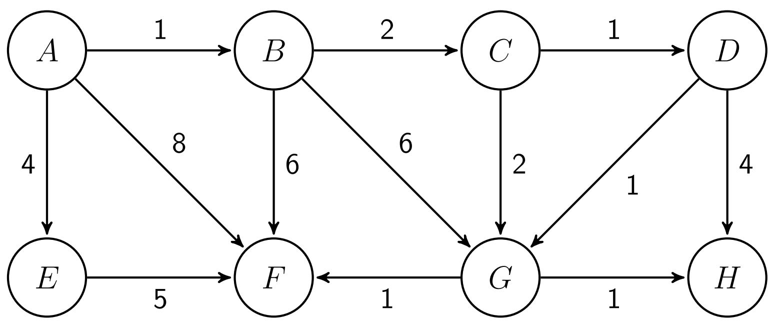
CS325 Winter 2013: HW 3

Daniel Reichert
Trevor Bramwell
Lance Stringham

January 31, 2013

4.1

Problem: Suppose Dijkstraś algorithm is run on the following graph, starting at node A.



(a) Draw a table showing the intermediate distance values of all the nodes at each iteration of the algorithm.

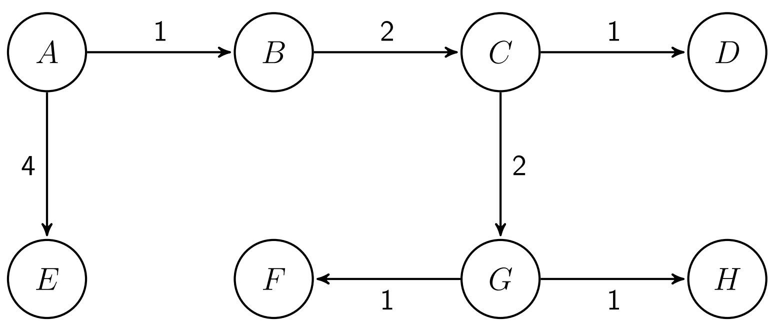
(b) Show the final shortest-path tree.

Solution:

(a)

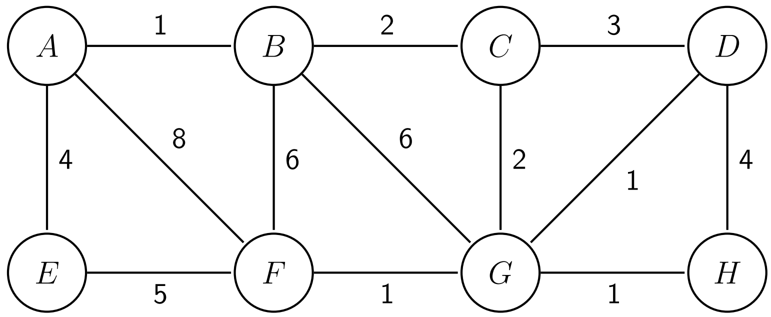
	Iteration						
Node	0	1	2	3	4	5	6
A	0	0	0	0	0	0	0
B	1	1	1	1	1	1	1
C	∞	3	3	3	3	3	3
D	∞	∞	4	4	4	4	4
E	4	4	4	4	4	4	4
F	8	7	7	7	7	6	6
G	∞	7	5	5	5	5	5
H	∞	∞	∞	8	8	6	6

(b)



5.2

Problem: Suppose we want to find the minimum spanning tree of the following graph.



(a) Run Prim's algorithm; whenever there is a choice of nodes, always use alphabetic ordering (e.g., start from node A). Draw a table showing the intermediate values of the cost array.

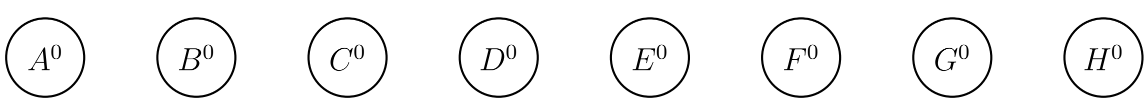
(b) Run Kruskal's algorithm on the same graph. Show how the disjoint-sets data structure looks at every intermediate stage (including the structure of the directed trees), assuming path compression is used.

Solution:

(a)

Set S	A	B	C	D	E	F	G	H
$\{\}$	$0/\text{nil}$	∞/nil	∞/nil	∞/nil	∞/nil	∞/nil	∞/nil	∞/nil
A		$1/A$	∞/nil	∞/nil	$4/A$	$6/B$	$6/B$	∞/nil
A, B			$2/B$	∞/nil	$4/A$	$6/B$	$2/C$	∞/nil
A, B, C				$3/C$	$4/A$	$6/B$	$2/C$	∞/nil
A, B, C, D					$4/A$	$6/B$	$1/D$	$4/D$
A, B, C, D, E						$5/E$	$1/D$	$4/D$
A, B, C, D, E, F							$1/D$	$4/D$
A, B, C, D, E, F, G						$1/G$		$1/G$
A, B, C, D, E, F, G, H								$1/G$

(b)



5.5

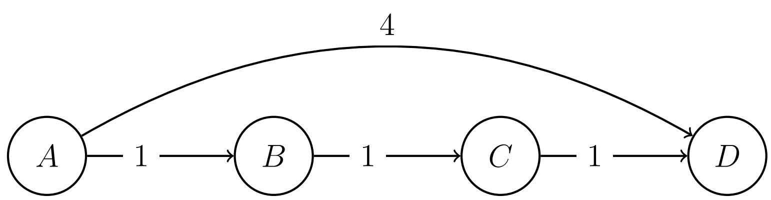
Problem: Consider an undirected graph $G = (V, E)$ with nonnegative edge weights $w_e \geq 0$. Suppose that you have computed a minimum spanning tree of G, and that you have also computed shortest paths to all nodes from a particular node $s \in V$. Now suppose each edge weight is increased by 1: the new weights are $w_e = w_e + 1$.

(a) Does the minimum spanning tree change? Give an example where it changes or prove it cannot change.

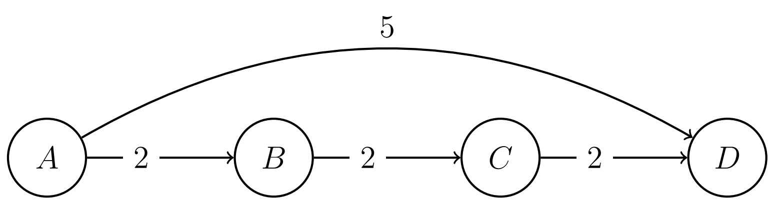
(b) Do the shortest paths change? Give an example where they change or prove they cannot change.

Solution:

Proof. Part A: Proof by counter example. Consider the graph:



In this graph the shortest path from A to D is from A to B to C to D with a weight of 3. Now consider the same graph, but with all of the edges having a weight increased by 1.



The previously shortest path now has a weight of 6, where the direct route from A to D only has a weight of 5. Thus the shortest path has changed and the counter example is proved. □

Proof. Part B: □

5.7

Problem: Show how to find the *maximum* spanning tree of a graph, that is, the spanning tree of largest total weight.

Solution:

Proof. □

5

Problem: Consider the Change Problem in Austria. The input to this problem is an integer L. The output should be the minimum cardinality collection of coins required to make L shillings of change (that is, you want to use as few coins as possible). In Austria the coins are worth 1, 5, 10, 20, 25, 50 Shillings. Assume that you have an unlimited number of coins of each type. Formally prove or disprove that the greedy algorithm (that takes as many coins as possible from the highest denominations) correctly solves the Change Problem. So for example, to make change for 234 Shillings the greedy algorithms would take four 50 shilling coins, one 25 shilling coin, one 5 shilling coin, and four 1 shilling coins.

Solution:

Proof. Proof by counter example. When the Shilling total is equal to 40, the optimal solution would use two 20 shilling coins, so $n = 2$ where n is the number of coins. According to the greedy method of taking as many coins as possible from the highest denomination available, a shilling total of 40 would have change made with one 25 shilling coin, one 15 shilling coin, and one 5 shillingcoin making $n = 3$ which is not optimal. □

6

Problem: Consider a long quiet country road with houses scatter very sparsely along it (We can picture the road as a long line segment). You want to place cell phone base stations at certain points along the road, so that every house is within four miles of one of the base stations. Give an efficient algorithm that achieves this goal, using as few stations as possible. Show that the algorithm achieve the optimal solution using the "stay ahead" argument.

Solution:

Proof. □