CMSC 641 Design & Analysis of Algorithms Spring 2023
Section 01 Homework 7

Faisal Rasheed Khan

VB02734

vb02734@umbc.edu

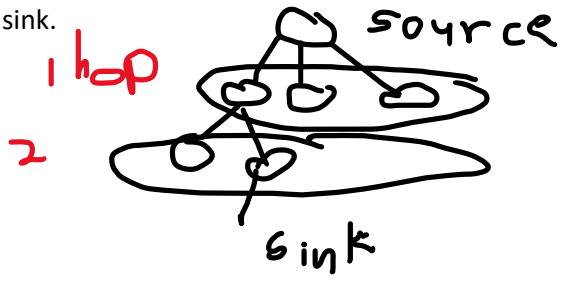
1 Good Augmenting Paths

Any algorithm for maximum flow that uses the Ford-Fulkerson method has to cope with the problem of picking a "good" augmenting path. If a "bad" augmenting path is chosen, the flow will increase, but only a little. This will lead to numerous iterations and a slow algorithm. What if there aren't any good augmenting paths? what if they just don't exist? The Max Flow Min Cut Theorem won't help us here because if there are still bad augmenting paths, then we have not achieved maximum flow yet. For this problem, you will show that there is always a sequence of at most E augmenting paths (starting from zero flow) that leads to the maximum flow.

Assignment:

1. You are given a flow network G. Suppose that you "cheat" and peek at a max flow f. (I.e., you look at the answer before doing the problem.) How would you pick a good augmenting path for the zero flow in G? (Remember that you are trying to minimize the number of iterations in the Ford-Fulkerson method.)

A. In order to pick a good augmenting path we apply Breadth-First Search on the graph, each vertex hop distance is stored from the source, and select augmenting path as min link path from source to sink.



Augmenting saturates the edge by sending max flow.
Find different Augmenting path, if the edge gets saturated the link distance increases monotonically.
As you are Augmenting, link distance gets higher and higher to any node. At some point link distance from source to sink becomes ∞, and then BFS stops.

2. How would you update the max flow f so you can pick another good augmenting path? This augmenting path would augment the good path you picked above and still result in a valid flow for G. Note: Do this without constructing a residual graph.

A. In order to update the max flow from the current augmenting path select the minimum capacity edge which is in the current augmenting path.
$c_f(p) = \{ \min c_f(u,v) \mid (u,v) \text{ is edge in } p \}$
the min capacity edge (u,v) is saturated.
Add the flow capacity of min edge (u,v) i.e. min $c_f(u,v)$ to all the edges in path $c_f(p)$ and subtract their original capacities with min $c_f(u,v)$.
Now the edge (u,v) is saturated the vertex of u becomes ∞
Now again compute the BFS, for the min link distance of all the vertices from source.
Now again you pick minimum augmenting path and repeat till there is no augmenting paths left.

3. Argue that you can keep picking good augmenting paths this way.

A. We know that after the edge gets saturated, distance of the link from source increases monotonically. By default we are saying that we are picking good augmenting paths. We need to prove this.
$\delta_f(s,v)$=minimum # of edges in a path from s to v in G_f
=link distance from s to v in G_f
=$\delta_f(v)$, since we only use $\delta_f(s,-)$
Lemma:
For each v∈V-{s,t}, $\delta_f(v)$ increases monotonically after each flow augmentation using BFS
Proof of Lemma(By Contradiction):
Let $f' = f \uparrow f_p$ be the first augmentation that causes $\delta_f()$ to decrease for some vertex v i.e.
$\delta_{f'}(v) < \delta_f(v)$ -----Eq 1
Out of all such v pick the one with smallest $\delta_{f'}()$ ------Eq 2
Let p' be a min link path in $G_{f'}$ from s to v
p': s---------->u--------->v-------->t
then $\delta_{f'}(v) = \delta_{f'}(u) + 1$ ---------Eq 3 Since p' is min link path
By Eq 2,
$\delta_{f'}(u) \geq \delta_f(u)$ ---------Eq 4
If $\delta_{f'}(u) < \delta_f(u)$, then we would have picked u instead of v
Two cases :
$(u,v) \in E_f(u)$ or $(u,v) \notin E_f(u)$
Both gives contradiction.

Therefore, Our Lemma holds true.

Therefore, we can claim that in this way we are picking good augmenting paths.

4. Argue that you will eventually reach max flow if you pick good augmenting paths this way.

A. We know that after the edge gets saturated, distance of the link from source increases monotonically. By default, we are saying that we are picking good augmenting paths. And also, we are maximizing the flow by saturating the edges. We need to prove this.

δ_f(s,v)=minimum # of edges in a path from s to v in G_f

=link distance from s to v in G_f

=δ_f(v), since we only use δ_f(s,-)

Lemma:

For each v∈V-{s,t}, δ_f(v) increases monotonically after each flow augmentation using BFS

Proof of Lemma(By Contradiction):

Let $f^{|}$ = f↑ f_p be the first augmentation that causes δ_f() to decrease for some vertex v i.e.

$\delta_f^{|}$(v) < δ_f(v) -----Eq 1

Out of all such v pick the one with smallest $\delta_f^{|}$() ------Eq 2

Let $p^{|}$ be a min link path in $G_f^{|}$ from s to v

$p^{|}$: s---------->u--------->v-------->t

then $\delta_f^{|}$(v)= $\delta_f^{|}$(u) + 1 ---------Eq 3 Since $p^{|}$ is min link path

By Eq 2,

$\delta_f^{|}$(u)≥ δ_f(u) ---------Eq 4

If $\delta_f^{|}$(u) $<$ δ_f(u), then we would have picked u instead of v

Two cases:

(u,v) ∈ E_f(u) or (u,v) ∉ E_f(u)

Both gives contradiction.

Therefore, Our Lemma holds true.

Therefore, we can claim that in this way we are picking good augmenting paths.

By saturating the edges, we are getting the link path increase monotonically, it is nothing but we are considering the maximum flow across the graph. We do this till no augmenting paths are left.

Therefore we can reach max flow while picking good augmenting paths.

5. Argue that you will pick at most E augmenting paths.

A. We In order to prove we pick at most E augmenting paths, we need to prove the running time of the Edmonds-Karp i.e. O(VE), where V is the vertex and E is the Edge.

Lemma:

Edmonds-Karp takes at most O(VE) iterations.

Proof:

Each edge (u,v) can be critical no more than $\frac{v}{2}$ times, i.e. each edge can be saturated at most $\frac{v}{2}$ times, otherwise link distance from s to v > v-1

At most E* $\frac{v}{2}$ augmentations, since each augmenting path saturates at least one edge-----Eq 1

So total iterations ≤ E* $\frac{v}{2}$ = O(VE)

Hence the Lemma is proved.

Consider Eq 1,

At most E* $\frac{v}{2}$ augmentations, since each augmenting path saturates at least one edge
Since V has E edges, according to our lemma stated some of the edges along with some vertex gets saturated and at most $\frac{v}{2}$ vertex are there.
After few edge saturations there will be fewer augmentations left.
The augmentation will be less as edges are saturated, therefore the upper bound will have at most E Augmentations.

2 Petulant Baristas

A few years from now, after some soul-searching, you discover that your true passion is not computer science after all. It's managing a coffee shop! You cash in your life savings and open your own espresso bar. After a while you find out that attracting customers is the easiest part of your job. Your real problem is getting competent employees to show up for work.

Although your coffee shop employs k baristas, there are b bad weeks in the year when many of them refuse to work. After haggling for a while, you and the baristas finally agree on these rules:

• Each barista will provide a list of at least t days in bad weeks when they will work.

• You will assign a barista to work during a bad week only on the days on that barista's list.

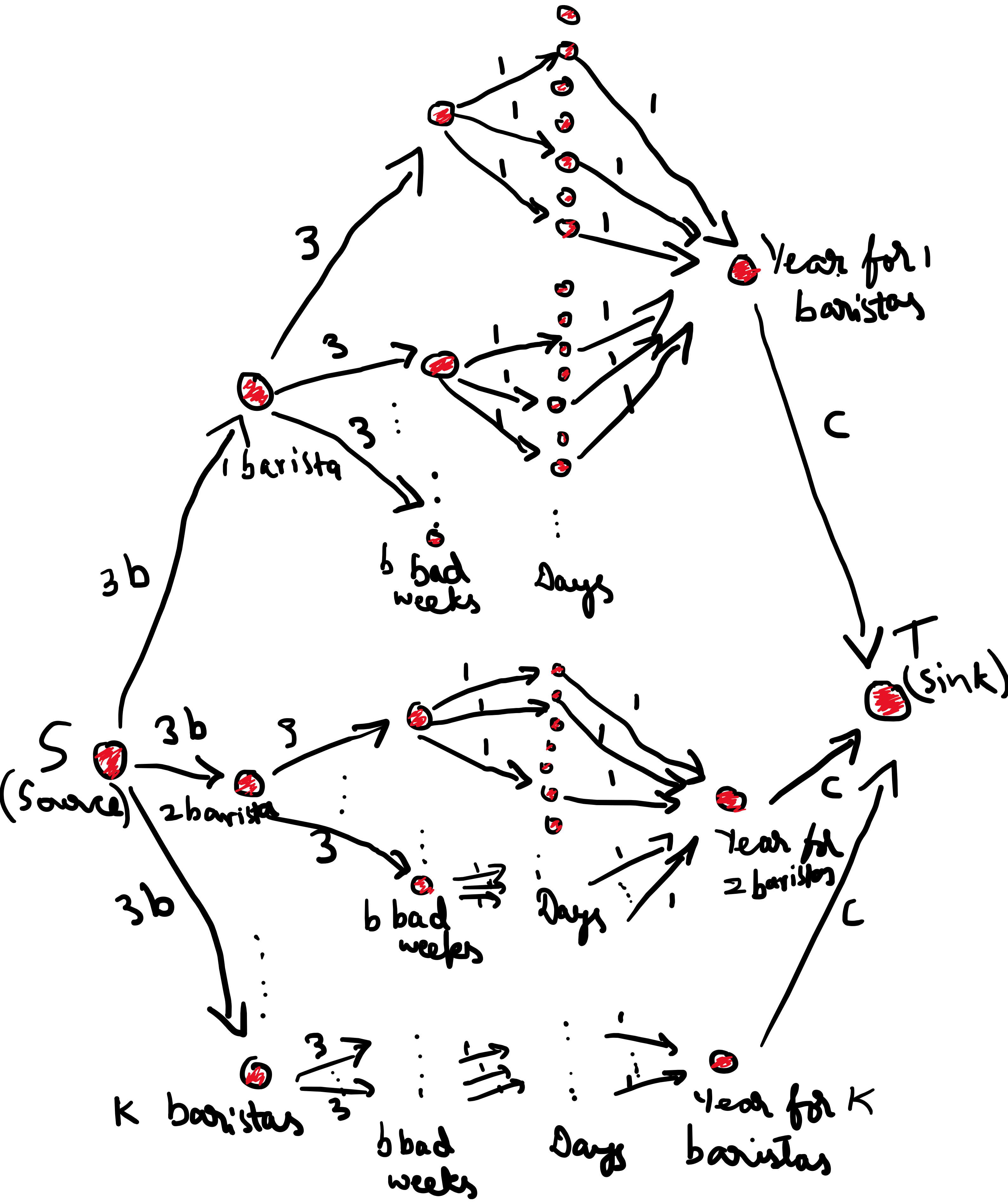
• No barista will have to work for more than c days during bad weeks in the entire year.

• No barista will have to work for more than 3 days during the same bad week.

The baristas turn in their lists, but you struggle to follow the agreed upon rules and have coverage for the bad weeks, because you need at least one barista to work on every day of the bad weeks. Fortunately, you remember enough algorithms to know that this problem can be solved with a flow network.

1. Using a combination of a diagram, descriptive labels and English sentences, describe how you can transform the problem described above into a flow network.

A. <u>Buliding a flow network:</u>
1 node per barista k
1 node per year for barista k
1 node per bad weeks per barista b*k
1 node per day per bad weeks per barista 7*b*k
We have 3*b capacity from source node S to each barista (i.e. 3 capacity limit over b weeks)
Each barista node has 3 capacity to b bad weeks
Each b bad weeks of each barista has max three 1 capacity nodes to days
Each days (considering all days of b bad weeks) of each b bad weeks of each barista max 3b 1 capacity nodes to year node of each barista
Each year node has c days capacity to sink node T



S
(Source)
3b
1 barista
2 barista
K baristas
3
b bad weeks
Days
Year for 1 baristas
Year for 2 baristas
Year for K baristas
C
T
(sink)

2. Suppose that a solution does not exist — that is, there is no way to assign at least one barista to every bad day, given the constraints described. Show how a maximum flow in the network you constructed can be used to determine that no solution is possible.

A. In order to conclude about the solution, we have max flow min cut theorem to find the maximum flow form the network graph.
Max flow algorithm tells us that the solution is possible or not.
Max Flow Min Cut Theorem:
Let f be the legal flow in the flow network G=(V,E). Then,

- f is maximum flow in G
- G_f has no augmenting paths
- |f| = cut capacity of some cut (S,T)

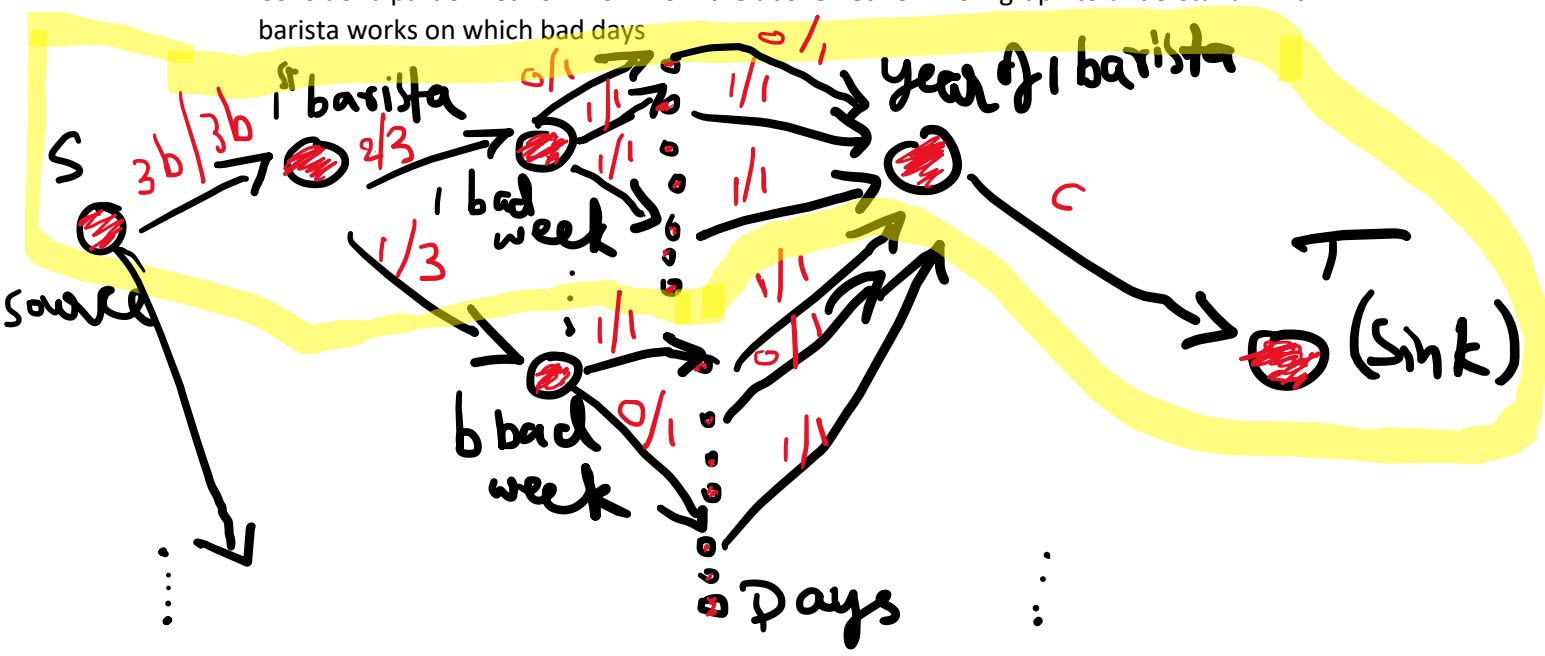
In Max Flow Min Cut theorem, cut(S,T) gives maximum flow f

Consider the sink node T of our network flow created.

Consider the cut R in our network flow,
If the Maximum Flow value f given by Max Flow Min Cut theorem(which runs on our network flow) is less than the cut capacity R, then solution is not possible.

3. Argue that your transformation works. That is, explain how a maximum flow of the flow network you constructed can be used to identify which baristas work on which bad days when a solution does exist.

A. By Applying Max Flow Min Cut Algorithm, it gives us all the information regarding which baristas work on which bad days.
Max Flow Min Cut Algorithm saturates the necessary edges and ensures that there are no augmenting paths left in the network.
Consider a part of network flow from the above network flow graph to understand which barista works on which bad days



Consider the highlighted section of the above network graph, the 1st barista which is assigned to the bad week days can be found by analysing the flow of the network.

If the edge is saturated i.e. flow zero (considering 1 barista) from the 1 bad week to the day node, then that bad day is not assigned to that barista(1 barista).(we are considering here ony 1st barista and 1st bad week)

If the edge has a flow i.e 1/1 from 1 bad week to the day node then that barista(1 barista) is assigned to that bad day.

3 The New Office

You are the chief operating officer of a company and you are considering moving your office to a new building across town. The new building has better facilities and lower heating and cooling costs. It is closer to most of your employees' homes, so a move would also reduce their commuting time. Happier employees have higher productivity, too.

Unfortunately, the CEO and founder of your company loves the view from his corner office in the old building and absolutely refuses to move. Since there is nothing more you can do to change the CEO's mind, you plan to move just a subset of your employees to the new building. Even with Zoom meetings, having your employees at two separate locations will incur some costs. So, you attempt to determine which employees you should move to the new building and which ones will stay behind with the CEO.

These are your parameters and constraints:

• Your company has n employees, including the CEO and yourself.

• For employee i, you have calculated the total benefit bi gained by the company, if you move employee i to the new building. Assume that bi is a non-negative integer dollar amount.

• Some employees work closely together and putting them in separate buildings would create a cost due to a reduction in productivity. For each pair of employees i and j, you have calculated a cost cij incurred by the company if you placed employees i and j in separate buildings. (The cost is the same whether you move employee i or employee j.) Assume that cij is a non-negative integer dollar amount.

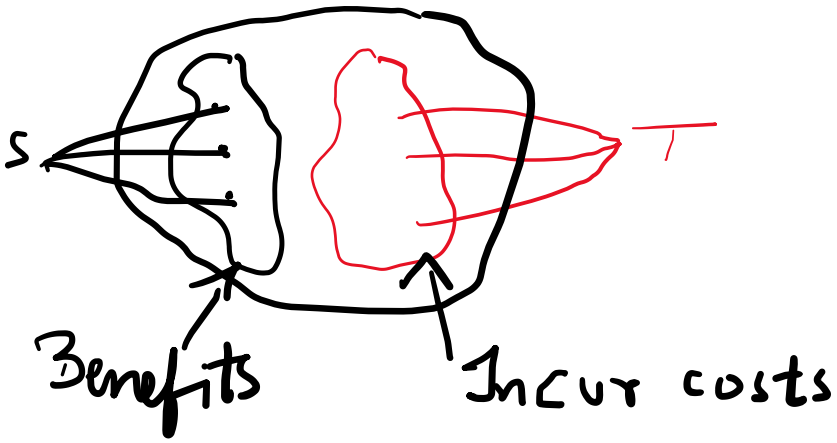
• As stated previously, the CEO, employee #1, stays in the old building.

You must use network flow to solve this problem. The solution produced by your algorithm must maximize the sum of the total benefits minus the sum of the total costs.

Note: You have cij for every pair of employees. If two employees do not work together, the corresponding cij may be very small or zero. Also, your employees tend to work on multiple projects, so there are no convenient "clusters" for you to move around. Let max flow do the work for you, instead of trying to solve the problem directly.

1. Using a combination of a diagram, descriptive labels and English sentences, describe how you can transform the problem described above into a flow network.

A.



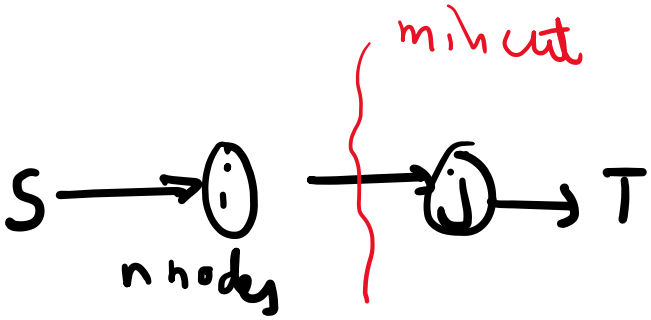
For each benefit generated, G_i , add edge (S,G_i)

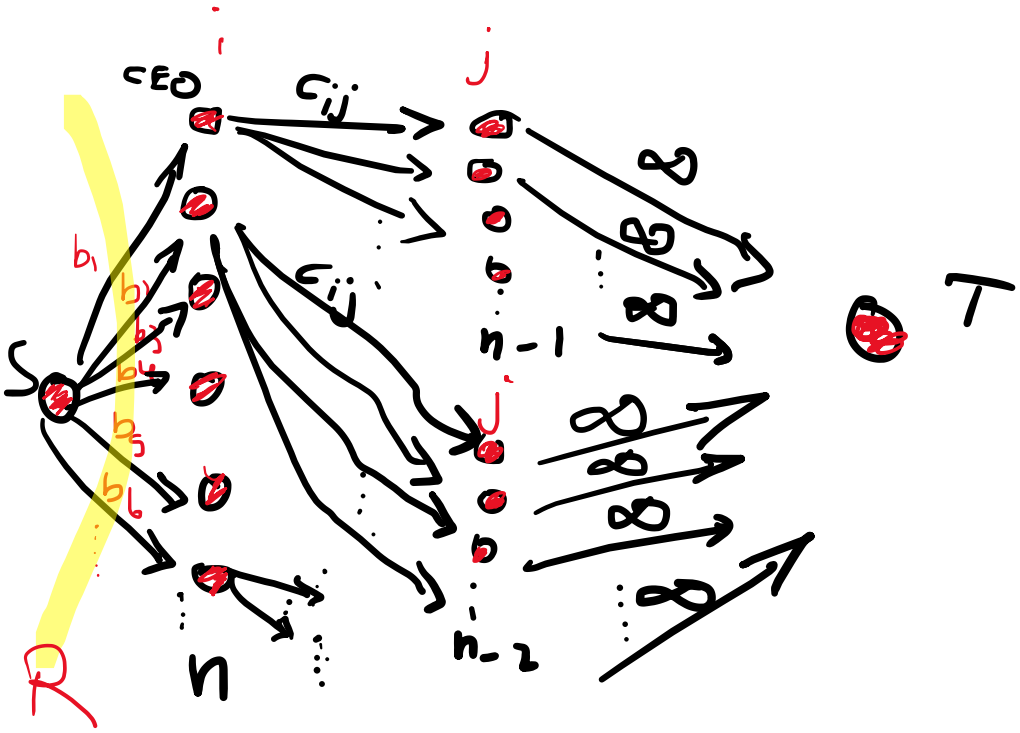
Set $c(S,G_i) = b_i$

For each incurring cost, G_i , add edge (G_i,t)

Set $c(G_i , G_j) = c_{ij}$

For edge from G_j to sink, add edge, $c(G_j , T) = \infty$





1 node per n employees

1 node per n-1 employees per each employee (CEO)

1 node per n-2 employees per each employee

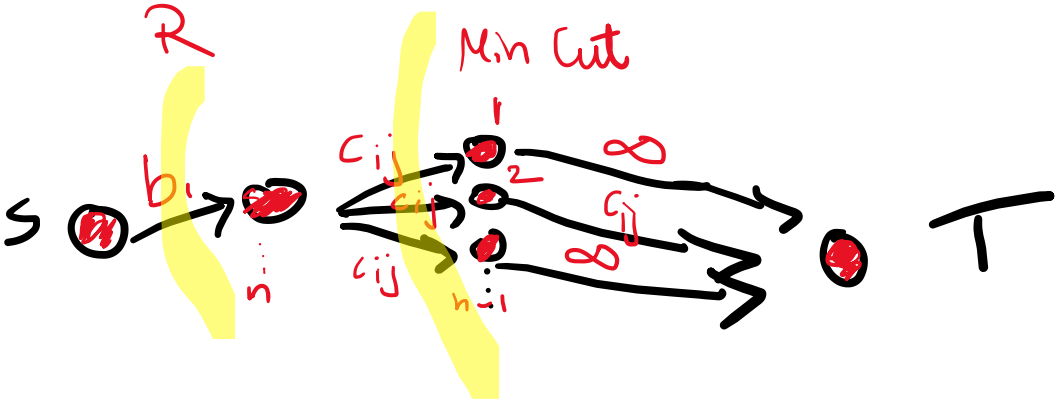
Each n employee has capacity b_i from source

Each n-1/n-2 employee has c_{ij} capacity from each employee n

Each sink nodes has capacity ∞ from n-1/n-2 employees

2. Argue that your transformation works. That is, explain how a maximum flow of the flow network you constructed can be used to identify the set of employees to move.

A. By Applying Max Flow Min Cut Algorithm, it gives us the set of employees to move.
Max Flow Min Cut Algorithm saturates the necessary edges and ensures that there are no augmenting paths left in the network.
Consider a part of network flow from the above network flow graph to understand which set of employees to move.
None of the dependency edges with ∞ capacity can cross cut from S to T (cannot have ∞ in min cut)
Consider a sub part of the above network flow



The edges which are not ∞ are considered, and the edges which have the value not ∞, their network flows are considered from where it is generated.
It is generated from the 2nd node, so the 2nd node is moved to the new building.

3. Explain why the solution produced by your algorithm maximizes the sum of the total benefits minus the sum of the total costs

A. Compute Max Flow of the network graph, with the help of Max Flow Min Cut Algorithm
Max Flow Min Cut Theorem:
Let f be the legal flow in the flow network G=(V,E). Then,

- f is maximum flow in G
- G_f has no augmenting paths
- |f|= cut capacity of some cut (S,T)

In Max Flow Min Cut theorem, cut(S,T) gives maximum flow f

Let (S,T) be the min cut found
Selected Employees = S – {s,subset of n employees except CEO)
Claim: Select Employees to move to other building such that company gets maximum benefits.
None of the dependency edges with ∞ capacity can cross cut from S to T (cannot have ∞ in min cut)
c(S,T) = benefits of employees not moved + expense of employees moved
= R – (benefits of employee moved – incur cost associated with the moved employees)
c(S,T) = R – profit
profit = R – c(S,T)
As much as we minimize cost, the higher will be the profit.

Therefore, minimize cost = Maximize profit
Therefore, it maximizes the sum of the total benefits minus the sum of the total costs.

References:

https://archive.org/details/introduction-to-algorithms-by-thomas-h.-cormen-charles-e.-leiserson-ronald.pdf/page/427/mode/2up

https://archive.org/details/AlgorithmDesign1stEditionByJonKleinbergAndEvaTardos2005PDF/page/n201/mode/2up

12.1-Network-Flow-post.pdf - Google Drive

12.2-Max-Flow-Min-Cut-post.pdf - Google Drive

13-Network-Flow-Applications-post.pdf - Google Drive

15.3 Edmonds-Karp with Proofs (optional).pdf - Google Drive