School of Computer Science
University of Guelph

CIS*3490 The Analysis and Design of Algorithms

Winter 2022
Instructor: Fangju Wang

Assignment 5 Guide

- Q1.1
  Please read pages 218–223 for AVL tree operations. Please see slides 27 – 29 in "06 Transform and Conquer" for constructing an AVL tree from a list. The details of rotations can be found in slides 24 – 26.

- Q1.2
  Please read pages 223–225 for 2-3 tree operations. Please also see slides 32 – 33 in "06 Transform and Conquer" for constructing a 2-3 tree from a list.

- Q2.1, Q2.2
  Please read pages 226–232 for heap operations. Please see slide 43–45 in "06 Transform and Conquer". Slide 43 explains heap construction, and slides 44 – 45 describe heap sorting in array representation.

- Q2.3
  Please see slides 47 – 49 in "06 Transform and Conquer" for heap as priority queue. To remove the largest number, swap the root with the last element in the heap, remove the largest from the array representing the heap, and then perform a downheap operation on the new root. To add a new number to the queue, add it to the heap as the last element and perform a upheap operation on the last element.

- Q3.1
  Please read pages 261–264 in the text for the maximum-flow problem and the augmenting path method. Please see slides 11 – 18 in "10 Iterative Improvement" for how to find augmenting paths. The method is based on depth-first search (DFS).

- Q3.2
  Please see slides 25 – 31 in "10 Iterative Improvement" for the method of shortest augmenting paths. The method is based on breadth-first search (BFS).

- Q4.1, Q4.2
  Please read 11.3 in the text, and see slides in "11 Limitations of Algorithm Power". Keep in mind that up to today not a polynomial algorithm has been found to solve any NP complete problem.

- Q5.1
  Please see slide 13 in "12 Coping with the Limitations of Algorithm Power" for using a backtracking algorithm to solve a Hamiltonian circuit problem. The algorithm creates a state space tree when traveling the graph. Keep in mind that the algorithm travels the graph with the help of the tree. When the algorithm visits a node, it marks the node as "visited". When the algorithm goes back from a node, it marks the node as "unvisited". The backtracking method is based on depth-first search (DFS).

- Q5.2
  Please see slides 22 – 29 in "12 Coping with the Limitations of Algorithm Power" for using a branch and bound algorithm to solve a problem. Please also see pages 438 – 440 in textbook for an example of branch and bound for TSP problem. The branch and bound method is based on breadth-first search (BFS).

- Q6.1
  Please see slides 8 – 13 in "8 Dynamic Programming" for the dynamic programming algorithm. You can also read pages 292–294 in the text.

- Q6.2
  Please see slides 35 – 36 in "12 Coping with the Limitations of Algorithm Power" for the Sahni's approximation scheme for solving a knapsack problem.