MONASH University

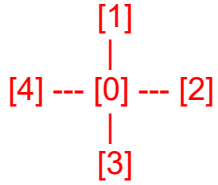
[Template for Applied Session submission]

Applied Session Week {Week 8}

Student ID	Student Name	Student Email
32837933	Chua Sheng Xin	schu0077@student.monash.edu

Tasks

{Answers
Question 1
A) Diagram: [0] -> [1] -> [2] -> [3] -> [4] -> (back to 0)
In a ring topology, nodes are connected in a circular manner. If the messages can only be sent in one direction and each message must be sent sequentially due to the restriction that only one message can be sent at a time, the root node (rank 0) should broadcast 4 messages to the other 4 nodes in the same direction sequentially to minimize time consumption. The messages can be sent to the closest connected nodes first followed by the next node in the direction until the message is delivered to the furthest node from the root node. In this order, the time that is taken would be 0.5 + 0.5*2 + 0.5*3 + 0.5*4 = 5 seconds since travelling to a node that is directly connected would require an additional 0.5 seconds.
B) In a star topology, all non-root nodes are directly connected to the root node. Thus, messages from the root can be sent directly to each node without having to relay them every other node.
Diagram:



Since the root node can send messages directly to each node, it will send a message to each node one after the other. The time taken would be 0.5*4 = 2 seconds since every message takes 0.5 seconds to send. This is significantly better than the ring topology in Part A since the root node is directly connected to every other node and this reduces the time from 5 seconds to 2 seconds.
C) If the network is upgraded to allow multiple messages to be sent simultaneously, I would not want to continue sending messages in one direction only in Part A. This is because in this case, the root node could send messages to multiple nodes at the same time. Instead of sending messages sequentially, the root can send a message to each node simultaneously. Since the messages travel in one direction in a ring topology, the furthest node (rank 4) could receive the message in 2 seconds rather than 5 seconds in Part A. Therefore under this upgraded network condition, sending in only one direction would be sub optimal.
D) I would design the program using the ring topology. This is because only one message needs to be sent through the entire network to visiting all the nodes and for this a ring topology is better suited than a star topology as the message can travel to each node in sequence without needing to return to the root node. The program would send the message from the root node (rank 0) to node 1, which would then forward it to node 2 and so on, until it reaches the last node (rank 4). Using a star topology for this would be inefficient as the message would have to return to the root node after each hop.

MONASH University

Question 2

A) The time spent on the network/fabric can impede the maximum speed-up of a program if it is large enough. Amdahl's Law states that the speed-up of a program is limited by the serial part that cannot be parallelized. The time spent on the network or fabric can contribute to this serial part if the latency involved in communication between processors is sufficient. When the network is slow, the time spent waiting for data to be transferred over the network/fabric adds to the serial part of the computation, thus reducing the speed-up. Even if the computational portion of the program is fully parallelized, poor network performance can limit the maximum speed-up achievable. Therefore the time spent on the network/fabric is an obstacle to the maximum speed-up of a program.

B) The percentage of serial code for Bob's program is 25%.

Amdahl's Law is given by:

Speed-up(N) = 1/(S + P/N), where N is the number of processors, S is the serial part of the program and P is the parallelized part of the program.

Let Speed-up(N) = 4 and P = 1 - S since S + P = 1.

4 = 1/(S + (1 - S)/N)

As N approaches an infinitely large number, (1 - S)/N is approximated to 0.

4 = 1/S

S = 1/4

S= 0.25

Hence, 25% of Bob's program consists of serial code while the other 75% is parallelizable.

C) We cannot confirm that Bob's program must be able to gain a speed-up larger than 4. This is because even if the network latency and bandwidth are improved, the maximum speed-up is limited by the serial part of the program in general. The serial part of the program consists not only of cost of network/fabric transmissions but more importantly the computations of the code according to Amdahl's Law. If the computation of serial part is significant enough, it might still prevent Bob's program from being able to gain a speed-up larger than 4 even with the improvement of the network/fabric with better latency & bandwidth.

Question 3: Partitioning

A) The partitioning of the problem is not ideal and the workload distribution is not efficient because Bob's partitioning strategy involves assigning one task per core but each task depends on previously computed values of the recurrence function. Since each function f(n) depends on f(n−1), f(n−2), and f(n−3), solving the problem in incremental order creates dependencies that prevent the tasks from being independent. For a machine with 64 CPU cores, each core has to wait for another to finish processing earlier values to begin the processing of the following code. This results in a non-ideal partitioning of the problem where idle cores that wait for others to finish exist and leads to inefficient workload distribution.

B)The recurrence relation/function in this example is not easy to be parallelized as it introduces dependencies between the results of different tasks. Each calculation of f(n) depends on the results of f(n−1), f(n−2), and f(n−3) hence it is impossible to compute all values of f(n) independently. This limits parallelism of the program because tasks cannot be executed simultaneously without completing the calculations for the smaller n values first.

C) A close-form expression will usually be a better choice for parallel computing because it eliminates dependencies between values and allows each instance of the function f(n) to be computed independently of others. This means that the problem can be fully parallelized across multiple cores for more efficient utilization of processors. Compared to

recurrence relations, close-form expressions avoid sequential bottlenecks and are far more suitable for parallel computing.
}

{Screenshots}

{References
I have used ChatGpt in generating answers.}