

ArsDigitaUniversity
Month5:Algorithms -ProfessorShaiSimonson

ProblemSet2 –DataStructuresandGraphAlgorithms

1.PracticewithRed -BlackTrees

- a. ShowthesuccessiveRed -Blacktreesatareconstructedbyaddingthekeys10,20,30,40,50,60,70,80,90,100intoaninitiallyemptytree.
- b. Showhowtodeletethevalues60,70and90assumingthetreedoesNOThavetomaintain theRed -Blackproperties.

2.PracticewithGraphAlgorithms

- a. Exhibitagraphthathasnegativeedgesbutnonegative cycles,whereDijkstra’salgorithm computesanincorrectshortestpath.
- b. Anintervalgraphisanundirectedgraphwherethenodesareclosedintervalsonthereal numberline,andtheedgesconnectintervalsthatoverlap.Showthedepthfirstsearch tree withalltreeedges,andbackedges(see page483forareview)ofthefollowinginterval graph: {[1,3],[2,4],[6,9],[5,10],[8,9],[8,11],[3,6],[1,4],[3,7]}.Assumethatthe graphdatastructureordersthenodesastheyarelisted,and youstartyoursearchfromthe firstnode.
- c. Usingthegraphonpage499,showtheshortestpathtreeconstructedbyDijkstra’s algorithmandthebreadthfirstsearchtreeconstructedbyabreadthfirstsearch.Assume thatyoustartfromnode *a*.

3.Short estPathAlgorithm

DoestheBFS -scanningshortestpathalgorithm(similartoBellman -Ford)workonagraphwitha negativecostcycle?Ifyes,provethatitdoes,otherwiseexplainwhathappensandwhy.Use examples.

4.TopologicalSorting

Writecodet otopologicallysortagivendirectedgraphbyrepeatedlyfindingavertexofin - degree0anddeletingitfromthegraph.Makesureyouralgorithmrunsinatmost $\theta(n^2)$ time.

5.FindingaCycleinanUndirectedGraph

WriteanalgorithmusingDFSthatd etermineswhetherornotanundirectedgraphhasa cycle.Youralgorithmshouldprintthecyclethatitfinds.Analyzethetimecomplexity ofyouralgorithm.(Notethatalthoughthepreviousproblemcanbeusedtodetect whetherthereisacycle,itdoes notidentifythecycle).Hint:UseastackinyourDFS algorithmtokeeptrackoftheverticesintheorderthattheyarevisited.Becarefulto distinguishbetweenabackedge,andatreeedgeinthereversedirection.

6. Depth First Search

You are given a graph G where each vertex has an additional value E, B or W . (The array represents a final position in a game where E is empty, B is black and W is white.) The score for black is the total number of values labeled B plus all the values labeled E which are *surrounded* by B 's. The score for white is the total number of values labeled W plus all the values labeled E which are *surrounded* by W 's.

A node x is *surrounded* by B if and only if every path from x eventually reaches a B and never reaches a W . Being surrounded by W is defined analogously.

- Write a program to accept a graph representing the final position in the game, and output the final score.
- Test your program by creating and sending in a 7 by 7, 2-dimensional array holding E 's, B 's and W 's. The array should be converted to a graph, where the nodes are positions in the array, and the edges are connections between positions. Every position in the array is connected to four other positions, to the left, right, above and below. Positions on the edges are connected to only three other positions, the corners only to two. Diagonals are not connected, and the connections do not wrap around the sides of the board.

7. The Circus Problem

Describe an algorithm to solve the problem below. You don't need to code it - but be specific about data structures and procedures.

A circus is designing an act consisting of a tower of people standing on each other's shoulders. For practical and aesthetic reasons, each person on top of another must be both shorter and lighter. Given the heights and weights of each person in the circus, what is the largest number of people in such a tower?

Example: **Input:** 66510070150569075190609568110

Output: The longest tower is length 6 and includes from top to bottom:
56,90 60,95 65,100 68,110 70,150 75,190