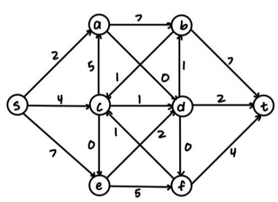
# CMSC 27200 - Problem Set 6

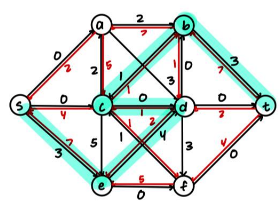
Sohini Banerjee

February 24, 2024

## 1

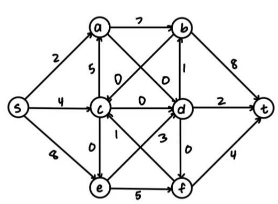
As indicated, the final iteration demonstrates that there is no path from $s$ to $t$ in the residual graph, so we have found the maximum flow from $s$ to $t$. Based on Ford-Fulkerson and the min-cut / max-flow theorem, the maximum flow is 15.

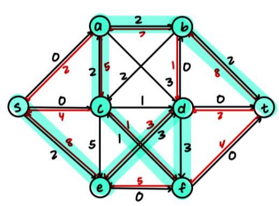




Flow path: s→e→d→c→b→t

capacity: 1

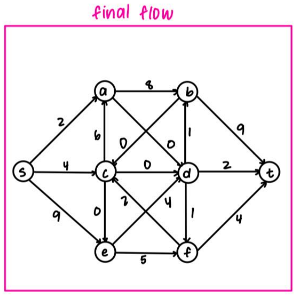


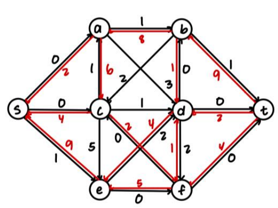


Flow path: s→e→d→f→c→a→b→t

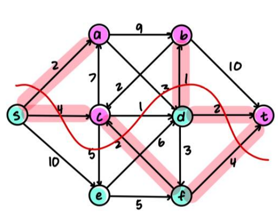
Capacity: 1

final flow





Maximum flow: 15



Minimum cut: L={s,e,d,f}, R={a,b,c,t}

2+4+1+2+2+4=15

# 2

## Algorithm

We are given the maximum flow $F$ on $G$ from $s$ to $t$.

1. Determine what the residual graph is from the current flow.
2. Determine if edge $e$ was at full capacity.
   (a) If edge $e$ was not at full capacity, terminate.
   (b) If edge $e$ was at full capacity, run BFS on the residual graph to determine whether there is a path from $u$ to $v$ in the residual graph.
      i. Suppose there is a path from $u$ to $v$ in the residual graph. Call the path from $u$ to $v$ $P$. For each edge $e'$ in $P$, increase the flow on $e'$ by 1. Reduce the flow on $e'$ by 1 since it exceeded the new capacity and terminate.
      ii. Suppose there is no path from $u$ to $v$ in the residual graph. There must be a path from $u$ to $s$ and $t$ to $v$ in the residual graph. Call the path from $u$ to $s$ $X$ and the path from $t$ to $v$ $Y$.
         A. For each edge $e'$ in $X$, increase the flow on $e'$ by 1.
         B. For each edge $e'$ in $Y$, increase the flow on $e'$ by 1.
         C. Reduce the flow on $e'$ by 1 since it exceeded the new capacity.
         D. Run 1 iteration of Ford-Fulkerson and terminate.

## Proof

Overview: Denote the initial capacity of $e$ as $c(e)$, decreased capacity as $c'(e) = c(e) - 1$, initial flow on $e$ as $f(e)$, and final flow on $e$ as $f'(e)$. If the initial flow on $e$ exceeds its new capacity, we have an extra unit of flow coming into $e$ that we either have to route along another path or remove from $F$.

**Case 1**: Edge $e$ was not at full capacity.

We know that $f(e) < c(e)$, so $f(e) \leq c(e) - 1 = c'(e)$ since all capacities and flows are integers. This means $e$ can handle the initial flow $f(e)$ with the decreased capacity, so no further changes are needed.

**Case 2**: Edge $e$ was at full capacity and there is a path from $u$ to $v$ in the residual graph.

If there is a path from $u$ to $v$ in the residual graph, we are not routing maximum flow from $u$ to $v$ in the flow graph. Since $c(e)$ decreased, we have an extra unit of flow to route to $v$ that cannot go through edge $e$ since $c'(e) < c(e)$ and $e$ was at full capacity. Therefore, we route the extra unit of flow through the residual path from $u$ to $v$ so that $v$ still receives the extra unit of flow but through another path, not including $e$.

**Case 3**: Edge $e$ was at full capacity and there is no path from $u$ to $v$ in the residual graph.

If there is no path from $u$ to $v$ in the residual graph, we are routing maximum flow from $u$ to $v$ in the flow graph. We will prove the following 2 observations:

- We will show that there exists a path from $u$ to $s$ in the residual graph. Let $U$ denote the set of vertices reachable from $u$ in the residual graph. Since there is no path from $u$ to $v$ in the residual graph, $v \notin U$. Since there was at least 1 unit of flow from $u$ to $v$ in the flow graph, there must be flow coming out of $U$ in the flow graph. Now, suppose there is flow coming into $U$ from a vertex $n$. Then, there is a path from $u$ to $n$ in the residual graph, so $n \in U$, which is a contradiction. Therefore, there is no flow coming into $U$, but there is flow coming out, which can only occur if $s \in U$. Since $s \in U$, by definition, there is a path from $u$ to $s$ in the residual graph.

- We will show that there exists a path from $t$ to $v$ in the residual graph. Let $V$ denote the set of vertices reachable from $v$ in the flow graph. Since there is no path from $u$ to $v$ in the residual graph, there is no path from $v$ to $u$ in the flow graph, so $u \notin V$. Since there was at least 1 unit of flow from $u$ to $v$ in the flow graph, there must be flow coming into $V$ in the flow graph. Now, suppose there is flow coming out of $V$. Then, there must be a vertex $n$ such that flow enters from $n$ to $V$ in the residual graph. This means there is a path from $v$ to $n$ in the flow graph, so $n \in V$, which is a contradiction. Therefore, there is no flow coming out of $V$, but there is flow coming in, which can only occur if $t \in V$. Since $t \in V$, by definition, there is a path from $v$ to $t$ in the flow graph and $t$ to $v$ in the residual graph.

Since there is a currently 1 extra unit of flow flowing through $e$ that exceeds capacity that cannot be routed along another path, we must reduce 1 unit of flow entering $u$ and 1 unit of flow exiting $v$. For the former, we can add 1 unit of flow along the path from $u$ to $s$ on the residual graph, which reduces flow from $s$ to $u$ on the flow graph by 1 unit. For the latter, we can add 1 unit of flow along the path from $t$ to $v$ on the residual graph, which reduces flow from $v$ to $t$ on the flow graph by 1 unit. Finally, we reduced the flow by at most 1 unit, so we can run 1 iteration of Ford-Fulkerson to ensure we have the proper flow. It will not take more than 1 iteration since each iteration improves our flow by at least 1 unit and we are only a maximum of 1 unit under the maximum flow $F$.

## Runtime

- Computing the residual graph takes $O(|V| + |E|)$ time.
- Determining if edge $e$ was at full capacity takes $O(1)$ time.
- Running BFS to determine whether there is a path from $u$ to $v$ in the residual graph takes $O(|V|+|E|)$ time. Similarly, finding a path from $u$ to $s$ and $t$ to $v$ in the residual graph using BFS takes $O(|V|+|E|)$ time.
- Changing the flow along a path takes $O(|V| + |E|)$ time. We do this a maximum of 2 times (for the case of no residual path from $u$ to $v$ where we change flow along 2 different paths).

Therefore, the overall algorithm takes $O(|V| + |E|)$ time.

# 3

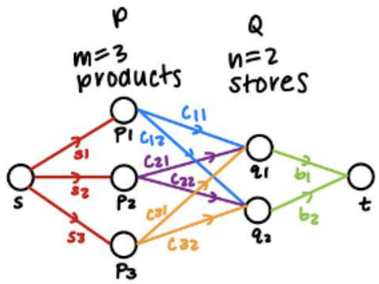
## Algorithm

We can model this problem as a maximum flow problem. Let $P$ be the set of $m$ vertices for each product. Let $Q$ by the set of $n$ vertices for each store. Create a source node $s$ and sink node $t$. Call this graph $G$. Add the following edges:

- Add an edge of weight $s_i$ from $s$ to each $p_i$ where $i \in [m]$.
- Add an edge of weight $b_j$ from each $q_j$ to $t$ where $j \in [n]$.
- Add an edge of weight $c_{ij}$ from each $p_i$ to each $q_j$ where $i \in [m]$ and $j \in [n]$.

Run Ford-Fulkerson to find the maximum flow from $s$ to $t$. To extract a sales plan, let $x_{ij}$ be the flow on the edge from $p_i$ to $q_j$. Then, let the total sales $T = \sum_{i=1}^{m} \sum_{j=1}^{n} x_{ij}$.

## Graph




## Proof

We need to prove 2 things to demonstrate correctness:

- If there is a sales plan with total sales $T$, then there is a flow with value $T$.
- Given an integer-valued flow with value $T$, we can extract a sales plan with total sales $T$.

**Part 1 Proof**: If there is a sales plan with total sales $T$, then there is a flow with value $T$.

If we have a sales plan with total sales $T$, the following 3 conditions must hold:

- For all $i \in [m]$, $\sum_{j=1}^{n} x_{ij} \leq s_i$.
- For all $i \in [m]$ and $j \in [n]$, $x_{ij} \leq c_{ij}$.
- For all $j \in [n]$, $\sum_{i=1}^{m} x_{ij} \leq b_j$.

We can find a flow with value $T$ by doing the following.

- The first condition lets the flow from $s$ to $p_i$ be $\sum_{j=1}^{n} x_{ij}$.
- The second condition lets the flow from $p_i$ to $q_j$ be $x_{ij}$.
- The third conditions lets the flow from $q_j$ to $t$ be $\sum_{i=1}^{m} x_{ij}$.

The flow in this network is $\sum_{j=1}^{n} \sum_{i=1}^{m} x_{ij}$, as this is the total products for all stores combined. By definition, a sales plan with total sales $T$ means the sum of all products across all stores is $T$, and since each edge the flow does not exceed the capacity, we have a flow with value $T$.

**Part 2 Proof**: Given an integer-valued flow with value $T$, we can extract a sales plan with total sales $T$.

If we are given an integer-valued flow with value $T$, the following 3 conditions must hold:

- The flow from $s$ to $p_i$ does not exceed its capacity $s_i$.
- The flow from $p_i$ to $q_j$ does not exceed its capacity $c_{ij}$.
- The flow from $q_j$ to $t$ does not exceed its capacity $b_j$.

We can find a sales plan with total sales $T$ by letting $x_{ij}$ be the flow from $p_i$ to $q_j$. Given a flow, we know the following as well:

- The first condition means that for all $i \in [m]$, $\sum_{j=1}^{n} x_{ij} \leq s_i$.
- The second condition means that for all $i \in [m]$ and $j \in [n]$, $x_{ij} \leq c_{ij}$.

- The third condition means that for all $j \in [n]$, $\sum_{i=1}^{m} x_{ij} \leq b_j$.

The total flow coming into $t$ in the given flow is $T$. We also know that $T = \sum_{j=1}^{n} \sum_{i=1}^{m} x_{ij} = T$ by definition of a sales plan. Therefore, since all conditions 3 conditions for a sales plan are met, there exists a sales plan with total sales $T$.

**Final Argument**: Let $T_{max}$ be the maximum total sales. Then, there exists a sales plan with total sales $T_{max}$. Part 1 means that there is a flow with value $T_{max}$, which implies that the maximum flow has a value of at least $T_{max}$. Second, Ford-Fulkerson finds an integer-valued maximum flow of flow of at least $T_{max}$. Part 2 says the sales plan we take has a value of at least $T_{max}$, but since $T_{max}$ is the maximum possible sales, the value must be $T_{max}$.

### Runtime

The maximum flow of $G$ is $S = \sum_{i=1}^{m} s_i$ since we cannot supply more than the total number of each product there is a supply for. This does not mean there exists a flow with value $S$, only that any flow has value of at most $S$, as it is an upper bound. In $G$, there are $E = m + mn + n$ edges. Creating the graph takes $O(|V| + |E|)$ time or $O((m+n) + (m+mn+n))$, which is $O(mn)$. Furthermore, Ford-Fulkerson takes $O(|F| \cdot |E|)$ or $O(S \cdot (m+mn+n))$ time, which is $O(Smn)$ time. Clearly, the time complexity is dominated by running Ford-Fulkerson, so the overall time complexity is $O(Smn)$, which is polynomial in $S, m, n$.

## 4a

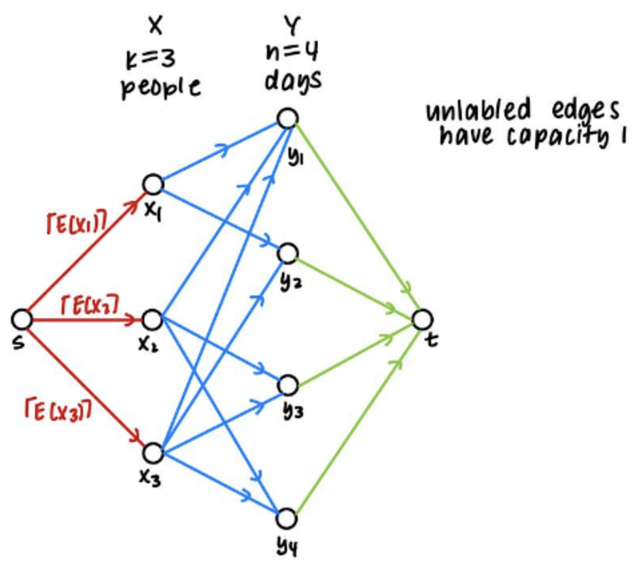
### Algorithm

We can model this problem as a maximum flow problem. Let $X$ be the set of $k$ vertices for each person. Let $Y$ be the set of $n$ vertices for each *day*. Create a source node $s$ and sink node $t$. Call this graph $G$. Add the following edges:

- Add an edge of weight $\lceil E(x_i) \rceil$ from $s$ to each $x_i$ where $i \in [k]$.
- Add an edge of weight 1 from each $y_j$ to $t$ where $j \in [n]$.
- Add an edge of weight 1 from each $x_i$ to $y_j$ where $i \in [k]$ and $j \in [n]$ such that person $i$ goes to work on day $j$.

Run Ford-Fulkerson to find the maximum flow from $s$ to $t$. To extract a driving schedule, we say that person $i$ drives on day $j$ if there exists an edge from $x_i$ to $y_j$ and the flow on this edge is 1.

## Graph




## Proof

We need to prove 3 things to demonstrate correctness:

- The maximum flow is $n$.
- There exists a valid driving schedule.
- Given an integer-valued flow with value $n$, we can extract a valid driving schedule.

**Part 1 Proof**: The maximum flow is $n$.

We can produce a cut of flow $n$ by passing through edges from $y_j$ to $t$, each of which have a capacity of 1. This produces a cut of flow $n$, meaning the maximum flow of $G$ is at most $n$. Therefore, if we find some flow with value $n$, we know that $n$ must be the maximum flow on $G$.

Suppose that for every edge between some $x_i$ and $y_j$, the flow across the edge is $\frac{1}{|S_j|}$. This satisfies the capacity of 1. In other words, the flow from $x_i$ to $y_j$ is what person $i$ would drive on day $j$ if every person carpooling for day $j$ drove an equal amount. We get the following:

- The total flow entering $x_i$ is $\sum_{j \in n: i \in S_j} \frac{1}{|S_j|}$, which is equivalent to $E(x_i)$, or the expected number of times $x_i$ drives. This satisfies the capacity of $\lceil E(x_i) \rceil$.
- The total flow exiting $y_j$ is $(\frac{1}{|S_j|}) \cdot |S_j| = 1$, which satisfies the capacity of 1. This means day $y_j$ has a driver for the entire ride (in this case, drivers carpooling that day drive an equal amount of the ride).

From above, all the capacities are satisfied on the graph, so we have a total flow of $n$ entering $t$, where each day $y_j$ supplies 1 unit of flow. Since we found some flow of value $n$ and we know there cannot be a flow of more than $n$, we know that $n$, or the number of days, is the maximum flow.

**Part 2 Proof**: There exists a valid driving schedule.
Based on Theorem 7.14 (textbook), since we have some maximum flow of $n$ on $G$, there exists some integer-valued maximum flow of $n$ since all capacities are integers. We can find this using Ford-Fulkerson such each

person drives at most $\lceil E(x_i) \rceil$ days and there is a single driver for each day. This produces a valid driving schedule, where each person drives an integer amount of days.

**Part 3 Proof**: Given an integer-valued flow with value $n$, we can extract a valid driving schedule.

If we are given an integer-valued flow with value $n$, the following 3 conditions must hold:

- The flow from $s$ to $x_i$ does not exceed its capacity $\lceil E(x_i) \rceil$.
- The flow from $x_i$ to $y_j$ such that person $x_i$ goes to work on day $y_j$ does not exceed its capacity 1.
- The flow from $y_j$ to $t$ where is exactly 1.

We can find a valid driving schedule by letting person $x_i$ drive on day $y_j$ if the flow from $x_i$ to $y_j$ is 1. Furthermore, since the flow out of $y_j$ is 1, the flow coming into $y_j$ is also 1, so no other person other than person $x_i$ must be driving on day $y_j$. Therefore, given an integer-valued flow with value $n$, we can extract a valid driving schedule. Overall, to extract a driving schedule, we say that person $i$ drives on day $j$ if there exists an edge from $x_i$ to $y_j$ and the flow on this edge is 1.

## Runtime

The maximum flow of $G$ is $n$ since there is 1 driver per day. In $G$, there are $E = k + kn + n$ edges. Creating the graph takes $O(|V| + |E|)$ time or $O((k+n)+(k+kn+n))$, which is $O(kn)$. Furthermore, Ford-Fulkerson takes $O(|F| \cdot |E|)$ or $O(n \cdot (k+kn+n))$ time, which is $O(kn^2)$ time. Clearly, the time complexity is dominated by running Ford-Fulkerson, so the overall time complexity is $O(kn^2)$, which is polynomial in $k$ and $n$.

# 4b

## Algorithm

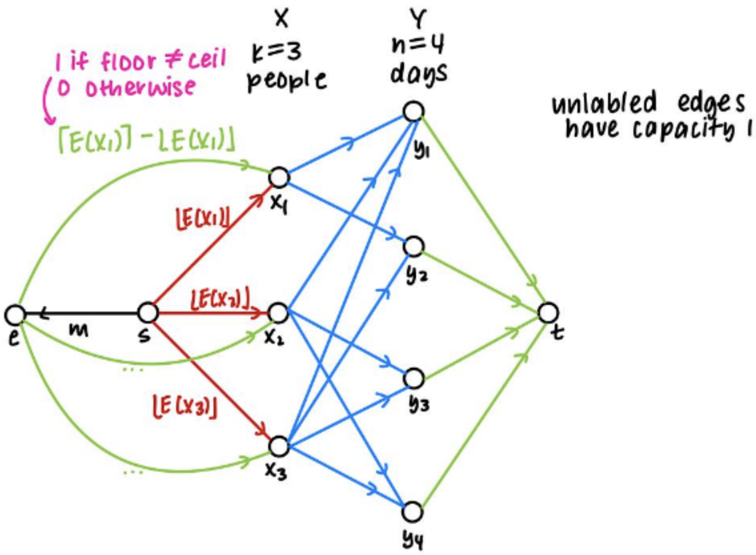
We can modify our initial graph to ensure that we produce a valid driving schedule where each driver drives a minimum of $\lfloor E(x_i) \rfloor$ and maximum of $\lceil E(x_i) \rceil$ days.

Let $m = n - \sum_{i=1}^{k} \lfloor E(x_i) \rfloor$. This represents the total number of "extra drives". In other words, this is the number of days we need a driver for after everyone has driven their minimum requirement.

- Create a vertex $e$.
- Add an edge of weight $m$ from $s$ to $e$.
- Add an edge of weight $\lceil E(x_i) \rceil - \lfloor E(x_i) \rfloor$ from $e$ to each $x_i$ where $i \in [k]$.
- Let $\lfloor E(x_i) \rfloor$ be the new weight on the edge from $s$ to each $x_i$ where $i \in [k]$.

Run Ford-Fulkerson to find the maximum flow from $s$ to $t$. Repeat the above procedure to extract a driving schedule since the graph is identical for all flows leaving $x_i$ and beyond.

## Graph




## Explanation

We need to explain 2 things to demonstrate correctness:

- A flow exists that satisfies the minimum capacities.
- Running Ford-Fulkerson on the modified graph will produce a schedule that satisfies the minimum and maximum capacities.

**Part 1 Explanation**: From above, we showed that there exists a flow of $n$ such that each driver drives for $E(x_i)$ days. We know that $\lfloor E(x_i) \rfloor \leq E(x_i)$, so for the edge from $s$ to each $x_i$, we are either keeping the flow the same or decreasing it, meaning the new flow is at most $n$. Note that here we are not adding flows from $e$ to $x_i$, only using the minimum capacity flows from $s$ to $x_i$. Using minimum capacities produces an integer-valued flow, so the schedule is legitimate so far but may be incomplete. We consider the two cases:

- If the flow is $n$, we are done as satisfying the minimum capacity also satisfies the maximum capacity.
- If the flow is less than $n$, some days do not have drivers, so we need to find a way to pick people to drive these days, which is explain below.

**Part 2 Explanation**: Running Ford-Fulkerson on the modified graph will produce a schedule that satisfies the minimum and maximum capacities.

We can cap the number of extra drives, $m$. These are the number of total drives beyond the sum of minimum capacities, which may be needed to supply a driver for each day. A driver is a candidate for an extra drive when $\lceil E(x_i) \rceil = \lfloor E(x_i) \rfloor + 1$ instead of $\lceil E(x_i) \rceil = \lfloor E(x_i) \rfloor$ (i.e. if their minimum and maximum capacities are the same, they would just drive their minimum capacity). If not, they have the potential to contribute an extra day of driving since it would satistify their maximum capacity.

By creating the edge from $s$ to $e$ with weight $m$, we ensure that we supply only the number of extra drives needed to attain a flow of $n$. More specifically, by drawing a cut through edges from $y_j$ to $t$, we know there is a cut of flow $n$, so the maximum flow cannot be greater than $n$. If we prove there is a flow of value $n$ on this graph, we know $n$ is the maximum flow and that Ford-Fulkerson will find an integer-valued flow of value $n$.

Let us create a fractional flow. We can break this into two cases:

- For each $x_i$ such that $\lceil E(x_i) \rceil = \lfloor E(x_i) \rfloor$, there is no edge with non-zero capacity from $e$ to $x_i$. Thus, we send $E(x_i)$ units of flow from $s$ to $x_i$. This makes the flow leaving $x_i = E(x_i)$, which is valid since $E(x_i)$ here must be an integer.
- For each $x_i$ such that $\lceil E(x_i) \rceil = \lfloor E(x_i) \rfloor + 1$, we send $\lfloor E(x_i) \rfloor$ units of flow from $s$ to $x_i$ and $E(x_i) - \lfloor E(x_i) \rfloor$ units of flow from $e$ to $x_i$. This makes the flow leaving $x_i = \lfloor E(x_i) \rfloor + (E(x_i) - \lfloor E(x_i) \rfloor) = E(x_i)$.

With these cases, we have reduced our flow to the fractional flow of the previous problem. In other words, the flow exiting each $x_i = E(x_i)$. As we showed before, this means the total flow of the graph is $n$ and Ford-Fulkerson will produce an integer-valued flow for $n$ on this graph. Therefore, $n$ is the maximum flow.

As indicated before, Ford-Fulkerson will return an integer-valued maximum flow on this modified graph. However, the maximum possible flow leaving $s = \sum_{i=1}^{k} \lfloor E(x_i) \rfloor + m = \sum_{i=1}^{k} \lfloor E(x_i) \rfloor + (n - \sum_{i=1}^{k} \lfloor E(x_i) \rfloor) = n$. Therefore, the integer-valued maximum flow will have filled capacity on all the edges, including the edges of minimum capacity. This proves that there always exists a solution.

## Example

We can demonstrate this briefly using the example given.

We know that $E(x_1) = \frac{5}{6}$, $E(x_2) = \frac{4}{3}$, and $E(x_3) = \frac{11}{6}$. The minimum and maximum ranges are as follows: $[0, 1]$ for $x_1$, $[1, 2]$ for $x_2$, and $[1, 2]$ for $x_3$.

- $x_1$: route 0 from $s$ to $x_1$ and $\frac{5}{6} - 0 = \frac{5}{6}$ from $e$ to $x_1$
- $x_2$: route 1 from $s$ to $x_2$ and $\frac{4}{3} - 1 = \frac{1}{3}$ from $e$ to $x_2$
- $x_3$: route 1 from $s$ to $x_3$ and $\frac{11}{6} - 1 = \frac{5}{6}$ from $e$ to $x_3$

Therefore, $e$ supplies $\frac{5}{6} + \frac{1}{3} + \frac{5}{6} = 2$ units of flow. In other words, $e$ supplies 2 extra drives, which we get from $m = n - \sum_{i=1}^{k} \lfloor E(x_i) \rfloor = 4 - (0 + 1 + 1) = 2$.