UNIVERSITY OF LONDON
IMPERIAL COLLEGE OF SCIENCE, TECHNOLOGY AND MEDICINE

EXAMINATIONS 2002

MEng Honours Degree in Information Systems Engineering Part IV
MSci Honours Degree in Mathematics and Computer Science Part IV
MEng Honours Degrees in Computing Part IV
MSc in Advanced Computing
for Internal Students of the Imperial College of Science, Technology and Medicine

This paper is also taken for the relevant examinations for the Associateship of the City and Guilds of London Institute
This paper is also taken for the relevant examinations for the Associateship of the Royal College of Science

PAPER C429=I4.10

PARALLEL ALGORITHMS

Thursday 25 April 2002, 10:00
Duration: 120 minutes

Answer THREE questions

Paper contains 4 questions
Calculators not required

1. a. i) Define the 2-*dimensional rectangular mesh without wrap-around* interconnection network topology.

ii) Describe how such a $2p$-processor mesh can be constructed from two p-processor meshes, where $p = ab$ for positive integers a, b.

b. i) Describe a checkerboard-based parallel algorithm to multiply two $n \times n$ matrices, using block-striped partitioning of rows. Assume that initially both matrices are already striped across p processors, where p divides n.

ii) Calculate the asymptotic parallel run-time of your algorithm on a p-processor square mesh of the type defined in part a, with a bi-directional, concurrent communication facility.

c. Devise a recursive algorithm to invert a product of m $n \times n$ matrices, $M_0 M_1 ... M_{m-1}$, on the specified p-processor mesh, where $p = 2^k$ for positive integer k and p divides m. Calculate the number of steps in the algorithm, detail the operation of each step and show that the asymptotic parallel execution time at large k and n is $\Theta(n^3)$ if $k = O(n)$.
Hint: For square matrices A, B, $(AB)^{-1} = B^{-1} A^{-1}$

The three parts carry, respectively, 20%, 40% *and* 40% *of the marks.*

2. a. i) Describe the *Jacobi method* for solving a set of linear equations and explain where a parallel algorithm can significantly speed up the computation.

ii) How is the rate of convergence of the Jacobi iteration increased in the *Gauss-Seidel method* by using more up-to-date data, where available, and how does this affect the speed-up gained in the Jacobi parallel algorithm?

iii)Given a cost-optimal algorithm with negligible communication latency for the parallel computations, estimate by how much the rate of convergence must increase in the Gauss-Seidel method to make it faster than the Jacobi method on p processors.

b. Suppose you have a set of equations $Ax = b$, where $A = (a_{ij})$ is a $2n \times 2n$ matrix with all its diagonal elements non-zero and elements $a_{ij} = 0$ for $0 < j < i < n$ and for $n < j < i < 2n$ in the lower triangle. Describe a SOR-type of algorithm that is equivalent to Gauss-Seidel by implementing each iteration in two phases.

c. List the main steps in mapping a partial differential equation onto a set of linear equations, using the *finite differencing* method. Use the equation for the function $u(x,y)$,

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f(x,y)$$

defined in a rectangle in the positive quadrant with two sides on the axes, for given function $f(x,y)$, to illustrate your answer.

The three parts carry, respectively, 40%, 30% *and* 30% *of the marks.*

3. In an all-to-all vector broadcast, each of p processors broadcasts a vector of n values to every other processor, producing a vector of np results on every processor. So on a four processor parallel machine in which the processors initially hold the vectors {0,1}, {2,3}, {4,5} and {6,7} respectively, an all-to-all broadcast results in the vector {0,1,2,3,4,5,6,7} on all nodes.

a) Give pseudocode for an efficient all-to-all vector broadcast procedure `all_to_all_broadcast(node, input_vector, d)` which can be run on each node of a hypercube with bi-directional links and store-and-forward routing. Here `d` is the dimension of the hypercube, `node` is the processor's label/number and `input_vector` is the n-element input vector on the processor. Your procedure should store the result of the broadcast in a vector called `result_vector`.

b) Derive a general expression for the parallel run time of your algorithm when broadcasting n element vectors on a hypercube with p processors. You may assume each vector element is one word big, message start-up time is t_s, hop time is negligible, per-word transfer time is t_w and the time to concatenate two vectors of length n is nt_a. What is the parallel cost?

c) Derive the run-time for a sequential (1 processor) implementation which combines p vectors of length n into p vectors of length np on a single processor. You may assume that the type taken to copy a vector of length n is nt_a. Are there conditions under which the parallel algorithm will be cost-optimal?

d) Predict the speedup and efficiency of the algorithm when broadcasting vectors of length 64 on a 16 processor hypercube that has t_s = 100ns, t_w = 50ns and t_a = 120ns.

e) To which MPI primitive does this all-to-all broadcast correspond?

The five parts carry, respectively, 20%, 25%, 25%, 20% and 10% of the marks.

4.a) State whether each of the following statements is True or False. In each case, give a *brief* (one sentence) justification for your answer.

i) An EREW PRAM with n^3 processors can compute the product of two $n \times n$ matrices in $O(1)$ time.

ii) Consider a parallel machine with an interconnection network that supports cut-through routing and has a negligible hop time. The performance of MPI applications running on such a machine will appear to be independent of how MPI processes are mapped onto processing nodes.

iii) The distance in hops between the processors numbered 13 and 27 in a 5 dimensional hypercube is 2.

iv) A cubic 3D mesh (without wraparound) with p processors has a diameter of $3(\sqrt{p} - 1)$ and contains $3(p - p^{2/3})$ wires.

b) i) In the context of a parallel depth-first tree-searching algorithm, describe the problems that arise if subtrees are allocated to processors in a static manner.

ii) In a dynamic load balancing scheme, a processor that is out of work requests more work (i.e. a subtree to search) from a busy processor. Describe the operation of the Asynchronous Round Robin (ARR) and the Global Round Robin (GRR) dynamic load balancing schemes. For each scheme, derive the worst-case value of $V(p)$, the minimum number of requests for work which is guaranteed to include at least one request to each of p processors.

iii) If w units of work are always split such that each partition is guaranteed to have at least αw units of work after the splitting ($0 < \alpha < 1$), derive an upper bound on the total number of requests for work in the Asynchronous Round Robin scheme. You may assume a given initial work load of W units and a minimum splitting threshold of ε work units.

The two parts carry, respectively, 40% and 60% of the marks.

End of paper