EECS 358 HW3

Chenhui Zhou
Wenjie Zhang

June 2016

1 Problem 1

(a)
- hypercube: Initial partition the matrix and vector x then distribute the full vector among all the processors by all-to-all broadcast which takes $t_s log(p) + t_w n/p(p-1)$ times and for each professor it need n^2/p times to compute so the total time is:$n^2/p + t_s log(p) + t_w(n - n/p)$
- torus: The procedure is similar, first partition the matrix and vector x, then do the all to all broadcast which takes $2t_s(\sqrt{p}-1) + t_w n/p(p-1)$ and each professor need n^2/p to compute so the total time is $n^2/p + 2t_s(\sqrt{p}-1) + t_w n(n - n/p)$

(b) For a ring communication structure we need move data to its "right" neighbor. Once the node get data that it needs, it just store the data. If not, the data will continue to move right in the next round. So totally it takes $p-1$ steps to complish this job. And every time we move n^2/p data. And the total time is $n^2/2p + (p-1) \times (t_s + t_w \times n^2/p)$

2 Problem 2

(a)
- The algorithm description.
  In this problem we get number of points that randomly distributed. And the goal is that we need to find a number of quadrants (this number will be given as an argument to the program) in a 2-dimensional coordinate system. And at the final state each quadrant will contain equal number of points. To attain this result. We use the algorithm that each time we split the largest coordinate into two parts. So at each step we first calculate the "X-Width" and "Y-Width",if the X-Width" is larger than "Y-Width" we bisection the X coordinate, otherwise, we bisection the Y-coordinate. Every time we do a bisection we need to use quick select to get the mid number of the coordinate we split, and at the meantime we swap all points which corresponding X-value or Y-value is smaller than mid to the "left" or "bottom", and all points which corresponding X-value or Y-value is larger than mid

to the "right" or "up". Once we accomplish a split we get twice quadrants. We do the job above recursively until we reach the termination. The termination of the algorithm is when we reach the $numquadrants$. And by the time we reach the termination we record the split points and out the left-high, left-low, right-high, right-low points for each part.

All of the things mentioned above is done by processor 0. And for the part of compute the total cost. We use parallel method. Here we broadcast the whole X, Y axis to each processor and we use static interleaved way to give each processor a specific parts to compute, and the result is store in each processor as localcost. And at the end we use $MPIReduce$ to get the the sum of localcost as globalcost.

(b) The Rumtime and Costs:
64 quadrants:
1 processor:75.285402 300481983965360704.000000;
2 processor:37.855869 300481983965757632.000000;
4 processor:19.229993 300481983965806592.000000;
8 processor:13.002476 300481983965854336.000000;
16 processor:7.595588 300481983965814848.000000;
32 processor:5.845570 300481983965723264.000000;
64 processor:3.097618 300481983965723648.000000;
128 proessor:3.985593 300481983965723648.000000;
128 quadrants:
1 processor:40.006643 115665026377093680.000000;
2 processor:22.943064 115665026376846304.000000;
4 processor:12.984430 115665026376940464.000000;
8 processor:7.388859 115665026376887648.000000;
16 processor:4.940741 115665026376850464.000000;
32 processor:4.288545 115665026376855360.000000;
64 processor:2.539549 115665026376861888.000000;
128 proessor:3.352604 115665026376864688.000000;
256 quadrants:
1 processor: 19.217739 37545464583307400.000000;
2 processor: 10.203188 37545464583275720.000000;
4 processor: 6.585615 37545464583249184.000000;
8 processor: 4.275642 37545464583249424.000000;
16 processor:3.221633 37545464583251776.000000;
32 processor:3.672732 37545464583252752.000000;
64 processor:2.147990 37545464583254384.000000;
128 proessor:3.094778 37545464583254032.000000;