

Student Name: Duc Tu Luong
Last 3-digit ID: 122

Homework #5

1. Ford-Fulkerson Algorithm

Credit: <http://www.geeksforgeeks.org/ford-fulkerson-algorithm-for-maximum-flow-problem/>

a) Pseudo code:

Let G be the node – link graph
Let rG be the residual graph and let $rG = G$ initially
Let E be the number of edges
Let $f(e)$ be the flow of edge e
Let $C(e)$ be the capacity of edge e
Let $maxFlow$ be the maximum possible flow of graph
Let $f(e) = 0$ for all edges and let $maxFlow = 0$ initially

While there is an $s - t$ path P from s to t using BFS

An $s - t$ path exists if $f(e) < C(e)$ for every edge e on the path

If no path found

return $maxFlow$

Else

Find minimum residual capacity of the edges

along the path P filled by BFS and assign to $pathFlow$

Update residual capacities of the edges and
reverse edges along the path

For each edge in path P

Let u be the start node of the edge

Let v be the end node of the edge

$rG[u][v] = rG[u][v] - pathFlow$

$rG[v][u] = rG[v][u] + pathFlow$

Endfor

Endif

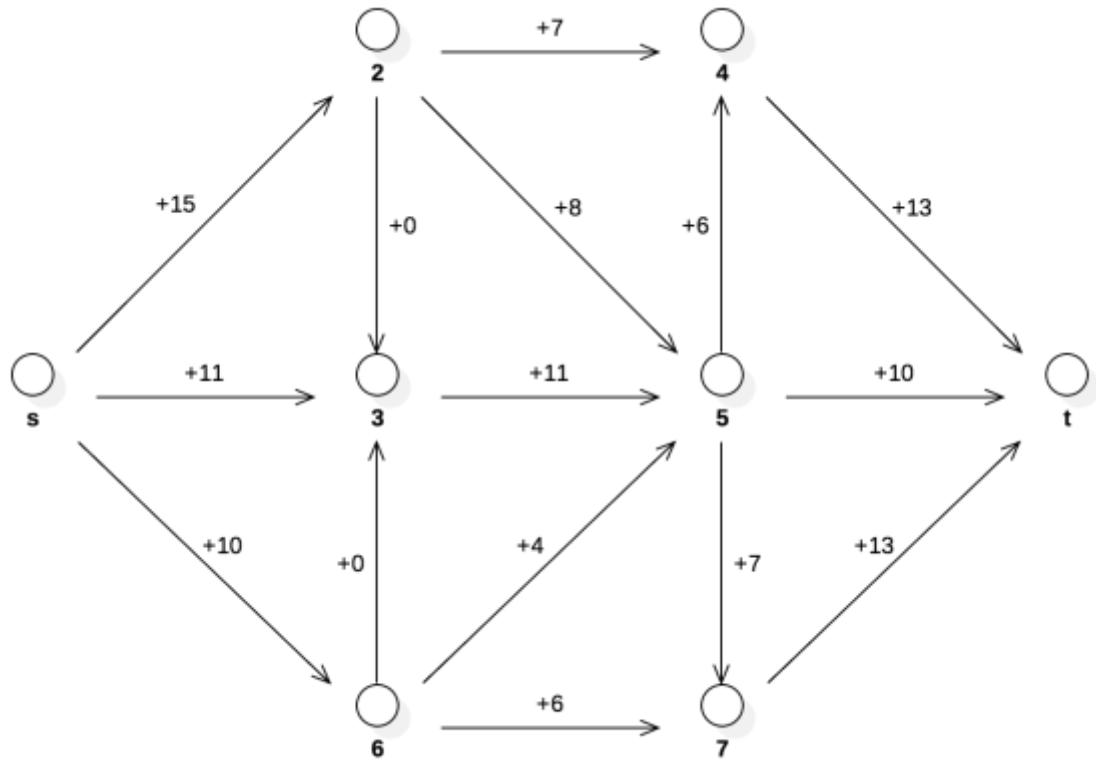
$maxFlow = maxFlow + pathFlow$

Endwhile

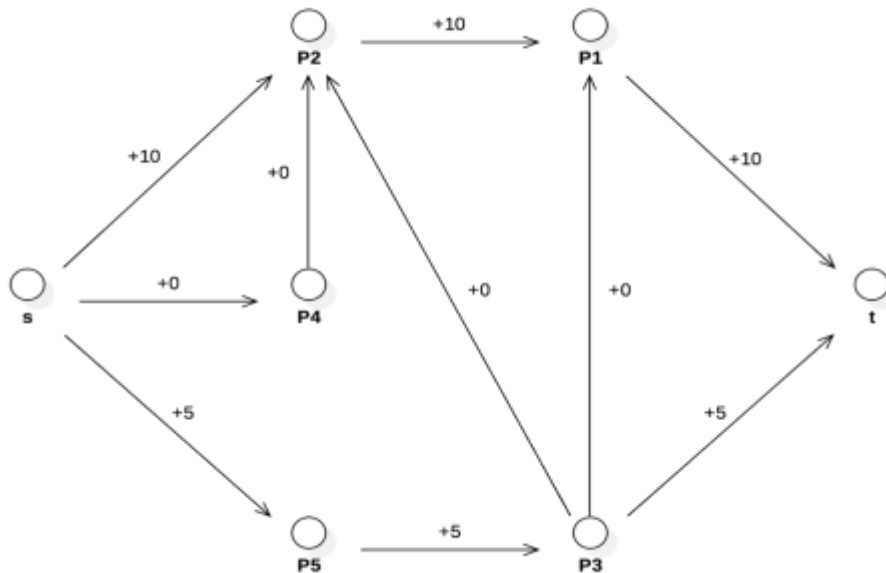
return $maxFlow$

b) Let F_m be the max flow, E is the total number of nodes in graph. When we iterate while there is still augmenting path, the worst case is that we only add 1 unit in every iteration, as a result, the time complexity is $O(F_m * E)$

c) The maximum possible flow is 36, the flow graph is as following:



2. We model the problem using network flow diagram. The network flow of profit is as following:



The maximum profit we can make is 15, we will take P1, P2, P3, P5 to generate this profit.

3. a)

Problem:

- n : number of doctors
- D : set of vacation days; $|D| = d$
- $S_i \subseteq D$: set of vacation-days doctor i can work
- k : vacation periods
- $D_j \subseteq D$: days in period j

Give a polynomial algorithm which returns assignment of doctors to vacation days, or reports if no such assignment exists; and satisfies following constraints:

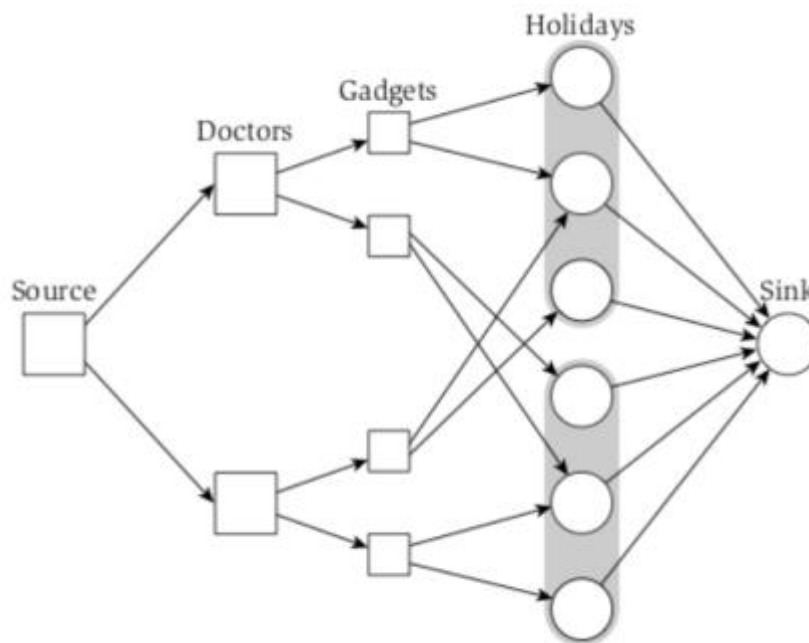
- For a given parameter c , each doctor should be assigned to work at most c vacation days total, and only days when he or she is available.
- For each vacation period j , each doctor should be assigned to work at most one of the days in the set D_j .

Solution:

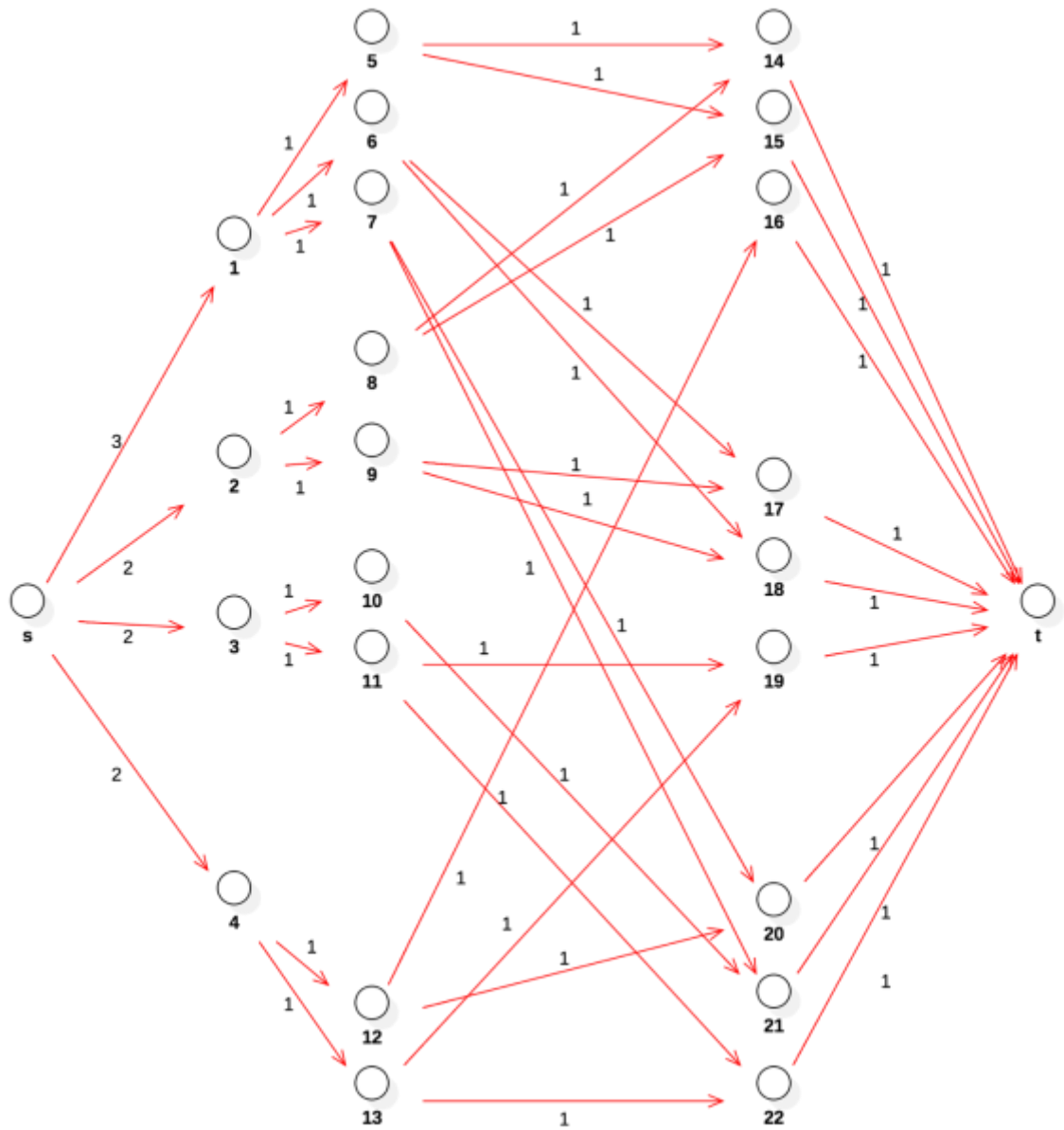
We use network flow model to solve this problem. We have a node u_i representing each doctor attached to a node v_l representing each day when he or she can work; each edge has capacity of 1.

We attach a super-source s to each doctor node u_i by an edge of capacity c , and we attach each day node v_l to a super-sink t by an edge with upper and lower bounds of 1. Suppose there are d vacation days total; we put a demand of $+d$ on the sink and $-d$ on the source.

We include a new node w_{ij} with an incoming edge of capacity 1 from the doctor node u_i , and with outgoing edges of capacity 1 to each day in vacation period j when doctor i is available to work. The network flow model is as following



b) Problem instance with 4 doctors and 3 vacation periods. The network flow diagram is as following



- Vertices from 1 to 4 represent 4 doctors.
- Vertices from 5 to 13 represent “gadgets” for 4 doctors, these gadgets guarantee each doctor only cover 1 day in each vacation period.
- Vertices from 14 to 22 represent vacation days, each 3-day block represents each vacation period.

c) We encode the network flow diagram into a matrix with values of each edge as following

```
new int[][]{ /* s 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 t */
/* s */ {0, 3, 2, 2, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
/* 1 */ {0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
/* 2 */ {0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
/* 3 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
/* 4 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0},
/* 5 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0},
/* 6 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0},
/* 7 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0},
/* 8 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0},
/* 9 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0},
/* 10 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0},
/* 11 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0},
/* 12 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0},
/* 13 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0},
/* 14 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0},
/* 15 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1},
/* 16 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1},
/* 17 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1},
/* 18 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1},
/* 19 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1},
/* 20 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1},
/* 21 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1},
/* 22 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1},
/* t */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1},
};
```

When applying to question 1 implementation, the result is as following:

Maximum flow is 9, which is all vacation days are covered, the details are as following:

- Day 1 of vacation period 1 is covered doctor 1
- Day 2 of vacation period 1 is covered doctor 2
- Day 3 of vacation period 1 is covered doctor 4

- Day 1 of vacation period 2 is covered doctor 1
- Day 2 of vacation period 2 is covered doctor 2
- Day 3 of vacation period 2 is covered doctor 3

- Day 1 of vacation period 3 is covered doctor 1
- Day 2 of vacation period 3 is covered doctor 3
- Day 3 of vacation period 3 is covered doctor 4

4. a)

Problem:

n users of the website

k groups from $\{G_1, \dots, G_k\}$ of the demographic groups, which users belong to

m advertisers

Give an efficient algorithm to decide if there is a way to show a single ad to each user so that the site's contracts with each of the m advertisers is satisfied for this minute? (That is, for each $i = 1, 2, 3, \dots, m$, can at least r_i of the n users, each belonging to at least one demographic group in $X_i \subseteq \{G_1, \dots, G_k\}$, be shown an ad provided by advertiser i ?), and if so, to choose an ad to show each user.

Solution:

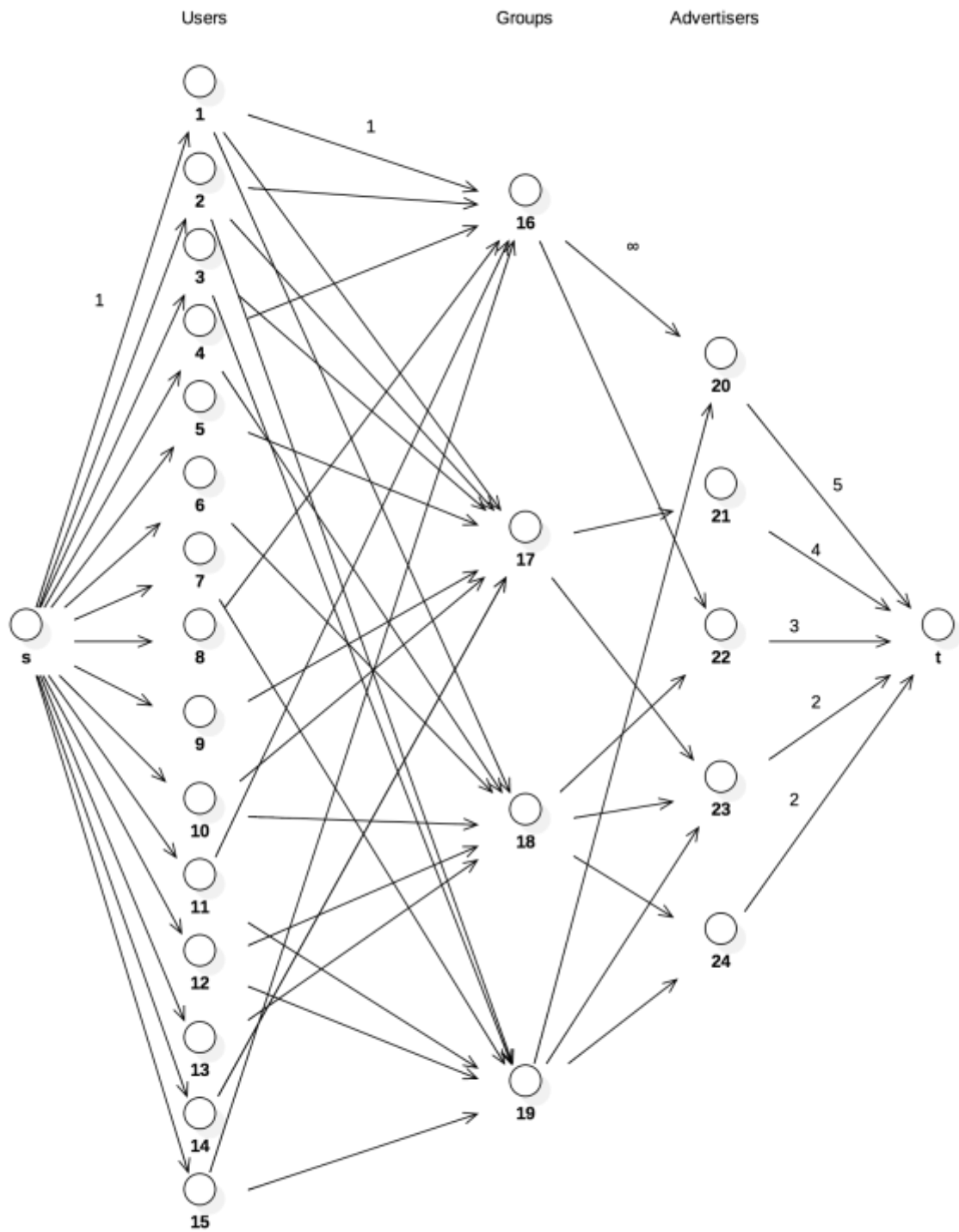
We design a flow network $G = (V, E)$ as following:

- There is a source s
- Vertices $u_1 \dots u_n$ for all users
- Vertices $g_1 \dots g_k$ for all demographic groups
- Vertices $v_1 \dots v_m$ for all advertisers
- There is a sink t
- There is an edge of capacity 1 from u_i to each group g_j for which user i belongs to a demographic group.
- There is an edge of capacity ∞ from each group g_j to at least one advertiser v_l .
- There is an edge with a capacity of 1 from s to each u_i for each i .
- There is an edge with a capacity of the site contracts with each of m advertisers which is r_j of the n users from v_j to t for each j .
- The source s has supply of n
- The sink t has demand of n

If there is a valid circulation in this graph, meaning that advertiser v_l shows ads to a subset of vertices of $g_1 \dots g_k$, in which a subset of $u_1 \dots u_n$ users belong to, which mean that advertiser shows ads to appropriate people. We will find out the maximum flow in this graph, if this maximum flow is equal to n , it means advertisers show their ads to all site's users with appropriate content.

b) Problem instance with $k=4$, $n=15$, $m=5$ is represented as following network flow diagram

- Each edge from s to vertices 1 to 15 has capacity of 1 (source to each user vertex)
- Each edge from vertices 1 to 15 to vertices 16 to 19 has capacity of 1 (user vertices to groups)
- Each edge from vertices 16 to 19 to vertices 20 to 24 has capacity of ∞ (groups to advertisers)
- Each edge from vertices 20 to 24 to sink t has capacity of site's contract to that advertiser (advertisers to sink t).



c) We encode the network flow diagram into a matrix with values of each edge as following, we use value 100 to encode ∞

```
int graph[][] =
new int[][]{ /* s 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 t */
/* 1 */ {0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
/* 2 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0},
/* 3 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0},
/* 4 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0},
/* 5 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0},
/* 6 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0},
/* 7 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0},
/* 8 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0},
/* 9 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0},
/* 10 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0},
/* 11 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0},
/* 12 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0},
/* 13 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0},
/* 14 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0},
/* 15 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0},
/* 16 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 100, 0, 100, 0, 0, 0},
/* 17 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 100, 0, 100, 0, 0},
/* 18 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 100, 100, 0},
/* 19 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 100, 0, 0, 100, 100, 0},
/* 20 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 5},
/* 21 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 4},
/* 22 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3},
/* 23 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2},
/* 24 */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2},
/* t */ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}
};
```

When applying to question 1 implementation, the result is as following:

The maximum flow is 15, which means all users are shown appropriate ads.

- Node 20 to sink t has flow of 5/5, which means the site's contract between advertiser 20 has been met.
- Node 21 to sink t has flow of 4/4, which means the site's contract between advertiser 21 has been met.
- Node 22 to sink t has flow of 3/3, which means the site's contract between advertiser 22 has been met.
- Node 23 to sink t has flow of 2/2, which means the site's contract between advertiser 23 has been met.
- Node 24 to sink t has flow of 1/2, which means the site's contract between advertiser 24 has NOT been met.