15-750 Graduate Algorithms (Spring ’11)
Manuel Blum, Liu Yang and Harsha Simhadri

Assignment 4 Due date: 4th April, 2011.

Policies:

• It is strongly recommended that you try to solve the problems yourself before consulting other sources. You may use any source to solve the problems, but please cite your sources.

• For questions about problems 1, 2 and 3, contact Harsha.

• Typesetting diagrams is not required for problems 1, 2. Please make sure your diagrams are clearly labelled and type the rest of your answer. Send a pdf copy to harshas@cs.cmu.edu.

1 Finish your trip (before hail breaks loose) [30 points]

It is a spring break morning, and you get out of your bed hoping for a pleasant bike ride to a favorite coffee place. It being Pittsburgh, you step out of your apartment to find that overnight precipitation and low temperatures have left icy patches on the road. Determined to ride your bike, you set about making your way avoiding the icy patches. Your goal is to figure out the length of the shortest path to your morning cup.

We model the problem as follows: you start at point $s$ on a plane and intend to get to point $t$ on the same plane (Pittsburgh suddenly became a flat world!). On this plane are some convex polygons of finite size (representing the icy patches). Further, no two polygons share a common point and $s, t$ are outside all the polygons. The input to the problem are the coordinates of $s, t$, and the list of polygons. For each polygon, the coordinates of it’s vertices are listed in some order. The union $V$ of the vertex sets of these polygons has cardinality $n$. Your goal is to write an algorithm, that runs in time $O(n^2 \log n)$, to find the length of the shortest path in the plane from $s$ to $t$ that does not overlap with the interior of any polygon (see figure for an example of a valid path). You may assume that:

• points in $V \cup \{s, t\}$ are distinct and have integral co-ordinates.
• asking for the square root of an integer will give you a floating point number that is at most $\epsilon$ away from the real answer.

• floating point arithmetic does not result in loss of accuracy.

Your algorithm’s answer has to be accurate to within $\pm n^2 \epsilon$.

You may first (not necessarily) want to solve the following problem or some variant: Given a set $A$ of $n(n-1)/2$ line segments in a plane representing the set of all line segments between some $n$ points with integral co-ordinates, and a set $B$ of $n$ line segments representing the boundaries of a set of convex polygons, identify all the segments in $A$ that do not cross any segment in $B$.

[Extra credit] Give an algorithm that runs in time $O(n^2)$ or better.

2 Directed triangles (draw enough of them to fill your board) [30 pts]

Define a Tournament Graph $G = (V, E)$ on $|V| = n$ vertices and $|E| = \binom{n}{2}$ edges to be a directed graph with no self-loops and exactly one (directed) edge joining every pair of distinct vertices. Such a graph can be represented by an $n \times n$ adjacency matrix $A$ with entries from the set $\{0, 1, -1\}$: $A_{ii} = 0$ for all $i$, $A_{ij} = 1$ if there is a directed edge from vertex $v_i$ to vertex $v_j$, and $A_{ij} = -1$ if there is a directed edge from vertex $v_j$ to vertex $v_i$. Define a
(Directed) Hamilton Path (HP) in a directed graph to be a directed path \( v_1 \rightarrow v_2 \rightarrow \cdots \rightarrow v_n \) that visits every vertex exactly once. For example, \( 1 \rightarrow 4 \rightarrow 6 \rightarrow 5 \rightarrow 3 \rightarrow 2 \) is a directed HP in this adjacency matrix:

\[
\begin{array}{ccccccc}
    & 1 & 2 & 3 & 4 & 5 & 6 \\
1 & 0 & -1 & 1 & 1 & 1 & 1 \\
2 & -1 & 0 & -1 & 1 & -1 & 1 \\
3 & -1 & 1 & 0 & -1 & -1 & 1 \\
4 & -1 & -1 & 1 & 0 & -1 & 1 \\
5 & -1 & 1 & 1 & 0 & -1 & -1 \\
6 & -1 & -1 & -1 & -1 & 1 & 0 \\
\end{array}
\]

(a) [10 points] Prove that every Tournament Graph (TG) has a Hamilton Path (HP). Also give an efficient (polynomial time) algorithm to find a Hamilton Path.

(b) [15 points] Give a lower bound on the number of elements in the adjacency matrix that any algorithm computing a HP in TG should probe. Do not count matrix entries that are not probed and any other data structure manipulations. Give an algorithm (for HP in TG) that makes the (asymptotically) same number of probes as your lower bound into the adjacency matrix.

(c) [5 points] Give a data structure that enables you to achieve the above running time when all steps (not just probes) are counted. Specify clearly the data structure and what counts as a step.

### 3 Reuse, Recycle [30 points]

For all the following problems, assume that fork and join operations are free. All other “usual” operations cost 1 unit each.

(a) [10 points] Describe a parallel algorithm to add two \( n \)-bit positive integers. Binary operations on bits costs 1 unit each. Your algorithm should cost \( O(n) \) work and \( O(\log n) \) depth. Hint: Use prefix sums.

(b) [2 points] Given two sorted arrays of integers, describe a parallel algorithm to merge them into one sorted array. Your algorithm should cost \( O(n) \) work and \( O(\log n) \) depth, where \( n \) is the sum of the lengths of the input arrays.
(c) [3 points] Describe a deterministic parallel algorithm to sort \( n \) integers that costs \( O(n \log n) \) work and depth \( O(\log^2 n) \).

(d) [15 points] Describe a deterministic algorithm to find the median of \( n \) integers that costs \( O(n) \) work and \( O(\log^3 n) \) depth. *Hint:* Revisit the selection algorithm of Blum et al.