Assignment 3
Comp 428

Maximilien Malderle

Id: 26562906

Nr 2:

The advantages of this search strategy, is that a tree can be searched much more quickly, by having multiple processes working on the same structure simultaneously. An efficient scheduler makes sure that each branch can be handled by its own process. This massively reduces the minimum search time, due to only sub sections of the structure being handled. A possible issue, would be having too many processors for a tree, leaving many with a work load that is too fine grained and in turn having many idle processors. There is no guarantee that this algorithm is more efficient than its serial equivalent, many processors are dispatched, guaranteeing a lower maximum search time, but not providing consistent outperformance.

Nr 3:

Case i:

Where no process j sends to process i where $i < j$

No matter what happens no process will be set to black (invalid), meaning that whenever the token is sent by a process, it will never be turned to black (invalid). So when the token arrives at P0, the program will terminate successfully, when P0 becomes idle.

Case ii:

Where there is a process j sends a job to process i where $i < j$.

Whenever this occurs process j is turned to black, the token is moved from j to i, meaning that it will need to pass process j again, thus turning the process to white (valid) and the token to black (invalid). This means that the token will need to pass P0 and Pi again, assuring that all ongoing processes have the ability to update the token again, restarting another pass. Whenever a process does not send a message back, all processes will be white (valid) meaning that the token will not be turned to black (invalid), so all processes will be done when the token is received by P0 and P0 is idle.

Nr 4:

a)

E max = 1/ (2 - (1/n))

E max = 1/(2 - (1/6)) = 0.5454

b)

Yes the process can be more efficient. This is done by keeping all the processing elements on a local process. The Strategy here would be to run one longest common subsequence on one processor and the other on a separate processor. If F[I,j] is of a comparison between A[i] and B[i], then process 1 will focus on the comparison where the result is in A and process two will focus on the comparison where the result is in B. This means that each process will complete in time n.

Speed up = n2 / n = n

Efficency = n/n = 1