High Performance Computing for Scientists and Engineers

Prof. Somnath Roy
Department of Mechanical Engineering
Indian Institute of Technology, Kharagpur

Module - 01
Fundamentals of Parallel Computing
Lecture - 02
Architecture for Parallel Computing

Hello everybody. I welcome you to the second class of our course High Performance Computing for Scientists and Engineers and we are discussing the first module which is Fundamentals of Parallel Computing. As the name suggest, high performance computing is a technology which is being often deployed by people who are working in different backgrounds of science and engineering and other application areas. So, this group of researchers or engineers or application people they often do lack a background in computer science.

However, we understand that high performance computing is something which is of course, one level complicated than simple computing or using a single computer using your own desktop or even using a very high end personal computing system and it is usually multiple processors and multiple computers connected together which need to communicate across themselves which need to be deployed to solve a problem in using a synchronization among themselves.

So, the high-performance computing infrastructure typically require certain amount of understanding of computer architecture and some computer science basics. But we understand that the people who will utilize this high-performance computing infrastructure are people without computer science basics.

Therefore, while discussing high performance computing especially while discussing different algorithms in high performance computing and seeing that how simple program which is supposed to be done on a single PC can be further developed into a high-performance computing program or a parallel program.

It is important that we also appreciate some of the basics of the fundamentals of parallel computing infrastructure including its architecture, including its hardware and including the software which gives the foundation of parallel computing. It is also important that we appreciate that what is the performance of a parallel computing system we which we are using combining all the hardware software and other back backend issues and we also understand that how when we develop a parallel computing program when we write an algorithm which is supposed to work in a parallel computing architecture.

Is it utilizing all the benefits of the system that again includes the hardware and the backend software's, is it including all the benefits of that? If even if I am an application engineer, but I am not quite sure about these basics I might end up doing things wrong and my application can be hampered.

So, it is important that we understand some of the basics or fundamentals of parallel computing including both hardware and backend software, algorithms for parallel computing, some of the basics on the architecture also.

But again while saying so, we understand that it is the job of the computer science background scientists and engineers to provide support for high performance computing and to come up with new developments in the area of high performance computing which can be utilized by the people who are coming from the application side.

So, people with computer science background they also need to understand that what is the performance of scientists and engineers working in HPC or High-Performance Computing. So, it's important to have a bridge between them and therefore, we have developed in this course also for people with computer science background and for people without CS background it is important to revisit some of the basics of parallel computing. And that is why we are discussing this particular module fundamentals of parallel computing and we cannot avoid having some discussions on the architecture of parallel computing.

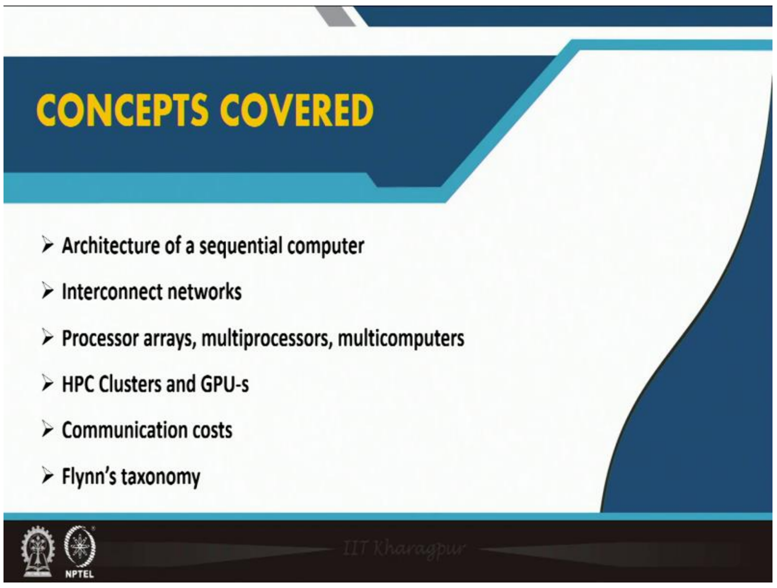
Again, while saying so, I must acknowledge the fact that computer architecture is a very vast subject, it is at least a semester along undergraduate course in computer science and engineering and it has many developments and it can go to much higher level. However, we have picked up only those areas which are absolutely essential in order to understand parallel computing. So, our focus is to learn high performance computing form from an application scientist or application engineer's perspective.

When we have this job setup, what are the basics we do need? And in order to learn that basics we need to revisit or we need to revise some of the basics of parallel computing including its architecture, including the memory models, including the programming models, including some of the parallel algorithms and including some performance metric issues.

So, our focus is to learn high performance computing for from an application point of view therefore, at least three fourth part of this syllabus of this course will be spent on learning programming using different high-performance computing infrastructure shared memory distributed memory GPU that is, learning open MP learning MPI and learning CUDA programming.

Again, as this program will run on a high-performance computing infrastructure, we need to learn about the architectures of high-performance computing infrastructure and we are starting that discussion. So, in this particular set of discussions in today's class and the subsequent classes we will learn about architecture for parallel computing and these are the topics we will go through it.

(Refer Slide Time: 06:54)



CONCEPTS COVERED

- Architecture of a sequential computer
- Interconnect networks
- Processor arrays, multiprocessors, multicomputers
- HPC Clusters and GPU-s
- Communication costs
- Flynn's taxonomy

NPTEL
IIT Kharagpur

We will first focus on architecture of a sequential computer. Then we will see what is an interconnect networks because we understand that if multiple processors are working together, it is required that they need they communicate in between them, it is required that they do their job with a synchronized manner.

It might be often desired that one processor sends some data to another processor and another receives, (Refer Time: 07:19) which will be done through an interconnect networks, which are basically networks which connects multiple processors. So, we will learn about the networks then. Then we will see 3 important components of HPC infrastructure, one is processor array another is multiprocessor and multi computers.

Again, we will focus more on multi processors and multi computers because when we do an application using HPC in today’s world, we are either using multi processors or using multi computers and then we will see the large HPC platforms which are HPC clusters. We will also quickly see what is in a GPU or Graphics Processing Unit and when we will do CUDA programming later we will see GPUs in much detail.

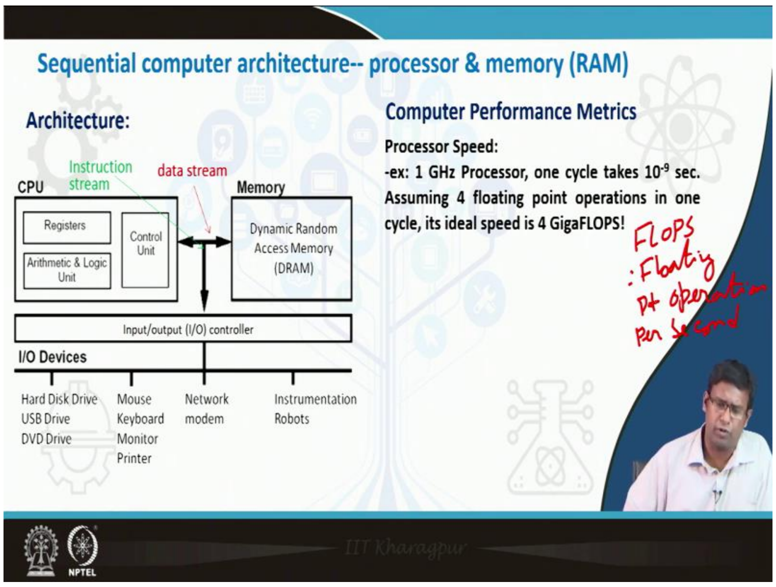
We look into communication costs which is extremely vital in understanding high performance computing because as I say high performance computing means multiple computers or multiple processors some way connected working together to solve different parts of the same problem.

While doing that they need to communicate across themselves, they need to do synchronization, they need to send data from one processor to another processor or they need to write data to a common global memory space which requires certain amount of computation communication among themselves, which has a time delay which has some cost in terms of computing.

So, we will see what is communication cost for an HPC program and what are the different components of the communication cost when we are running our own HPC parallel program, when we are solving a large problem in an HPC system even if the program is developed by somebody else. We should be very careful about the communication costs because this when we are running the sequential program or when we are running program in a single simple PC, this cost was not there it’s a single computer which is solving the problem in communication with the memory.

So, it did not have any communication cost, but when we are running many computers together in parallel to solve a same problem, we are introducing this cost which was not there earlier. Therefore, it is also important that we keep a check on the communication cost and we will look into it in detail. Professor Flynn long back has specified certain taxonomy to classify different architecture of parallel computers, we will go through this Flynn’s taxonomy and we will see that they are very much valid even in today’s world.

(Refer Slide Time: 10:02)



Sequential computer architecture-- processor & memory (RAM)

Architecture:

Computer Performance Metrics

Processor Speed:

-ex: 1 GHz Processor, one cycle takes 10^{-9} sec. Assuming 4 floating point operations in one cycle, its ideal speed is 4 GigaFLOPS!

FLOPS : Floating Pt operation Per Second

So, first we look into architecture of sequential computer. When we say sequential computer it’s a simple computer that we discussed. It is a single CPU and a single memory unit. Sequential means give any job it will process the instructions in a sequential manner one by one and there is no parallel processing of the information of the instruction, you give an instruction it will process it and next instruction will be followed only once the first instruction is finished.

So, even when we talk about sequential computer, we are not discussing about today’s multi core computer, we are discussing about a computer which is one CPU, inside which there is one arithmetic logical unit which is connected with a register and a control unit. So, this registers they take small amount of memories and as the arithmetic logical unit need to work on these memories and give the output. And this sequential computer CPU takes data from a dynamic random-access memory or dram.

So, when you go to buy a CPU or when we go to buy a computer, we look for the CPUs in terms of CPU speed that is how many gigahertz processing cycles that CPU will give and what is the size of this dynamic RAM and what is its speed . Then this connected to the CPU memory arrangement this is what is in the motherboard of the computer, this is connected to an I/O or Input Output controller ,to input output devices, hard disk drives, the mouse keyboard monitor etcetera network modem and if it is in an embedded system it is connected to instrumentation ,to robots or to some of the actuators or sensors which are working.

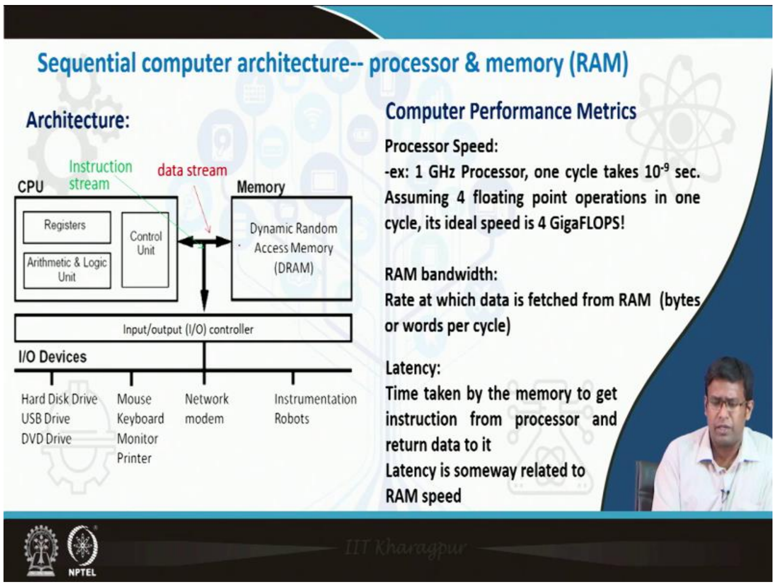
Now, when this is working, there is an instruction stream which follows the input device. Basically, it launches the program through the control unit and an instruction stream goes to CPU, it takes some data from the memory and processes the from the main memory RAM and then processes the instruction here. These two streams of process are important, one is the data stream the data is fetched from memory goes to the CPU and CPU returns the data back, another is instruction stream that this set of instructions are given to the CPU, it processes those instructions on the memory date on the data stream on the data that it has fetched from the memory.

So, if we look into the performance of the sequential computer, there are two performance matrix and one is the processor speed. As I said when we go to market and try to buy a computer we look into the processor speed that how many giga Hertz is the speed of the processor and if we say 1 giga Hertz processor; that means, one cycle of electric pulse to this processor takes 10 to the power minus 9 seconds. So, in 1 second there are 10 to the power 9 cycles and we assume that in one cycle of electricity through the processor there are 4 floating point operations done.

Floating point operations means you simply take numbers do something with the number minus ,addition, division, multiplication some operation with the number that is a floating-point operation, it is done again using the logical gates present in the CPU. And therefore, if there are 4 operations in one cycle, the 1 giga Hertz processor typically does 10 to the power 9 cycles in second 10 to the power 9 cycles in 1 second.

So, if there are four operations 4 floating point operations in one cycle, then one second it will do 4 into 10 to the power 9 floating point operations or 4 gigaflops, speed is called gigaflops. FLOPS is one of the performance matrices here which means floating point operation per second. In 1 second, it will do floating point it will do operation on 10 to the 4 into 10 to the power 9 floats, the speed is 10 to 4 gigaflops well. We actually do not get that high speed in a normal sequential computer and we will see why.

(Refer Slide Time: 15:02)



Sequential computer architecture-- processor & memory (RAM)

Architecture:

CPU
Registers
Arithmetic & Logic Unit
Control Unit
Instruction stream
data stream
Memory
Dynamic Random Access Memory (DRAM)
Input/output (I/O) controller
I/O Devices
Hard Disk Drive
USB Drive
DVD Drive
Mouse
Keyboard
Monitor
Printer
Network
modem
Instrumentation
Robots

Computer Performance Metrics

Processor Speed:
-ex: 1 GHz Processor, one cycle takes 10^{-9} sec. Assuming 4 floating point operations in one cycle, its ideal speed is 4 GigaFLOPS!

RAM bandwidth:
Rate at which data is fetched from RAM (bytes or words per cycle)

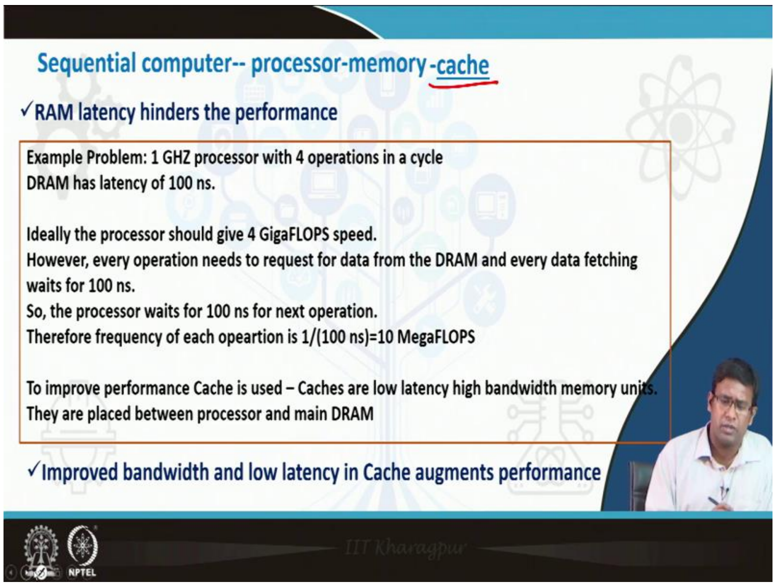
Latency:
Time taken by the memory to get instruction from processor and return data to it
Latency is someway related to RAM speed

So, if we talk about RAM there are two important performance metrics in a RAM one is the RAM bandwidth; that means, when the CPU control unit fetched data from the RAM, at one cycle again it does that in cycle it gives one set of instruction to the data stream and one packet of data comes here. And how much data is fetched in one cycle? So, it is given as how many bytes or how many words are fetched in each cycle that is the bandwidth.

And the next important thing is latency. Time taken by the memory to get the instruction from processor and return data from it. So, memory gets the instruction from the processor that it needs certain amount of data memory, returns the data to the processor and this is done through the connectors PCI bus etcetera .How much time is required for that? That time is called latency why is it called latency because when processor has asked for certain data, it goes into latent state till this data is obtained from the RAM and that that is why it is called latency because this is a time in which processor is actually not waiting it’s in the latent state and waiting for the time of for the data from the RAM.

And now when again we go to buy a RAM, we look about the RAM speed how many RPMs RAM etcetera. So, later this latency is somewhere related with the RAM speed looking into the RAM speed we can estimate that what should be their latency in the RAM in many cases there are other factors also.

(Refer Slide Time: 16:44)



Sequential computer-- processor-memory -cache

✓RAM latency hinders the performance

Example Problem: 1 GHZ processor with 4 operations in a cycle
DRAM has latency of 100 ns.

Ideally the processor should give 4 GigaFLOPS speed.
However, every operation needs to request for data from the DRAM and every data fetching waits for 100 ns.
So, the processor waits for 100 ns for next operation.
Therefore frequency of each opeartion is 1/(100 ns)=10 MegaFLOPS

To improve performance Cache is used – Caches are low latency high bandwidth memory units.
They are placed between processor and main DRAM

✓Improved bandwidth and low latency in Cache augments performance

So, if RAM latency is high it will hinder the performance; that means, the processor has to sit idle, though the processor can perform large number of operations in a segment, but if the RAM introduces some latency that time the processor cannot perform the operation.

It is waiting for the data, say the I ask the processor to do A is equal to B plus C. So, you will be asked from the RAM what are the values of B and C? The RAM will send these values the it will add the values and return the value A to the RAM. So, taking the values B and C and returning the value A, this will take certain amount of time during that time the processor cannot do anything else, it has not finished A is equal to B plus C. So, it is sitting idle.

So, if this time is high if RAM latency is high, performance will be degraded. So, we can see a small example problem, 1 gigahertz processor with 4 operations in a cycle is connected with a RAM and the dynamic RAM dram has a latency of 100 nanoseconds.

So, once in processor asked for some data, the RAM will take 100 nanosecond time to send the data to the processor. Now, the processor is 1 giga Hertz and does for a floating-point operation in one cycle and there are one gigahertz means there are 1 giga cycles in a second, ideally it should give 4 gigaflops speed we have seen it in last slide now for every operation. So, ideally there will be 4 into 10 to the power 9 floating point operations in us in once again, but for every operation the request for data from the dram and the data fetching it will take 100 nanoseconds.

So, one operation cannot be done unless this 100-nanosecond time is over. Therefore, once one operation is done ,the processor has to wait for 100 nanosecond and it will get the data and finish the next operation. So, though it can do 4 gigaflops operation flop operation in a second, for every operation it has to wait for 100 nanosecond therefore, in one second it can do only 1 by 100 nanosecond operation because it is once one operation is done it is waiting for 100 nanosecond to get the data.

Once one instruction set is launched, its waiting 100 nanosecond to wait to get the data. So, the frequency of each operation is 1 by 100 nanosecond in 1 second it can do only 10 into 10 to the power 6 or 10 megaflops operation. Therefore, from for from 4 gigaflops speed which was the theoretical speed considering no RAM latency, RAM latency reduces the speed to 10 megaflops much smaller the smaller by few 100 times.

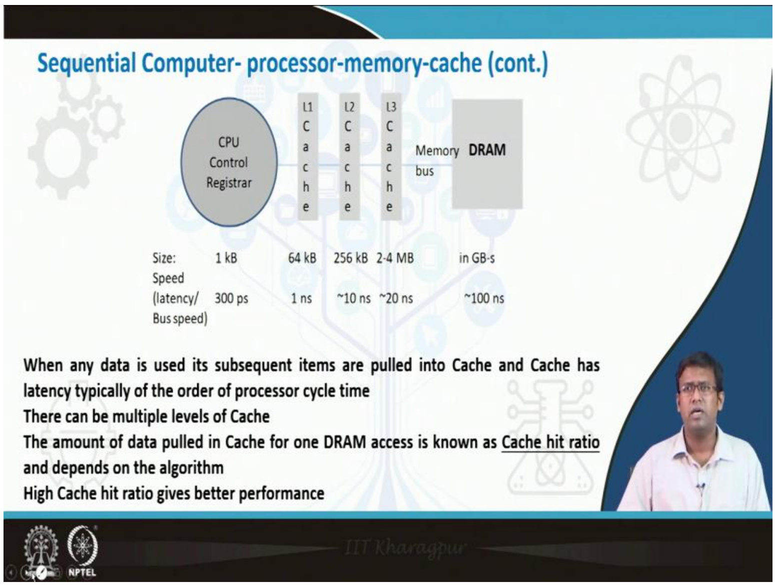
Now again 10 megaflops as we do not get 4 gigaflops speed in a normal CPU, we also do not get slow speed as 10 megaflops we geta speed in between them, you get a frequency in between 4 gigaflops to 10 megaflops. How is this is done? This is done by using something called cache to improve the performance cache is used.

Caches are low latency high bandwidth memory units; these caches are placed in between processors and RAMs. So, with some of the intuitions by the compiler and the programmer and hardware itself, some memory from the RAM is already brought by the cache and sits very close to the CPU.

So, if CPU ask for some data it does not need to go to the dram for that data it can take this data from the cache. Caches are low latency high bandwidth small memory units not the entire matrix will fit in the cache some part some rows or some elements in the row of the matrix will fit in the cache and when the CPU wants a memory, it does not need to every time go to dram, but it can take the memory from the cache.

And they have low latency, they have high bandwidth therefore, they improve the performance of the system improve bandwidth and low latency in cache augments of performance. And caches are extremely important in order to estimate the performance of a computer.

(Refer Slide Time: 21:26)



Sequential Computer- processor-memory-cache (cont.)

Size:	1 kB	64 kB	256 kB	2-4 MB	in GB-s
Speed (latency/ Bus speed)	300 ps	1 ns	~10 ns	~20 ns	~100 ns

When any data is used its subsequent items are pulled into Cache and Cache has latency typically of the order of processor cycle time
There can be multiple levels of Cache
The amount of data pulled in Cache for one DRAM access is known as Cache hit ratio and depends on the algorithm
High Cache hit ratio gives better performance

So, if we look into a sequential computer architecture, it's not a processor memory arrangement rather it's a processor cache memory arrangement. So, there is a CPU controller register, this register has a very small memory .Any variable coming from caches or dram is directly loaded to register and CPU can only work on the registered variables. The size of the memory is 1 kilobyte which is very small and latency is very small also it is 300 picoseconds.

Now, you have different layers of the cache, the first test cache is L1 cache which is small 64 kilobyte has a latency of 1 nanoseconds and then L 2 cache L 3 cache and then the main memory which is gigabyte in today's world we get terabytes of memory, but the latency is high 100 nanoseconds. When any data is used the subsequent items are pulled into cache and cache has latency typically of the order of the processor cycle time.

So, if we need to use any data for computing, first the data is coming from the RAM and the subsequent items if we are using an array the subsequent items in that address space are pulled into the cache. And therefore, when the CPU is working usually it works in a loop or following some algorithm, compiler also understands that what are the next elements that will be used for the calculation and it brings down this element to the cache.

So, if the CPU is working it does not need to wait for 100 nanoseconds to get the data from the dram rather only in one nanosecond latency, it can get up the get data from cache and thereby it can improve the performance. There can be multiple levels of cache, the amount of data pulled in cache for one-dram access is known as cache hit ratio.

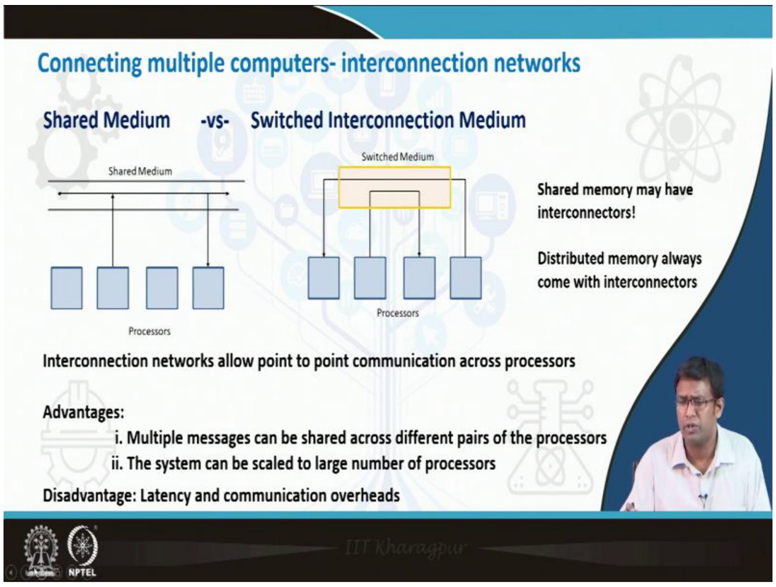
So,if one access to RAM is required by the CPU, how much data is already in the cache is called the cache hit ratio. So, for one particular data requirement from the CPU, how many times for one certain amount of data required by the CPU, how many of this data is residing in the cache determines the cache hit ratio?

If the cache hit ratio is high; that means, most of the data are already in cache which is deserved by the CPU the performance will be good. If the cache hit ratio is poor the performance will be bad high cache hit ratio gives better performance, I can give you one small example, if you look into Fortran and c programs. So, once any matrix is stored in c it stores a matrix row wise again once a matrix is stored in a Fortran program the matrix in the RAM is stored a column wise. So, if you are trying to access the matrix row wise.

If you are using c program row wise the data is pulled into the cache. So, next elements in the in the same row in the next elements in the same you are already residing in the cache. So, in C program if you are trying to access the data row wise is a very good cache hit ratio, but if you are trying to access the data across row wise in a Fortran program, the data is in the RAM the data is actually column wise therefore, cache as the next elements in the column not in the row there will be cache miss.

It looks for the data it will not find the data in the cache it will has to go to RAM. Therefore, cache hit ratio will be small and the program performance will be is poorer. So, in Fortran if you are trying to access some data if you are using a column wise data access through your program, you will get better cache hit ratio and the performance will be good.

(Refer Slide Time: 25:39)



Connecting multiple computers- interconnection networks

Shared Medium -vs- Switched Interconnection Medium

Shared Medium
Processors
Switched Medium
Processors

Shared memory may have interconnectors!

Distributed memory always come with interconnectors

Interconnection networks allow point to point communication across processors

Advantages:

i. Multiple messages can be shared across different pairs of the processors

ii. The system can be scaled to large number of processors

Disadvantage: Latency and communication overheads

Now, we come to multiple computer system. So, how multiple processors can be connected and there are two typical arrangements one is called a shared medium that there is a single bus in which many processors are connected.

There is a single bus ,it is connected to the main memory and other parts in the motherboard and many processors are connected there. And the other one is that different processors are there and they are connected via an interconnect switch. So, first one is called a shared medium, the second one is called the switch interconnection medium. Interconnection network allows point to point communication across the processors.

So, if we see here this processor can address some data, this processor can address some part of the data, but they cannot privately communicate across themselves. So, they have to be any way connected to the shared common address space, but if we see here the processors can point to point communicating in private communicate. If this processor communicates with this processor, the other two processors can do that without changing anything common between them let say it is a more flexible way of communicating across the processors.

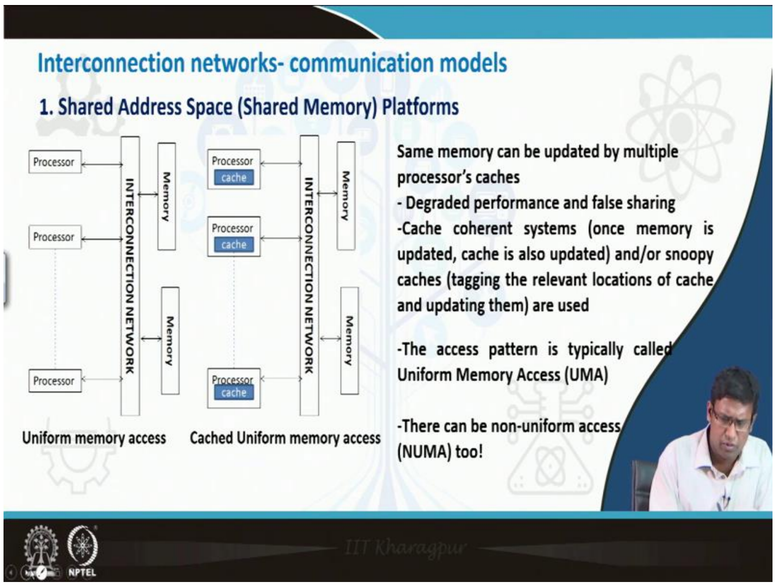
Also we can put many processors together if you using switch in an inter shared medium we can understand there is a common bus in which many CPUs are connected and there is a space issue there are heating heat generation issues there are electricity supply issue etcetera you cannot in really use large number of processors in a shared medium. And while saying large number of processor so, we think about really very very large number of processor. So, in a very efficient and leading HPC system, I have shown you in for in our first class discussion that there can be 100 or1000 computers which are connected together.

Interconnection networks allows point to point communication, the advantage is that multiple messages can be shared across different parts of the processor and the system can be scaled to a large number of processors as I said earlier. The disadvantage is that there will be more latency because the communication earlier was to a common shared medium, but now the communication is to a switch.

And this as we are discussing about the communication across CPU and RAM there is the latency now, there is a network switch which has more latency there will be more wait time for the CPUs to get the data. So, communication time will increase and communication overhead will increase.

This interconnector may be used in shared memory machines also, but distributed memory machines always come up with the interconnectors.

(Refer Slide Time: 28:40)



Interconnection networks- communication models

1. Shared Address Space (Shared Memory) Platforms

Same memory can be updated by multiple processor's caches
- Degraded performance and false sharing
-Cache coherent systems (once memory is updated, cache is also updated) and/or snoopy caches (tagging the relevant locations of cache and updating them) are used

-The access pattern is typically called Uniform Memory Access (UMA)

-There can be non-uniform access (NUMA) too!

So, you should also look into shared memory or shared address space machines, that there is one interconnect network which is connected with many CPUs and this can be through an interconnect meter, this can be through a shared address connected also and memories are connected with the interconnect network.

So, these memories though they are different pieces they can be assumed to a to virtually a contiguous piece of memory and the same memory is uniformly be accessible by all the processors which are connected to interconnect network. This is called a Uniform Memory Access or UMA machine and if the processors have cache this is called a cached uniform memory access. There is a difficulty if their processor sub cache this processor fits some data here the data is already residing in the cache and works on that.

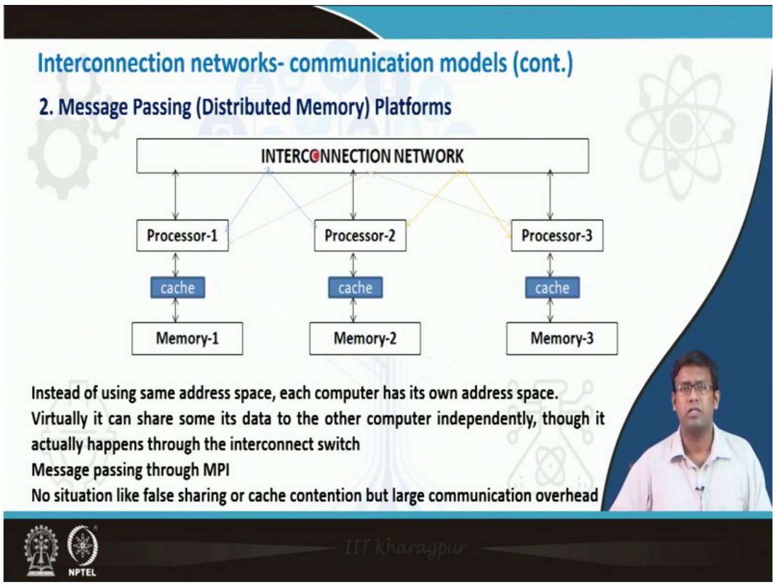
However, some same data is already residing in the cache of the next processor it has updated it. So, this has a non-updated value of the cached data and there is a conflict between them we will we will look into these conflicts in detail later. Same memory can be updated by multiple processors caches which is a significant issue here this will degrade the performance and in order to avoid that we need to use some cache coherent protocol.

They can be false sharing though the that particular memory is not being used by the other processor; however, they are using the same cache line and they are thinking that it is probably the same memory they are going to going to use and there will be contention in between them which is called false sharing.

We will again discuss false sharing in much detail and to avoid that we need cache coherent system that once memory is updated, cache is also updated. So, one processor updates something in the memo cache, it goes to the main memory and it is communicated to all other processors till then none of the processors can do anything else or snoopy cache that the relevant locations of the cache which is being updated at utilize. So, these protocols are required for cached shared memory system and in general shared memory systems in today’s world is cached through a shared memory system.

So, cache coherency is an important aspect in shared memory systems. The access pattern is typically called Uniform Memory Access UMA there can be non-uniform memory access also.Uniform memory access means same memory is visible uniformly by all the processors.

(Refer Slide Time: 31:15)


Interconnection networks- communication models (cont.)

2. Message Passing (Distributed Memory) Platforms

Instead of using same address space, each computer has its own address space.
Virtually it can share some its data to the other computer independently, though it actually happens through the interconnect switch
Message passing through MPI
No situation like false sharing or cache contention but large communication overhead

So, we look into Distributed memory platforms or message passing based platforms .There is an interconnect network and many processors with their own caches and own memories are connected to the same interconnection networks.

And these processors can communicate across themselves independently through the interconnection network. It’s not a same memory which is visible to all the computers each processor has its own local distributed sense of memory therefore, it’s also different memory elements and different caches given to all the processors.

We do not need to think about cache coherence false sharing etcetera instead of using same address space each computer has its own address space virtually it can share some of its data to other computer independently through interconnect networks. So, if some of the element data of this processor has to be given to the third processor, this should go through the interconnect network independently and it is not interfering any memory of any other processor, but though it actually happens through the interconnection switches and this is most popular and most widely used protocol for this is given through MPI your message passing interface calls, this is done through message passing interface.

So, no situation like false sharing or cache coherence are there, but as all the processors have different memory and substantial amount of data from one processor to another processor has to be moved through the interconnection network, there can be large communication over it. In our next discussion we will look these things into detail and we will try to understand what is communication overhead and what are the challenges its posing to a parallel computer the parallel computing application.

Thanks.