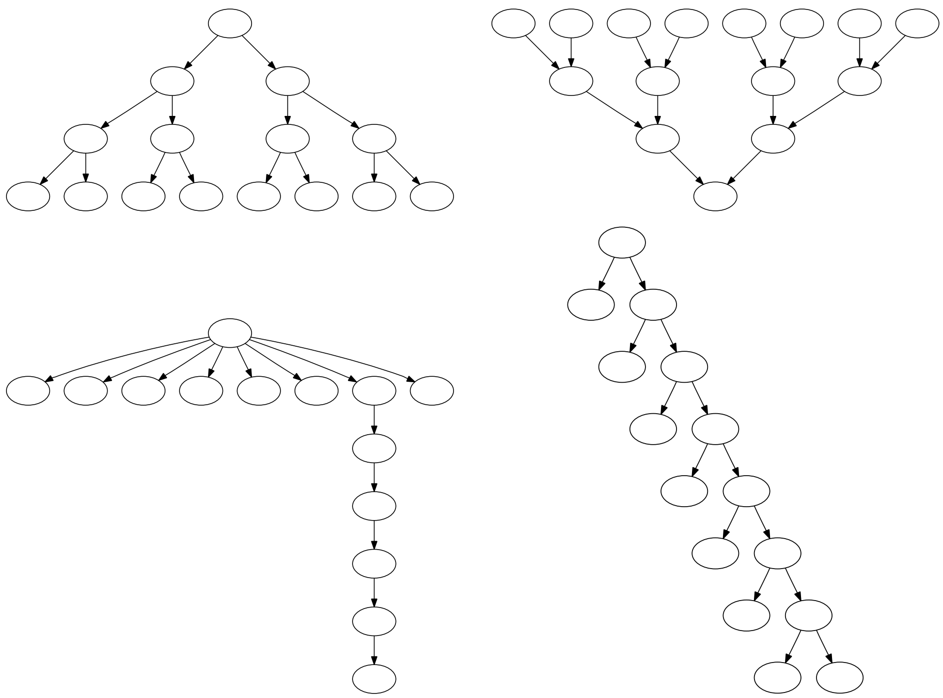
Department of Mathematics and Computer Science
University of Southern Denmark
Daniel Merkle

Exercises for Week 46

Parallel Computing, DM818 (Fall 2015)

Exercise 1

Task Graphs



For the task graphs given above, determine the following:

a) Maximum degree of concurrency.

b) Critical path length.

c) Maximum achievable speedup over one process assuming that an arbitrarily large number of processes is available.

d) The minimum number of processes needed to obtain the maximum possible speedup.

e) The maximum achievable speedup if the number of processes is limited to (a) 2, (b) 4, and (c) 8.

Exercise 2

Task Graphs

Let d be the maximum degree of concurrency in a task-dependency graph with t tasks and a critical-path length l. Prove that $\lceil \frac{t}{l} \rceil \leq d \leq t - l + 1$.

Exercise 3

Routing

Consider the routing of messages in a parallel computer that uses store-and-forward routing. In such a network, the cost of sending a single message of size m from P_{source} to $P_{\text{destination}}$ via a path of length d is $t_s + t_w \times d \times m$. An alternate way of sending a message of size m is as follows. The user breaks the message into k parts each of size m/k, and then sends these k distinct messages one by one from P_{source} to $P_{\text{destination}}$. For this new method, derive the expression for the time to transfer a message of size m to a node d hops away under the following two cases:

a) Assume that another message can be sent from P_{source} as soon as the previous message has reached the next node in the path.

b) Assume that another message can be sent from P_{source} only after the previous message has reached $P_{\text{destination}}$.

For each case, comment on the value of this expression as the value of k varies between 1 and m. Also, what is the optimal value of k if t_s is very large, or if $t_s = 0$?

Exercise 4

Task Dependency Graph

$$A = \begin{array}{|c|c|c|c|c|c|c|c|} \hline \bullet & \bullet & & & & & & \\ \hline \bullet & & & & & \bullet & & \\ \hline & & & \bullet & & & & \\ \hline & & & & \bullet & & & \\ \hline & & & \bullet & & & & \\ \hline & & & & \bullet & \bullet & & \\ \hline & & & & & & & \bullet \\ \hline & & & & & \bullet & \bullet & \\ \hline \end{array} \qquad B = \begin{array}{|c|c|c|c|c|c|c|c|} \hline & & \bullet & & & & & \\ \hline \bullet & & & & & & & \\ \hline & & \bullet & \bullet & & & & \\ \hline & \bullet & & & \bullet & & & \\ \hline & & & & & & & \\ \hline & & & & \bullet & & & \\ \hline & & & & & & \bullet & \\ \hline & & & & & \bullet & & \\ \hline \end{array}$$

Given are the two sparse matrices A and B. Consider the problem of sparse matrix-matrix multiplication. A dot corresponds to a non-zero entry. The computation is decomposed into 8 tasks. Let task i the owner of row $A[i,*]$ and of row $B[i,*]$. Task i has to compute row i of the result $C = A \cdot B$.

a) Draw the task interaction graph using directed edges. Draw an edge from task T_i to task T_j, if T_i requires data from T_j.

b) Suppose that task i owns column i of matrix B instead of row i for the computation. Draw the task-interaction graph for this case.

Exercise 5

LU factorization

$$\begin{pmatrix} A_{1,1} & A_{1,2} & A_{1,3} \\ A_{2,1} & A_{2,2} & A_{2,3} \\ A_{3,1} & A_{3,2} & A_{3,3} \end{pmatrix} \rightarrow \begin{pmatrix} L_{1,1} & 0 & 0 \\ L_{2,1} & L_{2,2} & 0 \\ L_{3,1} & L_{3,2} & L_{3,3} \end{pmatrix} \cdot \begin{pmatrix} U_{1,1} & U_{1,2} & U_{1,3} \\ 0 & U_{2,2} & U_{2,3} \\ 0 & 0 & U_{3,3} \end{pmatrix}$$

1: $A_{1,1} \rightarrow L_{1,1}U_{1,1}$	6: $A_{2,2} = A_{2,2} - L_{2,1}U_{1,2}$	11: $L_{3,2} = A_{3,2}U_{2,2}^{-1}$
2: $L_{2,1} = A_{2,1}U_{1,1}^{-1}$	7: $A_{3,2} = A_{3,2} - L_{3,1}U_{1,2}$	12: $U_{2,3} = L_{2,2}^{-1}A_{2,3}$
3: $L_{3,1} = A_{3,1}U_{1,1}^{-1}$	8: $A_{2,3} = A_{2,3} - L_{2,1}U_{1,3}$	13: $A_{3,3} = A_{3,3} - L_{3,2}U_{2,3}$
4: $U_{1,2} = L_{1,1}^{-1}A_{1,2}$	9: $A_{3,3} = A_{3,3} - L_{3,1}U_{1,3}$	14: $A_{3,3} \rightarrow L_{3,3}U_{3,3}$
5: $U_{1,3} = L_{1,1}^{-1}A_{1,3}$	10: $A_{2,2} \rightarrow L_{2,2}U_{2,2}$	

Given is the decomposition of the LU factorization into 14 tasks. (We assume that each of the 14 tasks requires the same unit amount of work).

a) Draw the task dependency graph.

b) Determine all critical paths.

c) Determine the average and the maximal degree of concurrency.

d) Describe/draw an efficient mapping of the task-dependency graph of the decomposition onto three processes.

e) Describe/draw an efficient mapping of the task-dependency graph of the decomposition onto four processes.

f) Which of the both mappings solves the problem faster?

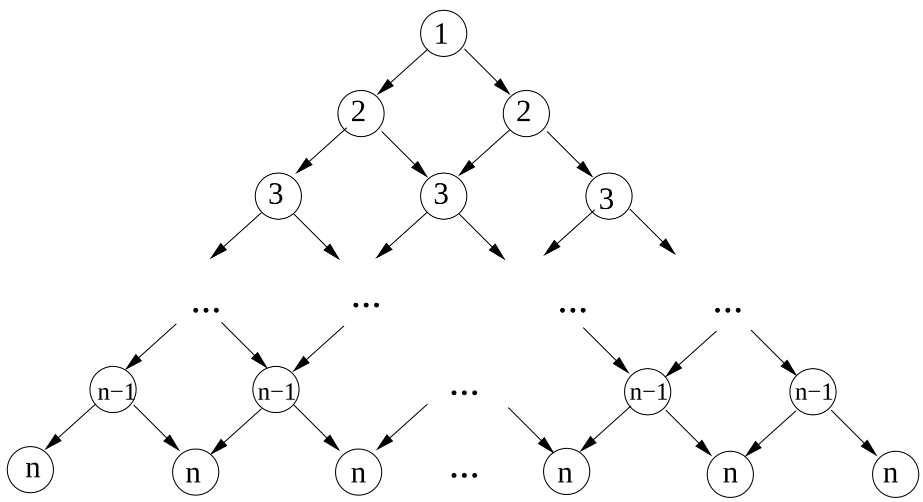
g) What is the maximal speedup that can be achieved and how many processes are necessary for that speedup? (to be discussed next quarter)

h) What is the maximal efficiency, that can be achieved, if $p > 1$ processes are used? Describe/draw the mapping that you used. (to be discussed next quarter)

Exercise 6

Task Dependency Graph

Given is the following task dependency graph:



a) Determine the maximal degree of concurrency.

b) What is the length of the critical path?

c) Determine the average degree of concurrency.

Exercise 7

Circular q-shift

Show that in a p-node hypercube, all the p data paths in a circular q-shift are congestion-free if E-cube routing (Section 4.5 of the course book) is used.

Hint: (1) If $q > p/2$, then a q-shift is isomorphic to a $(p - q)$-shift on a p-node hypercube. (2) Prove by induction on hypercube dimension. If all paths are congestion-free for a q-shift ($1 \leq q < p$) on a p-node hypercube, then all these paths are congestion-free on a $2p$-node hypercube also.

Exercise 8

Circular q-shift

Show that the length of the longest path of any message in a circular qshift on a p-node hypercube is $\log p - \gamma(q)$, where $\gamma(q)$ is the highest integer j such that q is divisible by 2^j.

Hint: (1) If $q = p/2$, then $\gamma(q) = \log p - 1$ on a p-node hypercube. (2) Prove by induction on hypercube dimension. For a given q, $\gamma(q)$ increases by one each time the number of nodes is doubled.