Cody Adams

CSC478-01

Winter 2010

Exam 1

1a) Sp=T1/Tp=20/11.25=1.77

Tp=S+(Q/P)= 10+(10/8)=11.25

Ep=1.77/8=0.222

1b) Sp=T1/Tp=20/4.25=4.71

Tp=S+(Q/P)= 2+(18/8)=2.25

Ep=2.25/8=0.588

2.) The four types of Parallel Random Access Machines (PRAM) are: Single Instruction, Single Data stream (SISD), Single Instruction, Multiple Data streams (SIMD), Multiple Instruction, Single Data stream (MISD), Multiple Instruction, Multiple Data streams (MIMD). SISD is the simplest type of machine, where a single processor works on a single data stream with no parallelization. A SIMD machine works with data level parallelization, where multiple processors work together on a set of data by segregating the data to be worked on in pieces. This is most common for display operations, where a large number of pixels all must have some operations done to them. The data is separated and spread across processors so the work can be done in a less time. A MISD machine works multiple instructions on the same data to achieve parallelization. A good example for this is the simple pipeline model in CPUs (Fetch, Decode, Execute, Memory Access, and Write Back) where the same piece of data is has different operations performed on it in a pipeline to achieve performance gains. Today’s traditional processors are MIMD machines, where multiple and independent cores work on separate and different data at the same time to achieve the highest maximum parallelization.

3a) Assuming that there is no time penalty for backtrack (ex. Visited nodes are pushed onto a stack with their child pointers and then popped off when needed) it takes visiting 11 arcs before finding the solution with DFS. 11 time units to find the solution.

3b) Once it is parallelized, it only takes 4 time units, yielding a speed-up of 11/4.

4a) S3 and S5 have a data flow dependency on S2. No other dependencies exist.

4b) All of the operations can be run in parallel if S2 is substituted into S3 and S5 for a.

5a) .4 \* 16 GFLOPS = 6.4 GFLOPS

.4 \* 700 GFLOPs = 280 GFLOPS

.2 \* 14 GFLOPS = 14 GFLOPS

300.4 GFLOPS

Using the harmonic mean;. The average execution rate is 36.176 GFLOPS.

5b) 1 000 000 000 GFLO / 35.176 GFLOPS = 28428473.96 secs or about 330 days (???—WOW!)

6.)

7a) Thread—a thread is a small piece of execution code that is the result of the fork of a running program, where two or more operations are running concurrently. Threads come from processes, but they differ from processes in that they share resources where processes do not.

7b) Race Condition—a race condition exists when two or more processes attempt to access the same piece of shared memory at the same time and strange dependencies are created. Since there is no guarantee which process will get to the data first, unpredictable behavior can easily occur. It is named after the idea that the two processes are ‘racing’ to affect the output first.

7c,d) Mutex— short for mutual exclusion, a mutex is a programming object that is used by a thread to ‘lock’ a resource so another thread cannot use it until the first thread is complete. This can lead to problems, though, such as lock contention(access attempt by another thread to a resource already locked by another process, priority inversion (a high priority thread waits for a lower priority one, since it has locked the resource), or starvation(situation where a thread cannot complete because cannot get enough resources, as they are all locked—the worst case resulting in deadlock).

7e) Granularity— the level of granularity is a relative measure of the amount a system can be broken down into smaller pieces. Granularity is generally either coarse or fine, and analogously, a system with coarse granularity consists of a smaller number of larger pieces, where a system with fine granularity has many, smaller pieces. Specific to parallel programming, granularity speaks to the communication to computation ratio. Fine granularity has many small, fast tasks that require much communication between them, whereas coarse granularity has larger, more time consuming tasks that require less inter-task communication.

7f) False Sharing is a problem that occurs when two threads are working on different data that resides in the same cache line. Even though both threads are looking for different data, any threads after the first will want for the entire line of cache to be available (unlocked) by the first thread before proceeding. To alleviate this, you can have threads work on a local variable rather elements of a related data set (that will be cached together as one line) and then put then results together. Essentially, you must remove dependency of several threads to the same cage line.

8.) If the summation operations are replaced with comparison operations, you can then use the algorithm to find the largest value in an array. That is;