Homework 6

Thomas Kim tsk389 51835

David Munoz dam2989 51840

CS331 Algorithms and Complexity

Problem Q1. Given integers $a_1, \ldots, a_n$, design a dynamic programming algorithm that determines whether there exists a partition of the numbers into 3 disjoint subsets P, Q, R such that the sum of the numbers in each set are equal. In other words,

$$\sum_{a_i \in P} a_i = \sum_{a_i \in Q} a_i = \sum_{a_i \in R} a_i$$

Memoization

Let the set of integers $a_1, \ldots, a_n$ be denoted A
Let the minimum element $a_i \in A$ be denoted $min(A)$
Let the sum of all elements $a_1, \ldots, a_n$ be denoted $sum(A)$
Create a table of dimension $min(A)$ by $sum(A)$
Each element of the table $e_{j,k}$ stores whether there exists a partition such that $\sum_{a_i \in P} a_i = j \wedge \sum_{a_i \in Q} a_i = k$

Algorithm $OPT(i, j, k)$

Let $A' := \{a_i, a_{i+1}, \ldots, a_n\}$
if $j = k = 0 \wedge i = n + 1$ **then**
    Fill in the table entry for coordinates (j, k) to be true
    return TRUE
else if $j \leq 0 \wedge k \leq 0 \wedge i \neq n + 1$ **then**
    Fill in the table entry for coordinates (j, k) to be false
    return FALSE
end if
return $OPT(i + 1, j - a_i, k) \vee OPT(i + 1, j, k - a_i)$

Call $OPT(1, \frac{sum(A)}{3}, \frac{sum(A)}{3})$

Correctness

Proof. □

Runtime

Proof. □

Problem Q2. Given positive integers n and W, design a dynamic programming algorithm that finds the number of possible n element sets $\{x_1, \ldots, x_n\}$ for which $\sum_i x_i^2 = W$ where each x_i is a nonnegative integer. **Memoization**

Create an n by W table
Each element (j, k) of the table will hold the number of unique j element sets for which $\sum_i x_i^2 = k$

Algorithm $OPT(i, j, k)$

if $i = 0$ **then**
    return 0
end if
return $OPT(i - 1, j - 1, k - i^2) + OPT(i - 1, j, k)$

Call $OPT(0, n, W)$

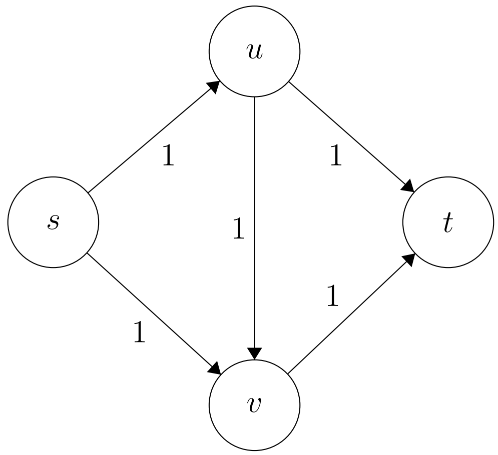
Correctness

Proof. □

Runtime

Proof. □

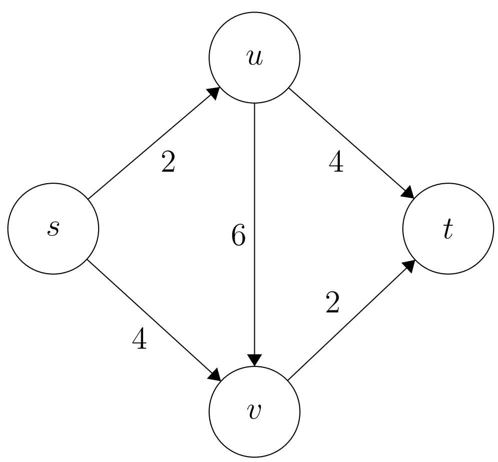
Problem Q3(a). List all the minimum $s-t$ cuts in the flow network in figure 7.24 (reproduced below).



Minimum cuts:

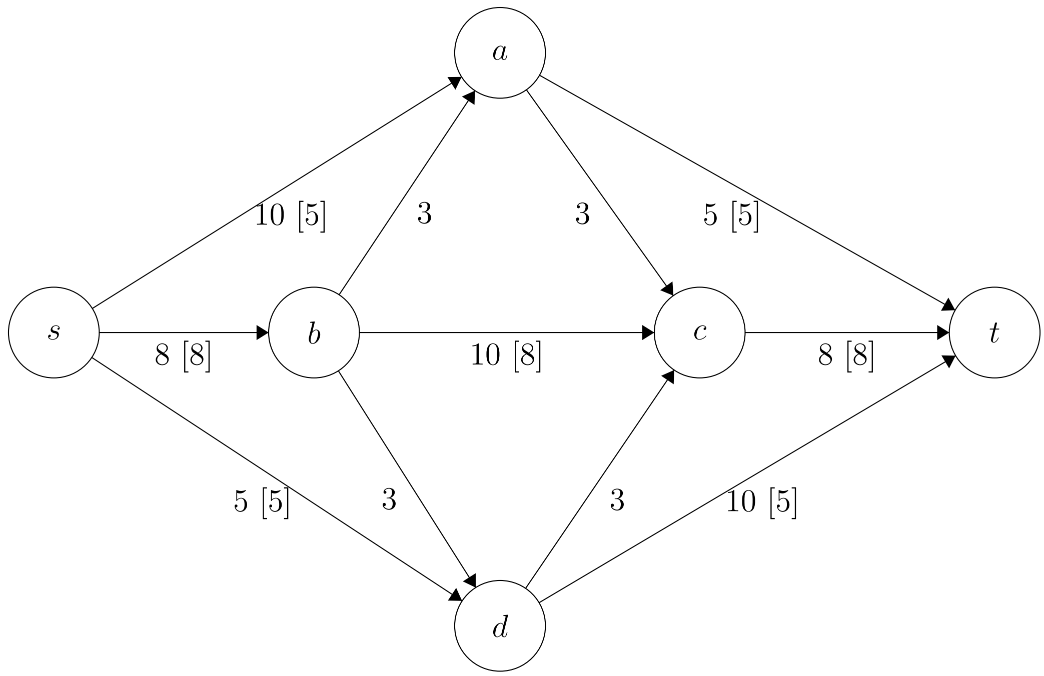
1. $\{s\}, \{u, v, t\}$
2. $\{s, v\}, \{u, t\}$
3. $\{s, v, u\}, \{t\}$

Problem Q3(b). What is the minimum capacity of an $s - t$ cut in the flow network in Figure 7.25 (reproduced below)?

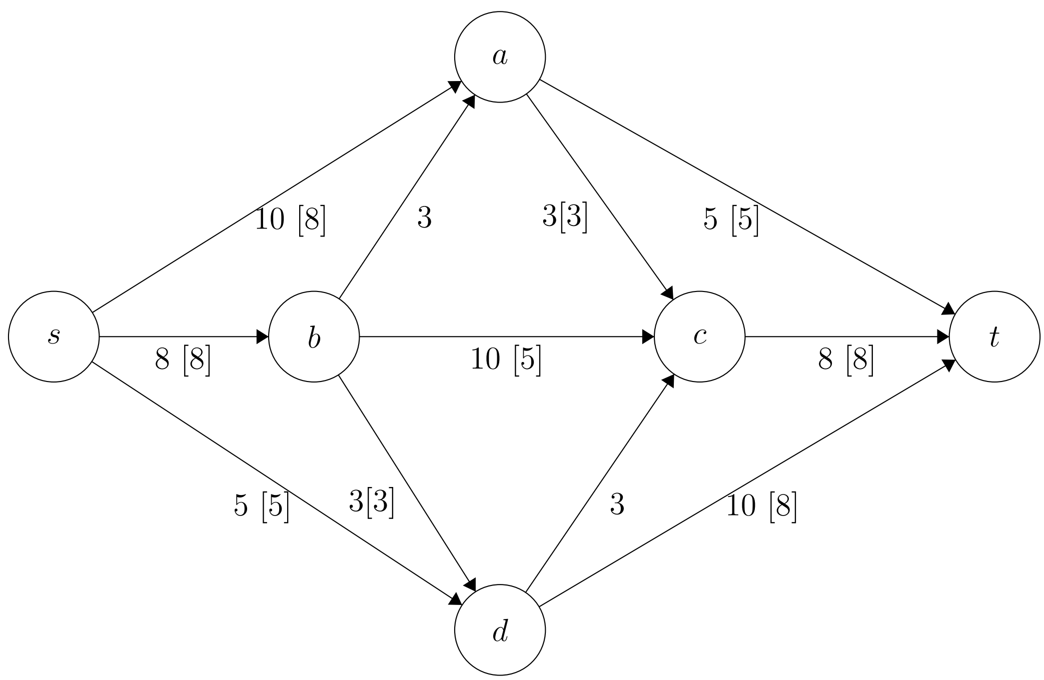


Minimum capacity: 4 (The cutset for $\{s, v\}$ contains edges (s, u) and (v, t) each of weight 2)

Problem Q4(a). What is the value of the $s - t$ flow depicted in figure 7.26? Is this a maximum (s, t) flow in this graph? The figure is reproduced below with additional node labels.



$v(f) = 18$, this is not a maximum flow.
There exists, trivially, a flow of higher value shown below.



Problem Q4(b). Find a minimum $s-t$ cut in the flow network pictured in figure 7.26, and also say what its capacity is.
A minimum $s-t$ cut is $\{s, a, b, c\}, \{d, t\}$ with a capacity of 21.

Problem Q5. Consider a set of mobile computing clients in a certain town who each need to be connected to one of several possible *base stations*. We'll suppose there are n clients, with the position of each client specified by its (x, y) coordinates in the plane. There are also k base stations; the positions of each of these is specified by (x, y) coordinates as well. For each client, we wish to connect it to exactly one of the base stations. Our choice of connections is constrained in the following ways.

- There is a *range parameter* r - a client can only be connected to a base station that is within distance r.
- There is also a *load parameter* L - no more than L clients can be connected to any single base station.

Your goal is to design a polynomial-time algorithm for the following problem. Given the positions of a set of clients and a set of base stations, as well as the range and load parameters, decide whether every client can be connected simultaneously to a base station, subject to the range and load conditions in the previous paragraph.

Create a graph $G(V, E)$
Let some edge $e(a, b)_w$ be a directed edge from a to b with weight w
Let the set of clients be denoted $C = \{c_1, c_2, \ldots, c_n\}$
Let the set of base stations be denoted $B = \{b_1, b_2, \ldots, b_n\}$
Define the function $valid(c_i)$ to return some set $B' = \{b'_1, b'_2, \ldots, b'_n\} \subset B$ such that c_i is within range $\forall b_i \in B'$
$V = \{s, t, c_1, c_2, \ldots, c_n, b_1, b_2, \ldots, b_n\}$
Initially $E = \{\}$
for $\forall v_i \in V$ **do**
    if $v_i \in C$ **then**
        $B' = valid(v_i)$
        $E = E \cup \{(s, v_i)_1, (v_i, b'_1)_1, \ldots, (v_i, b'_n)_1\}$
    end if
    if $v_i \in B$ **then**
        $E = E \cup \{(b_i, t)_{L_{b_i}}\}$
    end if
end for
Run Ford-Fulkerson on G to find maximum flow f
if $f = n$ **then**
    return TRUE
else
    return FALSE
end if

Correctness

Proof. □

Runtime

Proof. □