Assignment 2

CSCI 6104: Algorithms for Massive Data Sets
Due Nov 30, 2011

Banner ID: ____________________  Name: ________________________________

Question 1

Question 2

Total
**Question 1 (20 marks)** In class, we discussed Kumar/Schwabe's single-source shortest path algorithm. The correctness of this algorithm as described in class relies on no two adjacent vertices having the same distance from the source. Modify the algorithm so that this assumption is not needed. The I/O complexity should not be affected by this change (up to a constant factor). Argue that the I/O complexity is not affected. Prove that the algorithm you obtain is correct.

**Question 2 (20 marks)** A $d$-vertex colouring of a graph assigns one of $d$ colours to each of the vertices of the graph. Such a colouring is valid if no two adjacent vertices receive the same colour. Every graph whose vertices have degree no greater than $d$ has a valid $(d + 1)$-vertex colouring. Develop an algorithm that computes such a colouring using $O(sort(n + m))$ I/Os, where $n$ and $m$ are the numbers of vertices and edges in the graph. Argue briefly that your algorithm is correct and achieves the desired I/O complexity.

**Question 3 (20 marks + 10 bonus marks + 10 bonus marks)** A shortest path data structure for a graph $G$ is a data structure which, given two query vertices $x$ and $y$ reports the distance (and the corresponding shortest path) from $x$ to $y$. In general, such data structures are difficult to construct, at least if we want the data structure to be small and answer queries quickly. Now assume the given graph is a tree. Develop a data structure that uses linear space and can report the distance between any two vertices using $O(log_B N)$ I/Os. (Hint: The Euler tour technique is a good place to start.)

The first 10 bonus marks are for reducing the query complexity to $O(1)$ I/Os without increasing the size of the data structure.

The second 10 bonus marks are for adding the ability to report the path between $x$ and $y$ in the tree using $O(1 + K/B)$ I/Os, where $K$ is the number of edges in the path.