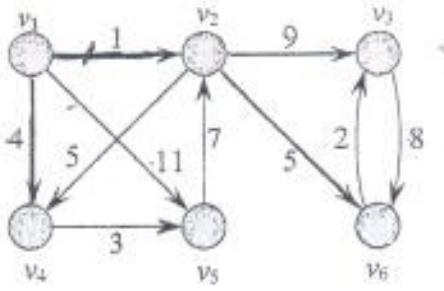
# CSCE 500 Midterm Exam #2

11/13/2019

1. The Dijkstra's algorithm (*DS*) solves the single-source shortest-path problem in a weighted directed graph $G = (V, E)$ without negative weighted edges or cycles, by edge relaxation at one vertex at a time until all vertices are examined. Given the graph $G$ below, follow *DS* to find shortest paths from vertex $v_1$ to all other vertices, with all predecessor edges shaded and estimated distance values from $v_1$ to all vertexes provided at the end. Also list the sequence of vertices at which relaxation takes place. **(12%)**

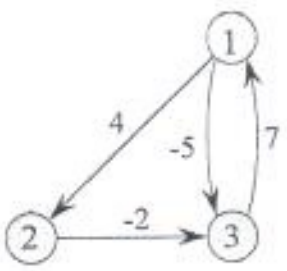
   What is the time complexity of *DS* for a general graph $G = (V, E)$, when candidate vertices are kept in an array? **(4%)**



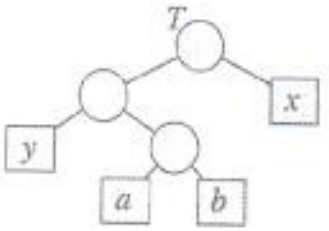

2. Follow depth-first search (*DFS*), starting from Node $v_1$ to traverse in the preceding graph. Mark (1) the type of every edge and (2) the discovery and the finish times of each node. (12%)

3. The Floyd-Warshall algorithm (*FW*) obtains all pairs of shortest paths in a weighted directed graph, with its pseudo code listed below. Consider the following graph, with its vertices labeled 1, 2, and 3. Derive all distance matrices $D^{(k)}$, for $0 \le k \le 3$, following *FW* so that the $d_{ij}^{(n)}$ element of final matrix $D^{(n)}$ denotes $\delta(i, j)$ for every vertex pair $<i, j>$. **(16%)**

```
FLOYD-WARSHALL(W, n)
D^(0) = W
for k = 1 to n
    let D^(k) = (d_ij^(k)) be a new n × n matrix
    for i = 1 to n
        for j = 1 to n
            d_ij^(k) = min(d_ij^(k-1), d_ik^(k-1) + d_kj^(k-1))
return D^(n)
```



4. Sketch a proof of the Lemma below, using the tree provided. (16%)
Let $C$ be an alphabet in which each character $c \in C$ has frequency $c.freq$. Let $x$ and $y$ be two characters in $C$ having the lowest frequencies. Then there exists an optimal prefix code for $C$ in which the codewords for $x$ and $y$ have the same length and differ only in the last bit.




5. A greedy algorithm (listed next) can be applied to determine one maximum set of compatible activities from an activity set.
Following the algorithm to derive one maximum compatible activity set out of the activity set given below. (12%)

GREEDY-ACTIVITY-SELECTOR$(s, f)$

$n = s.length$
$A = \{a_1\}$
$k = 1$
**for** $m = 2$ **to** $n$
  **if** $s[m] \geq f[k]$
    $A = A \cup \{a_m\}$
    $k = m$
**return** $A$

| $i$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $s_i$ | 1 | 3 | 0 | 5 | 3 | 5 | 6 | 8 | 8 | 2 | 12 |
| $f_i$ | 4 | 5 | 6 | 7 | 9 | 9 | 10 | 11 | 12 | 14 | 16 |

6. An optimal binary search tree (OBST) for a given set of keys with known access probabilities ensures the minimum expected search cost for key accesses. Given the set of four keys with their access probabilities of $k_1 = 0.24$, $k_2 = 0.19$, $k_3 = 0.11$, $k_4 = 0.15$, respectively, and five non-existing probabilities of $d_0 = 0.1$, $d_1 = 0.04$, $d_2 = 0.06$, $d_3 = 0.03$, $d_4 = 0.08$, construct OBST following dynamic programming with memoization for the given four keys and demonstrate the constructed OBST, which contains all four keys ($k_1$, $k_2$, $k_3$, $k_4$) and five non-existing dummies ($d_0$, $d_1$, $d_2$, $d_3$, $d_4$). (**28%**; show your work using the three tables, for expected costs: $e[i, j]$, access weights: $w[i, j]$, and $root[i, j]$, with $i$ in $e[i, j]$ and $w[i, j]$ ranging from 1 to 5, $j$ in $e[i, j]$ and $w[i, j]$ ranging from 0 to 4, and both $i$ and $j$ in $root[i, j]$ ranging from 1 to 4).

OPTIMAL-BST$(p, q, n)$

```
1  let e[1..n+1, 0..n], w[1..n+1, 0..n],
       and root[1..n, 1..n] be new tables
2  for i = 1 to n + 1
3      e[i, i − 1] = q_{i−1}
4      w[i, i − 1] = q_{i−1}
5  for l = 1 to n
6      for i = 1 to n − l + 1
7          j = i + l − 1
8          e[i, j] = ∞
9          w[i, j] = w[i, j − 1] + p_j + q_j
10         for r = i to j
11             t = e[i, r − 1] + e[r + 1, j] + w[i, j]
12             if t < e[i, j]
13                 e[i, j] = t
```