CS3101 Due: Wednesday 8-th of April 2015

Problem Set 3

CS3101 *Submission instructions on last page*

PROBBLEM 1. [25 points]

In this problem, we develop a divide-and-conquer algorithm for the following geometric task, called the *CLOSEST PAIR PROBLEM* (CSP):

Input: A set of n points in the plane

$$\{p_1 = (x_1, y_1), p_2 = (x_2, y_2), \ldots, p_n = (x_n, y_n)\},$$

whose coordinates are floating point numbers (positive, null or negative).

Output: The closest pair of points, that is, the pair $\{p_i, p_j\}$ with $p_i \neq p_j$ for which the distance between p_i and p_j, that is,

$$\sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}$$

is minimized.

For simplicity, we assume that n is a power of 2 and that all the x-coordinates x_i are pairwise distinct, as well are the y-coordinates y_i. Here's a high-level overview of the proposed algorithm:

1. Find a value x for which exactly half the points satisfy $x_i < x$ and half satisfy $x_i > x$, thus splitting the points into groups L and R.

2. Recursively find the closest pair in L and R. Let us call these pairs $\{p_L, q_L\}$ with $p_L, q_L \in L$ and $\{p_R, q_R\}$ with $p_R, q_R \in R$; we denote by d_L (resp. d_R) the distance between p_L and q_L (resp. p_R and q_R). Let d be the smallest of these two distances.

3. It remains to be seen whether there is a point in L and a point in R that are less than distance d apart from each other. To this end, discard all points with $x_i < x - d$ or $x_i > x + d$. Then, sort the remaining points by y-coordinate.

4. Now, go through this sorted list, and for each point, compute its distance to the *seven subsequent points* in the list. Let p_M, q_M be the closest pair found in that way.

5. The answer is $\{p_L, q_L\}$, $\{p_R, q_R\}$ and $\{p_M, q_M\}$, whichever is closest.

Question 1. [5 points] In order to prove the correctness of this algorithm, start by showing the following property: any square of size $d \times d$ (where d is as defined in the above overview of the proposed algorithm) in the plane contains at most *four points* of L.

Question 2. [5 points] Now show that the algorithm is correct. The only case which needs careful consideration is when the closest pair is split between L and R.

Question 3. [5 points] Write down the pseudo-code for the algorithm, and show that its work is given by the recurrence:

$$W(n) = 2W(n/2) + O(n\log(n))$$

Deduce that $W(n) \in O(n\log^2(n))$.

Question 4. [10 points] Propose a parallel version of this algorithm and show that its parallelism is limited to $\log(n)$.

PROBBLEM 2. [25 points]

In this problem, we consider the multiplication of two $n \times n$ matrices. To be simple, we assume that n $= 2^p$ for some integer p.

Question 1. [10 points] Write a PRAM algorithm (with the syntax introduced in class) using $O(n^3)$ processors for which $T(n, n^3) \in O(\log(n))$. For handling memory access conflict, consider the CREW-PRAM sub-model. What is the efficiency of this algorithm?

Question 2. [5 points] Following up on Question 1, can you reduce the number of processors to $O(n^3/\log n)$ still within the CREW-PRAM sub-model. What is the time complexity? What is the efficiency?

Question 3. [10 points] Design an algorithm to complete the operation (multiplication of square matrices of order n) with a parallel running time of $O(\log(n))$ in the EREW-PRAM sub-model. How many processors do you need? What is the efficiency?

PROBBLEM 3. [50 points]

Let A be a $n \times n$ invertible lower triangular matrix. A simple divide-and-conquer strategy to invert A is described below.

Let A be partitioned into $(n/2) \times (n/2)$ blocks as follows:

$$A = \begin{bmatrix} A_1 & 0 \\ A_2 & A_3 \end{bmatrix}, \tag{1}$$

where n is assumed to be a power of 2. Clearly A_1 and A_3 are invertible lower triangular matrices. From there, it is easy to show that A^{-1} is given by:

$$A^{-1} = \begin{bmatrix} A_1^{-1} & 0 \\ -A_3^{-1}A_2A_1^{-1} & A_3^{-1} \end{bmatrix}. \tag{2}$$

Therefore, we can obtain the inverse of A by recursively computing the inverses of A_1 and A_3, and by computing two $(n/2)\times(n/2)$ matrix products so as to generate the term $-A_3^{-1}A_2A_1^{-1}$. This divide-and-conquer method leads to the following questions.

Question 1. [25 points] Write a `CilkPlus`-like program to parallelize the above algorithm for matrices with `float` coefficients. You can make use of the code for matrix multiplication available in the collection of examples of the `MetaFork` project from `www.metafork.org`

Question 2. [10 points] Analyze the work and span of the matrix multiplication code that you are using.

Question 3. [10 points] Analyze the work and span of your parallel program for inverting lower triangular matrices.

Question 4. [5 points] Collect performance results with `Cilkview` for input random dense lower triangular matrices of order 2^i, for $i = 8, 9, 10, 11, 12$.

Submission instructions.

Format: For Problems 1 and 2, as well as for Question 4 of Problem 3, please submit a PDF file called `Assignment3.pdf` containing your answers. Problems 3 involves programming with `CilkPlus` and you should submit two input files: one text file called `Pb3.cpp` containing your source code and one makefile called `Makefile` specifying how to compile `Pbb3.cpp`. Also please explain how to run your code, for instance using a `README` file or using comments in the `Makefile`. In addition, each user defined function must be **documented.**

Submission: The assignment should be returned to the instructor by email.

Collaboration. You are expected to do this assignment *on your own* without assistance from anyone else in the class. However, you can use literature and if you do so, briefly list your references in the assignment. Be careful! You might find on the web solutions to our problems that are not appropriate. For instance, because the cache memory model is different. So please, avoid those traps and work out the solutions by yourself. You should not hesitate to contact the instructor if you have any question regarding this assignment. I will be more than happy to help.

Marking. This assignment will be marked out of 100. A 10 % bonus will be given if your answers are clearly organized, precise and concise. Messy assignments (unclear statements, lack of correctness in the reasoning, many typographical or language mistakes) may give rise to a 10 % malus.