International Journal of Computer Science
Scholarly Peer Reviewed Research Journal - PRESS - OPEN ACCESS
ISSN: 2348-6600
http://www.ijcsjournal.com Volume 5, Issue 1, No 20, 2017 ISSN: 2348-6600
Reference ID: IJCS-254 PAGE NO: 1618-1626
Alagappa University, Karaikudi, India 15th -16th February 2017
IT Skills Show & International Conference on Advancements in Computing Resources (SSICACR-2017)
http://aisdau.in/ssicacr ssicacr2017@gmail.com

A Study of parallel processing and its contemporary relevance

Mrs.P.Sudha
Department of Computer science
Thiruvalluvar University Model Constituent College of Arts and Science
Tittagudi, Cuddalore Dist, TamilNadu,India
st.blossomvalli@gmail.com
Mrs.S.Valli
Department of Computer science
Thiruvalluvar University Model Constituent College of Arts and Science
Tittagudi, Cuddalore Dist, TamilNadu,India
kannansudha2001@gmail.com

Abstract - Parallel processing is a type of computation in which many calculations or the execution of processes are carried out simultaneously.[1] Serial memory processing is the act of attending to and processing one item at a time, while parallel memory processing is the act of attending to and processing all items simultaneously. Large problems can often be divided into smaller ones, which can then be solved at the same time. Parallelism has been employed for many years, mainly in high-performance computing, but interest in it has grown lately due to the physical constraints preventing frequency scaling.[2] As power consumption by computers has become a concern in recent years,[3] parallel computing has become the dominant paradigm in computer architecture, mainly in the form of multi-core processors.[4] Parallel computing is closely related to concurrent computing—they are frequently used together, and often conflated, though the two are distinct: it is possible to have parallelism without concurrency and concurrency withoutparallelism.[5][6] Communication and synchronization between the different subtasks are typically some of the greatest obstacles to getting good parallel program performance.

Keywords— Parallel processing, Parallelism, Performance, SISD machine, SIMD machine, MISD machine, MIMD machine

INTRODUCTION

In computers, parallel processing is the processing of program instructions by dividing them among multiple processor with the objective of running a program in less time. In the earliest computers, only one program ran at a time. A computation-intensive program that took one hour to run and a tape copying program that took one hour to run would take a total of two hours to run. An early form of parallel processing allowed the interleaved execution of both programs together.

International Journal of Computer Science
Scholarly Peer Reviewed Research Journal - PRESS - OPEN ACCESS
ISSN: 2348-6600
http://www.ijcsjournal.com Volume 5, Issue 1, No 20, 2017 ISSN: 2348-6600
Reference ID: IJCS-254 PAGE NO: 1618-1626
Alagappa University, Karaikudi, India 15th -16th February 2017
IT Skills Show & International Conference on Advancements in Computing Resources (SSICACR-2017)
http://aisdau.in/ssicacr ssicacr2017@gmail.com

The computer would start an I/O operation, and while it was waiting for the operation to complete, it would execute the processor-intensive program. Parallelism is the process of processing several set of instructions simultaneously. It reduces the total computational time. Parallelism can be implemented by using parallel computers, i.e. a computer with many processors. Parallel computers require parallel algorithm, programming languages, compilers and operating system that support multitasking. Serial memory processing is the act of attending to and processing one item at a time, while parallel memory processing is the act of attending to and processing all items simultaneously.

Parallel processing is much faster than sequential processing when it comes to doing repetitive calculations on vast amounts of data. This is because a parallel processor is capable of multithreading on a large scale, and can therefore simultaneously process several streams of data. This makes parallel processors suitable for graphics cards since the calculations required for generating the millions of pixels per second are all repetitive.

TYPES OF PARALLELISM

Bit-level parallelism

Speed-up in computer architecture was driven by doubling computer word size—the amount of information the processor can manipulate per cycle. Increasing the word size reduces the number of instructions the processor must execute to perform an operation on variables whose sizes are greater than the length of the word. For example, where an 8-bit processor must add two 16-bit integers, the processor must first add the 8 lower-order bits from each integer using the standard addition instruction, then add the 8 higher-order bits using an add-with-carry instruction and the carry bit from the lower order addition; thus, an 8-bit processor requires two instructions to complete a single operation, where a 16-bit processor would be able to complete the operation with a single instruction.

Instruction-level parallelism

A canonical processor without pipeline. It takes five clock cycles to complete one instruction and thus the processor can issue sub scalar performance (IPC = 0.2 < 1).

A canonical five-stage pipelined processor. In the best case scenario, it takes one clock cycle to complete one instruction and thus the processor can issue scalar performance (IPC = 1).

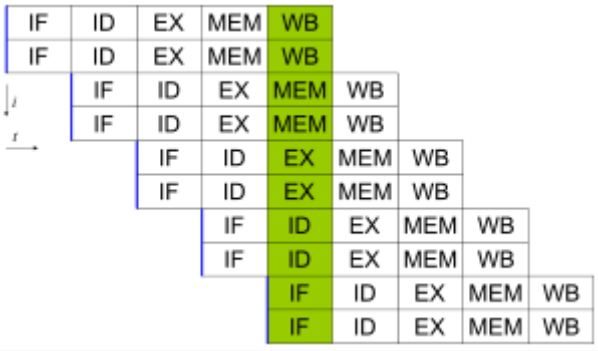
A computer program is, in essence, a stream of instructions executed by a processor. Without instruction-level parallelism, a processor can only issue less than one instruction per clock cycle (IPC < 1).

These processors are known as sub scalar processors. These instructions can be re-ordered and combined into groups which are then

International Journal of Computer Science
Scholarly Peer Reviewed Research Journal - PRESS - OPEN ACCESS
ISSN: 2348-6600
http://www.ijcsjournal.com Volume 5, Issue 1, No 20, 2017 ISSN: 2348-6600
Reference ID: IJCS-254 PAGE NO: 1618-1626
Alagappa University, Karaikudi, India 15th -16th February 2017
IT Skills Show & International Conference on Advancements in Computing Resources (SSICACR-2017)
http://aisdau.in/ssicacr ssicacr2017@gmail.com

executed in parallel without changing the result of the program. This is known as instruction-level parallelism. Advances in instruction-level parallelism dominated computer architecture from the mid-1980s until the mid-1990s.

All modern processors have multi-stage instruction pipelines. Each stage in the pipeline corresponds to a different action the processor performs on that instruction in that stage; a processor with an N-stage pipeline can have up to N different instructions at different stages of completion and thus can issue one instruction per clock cycle (IPC = 1). These processors are known as scalar processors. The canonical example of a pipelined processor is a RISC processor, with five stages: instruction fetch (IF), instruction decode (ID), execute (EX), memory access (MEM), and register write back (WB). The Pentium 4 processor had a 35-stage pipeline.[7]



A canonical five-stage pipelined superscalar processor. In the best case scenario, it takes one clock cycle to complete two instructions and thus the processor can issue superscalar performance (IPC = 2 > 1).

Most modern processors also have multiple execution units. They usually combine this feature with pipelining and thus can issue more than one instruction per clock cycle (IPC > 1). These processors are known as superscalar processors. Instructions can be grouped together only if there is no data dependency between them. Scoreboarding and the Tomasulo algorithm (which is similar to scoreboarding but makes use of register renaming) are two of the most common techniques for implementing out-of-order execution and instruction-level parallelism.

There are two approaches :

- Hardware
- Software

Difference between Hardware & Software Approaches:

Hardware Approaches:

It works on **Dynamic** Parallelism

In this **processor decides** at run time which instructions to execute in parallel

Example :

Pentium processor

Software Approaches:

It works on **Static** Parallelism

In this **compiler decides** which instructions to execute in parallel.

Example:

Itanium processor

International Journal of Computer Science
Scholarly Peer Reviewed Research Journal - PRESS - OPEN ACCESS
ISSN: 2348-6600
http://www.ijcsjournal.com Volume 5, Issue 1, No 20, 2017 ISSN: 2348-6600
Reference ID: IJCS-254 PAGE NO: 1618-1626
Alagappa University, Karaikudi, India 15th -16th February 2017
IT Skills Show & International Conference on Advancements in Computing Resources (SSICACR-2017)
http://aisdau.in/ssicacr ssicacr2017@gmail.com

Task parallelism

Task parallelisms is the characteristic of a parallel program that "entirely different calculations can be performed on either the same or different sets of data". This contrasts with data parallelism, where the same calculation is performed on the same or different sets of data. Task parallelism involves the decomposition of a task into sub-tasks and then allocating each sub-task to a processor for execution. The processors would then execute these sub-tasks simultaneously and often cooperatively. Task parallelism does not usually scale with the size of a problem.

Classified the computer systems based on parallelism in the instructions and in the data streams. These are:

- Single Instruction stream, Single Data stream (SISD) computers
- Single Instruction stream, Multiple Data stream (SIMD) computers
- Multiple Instruction stream, Single Data stream (MISD) computers
- Multiple Instruction stream, Multiple Data stream (MIMD) computers

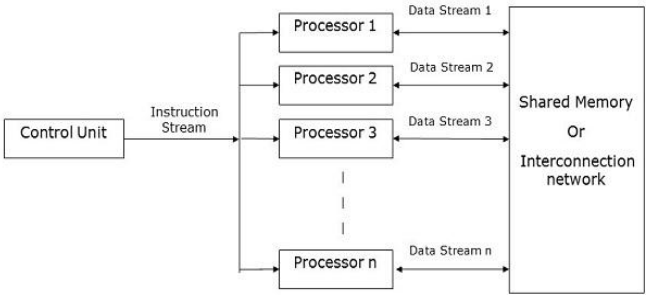
SISD COMPUTERS

SISD computers contain one control unit, one processing unit, and one memory unit.

In this type of computers, the processor receives a single stream of instructions from the control unit and operates on a single stream of data from the memory unit. During computation, at each step, the processor receives one instruction from the control unit and operates on a single data received from the memory unit.

SIMD COMPUTERS

SIMD computers contain one control unit, multiple processing units, and shared memory or interconnection network.



Here, one single control unit sends instructions to all processing units. During computation, at each step, all the processors receive a single set of instructions from the control unit and operate on different set of data from the memory unit.

Each of the processing units has its own local memory unit to store both data and instructions. In SIMD computers, processors need to communicate among themselves. This is done by shared memory or by interconnection network.

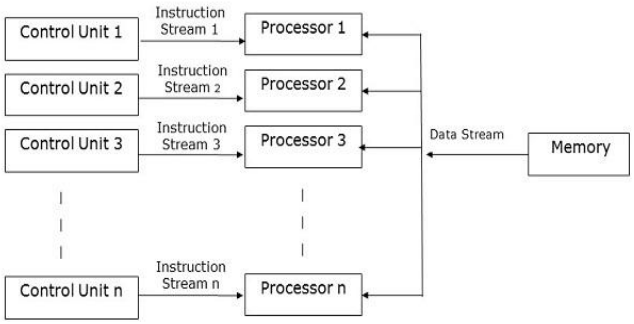
While some of the processors execute a set of instructions, the remaining processors wait for their

International Journal of Computer Science
Scholarly Peer Reviewed Research Journal - PRESS - OPEN ACCESS
ISSN: 2348-6600
http://www.ijcsjournal.com Volume 5, Issue 1, No 20, 2017 ISSN: 2348-6600
Reference ID: IJCS-254 PAGE NO: 1618-1626
Alagappa University, Karaikudi, India 15th -16th February 2017
IT Skills Show & International Conference on Advancements in Computing Resources (SSICACR-2017)
http://aisdau.in/ssicacr ssicacr2017@gmail.com

next set of instructions. Instructions from the control unit decides which processor will be active (execute instructions) or inactive (wait for next instruction).

MISD COMPUTERS

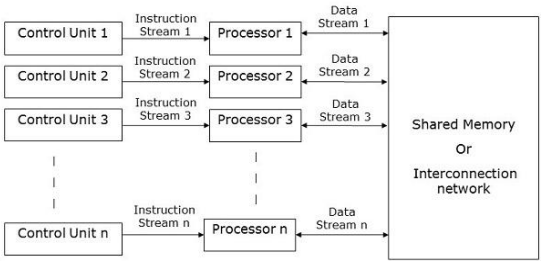
As the name suggests, MISD computers contain multiple control units, multiple processing units, and one common memory unit.



Here, each processor has its own control unit and they share a common memory unit. All the processors get instructions individually from their own control unit and they operate on a single stream of data as per the instructions they have received from their respective control units. This processor operates simultaneously.

MIMD COMPUTERS

MIMD computers have multiple control units, multiple processing units, and a shared memory or interconnection network.



Here, each processor has its own control unit, local memory unit, and arithmetic and logic unit. They receive different sets of instructions from their respective control units and operate on different sets of data.

PERFORMANCE

Performance Two main goals to be achieved with the design of parallel applications are:

- Performance: the capacity to reduce the time to solve the problem when the computing resources increase;
- Scalability: the capacity to increase performance when the complexity, or size of the problem, increases.

The main factors limiting the performance and the scalability of an application are:

- Architectural Limitations
- Algorithmic Limitations

FACTORS LIMITING PERFORMANCE

- Architectural Limitations:
  - Latency and Bandwidth
  - Data Coherency
  - Memory Capacity
- Algorithmic Limitations:

International Journal of Computer Science
Scholarly Peer Reviewed Research Journal - PRESS - OPEN ACCESS
ISSN: 2348-6600
http://www.ijcsjournal.com Volume 5, Issue 1, No 20, 2017 ISSN: 2348-6600
Reference ID: IJCS-254 PAGE NO: 1618-1626
Alagappa University, Karaikudi, India 15th -16th February 2017
IT Skills Show & International Conference on Advancements in Computing Resources (SSICACR-2017)
http://aisdau.in/ssicacr ssicacr2017@gmail.com

Missing Parallelism
Communication Frequency
Synchronization Frequency
Poor Scheduling

PERFORMANCE METRICS

There are 2 distinct classes of performance metrics:

- Performance Metrics for Processors:

Assess the performance of a processor using normally by measuring the speed or the number of operations that it does in a certain period of time.

- Performance Metrics of Parallel Applications:

Assess the performance of a parallel application normally by comparing the execution time with multiple processors and the execution time with just one processor. We are mostly interested in metrics that all.

PERFORMANCE METRICS FOR PROCESSORS

Some of the best known metrics to measure performance of a processor architecture:

- MIPS: Millions of Instructions Per Second.
- FLOPS: FLoating point Operations Per Second.
- SPECint: SPEC (Standard Performance Evaluation Corporation) benchmarks that evaluate processor performance on integer arithmetic
- SPECfp: SPEC benchmarks that evaluate processor performance on floating point operations
- Whetstone: synthetic benchmarks to assess processor performance on floating point operations
- Dhrystone: synthetic benchmarks to asses processor performance on integer arithmetic

PERFORMANCE METRICS FOR PARALLEL APPLICATIONS

There are a number of metrics, the best known are:

- Speedup
- Efficiency
- Redundancy
- Utilization
- Quality

There also some laws/metrics that try to explain and assert the potential performance of a parallel application. The best known are:

- Amdahl Law
- Gustafson-Barsis Law
- Karp-Flatt Law
- Isoeficiency Law

AMDAHL LAW

If P is the proportion of a system or program that can be made parallel, and 1-P is the proportion that remains serial, then the maximum speedup that can be achieved using N number of processors is **1/((1-P)+(P/N).**

Amdahl's law gives the theoretical speedup in latency of the execution of a task at that can be expected of a system whose resources are improved.

International Journal of Computer Science
Scholarly Peer Reviewed Research Journal - PRESS - OPEN ACCESS
ISSN: 2348-6600
http://www.ijcsjournal.com Volume 5, Issue 1, No 20, 2017 ISSN: 2348-6600
Reference ID: IJCS-254 PAGE NO: 1618-1626
Alagappa University, Karaikudi, India 15th -16th February 2017
IT Skills Show & International Conference on Advancements in Computing Resources (SSICACR-2017)
http://aisdau.in/ssicacr ssicacr2017@gmail.com

GUSTAFSON – BARSIS LAW

It gives the theoretical speedup in latency of the execution of a task at **fixed execution time** that can be expected of a system whose resources are improved.

Gustafson's law can be formulated the following way:

Slatency (s)=1-p+sp

- $S_{latency}$ → The theoretical speedup in latency of the execution of the whole task;
- p →The percentage of the execution workload of the whole task

KARP – FLATT LAW

The Karp–Flatt metric is a measure of parallelization of code in parallel processor systems. This metric exists in addition to Amdahl's law and Gustafson's law as an indication of the extent to which a particular computer code is parallelized. It was proposed by Alan H. Karp and Horace P. Flatt in 1990.

ISOEFICIENCY LAW

Maintain the same level of efficiency when we increase the number of processors one needs to increase the size of the problem.

ADVANTAGES

- Save time and cost
- Solve Larger Problems
- Concurrency

Can do many things simultaneously by using multiple computing resources

- Non Local resources

Can using computer resources on the Wide Area Network(WAN) or even on the internet.

APPLICATIONS

- Medical Applications

Parallel computing is used in medical image processing. Used for scanning human body and scanning human brain. Used in MRI reconstruction . Used for vertebra detection and segmentation in X-ray images. Used for brain fiber tracking

- Remote Sensing Applications

It is a software application that processes remote sensing data. Remote sensing applications read specialized file formats that contain sensor image data, georeferencing information, and sensor metadata. Computer analysis of such remotely sensed earth resources data has many applications in agriculture, forestry etc. Explosive amounts of pictorial information needs to be processed in this area.

- Genetic Engineering

It is the direct manipulation of an organism's genome using biotechnology for eg. Dna sequence analysis. Several of these analysis produce huge amounts of information which becomes difficult to

International Journal of Computer Science
Scholarly Peer Reviewed Research Journal - PRESS - OPEN ACCESS
ISSN: 2348-6600
http://www.ijcsjournal.com Volume 5, Issue 1, No 20, 2017 ISSN: 2348-6600
Reference ID: IJCS-254 PAGE NO: 1618-1626
Alagappa University, Karaikudi, India 15th -16th February 2017
IT Skills Show & International Conference on Advancements in Computing Resources (SSICACR-2017)
http://aisdau.in/ssicacr ssicacr2017@gmail.com

handle using single processing units because of which parallel processing algorithms are used

- Artificial Intelligence and Automation

  AI is the intelligence exhibited by machines or software. AI systems requires large amount of parallel computing for which they are used. Four types like Image processing, Expert Systems, Natural Language Processing(NLP) and Pattern Recognition

- Oceanography and Astrophysics

  Used to study wealth of ocean using multiprocessors having large computational power with low power requirements. ROMS were used originally but now MPI programming methods are used. Computational astrophysics refers to the methods and computing tools developed and used in astrophysics research.

VI. CONCLUSION

From the above study of parallel processing , we find out the main use of parallel processing over serial processing that in parallel processing each processing step is completed t the same time whereas not in serial. Using parallel processing we can reduce time for process the problem and also we can do many things simultaneously for getting best result. The main reason for parallel programming is to execute code efficiently, since parallel programming saves time, allowing the execution of applications in a shorter wall-clock time. As a consequence of executing code efficiently, parallel programming often scales with the problem size, and thus can solve larger problems. Using parallel processing some broad issues involved like the concurrency and communication characteristics of parallel algorithms for a given computational problem (represented by dependency graphs), Computing Resources and Computation Allocation: The number of processing elements (PEs), computing power of each element and amount/organization of physical memory used. • What portions of the computation and data are allocated or mapped to each PE. Parallel computing is more tightly coupled to multi-threading, or how to make full use of a single CPU. Distributed computing refers to the notion of divide and conquer, executing sub-tasks on different machines and then merging the results. With cloud computing, you are basically establishing a distributed architecture at a remote or virtual facility. You don;t need to buy any hardware or networking - you can rent the processing power you need. The emphasis is more on remote provisioning, and conforming to the various APIs that the could vendor makes available (data security, authentication, payments, etc.)

VII. REFERENCES

[1] Gottlieb, Allan; Almasi, George S. (1989). Highly parallel computing. Redwood City, Calif.: Benjamin/Cummings. ISBN 0-8053-0177-1.

International Journal of Computer Science
Scholarly Peer Reviewed Research Journal - PRESS - OPEN ACCESS
ISSN: 2348-6600
http://www.ijcsjournal.com Volume 5, Issue 1, No 20, 2017 ISSN: 2348-6600
Reference ID: IJCS-254 PAGE NO: 1618-1626
Alagappa University, Karaikudi, India 15th -16th February 2017
IT Skills Show & International Conference on Advancements in Computing Resources (SSICACR-2017)
http://aisdau.in/ssicacr ssicacr2017@gmail.com

[2] S.V. Adve et al. (November 2008). "Parallel Computing Research at Illinois: The UPCRC Agenda" (PDF). Parallel@Illinois, University of Illinois at Urbana-Champaign. "The main techniques for these performance benefits—increased clock frequency and smarter but increasingly complex architectures—are now hitting the so-called power wall. The computer industry has accepted that future performance increases must largely come from increasing the number of processors (or cores) on a die, rather than making a single core go faster."

[3] Asanovic et al. Old [conventional wisdom]: Power is free, but transistors are expensive. New [conventional wisdom] is [that] power is expensive, but transistors are "free".

[4] Asanovic, Krste et al. (December 18, 2006). "The Landscape of Parallel Computing Research: A View from Berkeley" (PDF). University of California, Berkeley. Technical Report No. UCB/EECS-2006-183. "Old [conventional wisdom]: Increasing clock frequency is the primary method of improving processor performance. New [conventional wisdom]: Increasing parallelism is the primary method of improving processor performance… Even representatives from Intel, a company generally associated with the 'higher clock-speed is better' position, warned that traditional approaches to maximizing performance through maximizing clock speed have been pushed to their limits."

[5] "Concurrency is not Parallelism", Waza conference Jan 11, 2012, Rob Pike

[6] "Parallelism vs. Concurrency"

[7] "The Microprocessor Ten Years From Now: What Are The Challenges, How Do We Meet Them? (wmv). Distinguished Lecturer talk at Carnegie Mellon University. Retrieved on November 7, 2007.

[8] A Survey Of Paradigms For Building And Designing Parallel Computing Machines Ahmed Faraz 1 Faiz Ul Haque Zeya 2 and Majid Kaleem 3 123Department of Computer and Software Engineering, Bahria University Karachi Campus, Stadium Road Karachi, Pakistan