Muntha et al., J Comput Sci Syst Biol 2017, 10:5
DOI: 10.4172/jcsb.1000262

Journal of
Computer Science & Systems Biology

Research Article Open Access

Comparative Analysis of Automatic Parallelization Techniques

Muntha SR, Prasad A, Gogineni K*, Nikhil L and Harshavardhan VL
VIT University, Vellore, Tamil Nadu, India

Abstract

With multicore design emerging as the chief design pattern for the microprocessor industry in a major way constant efforts are being made to improve its performance. The only way to extract maximum performance from a multicore based system as of now is through the automatic extraction of threads from sequential applications with the assistance of tools including, compilers and runtime optimizers. The increasing popularity of clusters and other similar forms of distributed computing calls for the requirement of automatic parallelization in compilers. This requirement for techniques through which automatic parallelization can be achieved has set of the development of various different methods with varying efficiencies. The objective of this paper is to test the efficiencies of two such techniques by performing a comparative analysis of the techniques based on various criteria like time complexity for instance. The techniques we are going to work on are namely the Scalar and Array Analysis technique and the Smith waterman technique. As mentioned previously these techniques will be analyzed using various criteria to arrive at a conclusion as to which one of them is superior performance wise.

Keywords: Parallel computing; Scalar analysis; Array analysis; Smith Waterman technique; Dynamic algorithm; Sequence alignment; Processing speed; Improved efficiency

Introduction

The intention of this project is to conduct a comparative analysis of the performance of two algorithms namely the "scalar and array analysis algorithm" and the "smith waterman algorithm". The origin of these algorithms can be traced back to two completely different fields, the former was derived from the fundamental principles of data structures and the later was derived from the fundamental principles used for gene sequencing but, both these techniques can be used as effective solutions for enhancing the field of automatic parallelization. Before discussion about the working and the performance of these algorithms here is a brief introduction on how these techniques might work.

Literature Review

Here are a few accounts of work done on this subject previously. These researchers have performed analyses on these algorithms using different parameter in order to gain more knowledge and different perspectives that could be associated with the algorithms. They also deal with the applications and use of these algorithms in the field of parallelization. This paper presents you with an algorithm which is capable of performing a detailed analysis.

Along which a given number of values flow. The give program is explained very simply in an imperative language. The iteration vector of the a given referencing statement is a function of the result which is the name of the statement which acts as a pre-cursor, this applies to any array or scalar function. The objective of the paper is to discuss the various methods of application which includes using array and scalar analysis for program verification. It aims on reconstructing and optimizing a compiler so that it can do the work effectively [1].

This paper deals with the survey conducted on the recorded developments done so far. It aims on providing very detailed and holistic information about the already existing approaches of alignment and sequencing of genes by applying the smith waterman algorithm. It also provides an account on the strengths and weaknesses of the implementation of this algorithm. It also provides details about how the older approaches are different than the existing approaches. It also provides some possible solutions to optimize the smith waterman algorithm, accelerate its functioning and make it much more effective [2].

This paper aims on provides an understanding on how process of parallelization is similar to the local sequence alignment and how smith waterman algorithm can be used in the filed o parallel computing. It also deals with methods to execute the smith waterman algorithm in a very effective and quicker and subjects a few simple yet effective ways to do so. One such method would include using the a GUP instead of a CPU to cope with the problem of the constrained speed of accessing the memory ("it is supposed that using GPUs in parallel for execution of the algorithm may speed up the process by 45 times approximately") [3].

This paper sheds light on how to use the approach of parallel programing to calculate the cell values in matrixes which are used in smith waterman algorithm while performing sequence alignment. It deals with the application of this bioinformatics based algorithm and also highlights the speed and efficiency concerns that might be caused by implementing the algorithm on a serial commuter. It focuses on the segments of algorithms which can possibly be elevated using techniques related to parallel computing. The mainly examine this approach using "Open MP" and "Cuda C" [4].

This paper highlights the increasing role of parallelization in the world of computing. It discusses about various different techniques that were used in parallelization in the past and compares them to the ones that are being used presently. It chiefly discusses in detail about one such algorithm used for parallelization i.e., the scalar analysis and array analysis. It discusses in detail how these techniques go hand in hand to analyze a given program. It also sheds some light on the working, advantages and disadvantages of the algorithm [5].

***Corresponding author:** Gogineni K, VIT University, Vellore, Tamil Nadu, India, Tel: 9249433595; E-mail: gkailashnath1998@gmail.com

Received November 10, 2017; **Accepted** November 15, 2017; **Published** November 18, 2017

Citation: Muntha SR, Prasad A, Gogineni K, Nikhil L, Harshavardhan VL (2017) Comparative Analysis of Automatic Parallelization Techniques. J Comput Sci Syst Biol 10: 126-129. doi:10.4172/jcsb.1000262

Scalar and Array Analysis

Scalar analysis

This analysis is actually a combination of two different types of analysis that work to gather in order to analyses and infiltrate data. The function of scalar analysis is to dissect the program and subsequently analyse variables of scalar nature and how they are inter dependent on each other. These dependencies can be defined using a quotation by the Hall et al. [5] which describes it as being the situation where a memory location written on one iteration of a loop is accessed (read or write) on a different iteration.

This analysis can also be used to check the interrelations between the array elements and their respective indices. This analysis also known as the scalar symbolic analysis is conducted by changing the available indices into simple and easily deducible equations that express the index pertaining to an array. This simplification makes the problem a very simple integer based programing problem, this problem as many possible solutions but the one which is more time efficient is chosen. One major con of this problem is that it can be applied to programs with a specific type of construct an example of this situation is that if a linear approach is used to access an array through an equation which acts as a multi-dimensional array then the analysis may fail to execute.

Array analysis

Array analysis can be considered as the counter part to the scalar analysis. Array based analysis to detect "privatize -able arrays" is one of the many methods of array analysis. The method used to allot a replica of the finished or working part of a given array every parallel instance which then further refers to it as the data without any intermediate dependencies for the portion in question is known as "privatization". The possibility of an array being privatized can only be determined after an equation to access the array after analysis.

The loop has to undergo a major transformation so as to be capable of parallelize a given segment of code. In case there is any discrepancy in the process of transformation, the whole analysis would fail which would result in the failure of the parallelization of the fore mentioned segment of the code. These analyses are the two most effective and power full tools that can be used to parallelize a given code based on analysis of ("scalar and array variables").

The Smith Waterman Algorithms

Smith water man algorithm was initially generated for finding the similarities between two DNA sequences in order to derive the evolutionary relationship between them. This algorithm searches throughout and along the DNA in order to find similarities between two sequences. This alignment of the sequences occurs at a small level in part by part fashion sequentially, this kind of alignment known as the "local alignment" is considered the best way to find all the variations and similarities between a couple of sequences. So, basically the function of the smith-water man algorithm is to optimally align the sequences.

Optimal local alignment

The process of comparing the given query sequence to the sequences available in the database at a character to character level is known as Optimal local alignment. The smith waterman algorithm is entirely dynamic in function and unlike Needleman-Wunsch algorithm which is used for global alignment (and is considered a precursor to "Smith-Waterman") this algorithm does alignment in a local fashion so that every small part of the DNA is optimized similarly. Since it is based on dynamic programing this algorithm works by dividing and sub dividing the problem and then finally putting together the solution of all these sub problems to attain a final answer covering the whole entire problem. Implementing this form of dynamic programing, the Smith-Waterman algorithm finds the optimized alignment of the given sequences, which might be staring and ending at any specific location in the sequences. The highlights of the Smith-Waterman Algorithm approach is:

- Same level of sequencing occurs at a given sub level.
- Used majorly by large bioinformatics based industries.
- Since the approach is dynamic the algorithm can be used to optimally sequence the sequences at every minor level.

Detailed Account of the Approach followed by the Algorithms

Scalar and array analysis

Scalar analysis: This analysis is very specific and only applicable to a certain niche program constructs. This analysis is generally used in synchrony with "array analysis", they are generally applied together so as to ensure complete productivity and efficiency. What basically happens in this analysis is that a program is broken down on order to analyses the inter dependencies that exists between scalar vectors. A dependency can be defined using a direct quotation by Hall et al. [5] ("when a memory location written on one iteration of a loop is accessed (read or write) on a different iteration").

In this analysis array elements can be checked using their respective unique indices, it is also commonly known as "scalar symbolic analysis, pertaining to the fact that is performed by manipulating and transforming the give indices of the array so that they become more simplified and solvable. This would basically mean that the problem becomes much less cumbersome to deal with and also becomes multi-dimensional making it more time saving.

Array analysis: The counter analysis is performed by the array based analysis. There are various ways to perform this analysis, the most popular one being analyzing the data in an array to detect the privatisable ones. Privatization can be defined as a method where each parallel instance is provided with a complete and exact replica of the functioning segment of an array so that it would reference it as the data with no dependencies to carry forward, which might be linked to that particular segment. The possibility of privatization of the array might be questionable, whether or not it can be done can only be determined if there is a particular equation that might get you to access the array.

Algorithm

Step 1: Scalar analysis divides the program into pieces to analyze scalar variables used in it.

Step 2: It sees for the dependencies, it does so by transforming indices to computable affine equations which gives the indices of array.

Step 3: Finds out the code segments which can be parallelized with the above dependence complications.

Step 4: The parts that aren't found to be parallelizable are left out for array analysis to make them parallelizable. This works over array data to find parallelizable arrays.

Step 5: Otherwise they won't parallelize (Figure 1).

Smith waterman algorithm

Initially this algorithm was designed to be applied in the field of bioinformatics but due to its dynamic nature and ability to improve performance it has found application in the field of computer science more specifically in the parallelization sector. Even though it is used to determine the alignment and similarity within sequences it can be used in high end GPU units, graphic processing in desktops and can also be used to deliver exceptional computational capabilities. They have been recently been exploited for the general purpose of computation by the "CUDA" programming environment on "Nvidia" GPUs and ATI based stream computation method on ("ATI based CPU'S").

It is based on an execution methodology which is based on parallel analysis of two strings. It provides a major boost in the execution time almost twice that of a regular algorithm. It is capable of reducing the time complexity from O(mn) to O(m+n) in case of two actively participating strings and form O(mnk) to O(m+nk) in case of multiple sequence alignment. This algorithm provides a framework that is capable of reducing time without compromising the sensitivity of the crucial tasks.

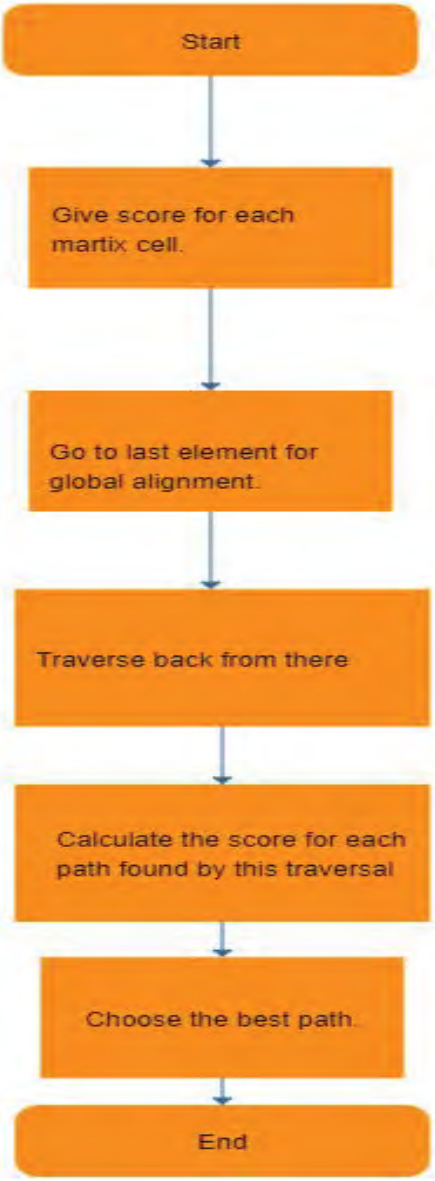

Figure 1: Yed representation of the algorithm.

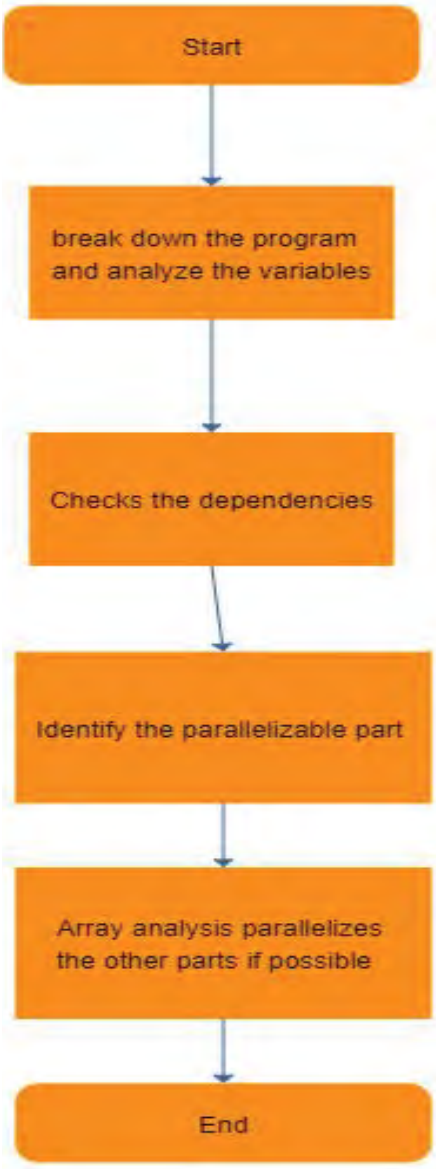

Figure 2: Yed representation of the algorithm.

Since it is a dynamic approach it goes on by working on small parts and further amalgamating them in order to get a result which is accurate in a very short period of time.

Algorithm

- Give scores to each of the cell in the matrix according to the formula.

- Once the scores are given go to the element in last row and last column for global alignment and for the cell having highest value for local alignment.
- From this selected element traverse to the previous cell that led to the current value of this cell.
- You will reach an end point while traversing. Whenever there is an arrow from right to left insert a gap in the lower strand and vice versa. Whenever it is going diagonally match the upper and lower strand.
- Then calculate the score obtained from the score table.
- If two alignments are obtained the one with the least score is optimal. Formula:

F=Max{H(x-1, y-1)+S, H(x-1, y)+G, H(x, y-1)+G}

Where, G is gap penalty; H is the value at the specified position; S is the score at the cell.

General scoring set values:

S is:

G=-2(For mis-match).

S=+2(For match).

G=-1(For insertion/deletion) (Figure 2).

Conclusion

So, we can finally conclude that the smith-waterman algorithm is a more effective and economical solution to increase the efficiency and reduce the time consumption in the field of parallelization. This algorithm follows an approach resembling the butterfly effect which basically means that it acts on the sub units and then eventually progresses to the higher levels whilst adding up the resultant values so that the final outcome could be reached very quickly and also free of any sort of errors or discrepancies. Dividing a given task into smaller sub parts would reduce the time consumption by a landslide, a task which generally would take a long time can now be completed in almost a fractional amount of the time without the quality of the outcome being compromised.

Apart from that this technique has a very easy operation principle; it follows the dynamic approach of comparative analysis at a local level which would mean the program would have little to know errors. It works on the basis of simple mathematical concepts. One other thing to be note here is the flexibility of the algorithm. It can be applied in various different fields to simplify and speed up work. There are almost no assumptions required while implementing this algorithm which would give a proper leverage to the user, important parameters can be calculated certainly and exactness. All in all we could also say that implementation of this algorithm would make processes of parallelization quicker, accurate and easier.

References

1. Feautrier P (1191) Dataflow Analysis of Array and Scalar References. Inter J Para Program 20: 23-53.
2. Prabhu AG, Aithal G (2014) Automatic Parallelization for Parallel Architectures Using Smith Waterman Algorithm - Literature Review. Int J Eng Inven 3: 1-11.
3. Khajeh-Saeed A, Poole S, Perot JB (2010) Acceleration of the Smith-Waterman algorithm using single and multiple graphics processors. J Comp Phy 229: 4247-4258.
4. Chaibou A, Sie O (2015) Comparative Study of the Parallelization of the Smith-Waterman Algorithm on OpenMP and Cuda C. J Comp Commu 6: 107-117.
5. DiPasquale N, Gehlot V, Way T (2005) Proceedings of MASPLAS'05 Mid-Atlantic Student Workshop on Programming Languages and Systems. University of Delaware, USA.