CSE 392 Spring 2014 Homework 3

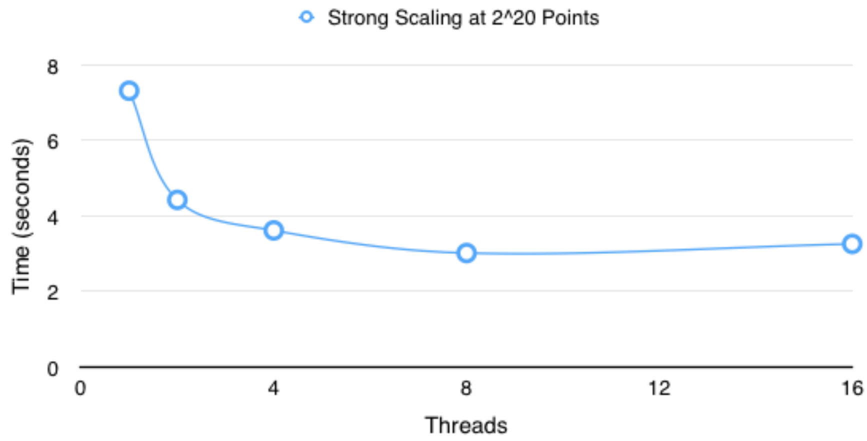
Abhishek Bhaduri and Aaron Stacy

1. See appended hand-written work
2. 2D N-body Simulation
    1. The main function of the algorithm, `nbody`, follows the pseudo-code given in lecture 15 slide 31 fairly closely:

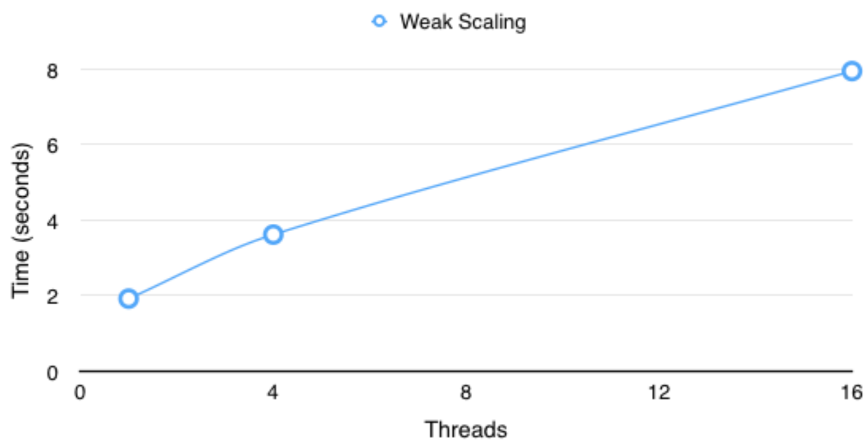
```
function nbody(pointsAndDensities, n, outputPotential)
  parfor i=1-n                                % W=N, D=1
    mids[i] = convertToMid(points[i])
  end

  [smids,idx] = parallelSort(mids)            % W=NlogN, D=logN * loglogN

  trees = []
  lengths = []

  parfor i=1:p                                % W=NlogN, D=logN
    myStart = n / p * i
    myEnd = myStart + n / p
    trees{i} = qtree()
    for j=myStart:myEnd
      trees{i}.insert(points[j])
    end
    lengths[i] = length(trees{i})
  end

  tree = []

  parfor i=1:p                                % W=N, D=1
    tree[sum(lengths(0:i-1))] = trees{i}.preOrder
  end

  tree = parallelSort(tree)                   % W=NlogN, D=logN * loglogN

  tree = removeDuplicates(tree)               % W=N, D=1

  [i, o] = eulerTour(tree)                    % W=N, D=logN

  treePrefixScan(tree, i, o, density)         % W=N, D=logN

  parfor i-1:N                                % W=NlogN, D=logN
    outputPotential[i] = evaluate(points(i), tree.root)
  end
end
```

    For parallel sort we used Intel TBB. For converting to morton ID's, inserting into the tree, and evaluating points we used the algorithms from the given Matlab implementation (`body.cpp`, `qtree.cpp`, and `euler.cpp`). For doing the Euler tour and prefix scan on the tree we used the algorithm from slide 28 of lecture 15.

    2. Scalability results:
        1. Strong scaling at ~ 1 million points



2. Weak scaling at a ratio of $2^{18} : 1$



3. We could efficiently estimate error by experimentally measuring the average error that center-of-mass approximation introduces, and then using that as a heuristic during the last step of the algorithm. This adds an $O(N)$ time step to the end of the algorithm.