Christian Dominic Angus

COT 4431 – Assignment 2 Part 1

1. **In how many ways can you classify parallel architecture?**

A generic parallel architecture will consist of three components, the processor, the memory, and the interconnection network, which according to a paper published by H.J. Siegel, is responsible for fast and reliable communication between the processing nodes in any parallel computer. This generic parallel architecture is then utilized either or both machine model, hardware based parallel architecture, and programming model, software based parallel architecture.

Parallel programming models are an abstract view of parallel computer architectures, where it allows the algorithms to easily express their structure in programs. Parallel machine models, according to a book published by Ian Foster, are a set of computers or nodes that are connected by an interconnection network, where each computer can execute their own program. The program may be able to access local memory and send and receive messages over the network, these messages are used to communicate with other nodes and read and write those messages into their own local memories.

2. **What are the different programming models that are used by these architectures above?**

There are four types of programming models; shared memory, message passing, data parallel, and hybrid. First is “shared memory,” where a small number of processors are attached to a single memory. The data is decomposed or divided into parts, where the partial sums of the

tasks are taken into the processors for evaluation, and then once all the data has been accounted for, they are then stored finally as a global sum. There are two classes of data in this programming model; logically shared and logically private. Logically shared data are the original values, the data before it is divided up into parts, and the global sum, the result of the processed data.

The second programming model is “message passing,” where messages can be sent between processors that are not necessarily on the same computer or machine. MPI or Message Passing Interface have created a de facto standard on how these messages are sent and received between processors, where if Processor A were to send a message to Processor B, B would need to receive the message first before it can send one to A, in order to avoid potentially causing errors where they may have sent and received messages at the same time. This programming model is the most scalable out of all four programming models.

The third one is “data parallel” programming model, where a single thread of control consists of parallel operations. Coordination between operations is done implicitly, meaning they are not done by the end user or the programmer, and statements are executed synchronously. Despite it being easy to understand, one of the main drawbacks is that not all problems fit into this model, one of the reasons being that they are difficult to map onto coarse-grained machines.

Lastly is the “hybrid” programming model, which can consist of other programming models such as message passing, data parallel models or shared memory models. Supercomputers are programmed this way to fully realize their capabilities and achieve full performance.

3. **What do we mean when we say, “parallel programming model”? What are the different components of such programming models?**

A parallel programming model is made up of languages and libraries that create an abstract view of the machine. These abstract views are important because these allows us to create parallel algorithms. It makes the algorithm abstract or as simple as needed by the program. It consists of four components; control, data, synchronization, and cost.

Control dictates how parallelism is performed, whether it is done because there are more processors available, the machine does the parallelism for you, implicit parallelism, or because the data is divided into parts, you are doing the parallelism yourself, explicit parallelism. Control also dictates the orderings of operations, which tasks take priority over another, which one is performed first, or which one is performed last.

Data controls what data is private and what data is shared. If data is shared between processors or compute units, the parallel programming model dictates how the data is accessed or communicated and what computing processor takes priority of writing on the memory address.

Synchronization oversees structuring which operations are done in the algorithm that ensures that the results we get from the parallel algorithm are correct. It also handles atomic operations, operations that operate independently from other operations. Synchronization also creates mechanisms that optimize which processors operate to fully take advantage of the parallelism capabilities of the computer.

Cost, lastly, oversees allocating resources. It oversees knowing how much it costs to move data from one processor to another, how much it is going to cost to access data from a

specific memory address, what is the cost of decomposing or breaking up the data. Essentially dictates how efficient the algorithm is going to be, because performing the algorithm operations is one thing but how much resource you spend or save is also important in parallelism.

4. **What is synchronization and what is a barrier? Why might they be needed in a parallel algorithm? Give examples**

Synchronization is where threads work together using global and shared data. A barrier then is a point in a program where threads stop and wait for other threads to reach the barrier before they can proceed with the rest of the task. Barriers are important in parallel algorithms because it ensures that threads do not run into situations where one thread may read a result before another thread can write it.

An example for applying barriers in parallel algorithms is if you were to write a program that rolls two dice on two separate threads and if both dice land on the same number, you win, else you lose. Your barrier then is that if one die has not yet finished generating a random number, the other cannot proceed to roll again, your program must wait until both dice have rolled before it can proceed and do the roll again.

5. **What kinds of programs would NOT be a good candidate to parallelize (at least trivially) on a multicore architecture?**

Video games previously did not fare well in multicore architecture, despite the advent of multicore gaming computers in the late-2000's, because games back then were not able to fully utilize multicore architecture.

Running a video game is data intensive, requiring for your computer to access assets from your drive storage, where your processor then needs to compute where the models needed to be generated and then your graphics card filled in the rest of the video game. Accessing all that data was resource intensive and game developers opted to focus their resources more on single-core processing, which is one of the reasons why SLI and Crossfire technology from NVIDIA and Radeon, respectively, have been phased out. Having multiple graphics cards on separate PCI-E slots caused games to stutter because the gap between both graphics cards was far too wide to be accessed quickly by the processor, memory and storage drive.

Nowadays, games are still leaning towards single-core processing, with AMD's recent 8-core CPU easily beating its 16-core brother in benchmarks performed by AnandTech, proving that focusing on single-core computing process will still give you a better overall performance in video games.

6. **What is meant by "race condition"? Explain it with the help of an example.**

Race conditions are locks in the programming model that ensure that one thread does not perform a task before the other, therefore avoiding getting incorrect results, as well as ensuring that the correct order of tasks is performed. Race conditions are used when a global shared variable is accessed by multiple threads at once. The process of locking and unlocking is done

implicitly, therefore the user has no control over which threads lock and unlock first or which thread performs the task first and which one performs the task next.

An example of a race condition is like purchasing a ticket for an event, let's say San Diego Comic-Con, which is well-known for their waiting room process ticket purchase system. Where essentially potential purchasers enter their website and are sent to waiting rooms, users are then selected at random to purchase tickets and are sent to a new webpage where they are allowed to purchase tickets. Once the user has purchased their tickets, a new set of users are allowed to purchase tickets.

This is essentially their race condition, with a finite number of tickets available for purchase, after a random set of users purchase their tickets the value for the number of tickets available are locked and updated before the next set of users can purchase their tickets. So that they do not sell tickets that may have already been purchased by previous users had they not done their waiting room system.

Works Cited

Bonshor, Gavin. “The AMD Ryzen 7 5800x3d Review: 96 MB of L3 3D V-Cache Designed for Gamers.” RSS, AnandTech, 30 June 2022, https://www.anandtech.com/show/17337/the-amd-ryzen-7-5800x3d-review-96-mb-of-l3-3d-v-cache-designed-for-gamers/4. Foster, Ian. “1.2 A Parallel Machine Model.” 1.2 a Parallel Machine Model, 1995, https://www.mcs.anl.gov/~itf/dbpp/text/node8.html.

Hagedoorn, Hilbert. “Nvidia Ends SLI Support and Is Transitioning to Native Game Integrations (Read Terminated).” Guru3D.Com, Guru3D.Com, 2020, https://www.guru3d.com/news-story/nvidia-ends-sli-support-and-is-transitioning-to-native-game-integrations-(read-terminated).html.

Saeed, Fahad. "Lecture 3 (Part 1) PowerPoint (PPT)." Canvas, 2020. PowerPoint Presentation.

Saeed, Fahad. "Lecture 3 (Part 2) PowerPoint (PPT)." Canvas, 2020. PowerPoint Presentation.

Saeed, Fahad. "Lecture 3 (Part 3) PowerPoint (PPT)." Canvas, 2020. PowerPoint Presentation.

Saeed, Fahad. "Lecture 3 (Part 4) PowerPoint (PPT)." Canvas, 2020. PowerPoint Presentation.

Saeed, Fahad. "Lecture 4 PowerPoint (PPT)." Canvas, 2020. PowerPoint Presentation.

Saeed, Fahad. "Lecture 5 PowerPoint (PPT)." Canvas, 2020. PowerPoint Presentation.

Siegel, H. J. “H. J. Siegel.” H. J. Siegel Homepage, 2008, https://www.engr.colostate.edu/~hj/.