Lecture 7: The max-flow min-cut and marriage theorems · 1MA020

Vilhelm Agdur[1]

[1] vilhelm.agdur@math.uu.se

14 November 2023

Continuing our exploration of things that can be done with weighted graphs, we move on to considering *flows* in weighted directed graphs, which model things like traffic flows. We see the connection between these and edge cuts of graphs, and prove the Ford-Fulkerson theorem.

Finally, we use our result about flows to prove the Hall marriage theorem.

In our exercises, we saw this figure:

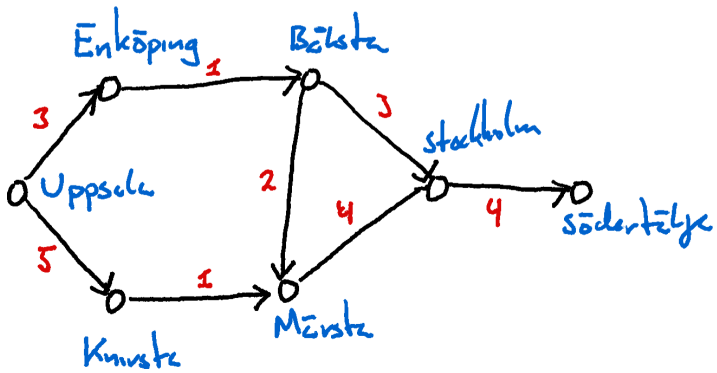

Figure 1: A hypothetical graph modelling public transport flow from Uppsala to Södertälje.

If we interpret the edge-weights as how many thousands of passengers can travel the route per hour, it makes sense to ask the question of how many can travel from Uppsala to Södertälje per hour. Intuitively, it is clear that the answer must be two thousand, because the traffic is bottlenecked by the Enköping-Bålsta and Knivsta-Märsta connections – the fact that five thousand per hour can get from Uppsala to Knivsta doesn't help at all, and upgrading the Bålsta-Stockholm route wouldn't improve things.

Let us now turn these intuitive considerations into actual rigorous mathematics. First, let us define the graphs we are working on:

Definition 1. A *weighted directed graph* G consists of a directed graph (V, E) and a weight function $w : E \to \mathbb{R}$. For it to be a *flow network* we additionally require that all weights be positive, that there be a distinguished *source* vertex s and *sink* vertex t, and that whenever $u \to v$ is an edge, $v \to u$ is not also an edge.[2]

We define a *capacity function* $c : V \times V \to [0, \infty)$ by that $c(v, v') = w(v \to v')$ whenever $v \to v'$ is an edge of G, and $c(v, v') = 0$ otherwise.

[2] Recall that in general, it is allowed in a simple directed graph to have both the edge $u \to v$ and the edge $v \to u$ – what is not allowed is multiples of the same edge, or loops. In this case, the restriction of not having such back-and-forth edges is not a genuine restriction, however: If we have such a pair of edges, we can get an equivalent flow network by introducing a "dummy vertex" in the middle of one of the edges, as in Figure 2.

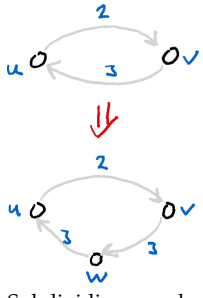

Figure 2: Subdividing an edge using a dummy vertex, to get around the

These flow graphs define the capacity of the network to handle traffic. Next, let us define the actual *flows*, which are the possible actual traffic situations. There are two natural constraints these should satisfy:

1. The flow through an edge can't be greater than the actual capacity of the edge, so we aren't putting more cars on the road than will actually fit.
2. Other than the source and the sink nodes, where we imagine vehicles are entering and exiting the graph, the flow into a vertex must equal the flow out of it. Trains do not magically vanish, nor do they appear out of nowhere or teleport.

Definition 2. A *flow* on the flow network G with capacity function c is a function $f : V \times V \to [0, \infty)$ which satisfies

1. the *capacity constraint* that
$$f(v, v') \leq c(v, v') \qquad \forall v, v' \in V,$$
2. and the *conservation constraint*
$$\sum_{w \in V} f(w, v) = \sum_{u \in V} f(v, u)$$
whenever $v \in V \setminus \{s, t\}$.

The *value* of a flow, denoted by $|f|$, is the net out-flow at the source, that is
$$|f| = \sum_{v \in V} f(s, v) - \sum_{w \in V} f(w, s).$$

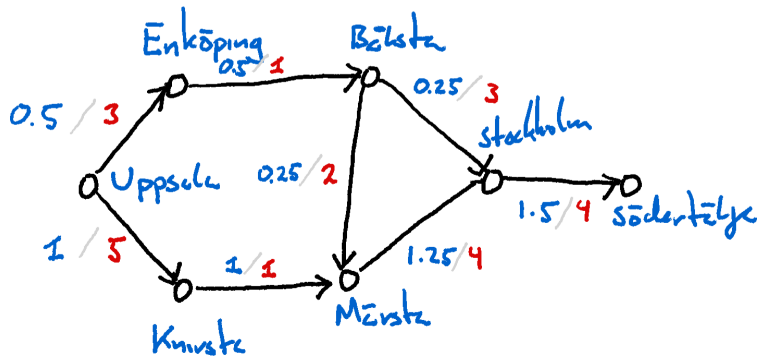

Figure 3: An example of a flow in the graph from Figure 1. The weights of the edges are given in red, and the flows through them in blue. This flow has value 1.5, which we can see both by computing the net out-flow from Uppsala and the net flow into Södertälje.

Remark 3. It follows from the conservation constraint that $|f|$ is also equal to the net in-flow at the sink.[3] Therefore, we can always assume that $|f| \geq 0$, since if it were not, we could just swap the roles of s and t.

[3] *Exercise* 1. Prove this.

s-t-cuts

Having defined what we mean by a flow, let us next formalize the intuitive notion of a bottleneck in the graph.

Definition 4. Let $G = (V, E, w, s, t)$ be a flow network with source s and sink t. An *s-t-cut* is a partitioning of V into two sets S, T such that $s \in S$ and $t \in T$. The *capacity* of the cut is

$$c(S,T) = \sum_{(v,v') \in S \times T} c(v,v') = \sum_{e \in E(S,T)} w(e).$$

Considering our intuition about the bottlenecks, it should be clear that any flow from s to t has to at some point pass from S into T, and so use some of the capacity of the cut. So the total flow cannot be greater than the capacity of the cut, that is, $|f| \leq c(S,T)$ for any flow f and any s-t-cut S, T.

In fact, it turns out that equality is only achieved in this inequality in the most extreme case.

Lemma 5. *Let G be a flow network with a flow f and an s-t-cut of V into S and T. If $|f| = c(S,T)$, then $|f|$ is maximal among all flows, and $c(S,T)$ is minimal among all s-t-cuts.*

Proof. As we saw, for any other flow f', we must have

$$|f'| \leq c(S,T) = |f|,$$

and so f is maximal. Similarly, for any other s-t-cut S', T' we have

$$c(S,T) = |f| \leq c(S',T')$$

and so S, T is minimal. □

Residual networks

A central construction in the theory of flow networks is the *residual network*, which as the name suggests encodes how much capacity is left over by a flow.

Definition 6. Let G be a flow network with capacity function c, and let f be a flow on G. The *residual capacity* c_f is a function from $V \times V$ into $[0, \infty)$ defined by[4]

$$c_f(u,v) = \begin{cases} c(u,v) - f(u,v) & \text{if } (u,v) \text{ is an edge in } G \\ f(v,u) & \text{if } (v,u) \text{ is an edge in } G \\ 0 & \text{otherwise.} \end{cases}$$

[4] Notice how we use the assumption that there are no back-and-forth edges in G here – otherwise the definition would not make sense.

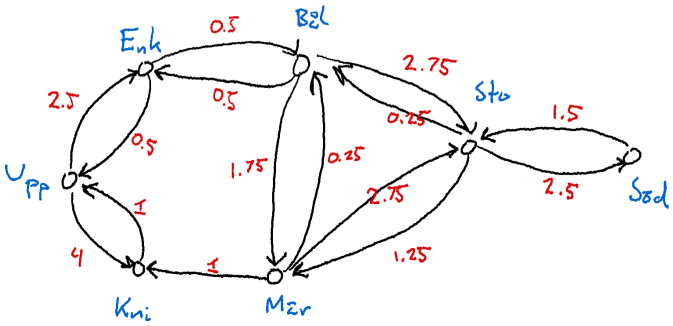

Figure 4: The residual network of the flow in Figure 3.

The *residual network* G_f is the weighted directed graph which has an edge $u \to v$ whenever $c_f(u,v) > 0$, and this edge has weight $c_f(u,v)$ if so.

We give an example of this in Figure 4, using the same flow network we have seen before.

How can we use these residual networks to find ways to improve a flow? The idea is that a walk from the source to the sink in the residual graph will correspond to a way of improving the flow. However, so far we have only defined walks in undirected graphs, so let us define what we mean here.

Definition 7. In any directed graph, a *directed path* consists of a sequence of vertices $v_0\, v_1\, \ldots\, v_k$ and a sequence of edges $e_1\, e_2\, \ldots\, e_k$, such that edge e_i points from e_{i-1} to e_i. We also call such a path a v_0-v_k-path.

If G is a residual network, a directed path P from s to t is called an *augmenting path*, and it has *residual capacity*

$$c_f(P) = \min_{e \in E(P)} c_f(e).$$

That these augmenting paths correspond precisely to ways to improve the flow is the content of our next lemma.

Lemma 8. *Let f be a flow in a flow network G such that G_f admits an augmenting path P. Then the flow f' defined by*

$$f'(u,v) = \begin{cases} f(u,v) + c_f(P) & \text{if } (u,v) \text{ is an edge in } P \\ f(u,v) - c_f(P) & \text{if } (v,u) \text{ is an edge in } P \\ f(u,v) & \text{otherwise} \end{cases}$$

satisfies $|f'| > |f|$.

Exercise 2. This proof is mostly just a slightly tedious unpacking of the definitions, which is not very illuminating to do during a lecture, but useful to do yourself for learning the definitions. Therefore, we leave it as an exercise. Remember that there are three things you need to check:

1. The capacity constraint on f',
2. the conservation constraint on f',
3. and that $|f'| > |f|$. (Note that this is a *strict* inequality!)

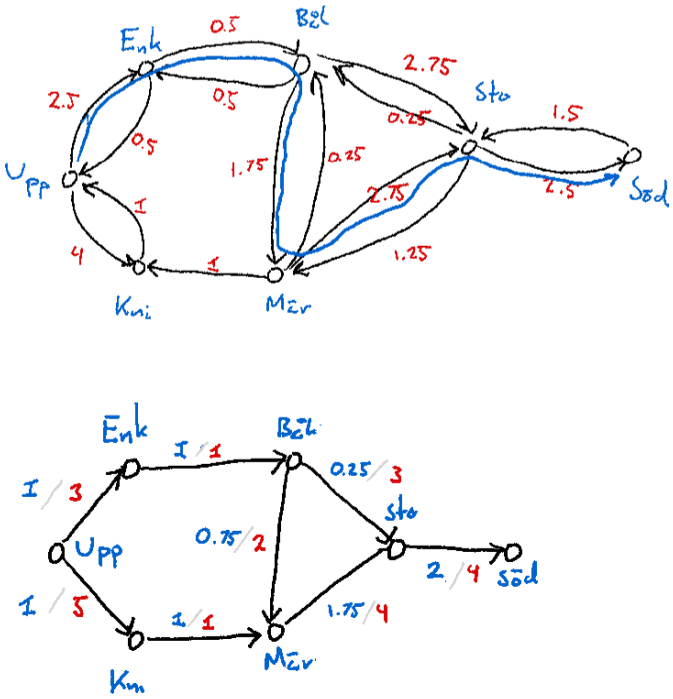

Figure 5: An augmenting path in the residual network of Figure 4, and the new flow f' gotten from it.

We see one example of improving a flow using an augmenting path in Figure 5.

Having now finally set up all the terminology and lemmata we need, we can finally state and prove the Ford-Fulkerson theorem.

Theorem 9 (Ford-Fulkerson)**.** *Let f be a flow on the flow network G. Then, the following are equivalent:*

1. *f is a maximal flow,*
2. *G_f contains no augmenting path, and*
3. *there is an s-t-cut S, T with $|f| = c(S, T)$.*

Proof. That 3. implies 1. is precisely the content of Lemma 5. To see that 1. implies 2., consider what the contrapositive of this implication is – it is precisely that the existence of an augmenting path implies there is a higher-value flow, which is exactly Lemma 8.

Therefore, the thing we need to show is that 2. implies 3. – so assume G_f does not contain an augmenting path,and let S be the set of vertices that can be reached from s by a directed path. Let

$T = V \setminus S$. We want to show that these S and T are the desired s-t-cut with capacity equal to $|f|$.

By assumption $t \in T$, since otherwise there would be an augmenting path. We also see that every arrow $u \to v$ from S to T in G must have its full capacity used by f, or there would be an arrow in G_f from u to v corresponding to the unused capacity. Likewise, every arrow $u \to v$ from T to S must have a flow of zero, or there would be an arrow in the opposite direction in G_f. Therefore we must have $c(S, T) = |f|$, as desired. □

Remark 10. Notice how this theorem does not actually say anything about the *existence* of a maximal flow. For finite networks, however, we can do a bit of analysis-style reasoning to prove that a maximum flow must exist. The details of this are left as an exercise.

The Ford-Fulkerson theorem very nearly gives us an algorithm, by just repeatedly finding augmenting paths and augmenting along them. It is still a bit unspecified,[5] however – specifying the details gives you for example the Edmonds-Karp algorithm.

[5] If you find your augmenting paths in a sufficiently poor way, and you have irrational values for some capacities, the algorithm might not terminate at all!

Definition 11 (Edmonds-Karp Algorithm)**.** Given a flow network G, let f be the zero-flow. Then, in each time step:

1. Construct G_f.
2. Find a shortest augmenting path in G_f using breadth-first search – if none exists, we are done and return f.
3. If we found such a path, augment f using it.

Analysing this algorithm shows that it will have a running time of $O\left(|V|^2|E|\right)$.

Remark 12. If you look through our proofs and constructions,[6] always assuming all capacities are actually integers, you will see that the proofs actually give the existence of an *integer* maximal flow, that is, a maximal flow where $f(e) \in \mathbb{Z}$ for every edge e.

[6]

Exercise 3. Actually do this, reading through the notes with an eye to keeping everything an integer.

The Hall marriage theorem

The final thing we are going to do in this lecture is to apply what we have just learned about flows to prove the Hall marriage theorem. We already saw in the exercise session that a *matching* in a graph $G = (V, E)$ is a subset M of the edges, such that no two edges in M share an endpoint. A matching in which every vertex is incident to some edge in the matching is called *perfect*.

The Hall marriage theorem concerns itself particularly with finding perfect matchings in *bipartite* graphs, so let us define what those are.

Definition 13. A graph $G = (V, E)$ is called *bipartite* if there is a partition of V into two sets A and B, such that $E \subseteq A \times B$. That is, all edges go between A and B, not within the two parts.

For a bipartite graph, it makes sense to relax the notion of a perfect matching to a *matching of A into B*, which is just a matching such that every vertex in A is incident to an edge in the matching.

In order for this to be possible, it must clearly be the case that for every subset $Q \subseteq A$, it has at least $|Q|$ neighbours in B, or we would trivially be unable to match Q into B. So, for any set of vertices Q, let us denote its set of neighbours by $N(Q)$. What the Hall marriage theorem says is that our necessary condition is in fact also sufficient.

Theorem 14 (Hall's marriage theorem)**.** *Let $G = (V, E)$ be a finite bipartite graph with $V = A \coprod B$. Then G contains a matching of A into B if and only if $|N(Q)| \geq Q$ for all $Q \subseteq A$.*

Proof. We have already seen the necessity of the condition, so what remains is to show that it is sufficient. So assume that $|N(Q)| \geq |Q|$ for all $Q \subseteq A$.

We create a flow network as follows: We introduce a new vertex s which is a neighbour to every vertex in A, and a new vertex t which is a neighbour to every vertex in B. We direct the edges so that they all point away from s towards t, and give the edges incident to s or t capacity 1, and the edges between A and B capacity ∞.[7]

[7] This is just for convenience – we could do it with a sufficiently large integer depending on G here instead, but the argument seems clearer if we don't have to worry about these capacities.

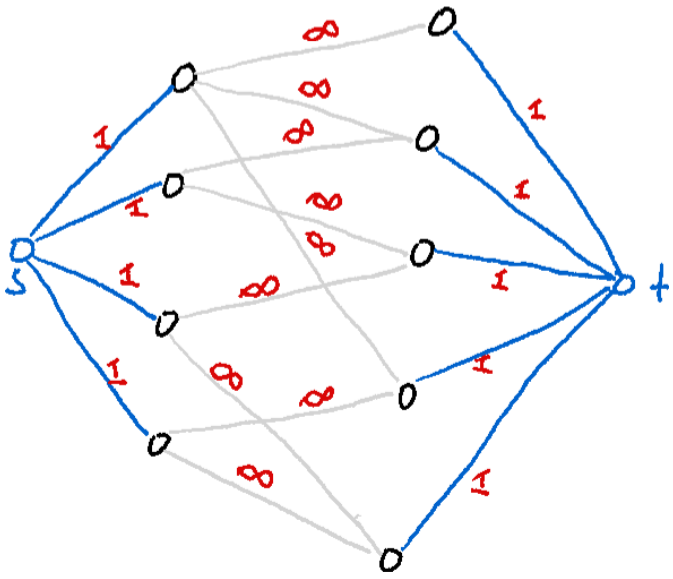

Figure 6: A flow network gotten from a bipartite graph as in the proof of the marriage theorem.

Now consider the easy s-t-cut where $S' = \{s\}$ and $T' = A \cup B \cup \{t\}$. It is easy to see that this cut has capacity $c(S', T') = |A|$, and hence $c(S, T) \leq |A|$ for any minimum cut (S, T).

In particular, this means that for any minimum s-t-cut S, T there cannot be an edge $u \to v$ with $u \in S \cap A$ and $v \in T \cap B$, since if so the capacity of the flow would be infinite. So any neighbour of $S \cap A$ must lie in $S \cap B$ – it must be in B by the bipartition, and must still be in S for the flow to still have finite capacity. Or, in other words, $N(S \cap A) \subseteq S \cap B$.

So, using this, let us compute a lower bound on $c(S, T)$:[8]

$$\begin{aligned} c(S,T) &= \sum_{u \in S, v \in T} c(u,v) \\ &= \sum_{v \in T \cap A} c(s,v) + \sum_{u \in S \cap B} c(u,t) \\ &= |T \cap A| + |S \cap B| \\ &\geq (|A| - |S \cap A|) + |N(S \cap A)| \\ &\geq |A| - |S \cap A| + |S \cap A| = |A|. \end{aligned}$$

Since we've already seen that a minimum cut has capacity at most $|A|$, this proves that minimum cuts in fact have a capacity of exactly $|A|$.

Using our remark about integer flows, we see that this implies that there exists an integer flow with value $|A|$. So this flow must fill every outgoing edge from s, and so by the conservation property each vertex in A must have outgoing flow 1. Since the flow is integral, all this flow must use a single edge.

Looking at the other endpoint of this edge in B, we see that this in fact can't have in-flow from any other vertex, since it can only have a maximum out-flow of 1 to t. So therefore the set of edges e from A to B such that $f(e) = 1$ form a matching of A into B. □

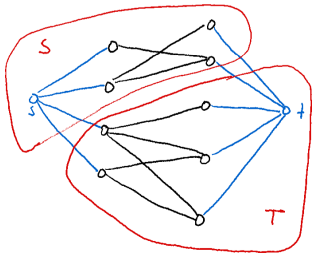

Figure 7: An example of a non-trivial minimal s-t-cut in a flow network gotten from a bipartite graph.

[8] The first equality is just the definition of the capacity of a cut.

The second equality uses our observation that there can't be any edges from $S \cap A$ into $T \cap B$ – so the only edges that can contribute to the capacity of the cut are edges from s to things in $T \cap A$, or edges from things in $S \cap B$ to t. It may be helpful to consider Figure 7 to understand what is going on here.

The third equality just uses that all the terms of the sums are 1.

For the first inequality, we in fact have $|T \cap A| = (|A| - |S \cap A|)$. The inequality part is that $|N(S \cap A)| \leq \cap S \cap B$, which is just our observation that $N(S \cap A) \subseteq S \cap B$.

The final inequality, that $|N(S \cap A)| \geq |S \cap A|$, is where we use the assumption of the theorem.

Exercises

Exercise 4. Prove that for a finite flow network $G = (V, E, w)$ there always exists a maximal flow, because the set of possible flows can be seen as a compact subset of $\mathbb{R}^{|E|}$, and the function sending f to $|f|$ is continuous.