International Journal of Research in Engineering, Science and Management
Volume-2, Issue-4, April-2019
www.ijresm.com | ISSN (Online): 2581-5792

Compiler Optimization and Performance Enhancement using Parallelism

Ravi Kumar[1], M. Rajath[2], S. N. Pratheek[3], A. Parkavi[4]
[1,2,3]*Student, Department of CSE, M. S. Ramaiah Institute of Technology, Bengaluru, India*
[4]*Assistant Professor, Department of CSE, M. S. Ramaiah Institute of Technology, Bengaluru, India*

***Abstract*: In this paper, we are going to describe the most important feature of the compiler that is parallelism. We have collected information about parallelism used in various fields of compiler architecture, fields like parallelism compiler optimization and their impact on the architecture design, JavaScript, embedded systems, Instruction-level Parallelism, etc. We have gone through several topics, which are co-related to the parallelism and its application on the real world; In safety critical system there should be need of parallel computation for the performance and the efficiency of the system in their workload by introducing the Ada. We are talking about parallelism in compilers and the steps we take to achieve parallelism in our discussion compilers are used in various domains of computer science and engineering our goal is to make the compilers more advanced and capable of performing better under any given situation. Parallelism is basically the process of performing several tasks at once simultaneously. Compilers are can be used to achieve parallel parsing of multiple instructions at the same time so our task is to implement this parallelism Architecture in compilers.**

***Keywords*: Ada, compiler, GPGPU, JavaScript, OpenMP, Parallelism, Polychronopoulos, SIMD, SUIF Quantum computing (QC).**

1. Introduction

Parallel processor in embedded systems are efficient and have potential for reducing computation time in the execution of the large numeric intensive real world application; meaning that parallel processor will have the capability to execute more number of instruction in less time at the same level. More computations can be performed simultaneously; parallelism is preferred over the thread based concept so it's quite useful because of the thread applications, lightweight concepts; nowadays increasing need for computational capabilities is handled well by the web browser, related to this JavaScript is the one of the most widely used programming language; we have gone through the General purpose of the Graphics processing units, how the graphics are correlated to the parallelism concepts, we have seen some more acceptable parameter for the graphics;

Parallelism- the architecture which emphasizes on simultaneous execution of stuff is our prime focus in this paper We have taken into account several parameters while implementing this feature onto the compilers starting from their basic architecture to the entire system on which it is built upon and the internal working of the compilers as well The phase wise analysis is done and we use it to bring out parallelism as a result we have to design our compilers to support this feature throughout the paper we have discussed how we have done this to obtain efficient computing machines in the real world scenario hence our primary focus is on improving the performance of compilers enhancing its computational capabilities.

2. Overview

The parallel processor will reduce the execution time of the large complex applications. Simultaneous execution of the programs and get the desired results with the parallel processor. Traditionally very small amount of parallelism will be introduced because of the problem of overhead, in order to overcome this problem and make the processor to compile and execute several programs to obtain desired results. typically the overhead problems include -synchronization, communication, scheduling activities hanging up over simultaneous execution of the programs and extra delays. So we need to develop the compiler which is capable of supporting parallelism and high synchronization. Some most recent researches are going on these topics for the detection of parallelism, develop parallel process, architectural design, handle the error, introduce the multiple loops.

Parallel computation in the Ada to manage with the performance parameters for the functionalism of the compilers which could be include the advance functionalities also. Because of its safety and critical systems. Nowadays the use of parallel programming have more benefits in this field, Recent works going on the Open MP to introduce in the Ada language to get the above-mentioned benefits. Also concern on the safety from the compiler perspectives. Overly we can assume that the race conditions and the program failure scenarios. Here recent researchers have introduced the languages which will compatible with the Ada programming. We can see the parallelism in web technologies. JavaScript is the most powerful and efficient web language for web programming, this include the front and back end of the web page or website. And it is widely used language for the variety of user applications such as gaming, VFX.

Performance and the energy usage include the parallelism in their hardware resources, this also includes the single instruction, multiple data vector parallelism mechanisms. Nowadays JavaScript supports parallelism in very limited manner and does the not directly exploit the single instruction, multiple data feature. And its asynchronous activities don't communicate through the shared memory concept in the area of JavaScript programming.

In this write up we have included the instruction level parallelism. This concept is not new but It's required nowadays to implement the compiler high-level languages compilation. Instruction-level parallelism technique will allow the sequence execution of the instruction. This is actually not suited for the high-level language to execute the instruction and language compressor. Parallelism in the instruction level execution will lead the high performance and reliability to the compiler system and make it flexible. Sequential program to be paralyzed, no need to write again and again the required code and execute; single execution will be able to manage all the programs and finally, it will use the technique of the parallelism. Keeping the hardware resources in mind it could be designed and implement the algorithm to select the required a portion of the code which could be paralyzed and start executing. By using this technique in the instruction level execution of the selected code will increase the performance by eliminating the complex processing, we will convert this into the paralyzed program and start to execute and check the performance.

Here partitioning application and then choosing the better algorithm for branch prediction analysis and optimization. The compiler code generator will be in the field of parallel microprocessor and micro architecture with the help of the compile code generator, in order to the to achieve this we have focused on four important aspects of parallelism such as-multiple operations effected by cycle, multiple result or overall distribution buses, multiple execution units and the pipe lined execution units. Before the code generation step there should be loop folding and migration of the executable code migration applied to the benchmark; GPGPU memory optimization with parallelism management is a wonderful topic which comes under the domain of parallelism of compilers , this ensures that the memory is optimized in a very good fashion in a truly intriguing way and the optimization techniques employed in this are very much useful for the usage of the compiler by the graphics processing unit hence the whole criteria for making use of the compiler to support this phenomenon is what the author wants to achieve through the paper.

Vector Parallelism in JavaScript in another really interesting domain where compilers are used JavaScript is a scripting language also makes of the compilers a SIMD (Single Instruction Multiple Data) A fixed vector width implies the simplicity in terms of consistent enhance performance and consistent semantics across vector architectures. The SIMD will allow the programmer to handle with the simple languages and direct control; it need very simple compiler but it should be compatible with the hardware of the machine and enhance the performance; Assessors and Mutators are also used such as extract lane, replace lane, Select, swizzle and shuffle which constitutes the entire concept as such the whole of the assessors including type conversions and other advanced concepts where these are made use of Compilers have many applications in many fields of engineering especially computer Science where they are subject to huge loads of instructions to handle in a real time fashion . JavaScript is one of the most widely used language for web development because of its various advantages such as security adaptability etc. The JavaScript does not use parallelism properly it currently only offers partial assistance to it, JavaScript cannot directly exploit SMID (Single Instruction, Multiple Data) instructions. To rectify this the implementation of SIMD language usability and compiler support with the hardware - is used it together add fine-grain vector parallelism to JavaScript. This specific step is a part of the final step of adoption by the JavaScript standards committee and these implementations of the code are available Program ability, portability, and ease of implementation are some of the key facts which are taken into consideration while influencing the design decision. Another potential avenue for future work that can be gleaned from the research conducted is related to this is to implement aggressive machine-specific JIT optimizations so that it may allow the possibility to utilize wider vectors when it is available in the underlying hardware The two major challenges for creating GPGPU programs where performance is at a high level, that the proper implementation of GPU memory hierarchy and judicious management of parallelism.; A feature which distinguishes this optimized compiler is that the understandability of the optimized code, which is useful for performance analysis and algorithm refinement is better compared to others With present advancements in VLSI technology and microprocessor designers can provide more micro-architectural parallelism to increase performance There are four major types of such parallelism they are multiple microcontroller operations issued per cycle, multiple result distribution buses, multiple execution units, and pipeline execution units. One of the most promising methods of designing high speed microprocessors are to use complicated and sophisticated compilers to take advantage of the parallelism both in the programs and in the micro architecture. The compiler and the parallel micro architecture work closely together to achieve high performance without using too many hardware components which in turn reduces the cost expenditure relating to hardware components which increases the efficiency of the production of the compiler. Parallel data paths alongside pipe-lining provides a degree of concurrency to the architecture of micro-controller. These properties assist in fetching and issuing some operations with executing, and distributing results for multiple micro-operations per cycle. For the given set of technology restraints, these experiments can be used to implement a cost-effective micro-controller architecture to execute the given set of programs at an increased speed.

3. Related work

Various fields of parallelism and purpose of the parallelism in the compiler

A. GPGPU compiler with parallelism to enhance the performance

In this discussion [1] authors will try to convey about the parallelism in the unique way of their expressing. Here they have tried to tell the information about optimization of compiler for general purpose computation on the graphics units. In this scenario they have introduced two major things which could be related to the high performance GPGPU programs , like effective utilization of GPU memory hierarchy and management of parallelism, To make it more efficient they have included and considered major two issues that is how to make the parallelism with real time application into the concurrent work items and spearheaded the workloads in a hierarchy of the thread blocks and the threads; another one major issue for this graphics optimization will be how to efficiently utilize the GPU memory hierarchy, given its dominant impact on the performance , they have got the technique that these two major issues will be quite coupled together and finding the optimal solution for this with optimization of the parallelism.

B. Safe parallelism

In this paper [2], will try to convey the information about the parallelism in the safer manner; they have tried to explain the safe parallelism in the unique way. Support the parallelism with Ada and OpenMP will lead to enhance the performance and benefits, Recent works going on the OpenMP to introduce in the Ada language to get the above mentioned benefits. Also concern on the safety from the compiler perspectives. Overly we can assume that the race conditions and the program failure scenarios. Here recent researchers are introduced the languages which will compatible with the Ada programming. In this paper they have found the solution of race condition means that the multiple threads are trying to execute the same code in the same time. They found the techniques which could detect the races;

C. Vector Parallelism in JavaScript

In this paper [3] authors explain about the parallelism in the JavaScript web language more efficient, they have proposed some more methodologies to implement the parallelism concept efficiently in their unique manner to the people which could be pretty fine; We can see the parallelism in the web technologies.

JavaScript is the most and best widely used we coding language in nowadays, and it has been sophisticated in implementing and also have the demand in the real world user applications in the current technology, like video, VFX, graphics, cryptography. parallelism in their hardware resources, this also include the single instruction, multiple data vector parallelism mechanism. Nowadays the JavaScript will support the parallelism in very limited and does the not directly exploit the single instruction, multiple data feature. And it's an asynchronous activities that don't communicate through the shared memory concept in area of JavaScript programming.

D. Parallelism and their Impact on Architecture Design

In this paper [4] author C. D. Polychronopoulos, who is the member of IEEE, he has tried explain the concept about the compiler optimizations for enhance parallelism and their Impact on the architecture design.

Main intention of this paper will be the detection of the parallelism and the effect, influence of this parallelism in the architecture designing; these can be solved in their unique methodology; in this scenario they considered the some techniques for the determination of the parallelism and their effect , they could able to provide the solutions , here cycle shrinkage of the compiler optimization which can be used to parallelize the certain types of loops; mean time they have tried to compute the run-time for the dependency for the analysis, and also handle the control statements;

At the end they have discussed about the barrier synchronization, this also one of the most serious sources of the run-time overhead in the field of the parallelism of the programs. To reduce these impact of the barrier they have tried to implement the concept called distributed barrier by using the shared registers; for implementation they have done a pretty fine method which could be easy in the understandable way;

E. To exploit instruction level Parallelism

In this paper [5] the author will try to explain the concept of instruction level parallelism. This concept is not new but It's required nowadays implement the compiler high level languages compilation. Instruction level parallelism technique will allow the sequence execution of the instruction. This is actually not suited for the high level language to execute the instruction and language preprocessor. Parallelism in the instruction level execution will leads the high performance and reliability to the compiler system and make it flexibility. Sequential program to be paralyzed, no need to write again and again the required code and execute; single execution will be able to manage all the programs and finally it will use the technique of the parallelism. Keeping the hardware resources in mind it could be design and implement the algorithm to select he required portion of the code which could be paralyzed and start execute. By using this technique in the instruction level execution of the selected code will increaser the performance by eliminating the complex processing, we will convert this into the paralyzed program and start execute and check the performance. Here partitioning application and the choosing the better algorithm for branch prediction analysis and optimization.

F. To Parallelizing Compilers

The main goal here [6] is to parallelize the compiler and to provide ways for the developer to keep a check on granularity. granularity of a process can be modified using the dynamic scheduling of the program.

G. *Automatic Parallelization*

In this paper [7] the main aim is to automate parallelization this can be achieved to a certain extent by making use of the hardware support, but we need to focus on the compiler aspect since the software industry is highly dependent on compiler technology to support the needs of their clients

H. *Parallelism in scalar programs*

In this paper [8] the idea is to achieve parallelism in scalar programs, programs are called as scalar when they can be represented using a single scalar number. We can introduce the concept of parallelism in this domain to optimize and enhance the computational capabilities of the compiler by increasing its overall performance.

I. *Parallelism and Data Locality*

In this paper [9] the goal is to achieve parallelism through optimization techniques of the compiler. Today's CPUs are on a single chip .Hardware caches make use of the observation that programs exhibit temporal and spatial locality hence we can use this concept to arrive at our goal to optimize the compilers through the concept of parallelism

J. *Compiler support for parallelism in quantum computation*

Quantum computing (QC) accelerates a variety of computationally [10] intensive benchmarks. Due to the challenges of DE coherence Hence Multi-SIMD QC architecture and effective schedulers are used. To achieve this, we need to develop efficient schedulers which also helps us to implement parallelism onto the compilers, thus improving their computational capabilities.

K. *The SUIF Compiler System*

From this paper [11] we can see that the SUIF system has the capability of performing compiling standard programs. From continuous development and experimentation with new compiler techniques the authors have succeeded in their goal to establish an interface between compiler passes so that other people involved in research can collaborate with the final goal being developed by the SUIF infrastructure

L. *Analysis of Parsing Techniques & amp*

The analysis of the research conducted in this paper [12] reports the uses of compiler for a real-time application with the goal being that of addressing the accessibility at the high level language which implements knowledge of compiler function and register's the allocation technique. The paper describes the techniques used for compiler parsing application and types the grammar.

M. *Parallel parsing on multi-core machines and issues*

In this paper [13] various problems in implementation of parsing algorithms on multi-core machines are discussed Through the research and experimentations conducted in this paper we can say that proper amount of concentrated efforts is necessary to explore the inherent property of parallel processing present in the multi-core machine's.

N. *Parallelization*

In this paper [14]a comparative study of all the techniques which have been used for automatic parallelization has been presented. The main motto of this paper is to provide basic insight into the techniques of non-manual parallelization and the present implementation of the technique in the construction of the complier.

O. *Concurrency and computation*

This paper [15]discusses the special issues of the compiler such as the trend in which the complier is used for computation in the parallel front . Many other key performance parameters are also discussed related to compiler optimization using parallelism

4. Literature

The [1]-[15] references will have the complete information about the parallelism. They have clearly mentioned the importance of the parallelism in nowadays computer systems for the programming world. [16] provided knowledge about the compiler architecture and the usage in the programming systems.

5. Conclusion

This paper presented an overview on compiler optimization and performance enhancement using parallelism.

References

[1] Yi Yang, ping Xiang, Jingfei King, Huiyang Zhou, "A GPGPU compiler for Memory optimization and parallelism Management," PLDI '10 Proceedings of the 31st ACM SIGPLAN Conference on Programming Language Design and Implementation, pp. 86-97, Toronto, Ontario, Canada, June 05 - 10, 2010.

[2] Sara Royuela, Xavier Mortorell, Eduardo Quinones, Luis Mifuel Pinho, "Safe Parallelism: Compiler Analysis Techniques for Ada and OpenMP," Ada-Europe International Conference on Reliable Software Technologies, Ada-Europe 2018: Reliable Software Technologies – Ada-Europe 2018 pp 141-157.

[3] I. Jibaja *et al*., "Vector Parallelism in JavaScript: Language and Compiler Support for SIMD," *2015 International Conference on Parallel Architecture and Compilation (PACT)*, San Francisco, CA, 2015, pp. 407-418.

[4] C. D. Polychronopoulos, "Compiler optimizations for enhancing parallelism and their impact on architecture design," in *IEEE Transactions on Computers*, vol. 37, no. 8, pp. 991-1004, Aug. 1988.

[5] Kumar R., Singh P.K. (2014) An Approach for Compiler Optimization to Exploit Instruction Level Parallelism. In: Kumar Kundu M., Mohapatra D., Konar A., Chakraborty A. (eds) Advanced Computing, Networking and Informatics- Volume 2. Smart Innovation, Systems and Technologies, vol. 28. Springer, Cham

[6] A Formal Approach to Parallelizing Compilers https://www-users.cs.umn.ed

[7] K. R. Varsha, "Automatic Parallelization: A Review," International Journal of Engineering Science and Computing, March 2017.

[8] Scott A. Mahlke Nancy J. Warter William Y. Chen Pohua P. Chang Wen-mei W. Hwu, "The Effect of Compiler Optimizations On Available Parallelism in Scalar Programs," in Proceedings of the 20th Annual International Conference on Parallel Processing, pp. 142-145.

[9] Optimizing for Parallelism and Data Locality, http://www.cs.utexas.edu/users/mckinley/papers

[10] Compiler Management of Communication and Parallelism for Quantum Computation, http://mrmgroup.cs.princeton.edu/papers
[11] Amit Barve1and Brijendra Kumar Joshi, “Issues in implementation of parallel parsing on multi-core machines.”
[12] Robert P. Wilson, Robert S. French, Christopher S. Wilson, Saman P. Amarasinghe, Jennifer M. Anderson, Steve W. K. Tjiang, Shih-Wei Liao, Chau-Wen Tseng, Mary W. Hall, Monica S. Lam, and John L. Hennessy, “The SUIF Compiler System: A Parallelizing and Optimizing Research Compiler,” Technical Report.
[13] Ch. Raju, Thirupathi Marupaka, Arvind Tudigani, “Analysis of Parsing Techniques & Survey on Compiler Applications,” International Journal of Computer Science and Mobile Computing, vol. 2, no. 10, pp. 115-125, October 2013.
[14] Special Issue: Current Trends in Compilers for Parallel Computers, Concurrency and Computation: Practice and Experience, 2007.