COT5405 Algorithm Final Solution Key

Yichi Zhang

December 2021

Question 1

Algorithm:

Algorithm 1: Find Change

1 **KnapSack**
Input: item count n, values $v_{1..n}$, weights $w_{1..n}$, times $t_{1..n}$, max weight W, max time T
Output: decisions $d_{1..n}$ where $\forall d_i \in 0, 1$, maximum value V
2 Create 3-d matrix V. Initialise with $\forall V_{i,j,k} = 0$
3 **for** $i = 0ton$ **do**
4   **for** $w = 0toW$ **do**
5   **for** $t = 0toT$ **do**
6   **if** $w_i > w \ or \ t_i > t$ **then**
7         $V_{i,w,t} = V_{i-1,w,t}$
8   **else**
9         $V_{i,w,t} = max(V_{i-1,w,t}, v_i + V_{i-1,w-w_i,t-t_i})$
10   **end**
11   **end**
12   **end**
13 **end**
14 $w = W$
15 $t = T$ **for** $i = 1ton$ **do**
16   **if** $V_{i,w,t} > V_{i-1,w,t}$ **then**
17     $d_i = 1$, $w = w - w_i$, $t = t - t_i$
18   **end**
19 **end**
20 **return** $d[]$, $V_{n,W,T}$

Proof of correctness:
Using DP method, we build 3-d matrix V, which each entry $V_{i,w,t}$ stands for optimal value for the first i items with maximum weight w and time t. Then we iterate through matrix to compute optimal solution and fill each entry. For the ith item we have choice of either picking it or not, corresponding to max function on line 9.

After that we simply backtrack and build up array of selection $d[]$.
Proof by induction:
Base case: $\forall V_{0,w,t} = V_{i,0,t} = V_{i,w,0} = 0$ trivially true.
Inductive hypothesis: if $V_{i-1,w,t}$ is correctly calculated as maximum value for first $(i-1)$ items with given maximum weight w and time t, then $V_{i,w,t}$ is as well.
Proof: we only have two choices, either put ith item in bag or not, which corresponds to two cases in line 9. According to the optimal substructure of this question it can be proven that max of the two is the optimal solution at i, w, t.
Time complexity analysis:
Nested for loop on line 3 through 5, time complexity is $O(nWT)$.

Question 2

Algorithm: For input graph $G(V, E)$, build weighted directed graph $G'(V, E', w)$ s.t. $\forall u, v \in V$, $(u, v) \in E \rightarrow (u, v) \in E', (v, u) \in E', w(u, v) = w(v, u) = 1$.
Proof of correctness: Partition of G separating s and t corresponds to a cut

Algorithm 2: Find Change

Ford-Fulkerson
    Input: Weighted graph $G'(V, E', w)$, source and target $s, t \in V$
    Output: Max flow value v
    $G'_f = G'$
    $v = 0$
    $\forall f(u, v) = 0$
    while *exists augmenting path p from s to t in G_f* **do**
        $v = v + 1$
        for $e \in p$ **do**
            $f(e) = f(e) + 1$
            remove e from G'_f
        end
    end

C on G'. Since $\forall e \in E' = 1$, minimum number of edges in cut $min(|C|) = maxflow(G')$ by max flow min cut theory.
Time complexity analysis: Since capacity of all edges is 1, the max flow is at most $O(|V|)$ in the form of $O(|V|)$ augmenting paths with flow 1, with each can be found in $O(|E|)$ time. So total time complexity is $O(|V||E|)$.

Question 3

To prove that given problem $BIPART$ is $NP-complete$, we need to prove it is in NP and in $NP-hard$.
$BIPART \in NP$: given any bipartition of A into A_1, A_2 to $BIPART$, where $A_1 \cup A_2 = A$ and $A_1 \cap A_2 = \emptyset$. It takes linear time to add up elements in either set and check if $\sum A_1 = \sum A_2 = \sum A/2$. Thus $BIPART \in NP$.
$BIPART \in NP-hard$: reduce $SUBSET-SUM$ to $BIPART$. For set $A = \{a_1, a_2 \cdots a_n\}$, let $a_0 = \sum A - 2t$ and $A' = A \cup \{a_0\}$, where t is the target for $SUBSET-SUM$. Suppose $\exists A_1 \in As.t. \sum A_1 = t$, then $A_2 = A - A_1$, $\sum A_2 = \sum A - t$, then $\sum(A_1 \cup \{a_0\}) = \sum A - t = \sum A_2$, we have a $BIPART$ for set A'. This shows $SUBSET-SUM$ reduces to $BIPART$.
Combining both NP and $NP-hard$ proof we know $BIPART$ is $NP-complete$.