Missouri University of Science & Technology — Department of Computer Science
Spring 2024 — **CS 2500: Algorithms**

Review Questions for Final Exam

Name: *Sid Nadendla* — **Email:** *nadendla@mst.edu*

Final exam will be a 2-hour long in-class exam on Thursday, May 3, 2024 at 3:30PM-5:30PM in CS 120. You may have 5 questions to solve, depending on the size of the questions. This handout presents review questions for the final examination.

Topic 1 Asymptotic Notation

1. If $f_1(n) = \Theta(g_1(n))$ and $f_2(n) = \Theta(g_2(n))$, prove that
$$f(n) = f_1(n) + f_2(n) = \Theta(g_1(n) + g_2(n)).$$
2. Our goal is to find both maximum and minimum elements of a given input array A of size n efficiently. A naive approach is to find the maximum using $n-1$ comparisons and then find the minimum element over the remaining entries using $n-2$ comparisons. Therefore, this algorithm takes about $2n-3$ comparisons. If you were to adopt a divide-and-conquer approach, what is the improvement in terms of the number of comparisons in closed form?

Topic 2 Sorting

1. Write the pseudocode for INSERTION-SORT and prove its correctness.
2. Demonstrate every iteration of MERGE-SORT on the input array $A = [1,\ 4,\ 7,\ 6,\ 3,\ 9,\ 5,\ 2,\ 4,\ 6]$.

Topic 3 Graph Search

1. Demonstrate Breadth-first search (BFS) on a graph, such as the one shown in Figure 1.
2. Demonstrate Depth-first search (DFS) on a graph, such as the one shown in Figure 1.

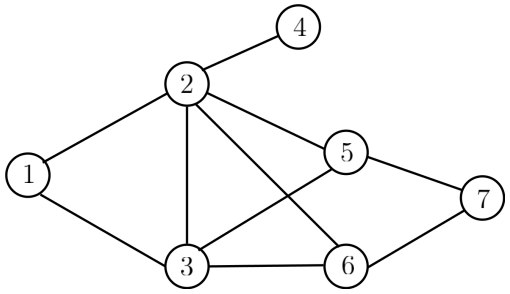

Figure 1: Unweighted Graph for Graph Search

Topic 4 Greedy Algorithms

1. Consider a job scheduling problem with n jobs. For any set of positive job weights $w_1, \cdots, w_n$ and positive job lengths $\ell_1, \cdots, \ell_n$, prove that the GREEDYRATIO algorithm outputs a schedule with minimum sum of weighted completion times.
   (Note: For your convenience, the pseudocode will be provided in this case.)

2. Demonstrate Prim's algorithm on the graph, such as the one shown in Figure 2, from some specific start node.

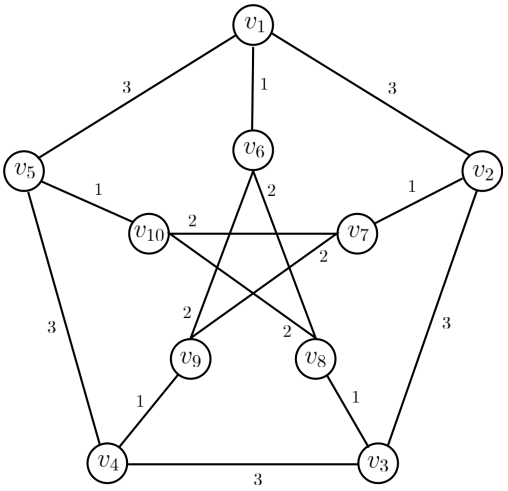

Figure 2: Weighted Graph for Minimum Spanning Trees

Topic 5 Dynamic Programming

1. Consider a 0-1 Knapsack problem with n indivisible items, where v_i and w_i are the value and weight of the i^{th} item respectively. If the size of the Knapsack is W,

   (a) Show that the Bellman equation (optimal substructure in value) for 0-1 Knapsack is

   $$V[i,j] = \max_{x_i \in \{0,1\}} v_i x_i + V[i-1, j-x_i w_i], \text{ for all } i = 1, \cdots, n \text{ and } j = 0, 1, \cdots, W.$$

   (b) Given the above Bellman recursion, design the pseudocode for the dynamic programming solution to 0-1 Knapsack problem.

2. Demonstrate the Bellman-Ford algorithm on the graph shown in Figure 3. Clearly show the value updates at each vertex at each and every iteration.

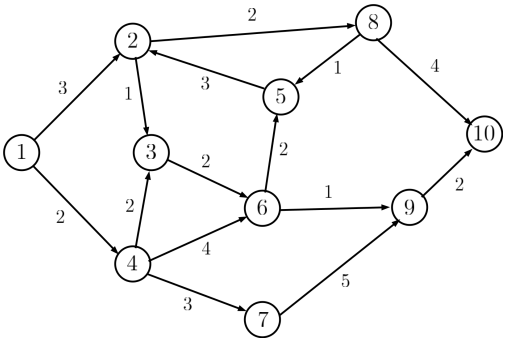

Figure 3: Weighted Graph for Bellman-Ford Algorithm