CS 575 Spr 2018 WA2: due Sun. Feb 25

Parallel Platforms & Communication Algorithms

Problem 1, Embeddings: [50 pts]

QI.1: Show the best possible embedding that you can think of, of a four dimensional hypercube (i.e., on with 16 nodes) into a 4×4 mesh with wraparound connections. What is the dilation and the congestion of this embedding. [5 pts]

QI.2: In the textbook, the first paragraph of Section 2.7.1 has a number of errors (e.g., is it clear which of G and G' is the process-graph and which is the processor graph?) Some errors are even more serious. Identify as many of the errors as you can , and provide a rewrite of the paragraph. [15 pts]

QI.3: In a k-d mesh, explain how each node can be uniquely labelled by a d-digit number in radix k. With this labeling, what are the neighbors of any given node. [5 pts]

QI.4: Sec 2.7.2 explains the cost-performance tradeoff in between the hypercube and the 2-D mesh under two different models of cost. In each case, the mesh turns out to be more powerful than the hypercube. However, in the lecture, Sanjay claimed that the hypercube was more powerful under one model, and less powerful in the other one. Explain Sanjay's mistake in the lecture. [10 pts]

QI.5: Derive the cost-performance tradeoffs between a p-node 3D mesh and a p-node hypercube of the same cost, where the cost is proportional to the number of wires . [15 pts]

Problem V, Communication algorithms: [50 pts]

QII.1: Write the pseudo-code of a parallel algorithm (unroll the recursion in to a loop program as done in the text) to perform the gather operation (think of it as a reduction with the concatenation operator). Make sure that the first communication is among processors/processes that differ in their least significant bits. [10 pts]

QII.2: Analyze the execution time of the algorithm on a topology-oblivious machine with N processors, using the $t_s + t_w m$ cost for message transmission. [10 pts]

QII.3: Now we want to implement the algorithm on a more realistic machine, the 2-D square mesh (so, N is an even power ot 2) with cut-through routing. In order to do this, we need to map the nodes (processes) in the above algorithm to the $\langle x, y \rangle$ coordinates of the mesh. We do this in the most obvious manner: node i, for $0 \leq i < N$ is mapped to $\langle \lfloor \frac{i}{\sqrt{N}} \rfloor, i \bmod \sqrt{N} \rangle$. This mapping simply consists of taking the $\log N$ bits that represent i, using the first half of them to determine the row index, x, and the rest to determine the column index, y. [10 pts]

QII.4.d: With cut-through routing, message transmission is still $t_s + t_w m$ if there is no congestion, regardless of the length of the path from source to destination. However, if at any step in the algorithm, there are k messages that need to traverse the same link, the cost becomes $t_s + kt_w m$. Note that k does not affect the startup (latency) just the message transmission time (bandwidth term). Analyze the execution time of the algorithm on this machine. [10 pts]

QII.5: Analyze your algorithm if the machine uses store-and-forward routing, so that the communication time is now $t_s + t_w ml$, where l is the number of links or hops that a message takes. [10 pts]