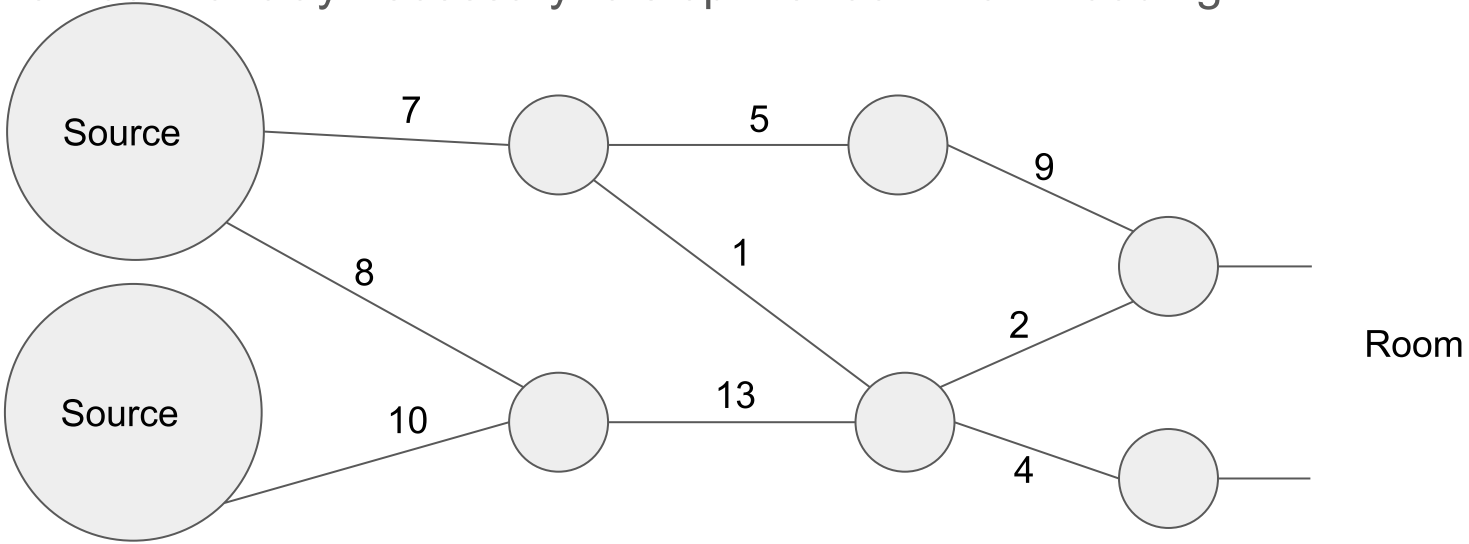
Network Flow Applications

Cutting Off The Water (Min Cut)

We have a series of pipes that are causing a room to flood. The water is flowing through some pipes with one-way valves. We can use some clay to block a pipe. Each pipe will have a specific amount of clay to block. We want to know the least amount of clay necessary to stop the room from flooding.

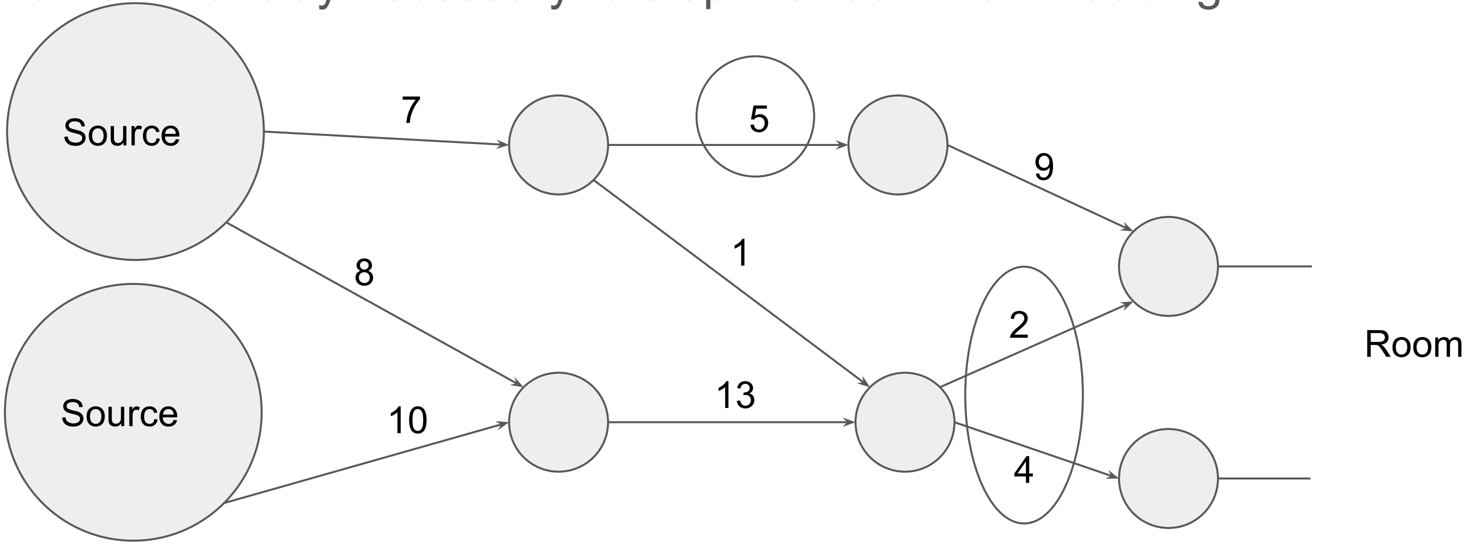
Cutting Off The Water (Min Cut)

We have a series of pipes that are causing a room to flood. The water is flowing through some pipes with one-way valves. We can use some clay to block a pipe. Each pipe will have a specific amount of clay to block. We want to know the least amount of clay necessary to stop the room from flooding.



Cutting Off The Water (Min Cut)

We have a series of pipes that are causing a room to flood. The water is flowing through some pipes with one-way valves. We can use some clay to block a pipe. Each pipe will have a specific amount of clay to block. We want to know the least amount of clay necessary to stop the room from flooding.



Solution

We can use the flow through the graph to find the minimum sum of values to cut off the source from the sink.

Solution

We can use the flow through the graph to find the minimum sum of values to cut off the source from the sink.

It’s as easy as that.

Correctness

A Max Flow produces a Cut. (max flow ≥ min cut)

Correctness

A Max Flow produces a Cut. (max flow ≥ min cut)

Proof. Let S be all the nodes that can be reached using an unsaturated edge from source s. By the FFA algorithm the sink, t, cannot be in S. The saturated edges leaving S form the cut, otherwise.

Correctness

A Max Flow produces a Cut. (max flow ≥ min cut)

Proof. Let S be all the nodes that can be reached using an unsaturated edge from source s. By the FFA algorithm the sink, t, cannot be in S. The saturated edges leaving S form the cut, otherwise.

An argument exists that flow could be coming into S from a node not in S. If flow is coming in from a node not in S, then there should be a residual edge that has capacity going to the node (the reverse edge). Thus the node would be in S.

Correctness

A Max Flow produces a Cut. (max flow ≥ min cut)

Proof. Let S be all the nodes that can be reached using an unsaturated edge from source s. By the FFA algorithm the sink, t, cannot be in S. The saturated edges leaving S form the cut, otherwise.

An argument exists that flow could be coming into S from a node not in S. If flow is coming in from a node not in S, then there should be a residual edge that has capacity going to the node (the reverse edge). Thus the node would be in S.

The sum of the capacities of these saturated edges represent the amount of flow that must be moving from s to t.

Correctness (cont.)

Flow greater than the min cut cannot be sent. (min cut ≥ max flow)

Correctness (cont.)

Flow greater than the min cut cannot be sent. (min cut ≥ max flow)

Proof. Generate the min cut. There will be a portion of the graph, S, reachable by the source s after the cut (note t will not be in S). The sum of the capacity of the edges leaving S will be the min cut.

Correctness (cont.)

Flow greater than the min cut cannot be sent. (min cut ≥ max flow)

Proof. Generate the min cut. There will be a portion of the graph, S, reachable by the source s after the cut (note t will not be in S). The sum of the capacity of the edges leaving S will be the min cut.

When sending flow from the source to the sink at some point the flow needs to be sent along one of these edges. We cannot send more flow than the sum of these edges without overfilling an edge capacity.

Correctness (cont.)

Flow greater than the min cut cannot be sent. (min cut ≥ max flow)

Proof. Generate the min cut. There will be a portion of the graph, S, reachable by the source s after the cut (note t will not be in S). The sum of the capacity of the edges leaving S will be the min cut.

When sending flow from the source to the sink at some point the flow needs to be sent along one of these edges. We cannot send more flow than the sum of these edges without overfilling an edge capacity.

Thus these two values are equal.

Group Problem Project (Bipartite Matching)

A group of students needs to complete some set of tasks to complete their project. Each student want to work on at most one task. Each task will be worked on by at most one task. Some students are incapable of working on certain tasks. Determine the maximum number of tasks the group can complete

Group Problem Project (Bipartite Matching)

A group of students needs to complete some set of tasks to complete their project. Each student want to work on at most one task. Each task will be worked on by at most one task. Some students are incapable of working on certain tasks. Determine the maximum number of tasks the group can complete

For example,

Group Problem Project (Bipartite Matching)

A group of students needs to complete some set of tasks to complete their project. Each student want to work on at most one task. Each task will be worked on by at most one task. Some students are incapable of working on certain tasks. Determine the maximum number of tasks the group can complete

For example,

S_1 can work on T_1, T_2, or T_3

S_2 can work on T_2

S_3 can work on T_2

Group Problem Project (Bipartite Matching)

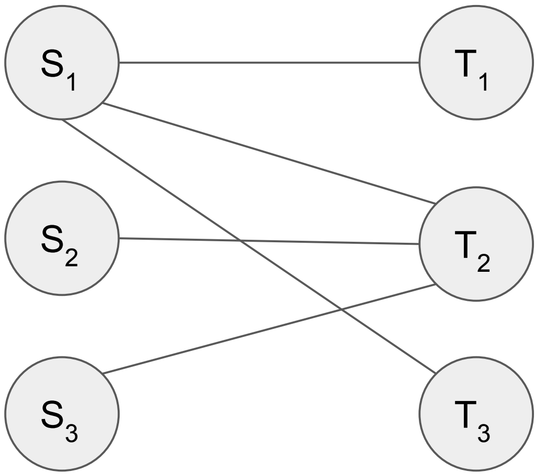
A group of students needs to complete some set of tasks to complete their project. Each student want to work on at most one task. Each task will be worked on by at most one task. Some students are incapable of working on certain tasks. Determine the maximum number of tasks the group can complete

For example,

S_1 can work on T_1, T_2, or T_3

S_2 can work on T_2

S_3 can work on T_2



Bipartite Graph

A graph is bipartite if the graph is 2 colorable, or there is no odd cycle.

Bipartite Graph

A graph is bipartite if the graph is 2 colorable, or there is no odd cycle.

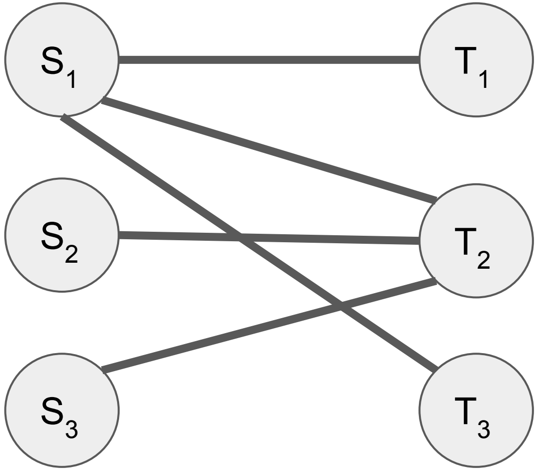
If you can partition the nodes into 2 groups where there is no edge going between 2 nodes within the same group, then the graph is bipartite.

Flow Solution

We want to saturate as many edges as possible.

Flow Solution

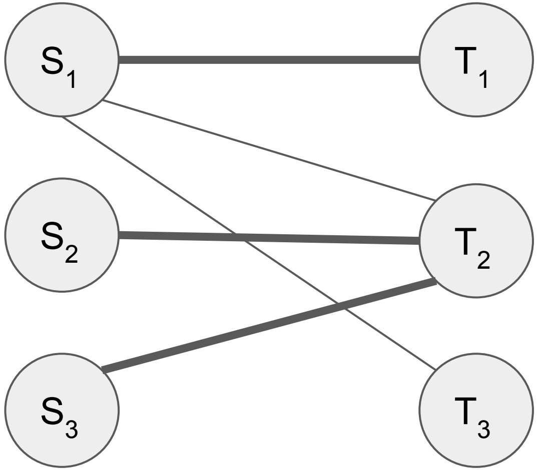
We want to saturate as many edges as possible.



Flow Solution

We want to saturate as many edges as possible.

Each student can only saturate one of their edges

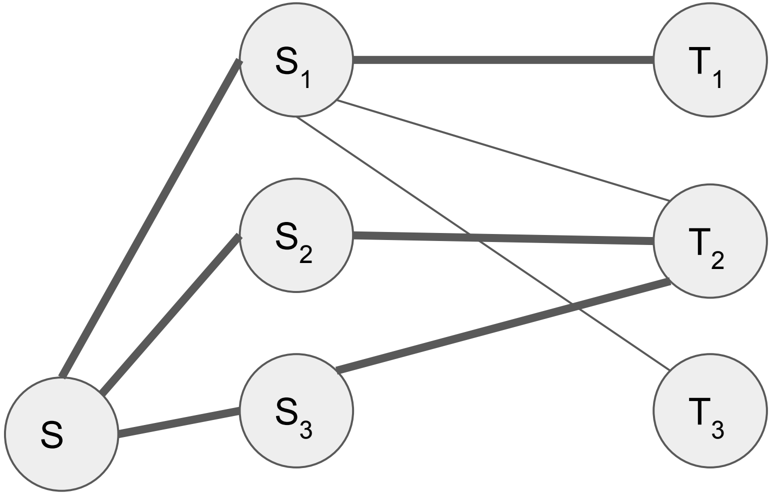


Flow Solution

We want to saturate as many edges as possible.

Each student can only saturate one of their edges

Connect to source with cap of 1



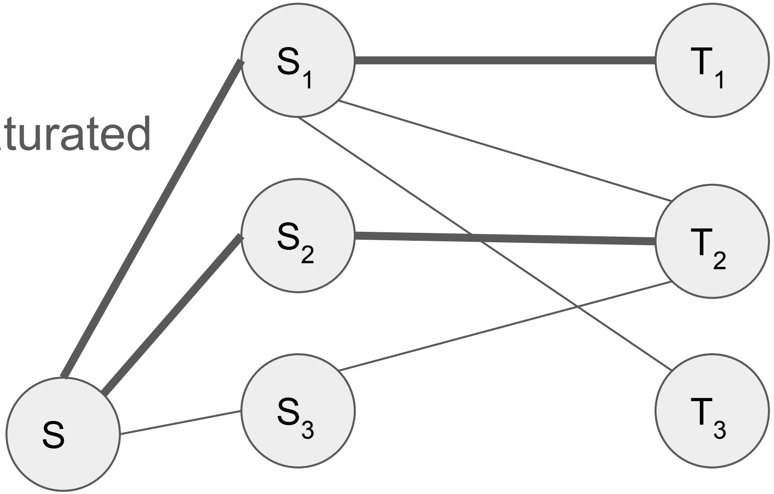
Flow Solution

We want to saturate as many edges as possible.

Each student can only saturate one of their edges

  Connect to source with cap of 1

Each task can only have 1 incoming edge saturated



Flow Solution

We want to saturate as many edges as possible.

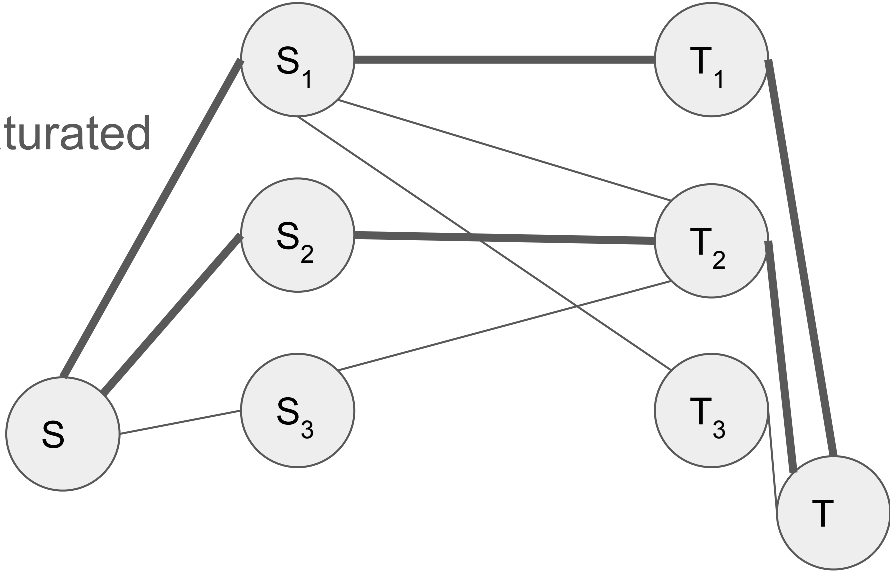
Each student can only saturate one of their edges

Connect to source with cap of 1

Each task can only have 1 incoming edge saturated

Connect to the sink with cap of 1

Make sure all the edges are directed.



Flow Solution

We want to saturate as many edges as possible.

Each student can only saturate one of their edges

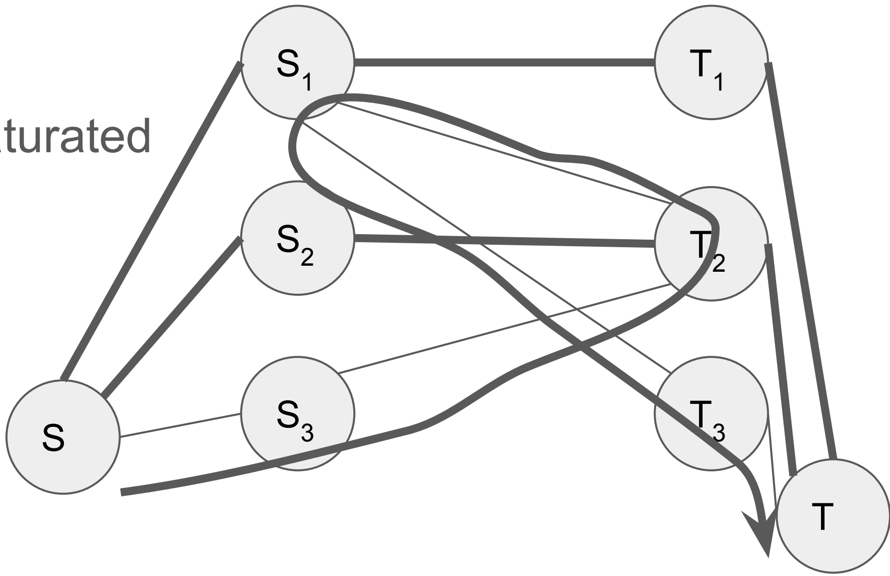
  Connect to source with cap of 1

Each task can only have 1 incoming edge saturated

  Connect to the sink with cap of 1

Make sure all the edges are directed.

  Don’t push flow backwards



Correctness

If we have a solution $[(S_{a1}, T_{b1}), (S_{a2}, T_{b2}), \ldots, (S_{an}, T_{bn})]$,

We can send flow down those edges.

Correctness

If we have a solution [(S_{a1}, T_{b1}), (S_{a2}, T_{b2}), …, (S_{an}, T_{bn})],

We can send flow down those edges.

If we have a max flow,

We choose the student-task pairs based on the saturated edges

Each student will have at most 1 assigned task thanks to the source edge cap

Each task will have at most 1 assigned student thanks to the sink edge cap

Correctness

If we have a solution [(S_{a1}, T_{b1}), (S_{a2}, T_{b2}), …, (S_{an}, T_{bn})],

We can send flow down those edges.

If we have a max flow,

We choose the student-task pairs based on the saturated edges

Each student will have at most 1 assigned task thanks to the source edge cap

Each task will have at most 1 assigned student thanks to the sink edge cap

What if some students could handle working on more than 1 task?

Trigger Warning (Bipartite Vertex Cover)

You will have a series of topics in a discussion. Some audience members might find some of the topics upsetting.

Trigger Warning (Bipartite Vertex Cover)

You will have a series of topics in a discussion. Some audience members might find some of the topics upsetting.

You probably should not do this IRL, but we can with the same amount of effort remove a topic or remove an audience member. What would be the least amount of effort required to not upset any “attending” audience member.

Trigger Warning (Bipartite Vertex Cover)

You will have a series of topics in a discussion. Some audience members might find some of the topics upsetting.

You probably should not do this IRL, but we can with the same amount of effort remove a topic or remove an audience member. What would be the least amount of effort required to not upset any “attending” audience member.

For example,

Trigger Warning (Bipartite Vertex Cover)

You will have a series of topics in a discussion. Some audience members might find some of the topics upsetting.

You probably should not do this IRL, but we can with the same amount of effort remove a topic or remove an audience member. What would be the least amount of effort required to not upset any “attending” audience member.

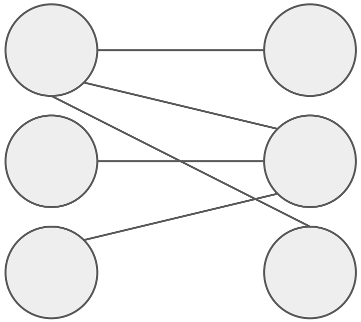
For example,
S_1 hates T_1, T_2, and T_3
S_2 hates T_2
S_3 hates T_2

Trigger Warning (Bipartite Vertex Cover)

You will have a series of topics in a discussion. Some audience members might find some of the topics upsetting.

You probably should not do this IRL, but we can with the same amount of effort remove a topic or remove an audience member. What would be the least amount of effort required to not upset any “attending” audience member.

For example,
S_1 hates T_1, T_2, and T_3
S_2 hates T_2
S_3 hates T_2

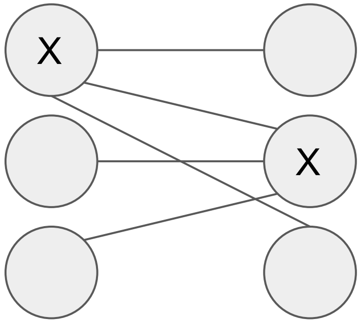


Trigger Warning (Bipartite Vertex Cover)

You will have a series of topics in a discussion. Some audience members might find some of the topics upsetting.

You probably should not do this IRL, but we can with the same amount of effort remove a topic or remove an audience member. What would be the least amount of effort required to not upset any “attending” audience member.

For example,
S_1 hates T_1, T_2, and T_3
S_2 hates T_2
S_3 hates T_2



Solution

Run flow on the same graph as the bipartite matching graph.

Solution

Run flow on the same graph as the bipartite matching graph.

Why is this correct?

Solution

Run flow on the same graph as the bipartite matching graph.

Why is this correct?

Each “match” contains 2 nodes.

Solution

Run flow on the same graph as the bipartite matching graph.

Why is this correct?

Each “match” contains 2 nodes.

1 of these nodes should be removed.

Cut the least amount of nodes such that no matches remain.

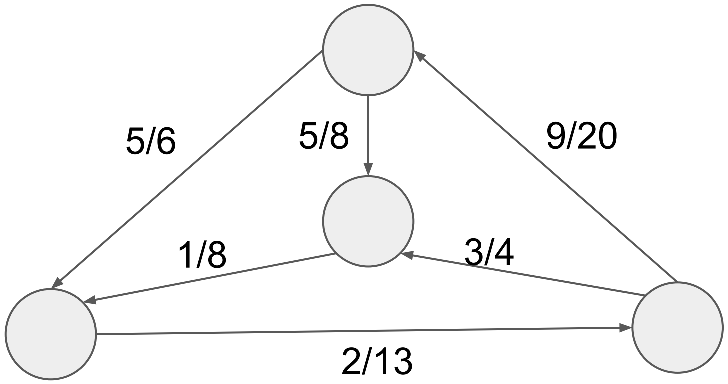
Non-Trivial Flow Applications

Resource Trading (Circulation with U/L Bounds)

Suppose we have some countries. Each country has some rules on which countries they can trade with (e.g. **A** can send at most 10 resources to **B**). Each country wants to ensure that the amount of resources they export is equal to the resources they import. Additionally, some countries NEED to trade with other countries (e.g. **C** must send at least 3 resources to **D**). Determine if it possible for all the trade requirements to be met.

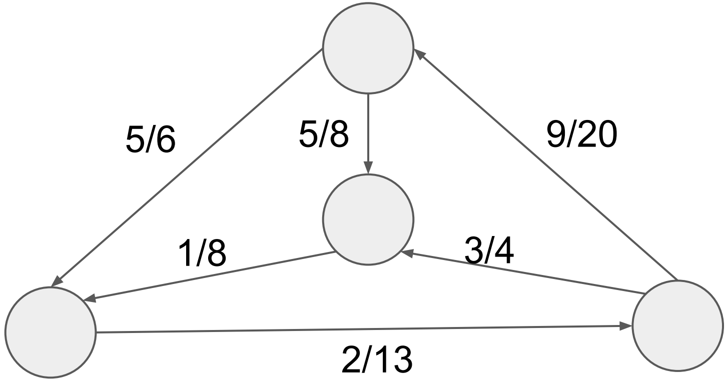
Resource Trading (Circulation with U/L Bounds)

Suppose we have some countries. Each country has some rules on which countries they can trade with (e.g. **A** can send at most 10 resources to **B**). Each country wants to ensure that the amount of resources they export is equal to the resources they import. Additionally, some countries NEED to trade with other countries (e.g. **C** must send at least 3 resources to **D**). Determine if it possible for all the trade requirements to be met.



Solution

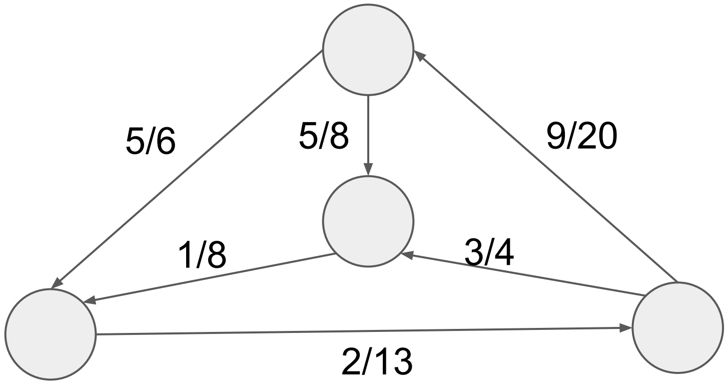
We must utilize some of the edges with a specified minimum.



Solution

We must utilize some of the edges with a specified minimum.

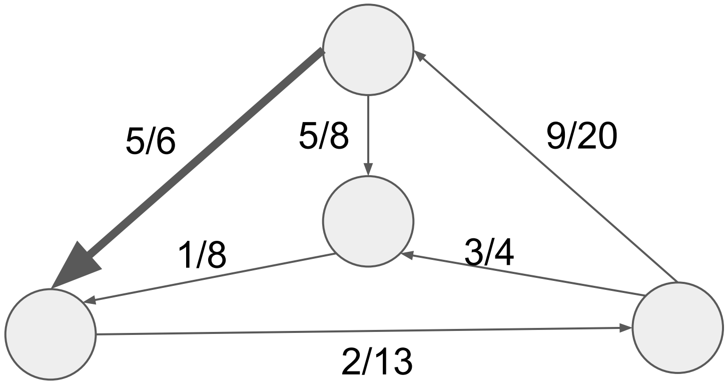
To ensure we use an edge we can connect corresponding nodes to source and sinks.



Solution

We must utilize some of the edges with a specified minimum.

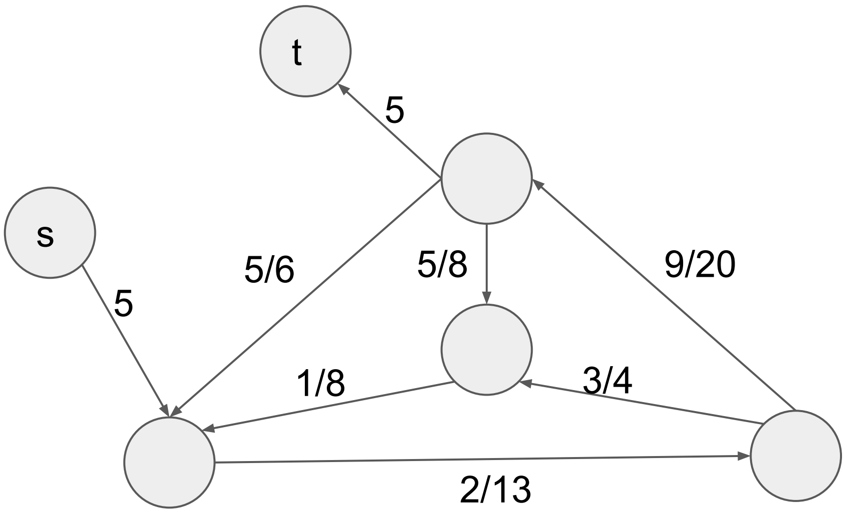
To ensure we use an edge we can connect corresponding nodes to source and sinks.



Solution

We must utilize some of the edges with a specified minimum.

To ensure we use an edge we can connect corresponding nodes to source and sinks.

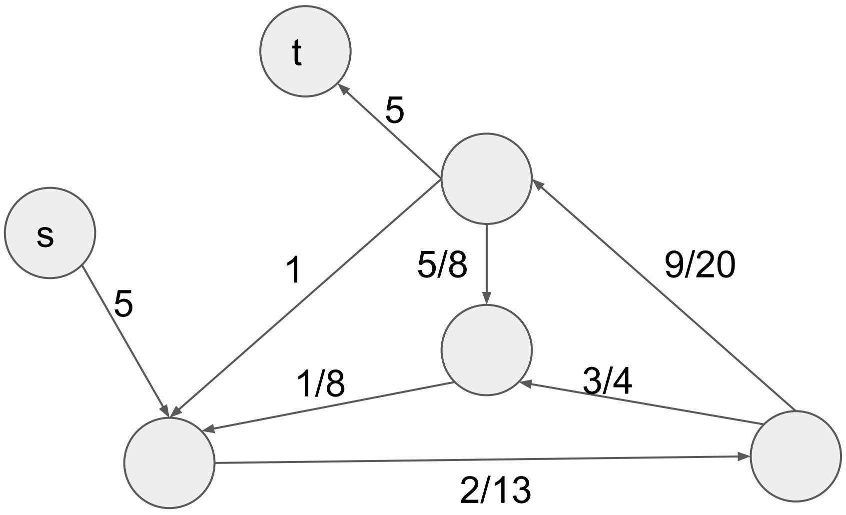


We must send 5 units.

Solution

We must utilize some of the edges with a specified minimum.

To ensure we use an edge we can connect corresponding nodes to source and sinks.

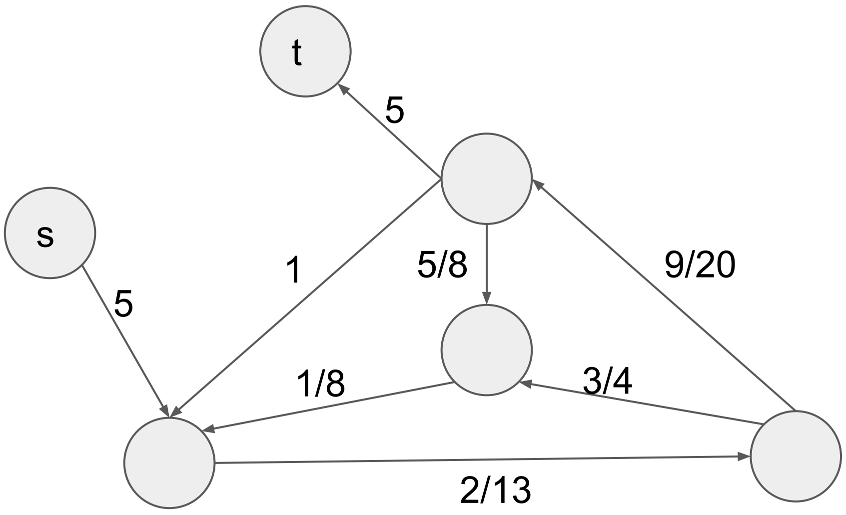


We must send 5 units.
We have 1 unit left we can send.

Solution

We must utilize some of the edges with a specified minimum.

To ensure we use an edge we can connect corresponding nodes to source and sinks.

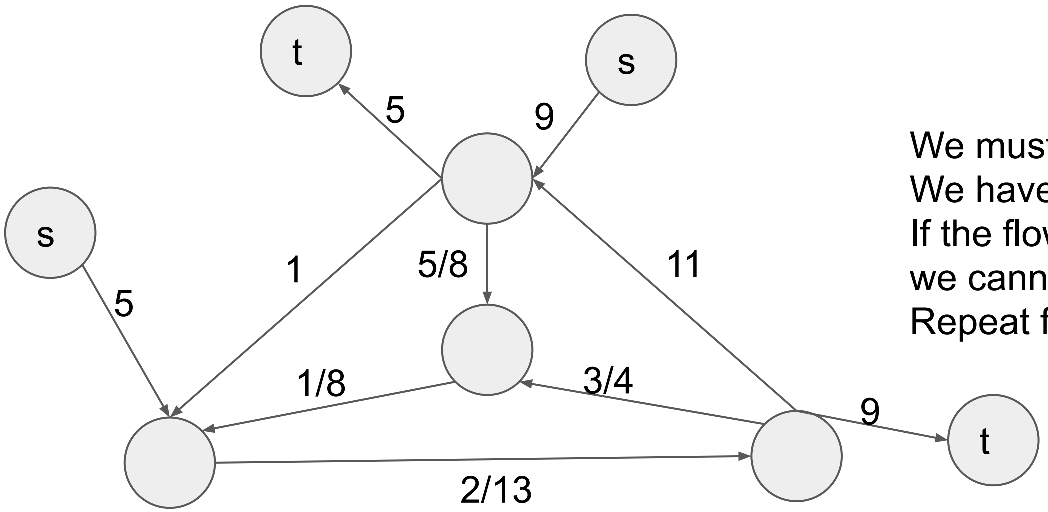


We must send 5 units.
We have 1 unit left we can send.
If the flow cannot saturate these edges,
we cannot met the requirements.

Solution

We must utilize some of the edges with a specified minimum.

To ensure we use an edge we can connect corresponding nodes to source and sinks.

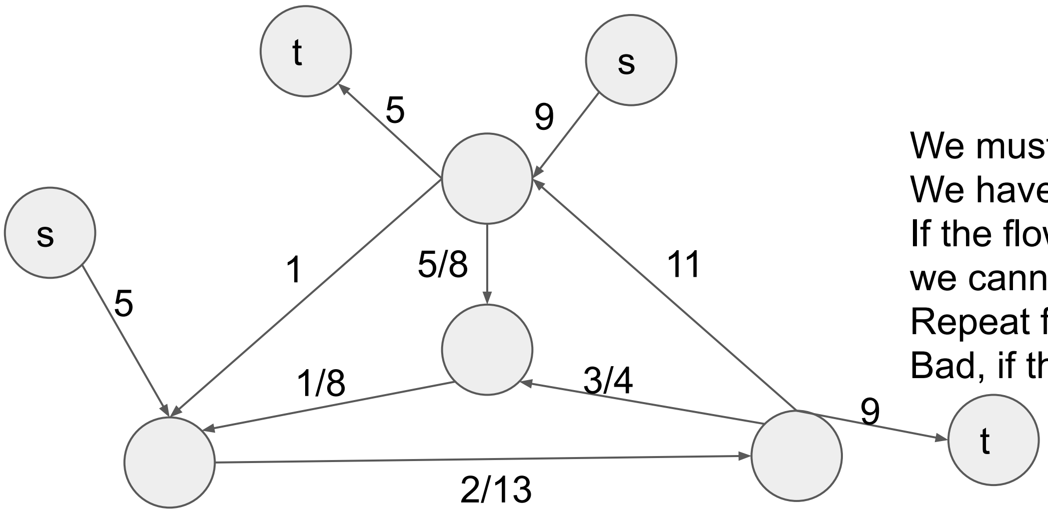


We must send 5 units.
We have 1 unit left we can send.
If the flow cannot saturate these edges, we cannot met the requirements.
Repeat for all edges.

Solution

We must utilize some of the edges with a specified minimum.

To ensure we use an edge we can connect corresponding nodes to source and sinks.



We must send 5 units.
We have 1 unit left we can send.
If the flow cannot saturate these edges,
we cannot met the requirements.
Repeat for all edges.
Bad, if the flow is not the sum of the mins.

Profit Maximization (Project Selection)

We have some projects we can work on. Each project will reward us with some money. Each project has some required machines to work on. Each machine has some 1 time cost. Some projects will share machines. We want to find the most money we can make with optimum purchasing.

Profit Maximization (Project Selection)

We have some projects we can work on. Each project will reward us with some money. Each project has some required machines to work on. Each machine has some 1 time cost. Some projects will share machines. We want to find the most money we can make with optimum purchasing.

What are the nodes?

Profit Maximization (Project Selection)

We have some projects we can work on. Each project will reward us with some money. Each project has some required machines to work on. Each machine has some 1 time cost. Some projects will share machines. We want to find the most money we can make with optimum purchasing.

What are the nodes?

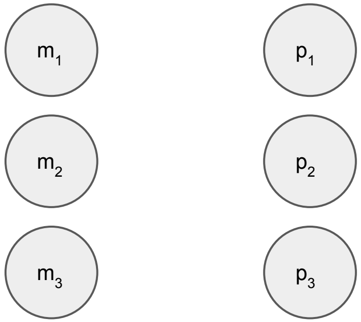
Machines AND projects

Profit Maximization (Project Selection)

We have some projects we can work on. Each project will reward us with some money. Each project has some required machines to work on. Each machine has some 1 time cost. Some projects will share machines. We want to find the most money we can make with optimum purchasing.

What are the nodes?

Machines AND projects



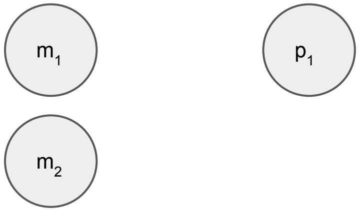
Profit Maximization (Project Selection)

We have some projects we can work on. Each project will reward us with some money. Each project has some required machines to work on. Each machine has some 1 time cost. Some projects will share machines. We want to find the most money we can make with optimum purchasing.

What are the nodes?

Machines AND projects

Suppose we have 1 project and 2 machines



Profit Max Flow Graph

Taxi Problem (Minimum Hiring Job Coverage)

Suppose we have a list of jobs a taxi company must fulfill. The taxi will start a some location at a specified time and will reach a destination at a specified time. The problem is that these jobs might need different taxis due to the time constraints. We know the time it takes to travel between the end point of one taxi job and the start point of another taxi job. What is the least number of taxis needed?

Taxi Problem Flow Graph

Image Segmentation

We have some pixels that have some probability of belonging to a foreground, and some probability of belonging to the background. The foreground is usually a clump of pixels. Pixels that are close to each other have a probability of belonging to the same group.

Image Segmentation

We have some pixels that have some probability of belonging to a foreground, and some probability of belonging to the background. The foreground is usually a clump of pixels. Pixels that are close to each other have a probability of belonging to the same group.

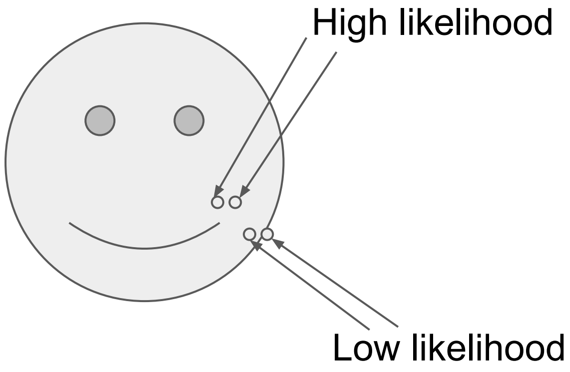


Image Segmentation

We have some pixels that have some probability of belonging to a foreground, and some probability of belonging to the background. The foreground is usually a clump of pixels. Pixels that are close to each other have a probability of belonging to the same group.

Maximize the probability product

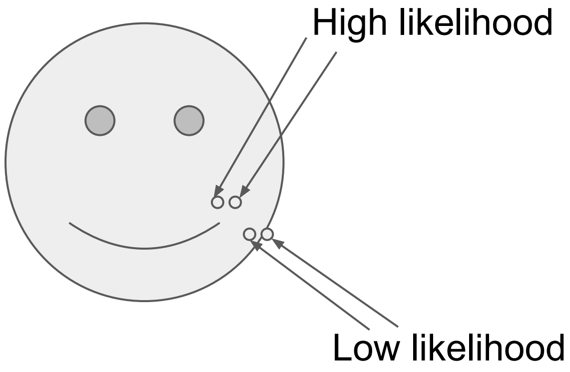


Image Segmentation

We have some pixels that have some probability of belonging to a foreground, and some probability of belonging to the background. The foreground is usually a clump of pixels. Pixels that are close to each other have a probability of belonging to the same group.

Maximize the probability product

Using the concepts of logs, the problem can become maximize a sum of values.

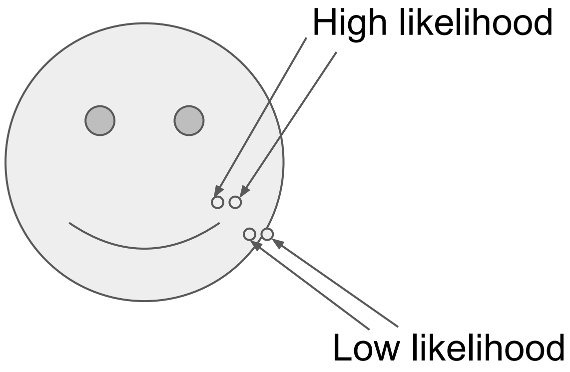


Image Segmentation Flow Graph

Baseball “Elimination”

Give a game schedule and the results of a few games. Determine which teams have the potential to finishing a season with the most number of wins.

Baseball Elimination Flow Graph