Mastermath midterm examination Parallel Algorithms. Retake. Solution.

Teacher: Rob H. Bisseling, Utrecht University

December 19, 2018

Each of the four questions is worth 10 points.

1. (a) The *cyclic distribution* of a vector $\mathbf{x}$ of length n over p processors is defined by assigning

$$x_i \longmapsto P(i \bmod p), \quad \text{for } 0 \leq i < n.$$

(b) The *block distribution* of $\mathbf{x}$ over p processors is defined by assigning

$$x_i \longmapsto P(i \text{ div } b), \quad \text{for } 0 \leq i < n,$$

where $b = \lceil n/p \rceil$ is the block size and the division operator is defined by i div $b = \lfloor i/b \rfloor$.

(c) An example is the smoothing operation

$$y_i = \frac{x_{i-1} + x_i + x_{i+1}}{3}.$$

(d) This operation has cost $3n/p + 2g + 2l$ for the block distribution, since first two boundary values have to be received for each block and then 3 flops are needed for every vector component. For the cyclic distribution, the cost is $3n/p + (2n/p)g + 2l$, because the values x_{i-1} and x_{i+1} must be obtained by communication, for all i.

2. (a) The evaluation of a polynomial f of degree $n-1$ in a variable x can be done in parallel by using Algorithm 1. We use Horner's rule for quick evaluation using the local coefficients, and then multiply by the power x^{sb} to obtain the correct block contribution.

Algorithm 1 Polynomial evaluation for processor $P(s)$.

get x from $P(0)$; ▷ Superstep (0)

$b := n/p$; ▷ Superstep (1)
$y_s = a_{(s+1)b-1}$;
for $i := (s+1)b-2$ **to** sb **step** -1 **do**
  $y_s := y_s \cdot x + a_i$;
$y_s = y_s \cdot x^{sb}$;

put y_s in $P(0)$; ▷ Superstep (2)

if $s = 0$ **then** ▷ Superstep (3)
  $y = 0$;
  for $t := 0$ **to** $p-1$ **do**
    $y := y + y_s$
  $f(x) := y$;

(b) The communication supersteps (0) and (2) are both $(p-1)$-relations, with total cost $2(p-1)g+2l$. Superstep (1) first performs a sequential polynomial evaluation using the local coefficients, performing 2 flops per loop iteration, i.e., in total $2n/p - 2$ flops. Then it computes the power x^{sb}. This can be done in $2\log_2 sb$ flops by repeated squaring: compute $x, x^2, x^4, x^8, \ldots$ and then multiply the needed powers (based on the binary expansion of sb) to compute x^{sb}. The largest cost is for processor $P(p-1)$, which performs at most $2\log_2(p-1)b = 2\log_2(n - n/p)$ flops. One extra multiplication finishes superstep (1). Superstep (3) costs p flops. The total BSP cost is thus

$$\frac{2n}{p} + p + 2\log_2\left(n - \frac{n}{p}\right) - 1 + 2(p-1)g + 4l.$$

3. (a) In the $q \times q$ square cyclic distribution over $p = q^2$ processors, row i resides in processor row $P(i \bmod q, *)$ and row $(i+1) \bmod n$ in $P((i+1) \bmod q, *)$. Note that this is true even for $i = n-1$, because $n \bmod q = 0$. This implies that $P(s,t)$ can simply put its part of the matrix into $P((s+1) \bmod q, t)$. Algorithm 2 gives the corresponding operations.

(b) The algorithm has a communication superstep (0), where all local data are sent to another processor, at cost $(n/q)(n/q)g + l =$

Algorithm 2 Row permutation for processor $P(s)$.

for all $i : 0 \leq i < n \wedge i \bmod q = s$ **do** ▷ Superstep (0)
  for all $j : 0 \leq j < n \wedge j \bmod q = t$ **do**
    put a_{ij} as $\hat{a}_{ij}$ in $P((s+1) \bmod q, t)$;

for all $i : 0 \leq i < n \wedge i \bmod q = s$ **do** ▷ Superstep (1)
  for all $j : 0 \leq j < n \wedge j \bmod q = t$ **do**
    $a_{ij} := \hat{a}_{(i-1) \bmod q, j}$;

$(n^2/p)g + l$. The algorithms also has a computation superstep (1), with n^2/p assignments (which are free in our model) and a synchronization, which costs l. The total cost is thus $(n^2/p)g + 2l$.

4. (a) The matrix update can be written as

$$a_{ij} := a_{ij} + x_i x_j, \quad \text{for } 0 \leq i, j < n. \tag{1}$$

This means that vector component x_i is needed by all processors in processor row $P(\phi_0(i), *)$ and also in those of processor column $P(*, \phi_1(i))$, which are $2q-1$ processors. Note that $P(\phi_0(i), \phi_1(i))$ is in both the processor row and the processor column. We should assign x_i to one of those $2q-1$ processors, so that it needs to be sent to only $2q-2$ other processors. Here, it does not matter which of those processors we choose.

Furthermore, we want to spread the vector evenly over all the processors, so that the vector broadcast is balanced, and hence we do not need the first (spreading) phase of a two-phase broadcast. This goal is achieved by defining

$$\phi_{\mathrm{x}}(i) = (i \text{ div } b, i \bmod q),$$

where $b = n/q$ is the block size. Note that then $\phi_{\mathrm{x},0} = \phi_0$.

(b) In superstep (0), each local value x_i is sent from its owner $P(\phi_{\mathrm{x}}(i))$ to all processors $P(s,t)$ with $s = \phi_0(i)$ or $t = \phi_1(i)$. In superstep (1), the update of equation (1) is carried out for all local matrix elements a_{ij}, i.e., those with $\phi_0(i) = s$ and $\phi_1(j) = t$.

(c) The maximum number of data words that a processor sends is $(n/p)(2q-2) = 2n/q - 2n/p$; the sending is perfectly balanced. The maximum number that a processor receives is $2n/q - n/p$; the receiving is a bit less balanced, since diagonal processors receive less than they send and others a bit (n/p) more. The BSP cost of superstep (0) is therefore $(2n/q - n/p)g + l$. The cost of superstep (1) is $2n^2/p + l$. The total cost is thus

$$\frac{2n^2}{p} + \left(\frac{2n}{\sqrt{p}} - \frac{n}{p}\right) g + 2l.$$