Midterm Solutions

B-405 Fall 2025

1 Problem 1

1.1 C

Simplify $f(n)$ we get $f(n) = 7n + 6$, so $f(n) = O(n)$

1.2 D

An undirected graph is connected means for every vertex pair (u, v), there exists a path (can be multiple hops) between them.

1.3 C

BFS cannot be used to find shortest path between a pair of vertices if the edge has weights. We should use Dijkstra algorithm instead.

1.4 B

One edge would bring 2 more degrees. Sum up all vertex degrees $1+1+2+2+3+3 = 12$. Edge number equals half of 12 which is 6.

1.5 B

To get topological ordering, we first find vertex C and B without incoming edges. After their contingent edges are removed, A and D has no incoming edges. So one possible topological ordering is C,B,A,D,E.

1.6 A

If every weight is the same, number of hops represents distance of a path. So BFS should be used.

2 Problem 2

2.1 a)

T_1 cannot be a BFS tree from vertex 1. In T_1, vertex 4 is traversed from vertex 3 which is not possible in a BFS tree. Vertex 4 should be traversed from vertex 2.

2.2 b)

T_2 cannot be a DFS tree since vertices 5, 6 and 7 should be traversed from vertex 4, instead of vertex 1.

3 Problem 3

3.1 a)

In the interval algorithm, we always choose the task with earliest finishing time.

- We first choose I_2
- After I_2 is done, I_1 has conflict with I_2, we can only choose I_3

- After I_3 is done, we choose I_4, and subsequently I_5 and I_6.

Therefore the optimal schedule is I_2, I_3, I_4, I_5 and I_6.

3.2 b)

The optimal schedule for this setting is I_2, I_4 and I_6.

3.3 c)

Algorithm 1 Lazy Interval Schedule

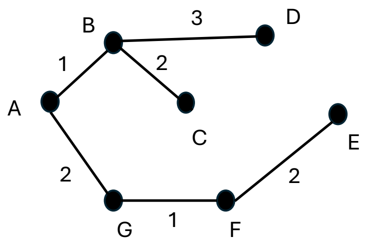
Sort all jobs with ascending finishing time
$curTime \leftarrow -\infty$
$Schedule \leftarrow \emptyset$
for each job (p_i, q_i) in sorted job set do
  if $curTime + 1 \leq p_i$ do
    $curTime \leftarrow q_i$
    Add job (p_i, q_i) into $Schedule$
return $Schedule$

Explanation: Following standard interval scheduling algorithm, sort all jobs with ascending finishing time. For each job, if the current time plus 1 (one unit of free time) $\geq$ the job's staring time, this indicates there's no conflict between the job and the previous schedule. So the current job will be appended into our final schedule. Besides, update current time to the job's finishing time.

Time complexity: Sorting takes $O(nlogn)$ time. The following loop takes $O(n)$ time as every job is iterated only once. So total time complexity is $O(nlogn)$.

4 Problem 4

4.1 a)



Path $A \rightarrow G \rightarrow F \rightarrow E$ has the minimum weight connecting A to E.

4.2 b)

Path $A \rightarrow B \rightarrow C \rightarrow D \rightarrow E$ has the minimum weight.

4.3 c)

Algorithm 2 Problem 4(c)

Run Dijkstra Algorithm from vertex s, denote the shortest path tree as T_s
Run Dijkstra Algorithm from vertex t, denote the shortest path tree as T_t
if $T_s(u) + T_t(v) < T_s(v) + T_t(u)$ do
  Return shortest path $s \rightarrow u$ and edge (u, v) and shortest path $v \rightarrow t$
Else
  Return shortest path $s \rightarrow v$ and edge (v, u) and shortest path $u \rightarrow t$

Explanation: If edge (u, v) must be used, there are only 2 options: either $s \to u \to v \to t$ or $s \to v \to u \to t$. To find the minimum weight path, we need to run twice Dijkstra algorithm, one from s and another one from t, so that we can obtain the shortest distance of $s \to u$, $s \to v$, $t \to u$ and $t \to v$. After these distances are obtained, compare two options and return the smaller one.

Time complexity: This algorithm takes twice the running time of Dijkstra algorithm, which is $O(E + V log V)$. To find the minimum path, it takes $O(V')$ which is negligble.