Vinh Nguyen - vinh.nguyen@ttu.edu

Assignment #2

CS4379: Parallel and Concurrent Programming
CS5379: Parallel Processing
Spring 2020

Due Date: 2/18, 12:30 p.m., please submit a soft copy via Blackboard (preferred) or a hard copy in class. Late submissions are accepted till 2/25, 12:30 p.m., with 10% penalty each day. No submissions accepted after 2/25, 12:30 p.m.

Please name your submission file starting as "LastName_FirstName_HW2".

Q1. What is the condition that two statements can execute in parallel?

ANSWER:
In order for two statements to be executed in parallel, their order of execution must not matter. In other words

statement1; statement2

must be equivalent to

statement2; statement1

In other words, there is no dependency between the statement (S1) and the statement (S2). Such dependencies can be listed as:

- True dependences
- Anti-dependences
- Output dependencies
- Loop-carried dependencies
- Control dependencies

Q2. Please give an example of output dependence and give a corresponding solution to remove the dependence.

ANSWER:
Statement S2 has an output dependence on statement S1 if

$$O(S1) \cap O(S2) \neq \emptyset$$

S1 has a write and is followed by a write to the same location in S2 (that is, write after write)
For example:

```
a = 1;
a = 2;
```

To remove output dependence, renaming of variables can be considered. For example

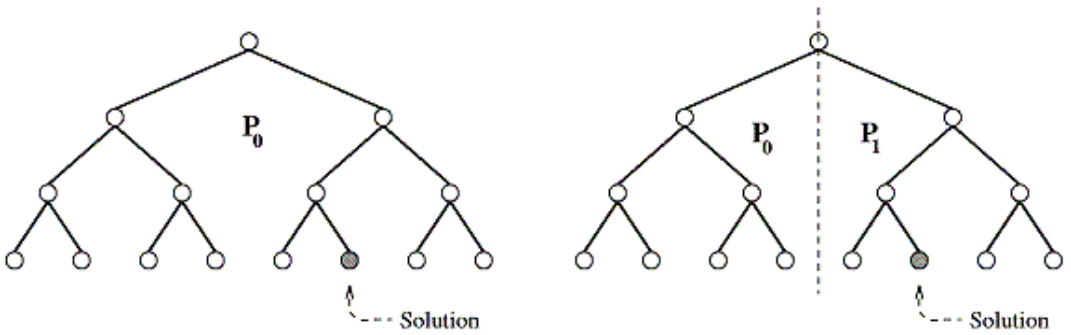
```
A = 5 * X
B = A / 4
```

A = 7 * Y

can be rewritten as

A2 = 5 * X
B = A2 / 4
A = 7 * Y

Q3. Consider the search tree shown in the following figure, in which the dark node represents the solution.



(a) DFS with one processor (b) DFS with two processors

(a) If a sequential search of the tree is performed using the standard depth-first search (DFS) algorithm, how much time does it take to find the solution if traversing each arc of the tree takes one unit of time? Note: DFS begins by expanding the initial node and generating its successors. In each subsequent step, DFS expands one of the most recently generated nodes. If this node has no successors (or cannot lead to any solutions), then DFS backtracks and expands a different node.

ANSWER:

Since the DFS algorithm only visits each node one time and expands the search from the left nodes. Time taken to reach the solution equals the number of edges counting from the left side to its current position. Thus

$$T_S = 11$$

(b) Assume that the tree is partitioned between two processing elements that are assigned to do the search job, as shown in figure b. If both processing elements perform a DFS on their respective halves of the tree,

- how much time does it take for the solution to be found? Since the two processing elements work independently, the amount of the required to find the solution equals the number of edges on the second half of the tree reaching to the dark node which is **4 units of time**
- What is the speedup? The speedup is calculated as the fraction of sequential search over parallel search

$$S_{speedup} = \frac{T_s}{T_p} = \frac{11}{4} = 2.75$$

- Is it a linear speedup, or a super-linear speedup, or a sub-linear speedup? This is a superlinear speedup since the speedup (2.75) is greater than the number of processing elements (2)

Q4. (a) Please derive the fixed-time scaled speedup formula (Gustafson's Law). Assume a sequential ratio α of a problem cannot be parallelized, and the number of processing elements (processors) is p. **(b)** Please give two scenarios of why we want to have scaled computing.

ANSWER:

(a). The Gustafson's law states that a task executed by the system can be splitted into two parts:

- a part that can NOT be benefit from the improvements of the system's resources
- a part that can be benefit from the improvements of the system's resources

The total workload of the system is the sum of the workload on the two parts

$$W' = aW + (1-a)pW$$

where a is the proportion of workload that cannot be parallelized and p is the number of processors. The speedup is defined as:

$$S'_p = \frac{Uniprocessor\ time\ of\ solving\ W'}{Parallel\ time\ of\ solving\ W'}$$

However, the parallel time of solving W' equals the best time of solving W in sequential algorithm on uniprocessor so that

$$S'_p = \frac{Uniprocessor\ time\ of\ solving\ W'}{Parallel\ time\ of\ solving\ W'} = \frac{Uniprocessor\ time\ of\ solving\ W'}{Uniprocessor\ time\ of\ solving\ W} = \frac{W'}{W}$$
$$= \frac{aW + (1-a)pW}{W} = a + (1-a)p$$

(b). Please give two scenarios of why we want to have scaled computing.

- s1: The first scenario is to solve a larger problem that a small machine can not fit. For example, a big dataset can not fit to a small cache on a uniprocessor. However, when the dataset is splitted into multiple data items, these items can be fitted into caches of multi processors.
- s2: The second scenario is to provide a real-time solution where time is matter. For example, weather forecasting, stock price forecasting. Data for this problem is usually big (terabyte). Responses should be available after a few hours or even minutes. Small computer may not fit to this case

Q5. Please list three reasons why scaled computing is desired.

ANSWER:

- R1. The first reason is the feasibility of solving larger problems. For example, a big data size can not fit into a cache of a uniprocessor but once it is splitted up into smaller items, they can be fit into caches of multiple processors
- R2. The second reason is to maintain efficiency of the system's resources. As an increasing number of processors will solve a larger problem, however the sources of overhead are also

increased. Scaled computing is desired to alleviate this problem while maintaining the efficiency (e.g., increase workload)

- R3. The third reason is the desire to provide a real-time solution such as weather forecasting, stock market prices… since these problems need to have the solution in a timely manner.

Q6. (a) Please study the following paper (can be found from the Blackboard) and write a 500 – 1,000 words of the summary of the paper. The summary should include: **1) problem statement**: what problem is studied in the paper; **2) solution**: what is the solution proposed in the paper; and **3) contribution**: what is the impact of the solution.

- M. Hill and M. Marty, "Amdahl's Law in the Multicore Era", IEEE Computer 2008

ANSWER:

The paper provided a quantitative study on how different multicore processor architecture affects the performance of the system in terms of speedup metric. They adapted the Amdhal's law in different designs and tried to answer the following research questions (or ***problem statement***) such as: 1) how many cores should be designed? 2) Should cores used a simple pipeline or powerful multi-issue pipeline design? 3) Should cores use the same or different microarchitectures? and 4) Should the designer be taken care of dynamic multicore chips?. To answer these research questions, the authors measured the speedup of the system from different schemes with different amounts of tasks that can be parallelizable (or ***the solution***). Before the measurement, the authors made two assumptions that a) a given size and technology generation of a multicore chip can contain at most *n* base core equivalents, where a single BCE implements a single baseline core and b) micro architects have methods for using resources of multiple BCEs to create a core with a greater sequential performance. The measurement was conducted in 3 scenarios such as symmetric multicore chips, asymmetric multicore chips and dynamic multicore chips. Each scenario had two schemes, one for a small number of r BCEs (n=16) and the other for larger numbers of r BCEs (n=256). The authors reported six results along with the implications for researchers (***the contribution***). Their finding suggested that 1) increasing the amount of parallelizable task (or *f*) will increase the speedup as the Amdhal's law still hold for multicore chips 2) new methods should be explored to increase core performance even at a higher cost because using more BCEs per core can be optimal, 3) researcher should look a way to design more powerful cores by showing an example when the number of chips (n = 256 chips) can help in the case of moving to denser chips, 4) asymmetric multicore chips should be investigated and it is shown to be more efficient than symmetric multicore chips, in addition scheduling and overhead challenges should be carefully taking into account because the Amdhal's law does not capture these issues, 5) new algorithms for speeding up sequential operations should be studied even though it may cause local inefficient because new algorithms may globally efficient as they reduce the sequential phase when the other chips (n-r) are idle, 6) new methods for approximating a dynamic multicore chips should be investigated because its performance is greater (never worse) than asymmetric chips in terms of speed up. Although it is difficult to apply under the Amdhal's assumption, they could flourish for software with substantial phases of intermediate-level parallelism. Although the quantitative results provided insight on each design, they are susceptible due to the complexity of the real-world problem. For example, it is challenging to scheduling software tasks on asymmetric and

dynamic multicore chips since it adds overhead or developing a parallel software is more difficult than developing a sequential software. Overall, the authors suggested the multicore chip designers should have a broad view of the entire chip's performance rather than looking at core efficiencies individually.

(b) Discussions. What new performance model (speedup/scalability) can you think of, or what new problems/ideas/hypothesis can you think of in terms of modeling and evaluating parallel computers (such as the paper you have just studied), or does the paper you have studied have any issues/shortcomings/limitations and how can you address them? Please provide a 300 – 600 words of discussion.

ANSWER:
Because the quantitative study result presented in this paper is suspect, the new performance model should consider the granularity of the complexity. For example, one can add overhead factor into the equation model or dynamic power so that the evaluation would be more accurately. Currently, the evaluation model works on the assumption that all processing elements have the same performance but this is not the real case in practice, even the same set of instruction can be resulted at a different amount of time, meaning that the evaluation model should take into account of latency (or error). For example, speedup can be written as

$$S = Ts/Tp + E \text{ (error)}$$

When doing experiment, the speedup should be measured as many time as possible so that the mean value of E is close to zero

The drawback of the study is the suspected quantitative result due to the complexity of the real world problem. Regarding the model, it does not take into account the effect on dynamic and static power, on and off chip memory system and interconnect design. To address this issue, sophisticated models were developed to validate the Amdhal's law to future systems, particularly embedded ones

If this is the first time for you to read a scientific paper, please consider reading the below two papers regarding "how to read a paper" first.

[1] "How to Read a Paper", By S. Keshav, University of Waterloo.
http://ccr.sigcomm.org/online/files/p83-keshavA.pdf

[2] "How to Read a Research Paper", By M. Mitzenmacher, Harvard University.
http://www.eecs.harvard.edu/~michaelm/postscripts/ReadPaper.pdf

THE END.