Homework #3

1. Let S be a packing obtained by the greedy algorithm, and let S' be an optimal packing. Assume that $\overline{M} < M$ and $\overline{M} \leq \frac{1}{2}C$. Note that $|S' - S| \geq 1$.

   Suppose for every $i \in S' - S$ that $w_i \leq \frac{1}{2}C$. Then the greedy algorithm would have added at least one element of $S' - S$ to S.

   Suppose for some $i \in S' - S$ that $\frac{1}{2}C < w_i \leq C$. Then the greedy algorithm's first choice would have been an object of weight at least $\frac{1}{2}C$.

   Since both cases contradict our assumption, either $\overline{M} = M$ or $M > \frac{1}{2}C$.

2. (a) Schedule the jobs in increasing order by deadline.

   (b) By induction on the number of jobs scheduled. Let S be a schedule obtained by the greedy algorithm. Suppose that some prefix of S consisting of i jobs can be extended to an optimal schedule S'.

   This is trivially true for the base case, when $i = 0$.

   Let j be the $(i+1)$st job in S. Let J be the set of jobs scheduled by S' between the ith job in S' and j.

   Since every job in J must have a deadline at least as late as that of j (or the greedy algorithm would schedule it before j), and S' schedules j to finish before its deadline, we can interchange the jobs in J with j.

   Therefore the prefix of S consisting of $i + 1$ jobs can be extended to an optimal schedule.

3. (a) Assume that G has two distinct minimum spanning trees, T_1 and T_2. Since T_1 and T_2 are trees on the same vertex set, they have the same number of edges. Therefore there exists an edge e of least cost which appears in exactly one of T_1 and T_2.

   Without loss of generality, suppose that $e \in T_1$. The graph $T_2 \cup \{e\}$ contains exactly one cycle, C. Since T_1 does not contain a cycle, there must be an edge $e' \in C$ such that $e' \notin T_1$.

   By choice of e, we know that $c(e') > c(e)$. Therefore the cost of the tree $T_2 \cup \{e\} - \{e'\}$ must be less than the cost of T_2. This contradicts our assumption that T_2 is a minimum spanning tree.

   Therefore G has a unique minimum spanning tree.

   (b) The claim is true. By the correctness of Kruskal's algorithm, G has a minimum spanning tree T_1 which contains e_1 and e_2. Assume that G has a second distinct minimum spanning tree T_2.

Again, there exists an edge e of least cost which appears in exactly one of T_1 and T_2. If $c(e) > c(e_1) = c(e_2)$ then remainder of the proof is the same as in 3 (a).

Otherwise, $e = e_1$ or $e = e_2$. In both cases, $e \in T_1$. Let e' be as in 3 (a). Since $e' \notin T_1$, we know that $e' \notin \{e_1, e_2\}$, so $c(e') > c(e)$. The remainder of the proof is the same as in 3 (a).

(c) The claim is false. One counter-example is a triangle with equal edge costs.

4. (a) Let $A[0 \ldots k, 0 \ldots |V|]$ be a matrix with every element initialized to $\emptyset$. We use $\oplus$ to denote the concatenation operator for sequences.

```
A[0,0] ← ⟨v_0⟩
B ← NO-SUCH-PATH

for i : 1...k
  for (v_j, v_j') ∈ E
    if A[i−1, j] ≠ ∅ and σ(v_j, v_j') = σ_i then
      A[i, j'] ← A[i−1, j] ⊕ ⟨v_j'⟩
      if (i = k) then
        B ← A[k, j']
      end if
    end if
  end for
end for

return B
```

(b) By induction on the rows, $A[i, j']$ contains a path $\langle v_0, \ldots, v_{j'} \rangle$ labeled by $\langle \sigma_1, \ldots, \sigma_i \rangle$, if such a path exists; otherwise it contains $\emptyset$.

In the base case, $A[0, j']$ contains the path $\langle v_0 \rangle$ if $j' = 0$; otherwise it contains $\emptyset$.

Suppose that the property holds for all rows up to the $(i-1)$st, for $i \geq 1$. Suppose that there exists a path $\langle v_0, \ldots, v_j, v_{j'} \rangle$ labeled by $\langle \sigma_1, \ldots, \sigma_i \rangle$. Then by the inductive hypothesis, $A[i-1, j]$ contains some path from v_0 to v_j labeled by $\langle \sigma_1, \ldots, \sigma_{i-1} \rangle$.

Inspecting the code, we see that when considering the edge $(v_j, v_{j'})$ we must assign $A[i-1, j] \oplus \langle v_{j'} \rangle$ to $A[i, j']$. This can only be overwritten by another such path.

Suppose that there does not exist a path $\langle v_0, \ldots, v_{j'} \rangle$ labeled by $\langle \sigma_1, \ldots, \sigma_{i-1} \rangle$. Then for every edge $(v_j, v_{j'})$, by the inductive hypothesis $A[i-1, j] = \emptyset$. We then make no assignments to $A[i, j']$.

We assign a path to B if and only if we assign it to $A[k, j']$ for some j'.

Therefore B contains a path beginning at v_0 labeled $\langle \sigma_1, \ldots, \sigma_k \rangle$ if such a path exists. Otherwise it contains NO-SUCH-PATH.

(c) Excluding the initialization of `A` (which is $O(k|V|)$), the algorithm uses $O(k|E|)$ time.