

**Applied Accelerated Artificial Intelligence**  
**Prof. Bharatkumar Sharma**  
**School of Computer Science and Engineering**  
**Indian Institute of Technology, Madras**


**Lecture - 08**

**Scheduling and Resource Management Introduction to scheduler and orchestration tools Part - 1**


So, welcome everyone to the next session of the Applied Accelerated Artificial Intelligence course, today I am going to take this session; my name is Bharatkumar and I work as a GPU Advocate at NVIDIA. The topic for today is primarily towards Scheduling and Resource Management.

So, I am going to take you through certain parts of how Artificial Intelligence is used in the real world scenarios and when you join enterprise or if you are working in this supercomputing domain how these use cases are applied and some of these concepts which will help you in understanding, the next set of lectures where we go into deep dive into how the AI clusters look like.


(Refer Slide Time: 01:06)



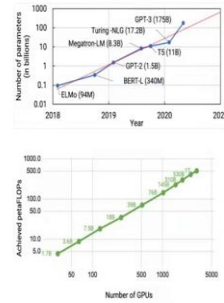
## What have we covered so far?




- Need for Speed
  - Accelerated Computing
  - Largest AI model ?
  - What does it require to train these models?



<https://developer.nvidia.com/blog/scaling-language-model-training-to-a-trillion-parameters-using-megatron/>





So, before we move forward, what have we covered so far? So, what have we covered the first lectures, couple of lectures were dedicated towards the need for speed. We talked about the significance of accelerated computing or GPU computing, what you see

here below is kind of a representation of how the CPU architecture and GPU architecture looks like.

On the left hand side the diagram which is more of a CPU representation you see, the fundamental difference as compared to the GPU which is on the right hand side if you can see the arrow is basically the number of cores. CPUs can come today with almost 80 cores nowadays, but with GPUs are more of throughput based architectures and they have 1000s of cores.

The latest GPUs have more than even 5000s of cores, what does it mean? When you have more number of cores on a particular processor fundamentally means that you can launch so many parallel operation on that particular processor. On GPUs you can practically launch 1000s of parallel operations on the GPU and these all are basically representing is what we call as accelerated computing or throughput based computing.

All artificial intelligence or high performance computing applications like computational fluid dynamics or whether it is molecular dynamics or as simple as even sorting can be converted into a parallel operation. All of these parallel operations can run very well on architecture like, GPU and we said that the GPUs can actually provide this. So, that you can run your model training much faster as compared to running it on the traditional architecture.

It is one of the pillars why deep learning was backed with a big bang in 2012 because the researchers were able to not just use the latest models, but also were able to train the network much more faster otherwise which would have taken them a very long duration. They wrote the parallel code on the GPU and the output of that was they basically won the ImageNet challenge and everybody started looking at using deep learning back again and today we are in the new revolution of AI computing for this overall industry.

From 2012 onwards where are we at this point of time? How big the models have become? As you know that the deep learning models have something called neurons, you have different set of parameters that needs to be trained via the cycle of back propagation and then forward propagation and making sure that our network kind of learns all those things. It is an iterative algorithm, but the number of parameters that you need to learn in AI, because it is primarily involving high dimensional data like images, speech and all those things and the number of parameters kept on increasing over years.

Let me show you what is the motivation behind it, you can see here from 2018 till 2021 is the x axis, what you see on the y axis is the number of parameters which are required to train that model which is in billions.

So, you can see here today, if you are talking about models which are in this space of speech you can see a GPT, this model has 175 Billion Parameters that needs to be trained in a network. So, the models which were Hello World Models where you used to have couple of convolution neural or couple of LSTM layers and all have now moved to really really complicated networks.

Artificial Intelligence is not about creating simpler tasks, because I believe those problems have been solved in fact, challenge like ImageNet, when we reach a particular accuracy level it was stopped after some time because we have surpassed the human capacity, but there are certain problems which still are ongoing and there is a lot of effort being going there.

You can see the number of parameters the model size is increasing very drastically in 2018 we were talking about 94 million parameters and now we are talking about 175 billion parameters in 2020 and we are still in 2022 where it is already known that we are going to cross trillion parameters. It requires a large amount of computation even a system like GPU actually cannot solve it.

Let us take an example from a point of view of the amount of computation which is required to train these models.

What you see again in this particular graph, on the x axis is the number of GPUs, the number of GPUs it was required to train it. And on the y axis what you see is the achieved petaFLOPS, which means the amount of FLOPS achieved to train that particular network, FLOPS; Floating Point Operations Per Second that is how you measure the efficiency or the compute power of any system. Peta is something so, you have megaFLOPS, gigaFLOPS, teraFLOPS and you keep on moving forwards and then we are talking about petaFLOPS here.

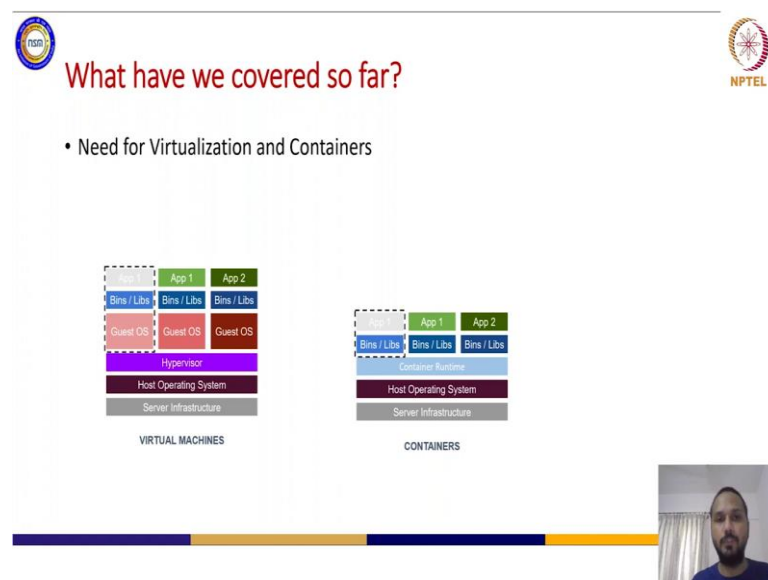
So, you can see here the amount of computation which was required could not be done in a single system, on a single GPU, it is not possible to do this, because if you try to train the network within a single system. You will end up either first of all you would not be

able to do it because the amount of memory, amount of computation required is just higher which a single system cannot handle or the second problem would be even if you are able to fit it within a single system you would be waiting decades for it to finish and give you any outputs.

Hence, the number of GPUs kept on increasing because you were not able to do it on a single system and you had to actually distribute the work within the system across the GPUs like 8 GPUs nowadays you get 16 GPUs within 1 node, but also take it around. 1 GPU has 1000s of cores and now you are talking about having few 1000s of GPUs there, that is the amount of computation which we are talking about in this era, where we are having models of that particular scale.

I am not saying that all the models require this kind of thing, but we will see the motivation that even if the model is not large why do you require a cluster and why do you require certain software stack, which is going to help you in making sure all your resources are kept good.

(Refer Slide Time: 08:32)



Also in the next couple of lectures what you went through was the Need for Virtualization and Containers. We already saw an example of where virtual machines and containers are used. Virtual machines as a technology was the base of cloud computing, the main reason for also coming out with cloud computing was to have a

central location and you could optimally use the resources and schedule them across multiple users.

Like, if you had a particular system which was very heavy end and one user could not use the whole system you could actually create multiple virtual machines on the same system, so that you can efficiently utilize the whole server or the whole machine much more efficiently. It was actually a resource optimization problem if you see it from a non-technology point of view, technology is the base on which you try to solve a problem; the problem that we are trying to solve was efficient utilization of the resources.

As you would have seen in the lectures you have the infrastructure, you have your host operating system there is an entity called as hypervisor which makes sure which will make sure that all the virtual machines they look as if one virtual machine is not even aware that there is another virtual machine which exists and they feel that the whole server is basically provided to them and hypervisor plays a very important role.

By the way I am not saying this is the only way of virtual machine as you know that you have gone through different types of hypervisors and different types of virtual machines, I am not talking on that part here.

You also went through in the last lecture on containers, the fundamental difference that you will see in a virtual machine and a container is you have the server infrastructure, you do have your host operating system, but you do not have this entity called as hypervisor which is acting like a OS over an OS and what you have is the container. The container acts like a bundle where all of your dependencies are bundled into one entity and it is running on a container runtime.

The main reason why containers became also really famous was because containers because they do not go through the hypervisor, there are higher chances that they almost provide a bare metal performance which means if you are going to run it in a non-hypervisor, non container environment you are directly running your application and if you are getting some performance and latency you will get almost the same if you run it through container runtime.

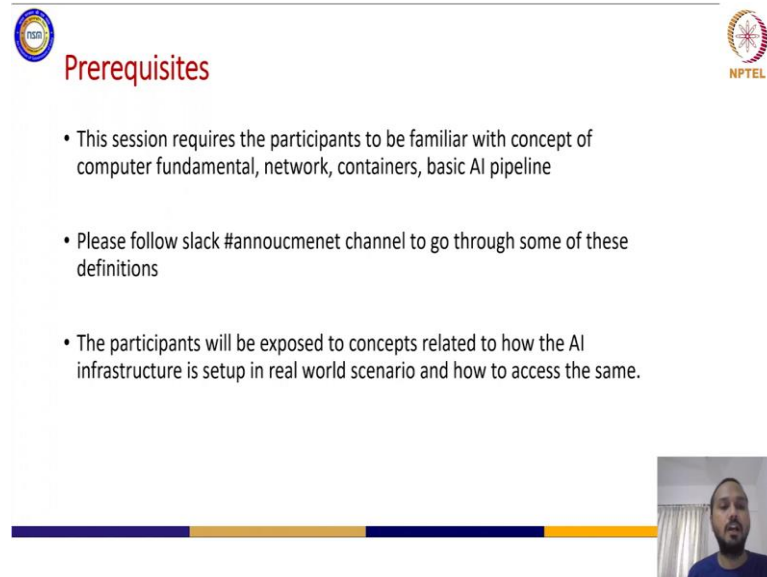
Because it is a very thin layer it helps you in bundling all of your dependencies and running it on the server infrastructure. So, you can achieve a lot of things which are there in a virtual machine also in a container environment, but container has its own problems like you cannot basically create a complete separation of concern where there is no guarantee that one container might not be able to access the resource of the other containers and things like this which is a different topic, but containers played a very important role in basically making sure that if I have if I have a system.

Suppose, if I implement a neural network I install all the dependencies, I can create a container and host it on one of the repositories and I will be able to run that container on my machine I can run it on another machine where I get access to, I can run it on the cloud I can run it anywhere practically without having to do any kind of a re installation anywhere.

So, it was not just about, can I do the installation on my own? Yes, you can and you can keep on doing it, but then the main problem which arises is a manual work of doing this results into it is much more error prone. Containers have been putting all the dependencies together and making sure that I can provide this container to anyone else and just run it without having to do any kind of installation and container played a very important role there.

So, if you see the journey we started with the fundamentals of why do we need for deep learning, why is it important to have a system like this, we went towards the software stack of it which is virtualization and container.

(Refer Slide Time: 13:13)



The slide features a header with two logos: a circular logo on the left and the NPTEL logo on the right. The title "Prerequisites" is centered at the top. Below the title, there is a bulleted list of three items. At the bottom right of the slide, there is a small video feed showing a man with a beard and a blue shirt.

### Prerequisites

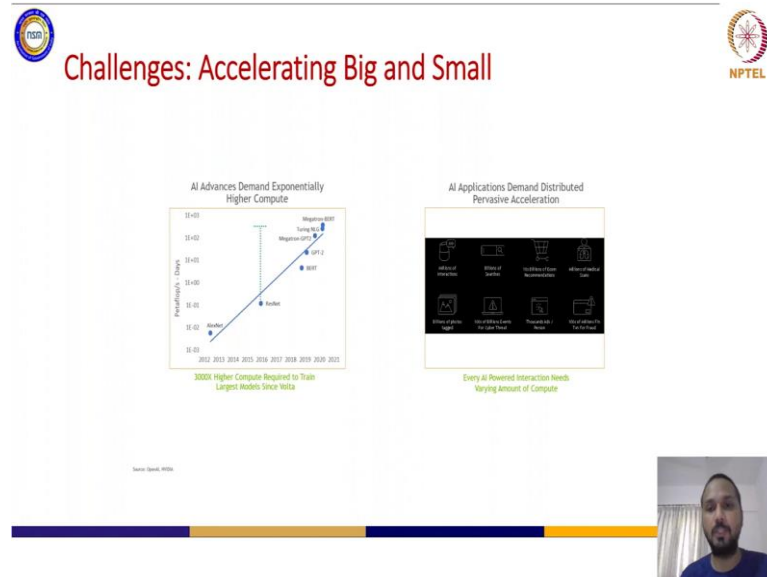
- This session requires the participants to be familiar with concept of computer fundamental, network, containers, basic AI pipeline
- Please follow slack #annoucmenet channel to go through some of these definitions
- The participants will be exposed to concepts related to how the AI infrastructure is setup in real world scenario and how to access the same.

And for this session we are going to talk and take you to the next level of what is and how you can solve some of those problems which I mentioned on the first slide in terms of how do you make sure that you are able to utilize a cluster environment.

So, this session requires the participants to be familiar with the concepts of computer fundamentals, network, containers and a basic AI pipeline. If you have not done some of this, you can go back and look at slack, we have posted some of the links on the announcement channel where you can learn some of these concepts as well. So, please follow the slack announcement channel to go through some of these definitions.

The participants in this particular lecture would be exposed to the concept related to how AI infrastructure is set up in the real world scenario and how to access them. We are not going to talk about the fundamental building blocks of creating your own supercomputing center which will be covered in the next set of lectures. But I am going to introduce you to some of those concepts which will help you in building these concepts for the next lectures.

(Refer Slide Time: 14:22)



So, when I talk about AI I talked about one particular challenge, in this slide what we are looking here is accelerating big and small challenge. On the left hand side is the part that we mentioned to you sometime back, that AI is advancing at an exponential rate, when I say advancing not just the accuracy is increasing the errors are going down, but the models are becoming more and more complex and you need to basically increase the compute power to train those networks.

You can see here again in 2012 the network which won the ImageNet challenge from AlexNet and then later on you have the ResNet you can see here there was a 3000 x gain just in 3 years from the time the previous model was released in 3 years. So, there was an increase in the amount of computation need by 3000 x to be at par with the latest accuracy standards which exist.

The second problem is the other side of the AI world problem which is another set which is also very well-known. As you know the number of sensors which are there on this earth are more than the number of humans which exists. The amount of data which is being pumped in by this sensors is huge, all that data needs to be analyzed by the network that you have created.

There are billions of searches happening on Google and all the other different search engines, there are so many sensors which are out there, there are so many interactions

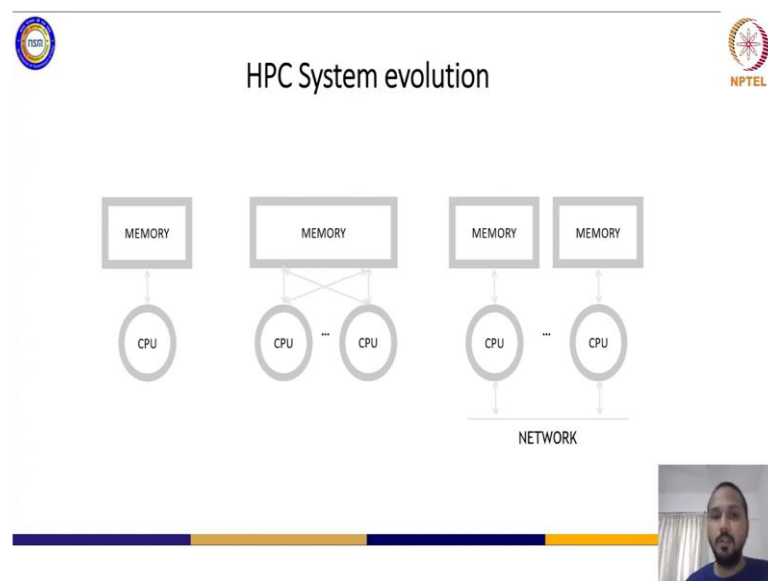


which are happening, the amount of videos getting uploaded, the photos getting tagged. Now this is a different problem altogether from the first one.

The first problem is about you training the network and the network itself is huge and you want to do it in smaller time. The second problem is basically you have a train network, but you need to basically scale out to make sure that you are able to answer the queries or do all of those interactions with this train model while inferencing across all of them.

Now that again requires you to scale out while the first one is scale up problem, the second one is referred to as scale out problem because you need to basically cater to so many users and that is the problem which the enterprise or the AI or even many of those agencies are whoever is deploying AI is trying to solve. It is not just about creating and improving the accuracy for a particular data set, but it is much more than that and this whole lecture is kind of focused on the practical aspects of it.

(Refer Slide Time: 17:06)



To solve this problem maybe I would give you a very small background towards how the evolution happened.

So, we have understood the fact that we need to solve the compute problem, but how it was solved before and how is it solved now is also the evolution that we are talking

about. In the earlier days we were ok with mega flop of performance, mega which is  $10^3$  and we used to have a single core CPU and you had a memory attached to itself.

And the amount of computation that you required was also very less, but then there was more compute requirements which came and one core was not enough to solve that compute requirement, that is when came the multi core era, where you do not get nowadays any processor which has 1 core you have multiple cores and all of them share the memory.

As I said the CPUs nowadays also come with 80 cores. So, you have 80 core machines all sharing the same RAM and you can run basically 80 or 160 or those many number of threads or those many tasks in parallel and make your application go 160 x or 180 times faster.

So, we went from a megaFLOP era to a gigaFLOPS era, which means I was able to do more number of operations, but as you can see the amount of computation which is required is increasing still. That is when the new era came which was the era of distributed computing.

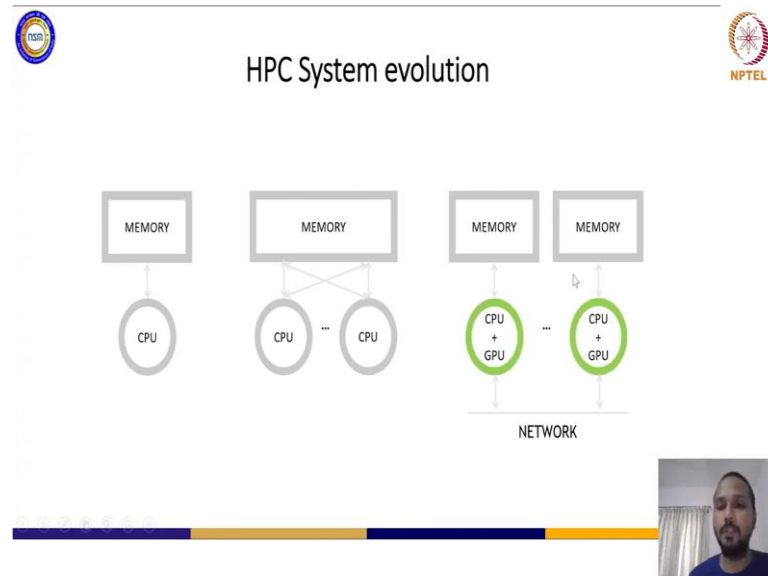
What it means is that, I do not have just one node which is having multiple core I attach I just stack all of these nodes together and I basically now connect all of them with the network and I write a code which distributes my work across this overall network and to different machines and then finally, accumulate the result into one particular base this is called as distributed computing.

And in this distributed computing the network which is required is not the same that you see in your houses, it is not the same 10 gigabits per second Ethernet, it is a very high throughput low latency networks like InfiniBand, which goes up to 400 GBPS and all also 200 GBPS and all it is a much more higher speed and low latency network, you cannot use your traditional Ethernet cables to make all of this work. So, there is a requirement which came.

The problem also to certain extent with this approach was that, now you need to distribute the work across network and the network sometimes might become a bottleneck unless you have distributed your work equally among all of the cores and if

you are spending more time on communication as compared to computation you will be bound by that.

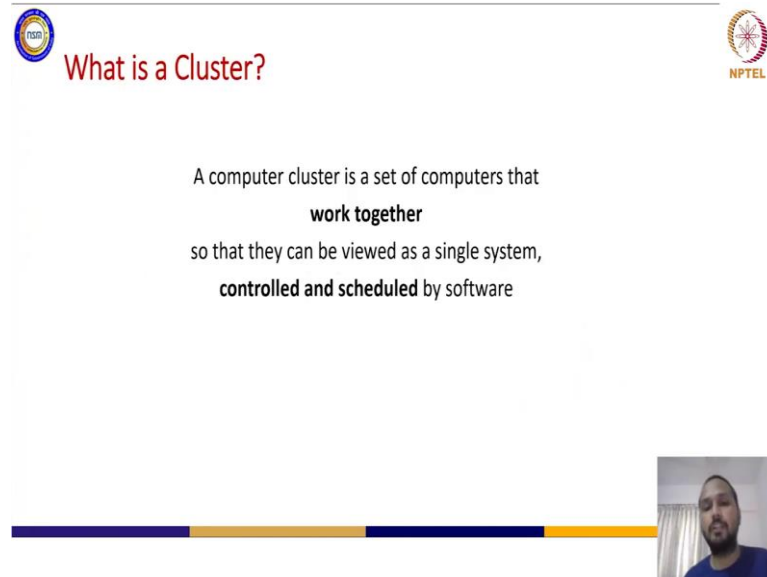
(Refer Slide Time: 20:03)



What GPU did was to your existing this particular part, what you get nowadays if you see most of the cluster is that you just do not get a CPU inside your node, but you also get a GPU. The advantage that you get out of that is now you can do more computation on a single node and you need to go and go across nodes much more lower required. So, that you are restricting and communicating much more lower as compared to how you would have gone across.

So, now, you have distributed computing, but within the distributed computing part you just do not have the CPU, you also attach a GPU and you do all those parallel operations and multitasking on CPU and GPU. And this is what I am talking about is what is called as a cluster, which means all of these nodes which are going to have they are going to solve a particular problem and you are going to communicate with each other using a high speed low latency network and there is a software stack which is required to maintain this cluster.

(Refer Slide Time: 21:06)



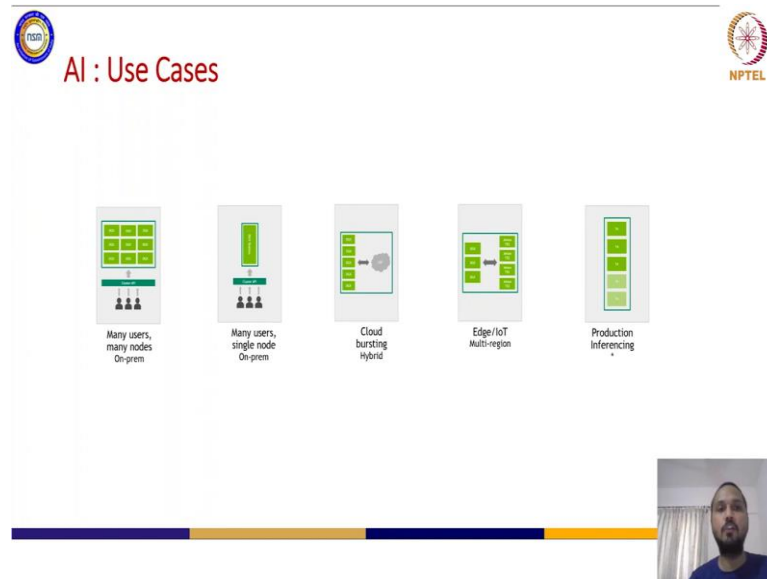
The slide features a title "What is a Cluster?" in red text. Below the title, a definition is provided: "A computer cluster is a set of computers that **work together** so that they can be viewed as a single system, **controlled and scheduled** by software". The slide includes logos for IIT Madras and NPTEL in the top corners. At the bottom right, there is a small video feed showing a man speaking. A decorative bar with alternating blue and yellow segments is positioned above the video feed.

So, if you look at the definition traditional definition of a cluster a computer cluster is a set of computers or nodes that work together and it may be viewed as a single system if required like if you are trying to solve the first set of problem which is you want to run a very large model across all of them.

You can consider that whole cluster as a one big single system and run your job across so many nodes or you can solve the other problem also, but they basically are required to work together and they need to be controlled and scheduled by a software, which means that you cannot just consider this cluster as one node which you have like your laptop and you are just running something on your laptop.

It is a system which is a heavy end system and it needs to be controlled and scheduled by a software to make sure that it is efficiently used the amount of power and all also which goes in powering up this cluster. So, we want to make sure it is very very efficiently used and that is why there are additional software stacks which come into the picture for this.

(Refer Slide Time: 22:13)



Let us look at the use cases where we need such kind of a system right. So, the first use case is when you have a case where you have; in your suppose you are in a college there are many students many users in your department and you also have many nodes on your premise right in your lab. So, what you are saying is a scenario where you have many users and all of them have also access to many nodes which is on premise which means it is there at your at your place at your disposal which you can see right.

Or there might be a case where you have only one system and there are many users which means that you want to access a system and there are many users and many users may be fighting for resources and they might even overlap the job with each other. The third type is more of the cloud bursting hybrid approach where you try to run things on your own machine as much as possible.

But once you exceed a particular threshold you have no other choice, but to start putting things onto the environment where the cloud becomes a natural extension to the work that you are doing on premise. So, you start with your own machines and when the workload increases you put or burst that particular part to the cloud vendors like maybe on Google cloud or Amazon cloud or whichever are the famous cloud vendors.

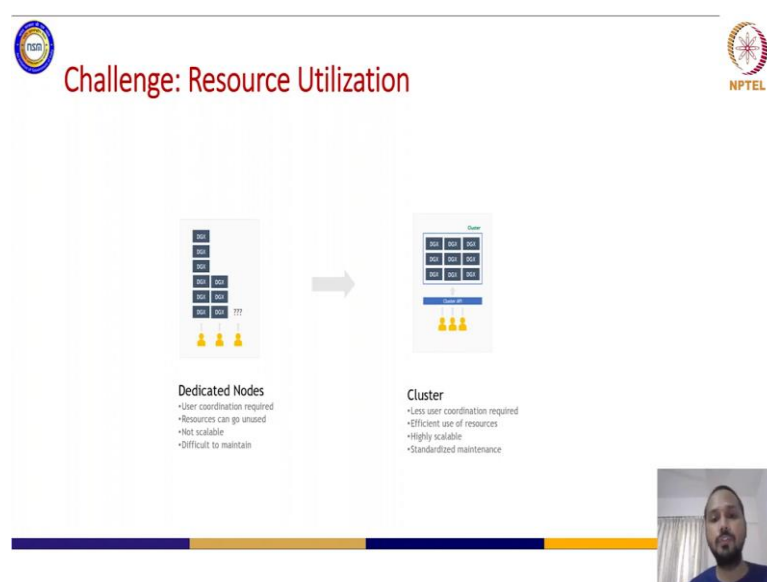
There is another scenario which we have not talked about so far, but which will be covered which is a very important part is the edge computing part which plays a very important role in terms of deployment where you have your machines where you are

doing all the training and then finally, you are going to deploy that to your embedded devices which may be scattered across and that is where you can also do the inferencing.

So, you can see here that based on the necessity of the AI application you have different requirements, but finally, it boils down to the part of how do I make sure that my job is transparent to me and I do not care where it is getting launched and I get a fair share of the resources which are there in front of me.

All of us face this issue especially you might many of you might be in this particular scenario where you have one machine where a department has bought and then you are trying to all of you are trying to access the same machine and you kind of are in a situation where there is a resource contention.

(Refer Slide Time: 24:53)



So, in order to solve some of this problem is what we are going to look at. So, the fundamental problem that we are trying to solve here is the resource utilization problem, where if you had your own dedicated node which means I have my own machine.

And then I can just do things on my machine and I do not care about the other folks they can get their own machines, but if I look at it from a point of view so, but when I talk about this scenario it is very rare generally in an environment like colleges or any other centers you would not get a dedicated node.

The user need to coordinate and among themselves and there might be a case where resource might even go unused if the coordination is not proper. It is a not scalable model and it is very difficult to maintain, you cannot just keep on adding machines and expect everyone to talk to each other and schedule their jobs.

While in a cluster environment where you have this particular software stack installed which I am going to talk about in certain duration, there is lesser user coordination required, you can use the resources much more efficiently like a particular user can ask; I need only these many cores, I need only this much memory and I need only 1 GPU so only give me this much, I do not even need the whole node to myself, so can you please give it to me and instances like that can be solved.

It is highly scalable which means that if you keep on getting more and more resources you can keep on adding and the users are not even aware that there are more nodes available, they just get the resources much more faster and it is very easier to maintain and that is what is the cluster environment which is generally used in no matter where the AI instances you will see in the case of a enterprise segment or if I talk about supercomputing where we have national supercomputing mission clusters spread across all of these IITs, IISC and all it is all cluster environment and all of us are required to understand the software to use these nodes.