SIGSOFT 20th International Symposium on the Foundations of Software Engineering, Nov. 11-16, ACM, Cary, North Carolina, pp: 54-54. DOI: 10.1145/2393596.2393660

Rajendrakumar, P., 2015. Molecular Marker Development Using Bioinformatic Tools. In: Sorghum Molecular Breeding, Madhusudhana, R., P. Rajendrakumar and J. Patil (Eds.), Springer, pp: 179-195.

Rallón, N.I., S. Naggie, J.M. Benito, J. Medrano and C. Restrepo *et al.*, 2010. Association of a single nucleotide polymorphism near the interleukin-28b gene with response to hepatitis c therapy in HIV/hepatitis c virus-coinfectedpatients. Aids, 24: F23-F29. DOI: 10.1097/QAD.0b013e3283391d6d

Rengasamy, V. and K. Madduri, 2015. Engineering a high-performance snp detection pipeline. Technical Report.

Rengasamy, V. and K. Madduri, 2016. Sprite: A fast parallel SNP detection pipeline. Proceedings of the 31st International Conference, ISC High Performance, Jun. 19-23, Springer, Frankfurt, Germany, pp: 159-177. DOI: 10.1007/978-3-319-41321-1

Sachidanandam, R., D. Weissman, S.C. Schmidt, J.M. Kakol and L.D. Stein *et al.*, 2001. A map of human genome sequence variation containing 1.42 million single nucleotide polymorphisms. Nature, 409: 928-933. DOI: 10.1038/35057149

Sanger, F. and A.R. Coulson, 1975. A rapid method for determining sequences in DNA by primed synthesis with DNA polymerase. J. Molecular Biol., 94: 441IN19447-446IN20448. DOI: 10.1016/0022-2836(75)90213-2

Savage, D., J. Batley, T. Erwin, E. Logan and C.G. Love *et al.*, 2005. SNP server: A real-time SNP discovery tool. Nucleic Acids Res., 33: W493-W495. DOI: 10.1093/nar/gki462

Smith, T.F. and M.S. Waterman, 1981. Identification of common molecular subsequences. J. Molecular Biol., 147: 195-197. DOI: 10.1016/0022-2836(81)90087-5

Trick, M., Y. Long, J. Meng and I. Bancroft, 2009. Single Nucleotide Polymorphism (SNP) discovery in the polyploid brassica napus using solexa transcriptome sequencing. Plant Biotechnol. J., 7: 334-346. DOI: 10.1111/j.1467-7652.2008.00396.x

Usher, C.L., R.E. Handsaker, T. Esko, M.A. Tuke and M.N. Weedon *et al.*, 2015. Structural forms of the human amylase locus and their relationships to SNPS, haplotypes and obesity. Nature Genet., 47: 921-925. DOI: 10.1038/ng.3340

Wang, L., Y. Wang and Q. Chang, 2016. Feature selection methods for big data bioinformatics: A survey from the search perspective. Methods, 111: 21-31. DOI: 10.1016/j.ymeth.2016.08.014

Westrop, S., A. Cocker, A. Boasso, A. Sullivan and M. Nelson *et al.*, 2017. Enrichment of HLA types and SNP associated with non-progression in a strictly defined cohort of hiv-1 controllers. Frontiers Immunol., 8: 746-746. DOI: 10.3389/fimmu.2017.00746

Zafalon, G., J. Visotaky, A. Amorim, C. Valêncio and L. Neves *et al.*, 2015. A parallel approach of coffee objective function to multiple sequence alignment. J. Phys. DOI: 10.1088/1742-6596/633/1/012084

Zhang, J., D.A. Wheeler, I. Yakub, S. Wei and R. Sood *et al.*, 2005. SNP detector: A software tool for sensitive and accurate SNP detection. PLoS Comput. Biol., 1: e53-e53. DOI: 10.1371/journal.pcbi.0010053