**Part 1 (15p) Race condition**

1.(2p) What is race condition?

Race condition is the computer behaviors where the resulted output is controlled by unexpected events. Usually it cause a bug in the program and need to be fixed.

(5p) Why race condition is difficult to reproduce and debug?

Race condition has two distinct conditions. One, it is nondeterministic. Second, the program is depending on the relative timing among the several threads. The problems may not shows using the debugging mode, so the programmer must go threw step by step (line by line) in order to find the bug.

(8p) How can it be fixed? Provide an example from your Project\_A3 (see spmd2.c)

Synchronized one threads to another threads. Synchronizing the thread would make the one threads result to visible to another. This may be equivalent as sharing the data among the threads. One example could be found from the Project A3. For example, using “Uniform Memory Access(UMA)” using OpenMP would allows to shares the memories among threads, and make visible the result information to another threads. Using this, we could task on thread by “check and act” using other threads.

**Part 2**

(15p) Summaries the Parallel Programming Patterns section in the “Introduction to Parallel Computing\_3.pdf” (two pages) in your own words (one paragraph, no more than 150 words).

“Patterns” is the way the computer to perform the reparative task efficiently. It has many types of patterns, but the programmer must consider pattern’s “strategies”, and “concurrent Execution Mechanisms Start with strategies, the programmer must know the what algorithm to use to run the program efficiently, and how would programmer implements those algorithm in parallel program. Using these two skills for strategies would make the “patterns” to run reparative job efficiently. Next is “Concurrent Execution Mechanism. This is related to the computer model (not programming) which is related to the computer Architecture. Programmer must know about the his/her own processor and threads, and parallel programming like OpenMP. There is two There is two main types for coordination pattern. Those are 1. message passing, and 2.mutual exclusion. These two are used in different scenario.

**Part 3**

(12p) In the section “Categorizing Patterns” in the “Introduction to Parallel Computing\_3.pdf” compare the following:

o Collective synchronization (barrier) with Collective communication (reduction)

Collective synchronization and Collective communication are both categories as Concurrent Execution Mechanisms. The collective synchronization is using a Barrier system, and collective communication use reduction system. The collective synchronization (a barrier) synchronize the threads across the all groups. The system would block all system until all group member (of system) response answer the call. The collective communication(reduction) has little different from barrier which the result of the program is returned to all group members and it will verify the result to only one member.

o Master-worker with fork join

Two of the patterns are categorized as “Program Structure”. Fork join parallelism process the program and merge the data at the one point to resume the sequential execution. This is idea where the processor work separately and merge the process in to one point.

The master worker parallelism allow program to perform simultaneous process across multiple threads (sometime machine) to finish the tasks.

**Part 4 (26p) Dependency: Using your own words and explanation, answer the following:**

(3p) Where can we find parallelism in programming?

Parallelism can be found from the program view. The program execute the code line by line in parallelism, and each(any) statements that use the tasks simultaneously represent the parallel program. Several distinct levels in parallelism are Block level, Loop level, routine level and process level.

(6p) What is dependency and what are its types (provide one example for each)?

Dependency are a program that the operations is require the previous operation result to compute the current operation. The result of the current operation is depending on the previous result. One example is flow dependency, output dependency and anti-dependency

Dependency example

Flow(true) dependency == a=2 , b=a

Output dependency a= 3 . a= 2

Anti dependency a=b b=2;

(3p) When a statement is dependent and when it is independent (Provide two examples)?

When previous/next statement does not impact the next/previous statement, then the statement is independent. Else, the statement is dependent. For example. A=1 b=2 is a independent since a and b are separated from each other. However, a=1 b= a is dependent because the b’s result is depend on a’s result.

(3p) When can two statements be executed in parallel?

When two processor run simultaneously. (not a sequential statement)

(3p) How can dependency be removed?

By rearranging the statement’s order and by eliminating the statement.

(8p) How do we compute dependency for the following two loops and what type/s of dependency?

(pictures from pdf)

The first(left) loop is a doall loop. All iterations of the loop is a dependent from i. This is a true(flow) dependent since the a[i] is constantly changing depend on ith number. The computer compute this program sequentially where a[1] ….. to a[100].

Second loop, on the other hand, is a dependent (more likely anti-dependent) where statement s1 impacts the statement s2. The computer would perform s1 first , then using that result to s2.

S1[1] => S2[1] => S1[2] => S2[2] ………