Parallel Computer Architecture

Hemangee K. Kapoor
Department of Computer Science and Engineering
Week - 01
Lecture - 01

Lec 1: Why do we need parallel architecture ?

Hello everybody. We are going to start this course with the first module called the introduction to parallel architectures. The first lecture in this module is why do we need parallel architecture? As we are studying a course on parallel computer architecture, the motivation behind this topic is very much important and we will see all those things in today's lecture. Before we begin, we let us have a quick recap about what is computer architecture? So, it is an engineering discipline which should focus on the design of computers and when we have to design something as an engineer, we need to take care of the technology constraints and what are the demands from our customers.

So, who are our customers? The software designers or rather the end user applications which will run on the computer. So, the software demands and the technology constraints will drive how we design our computer architectures. So, in earlier days, computer architecture was limited to only ISA. ISA is instruction set architecture.

It concentrated on what type of instructions should be designed, how complex or simple they should be, what type of operations they should support. Each vendor was concentrating on their own ISA development and with this ISA came a series of softwares which supported those ISAs and which ran on the machines developed for those particular ISAs. Any advancement in the designs were on how better the new version of ISA is compared to the older versions. However, as time passed ISAs designs got saturated and there was nothing more significant to be designed. So, minor improvements happened on the ISA and to give better and better performance to the end user, we had to think of something more.

The concentration shifted from ISA design to whole system design like microprocessor design to how are the systems organized, what are the different components I am going to give, how much memory I am assigning, what is the bandwidth, what this capacity?I can support and so on and so forth. So, the performance and capacity were different aspects that were getting added to the design. So, ISAs as I said, not much changes are happening to the ISA, only little small improvements are taking up in the ISA sphere. To understand the complexity of a computer system, I would divide that into three layers. A software level layer, computer architecture layer and a hardware layer.

So, what is a software layer? All the applications and the programs which a user writes. A user writes a program in a high level language which gets compiled by a compiler and the compiler translates it to language which is understood by the machine on which it is going to run. All these applications also take help from the libraries and the operating system for doing IO and resource management. The software layer also consists of the compilers and the libraries provided by the system as well as the operating system. You all know that operating system helps in running all your systems.

It does all the resource management in a fair and secure manner across multiple users. So, multiple users can have a transparent view of the system no matter how many users are using that computer. So, this is the software layer. Below this is the architecture layer where primarily we would concentrate on the ISA because the compilers are translating the high level language to that understood by the ISA designed by the ISA. Below the ISA definitely is the computer system organization which organizes the various modules which comprise a complete computer system.

So, where is the computer architect? He or she is standing at the junction between the ISA and the operating system. The architect has to make sure that we understand what are the demands of the application, how the operating system requires support from the hardware as well as the architect also has to understand on what hardware support the system is going to run? What are the different circuit implementations? What are the circuit technologies? What is the semiconductor physics? All that an architect has to understand, so that he or she can design systems which are more efficient and can help better and faster software development. Right, So, next we will see that the applications and hardware are kind of tightly connected to each other. As the ISAs did not change over time, we saw that the performance had to be still improved and the avenues of performance improvement shifted elsewhere.

So, the applications wanted more performance in the sense they want more memory capacity, more storage capacity, more processing speed and more IO bandwidth. So, users started demanding this and therefore, the applications demanded more performance. So, what was the performance requirement? We required more speed, we required more capacity and more IO bandwidth. So, this was the requirement from the software. So, the hardware had to come up with ideas to support this.

Similarly, the hardware systems also evolved over the time giving more and more memory capacity and all these aspects which I have underlined. So, the hardware made efforts to give you more and more of this. So, as hardware made efforts of that giving more opportunities, the applications also tried to improve. So, the applications demands and performances improved and forcing the hardware system to improve further and so on. So, you can see this is a cyclic system.

So, there is a synergy between what the application demands and what the hardware system is able to support. So, they are both helping each other to grow constructively. So, this is I would say a self-perpetuating cycle. So, applications demand performance requirements which the hardware is satisfying. At the same time hardware is creating further opportunities for the applications to perform better and better.

Next, we are going to see the reasons behind the multi-core revolution. However, I will just point out a few things that the current hardware evolution is dictating that the software should become parallel because of the emergence of multi-cores. Now that we have multi-cores to take benefit of this hardware, the software should be able to utilize these multiple cores which are available. So, what does this mean? The software has to become parallel. It should be able to give work to all these available cores so that the overall system performance will improve.

And so, why multi-cores? Because the technology constraints which we will see shortly are driving towards multi-core. In other words, single threaded applications can no longer be speeded up at the same rate which we were doing it earlier. So, future processors will have multiple cores, running multiple threads in parallel. So multiple cores and multiple threads. So this is why we would need parallel architectures and why we need to understand how their design developed and managed.

So some motivational examples of why applications would need parallelism. So this slide shows various examples in science and engineering which need parallel processing. For example, weather prediction, evolution of galaxies, oil reservoir simulations. So all these applications which I have listed, they take into account multiple factors, VLSI CAD, nuclear bomb simulation, drug development and so on. All of these have multiple factors.

There is lots of parallelism and lots of computation which should be done to complete the task. So these are physical systems which are to be modeled and the model could be either a 2D model or a 3D model in a computer program. Most of the tasks are related to number crunching. So we have to do lots of computations and when lots of computation is there, we need faster computers. And when there is lots of computation, if the applications are written in a neater way, if we can harness the true parallelism in the design and in the problem statement, we can use the multiple cores provided by a parallel architecture.

So such applications would benefit from a multi-core or a parallel system. In the commercial aspect, you must be aware that now we have lots of online transactions, decision support systems and so on. All of these would run on several data center servers or cloud servers in the recent times. So online transaction processing, decision support systems, they would need high end servers for running. What do they involve? They involve lots of data movement.

So the previous slide said we had lots of computation to do. This slide says I have computation, but along with computation, I also have data movement, not much number crunching, but more data movement. For example, the OLTP has got small queries, but several of them. The decision support system has got larger queries, but fewer larger queries. So each has got a different demand and depending on the demand, your system should have facilities to cover both these requirements.

So overall, these applications require throughput parallelism, that is there is no number crunching, but the throughput at which I am able to send out applications or send out results should be as fast as possible. Next is the domain of multimedia. We are all aware that lots of speech processing is happening, audio, video, data compression. These are very familiar applications to each one of us. And all of this again requires lots of computation.

And remember that these are the domains where I can divide the task and give it to multiple threads or multiple processes to compute. So there is sequentiality, but there is more parallelism than sequential computation, which I can harness in such applications. 3D graphics is another one and the most popular is the gaming applications. Gaming applications won't feel realistic if you did not have multi core systems supporting the execution. So all these, what does it involve? It involves number crunching, that is lots of calculation.

You have data movement. As I said, that's true parallelism. There is really lots of work which can be divided across processors to be computed and definitely throughput parallelism because I want output at a fixed or better constant rate. So every unit of time I should be able to finish more tasks. So with these applications which are familiar to us, what has happened is with this dramatic advancement in science, internet, entertainment, yes if you will understand that this is a new domain which has come up in past few years which require lots of compute powers, have increased the computational demand. So the computational demand has increased and there are other avenues coming up.

For example, decoding the human genome, accurate medical imaging. This medical imaging, then web searches, definitely a very good avenue to work and more realistic computer games. So the computer games have become more realistic and faster. You really feel that you are really on a battlefield if you are using that type of a game or you are really racing a car if you are running a game like that. So we have more realistic computer games.

So for having all this, we need better and betters hardware systems which will support that. So we are doing good but we can't rest on our laurels. There are more open problems coming up. So you would think that am I done with all the development which was supposed to be done? So for the new generation, there are new problems and new avenues to work upon.

So some open and ongoing problems. This is an ongoing, I would say climate modeling. So climate modeling involves interaction with several components. You have to see the interaction of the ocean, the atmosphere, the land and the ice caps. So climate modeling is one ongoing open problem.

Then protein folding is another one. Now how do I use this? So people in the biotechnology or biosciences and engineering use this idea to identify the type of diseases and the solutions to them. So our ability to study further configurations. So this configuration depends on the compute power. The more the compute power I have, the better I can predict the diseases or understand how they affect each other. Then the other new avenue is drug discovery.

So we are analyzing the genome and we want to understand the effect of a drug on a particular disease or to give personalized treatment. So you want to give a personalized treatment meaning depending on the individuals. So every individual has a different characteristic. If I can study the individual's characteristic and give a personalized treatment that will identify what type of a drug would help this particular individual, that would be a very beneficial discovery.

So for this again we need compute power. Energy research definitely we need more green and clean energy and to understand this we have to model the wind turbines, the solar cells, the batteries. So efficient models have to be developed to understand the efficacy of all these systems. The other domain is data analysis. We are generating tremendous amounts of data these days. Data is doubling almost every two years and what do I do with this data if I do not analyze it? So unless the data is analyzed it is useless.

I want to analyze this data. What do I do with it? For example, I have data related to DNA's. I can do little with this if I do not analyze to find out its effect on the diseases.

So for this I need to check the effect of a particular type of DNA on a disease. Then we are generating lots of data in particle colliders, then medical imaging, astronomical research and also web search engines. So all this data needs to be analyzed and we need again hardware support to do this.

So these and host of newer problems are there, will be coming up and they will be unsolved without the vast increase in the computational power which we are expecting to support. So with this I have given you a motivation of why do we need parallel architectures? Thank you. .