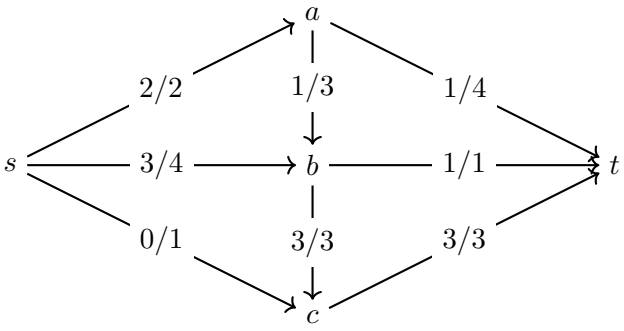
Math 482: Linear Programming[1] Mikhail Lavrov

Lecture 26: The Ford–Fulkerson Algorithm

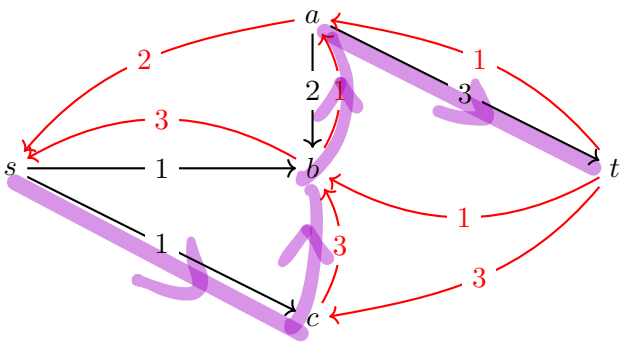
April 6, 2020 *University of Illinois at Urbana-Champaign*

1 Residual graphs, augmenting paths, and minimum cuts

Consider the following network with a feasible flow:



We want to find an augmenting path, so we construct a residual graph which places arcs along every possible path that an augmenting path would take. Each arc in the residual graph is labeled with its residual capacity: the maximum amount by which flow can be changed in that direction.



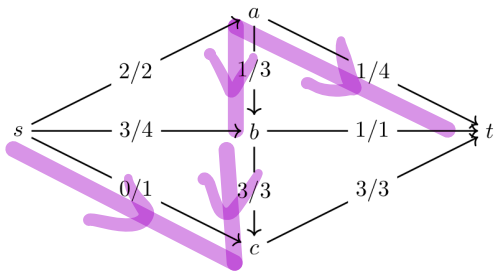
Exploring, we find that from s, we can get to b or c. From c, we can only go to b again, which as already possible; from b, we can also go to a. From a, we can return to s or get to t. We've found a path from s to t. Two, actually: we can go $s \to b \to a \to t$ or $s \to c \to b \to a \to t$ in the residual graph.

$S \to C \leftarrow b \leftarrow a \to t$

Let's choose the second path, for no particular reason. (Either one would lead us to an optimal solution.) In the actual network, this corresponds to the augmenting path

$$s \xrightarrow{0/1} c \xleftarrow{3/3} b \xleftarrow{1/3} a \xrightarrow{1/4} t$$

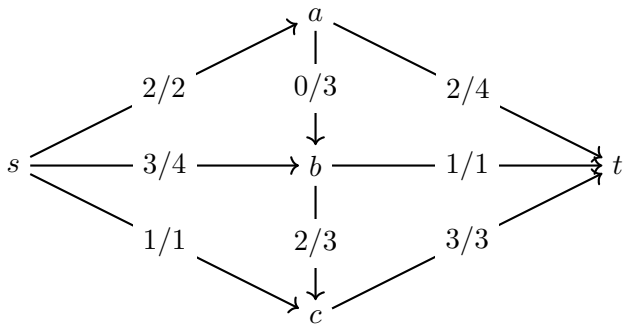
[1]This document comes from the Math 482 course webpage: https://faculty.math.illinois.edu/~mlavrov/courses/482-spring-2020.html



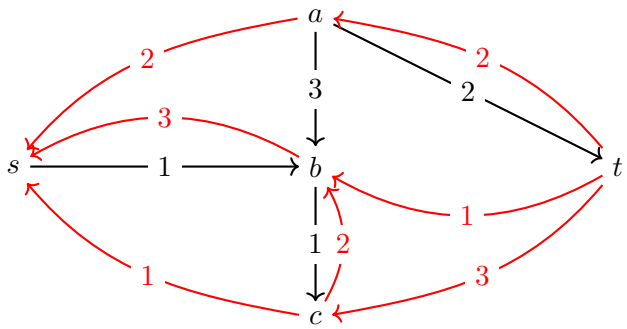
which we can modify by adding 1 to every forward edge, and subtracting 1 from every backward edge:

$$s \xrightarrow{1/1} c \xleftarrow{2/3} b \xleftarrow{0/3} a \xrightarrow{2/4} t$$

The new, updated, network flow is



Here's the new part. Let's also update the residual graph for this network flow: the new residual graph is



If we try to find a path from s to t in the new residual graph, we fail. From s, we can only go to b directly. From b, a new path to c opens up (and nowhere else). From c, we can only go back to b, which doesn't help.

This seems disappointing, but actually it's expected, and it's what we want to see. Here's why.

Theorem 1.1. *Suppose that there is no path from s to t in the residual graph (for some feasible flow $\mathbf{x}$ in some network). Then:*

- *The flow $\mathbf{x}$ is a maximum flow.*
- *Let S be the set of all nodes reachable from s in the residual graph (including s itself). Let T be the set of all other nodes. Then (S, T) is a minimum cut, and has capacity equal to the value of $\mathbf{x}$.*

So, in the example above, the value of the most recent flow we found is 6. The nodes reachable from s in the residual graph are $\{s, b, c\}$, giving us the cut with $\{s, b, c\}$ on one side and $\{a, t\}$ on the other. The capacity of this cut is $c_{sa} + c_{bt} + c_{ct} = 2 + 1 + 3 = 6$.

2 Proof of Theorem 1.1

Things work out in the example above: since the capacity of the cut (S, T) agrees with the value of $\mathbf{x}$, both are optimal.

Why is this true in general?

Three lectures ago, when we were proving that the capacity of a cut gives an upper bound on the value of a flow, we showed the following:

$$\sum_{j:(s,j)\in A} x_{sj} - \sum_{i:(i,s)\in A} x_{is} = \sum_{i\in S}\sum_{j\in T} x_{ij} - \sum_{i\in T}\sum_{j\in S} x_{ij} \leq \sum_{i\in S}\sum_{j\in T} c_{ij}$$

On the left, we have (by definition) the value of the flow $\mathbf{x}$. By some algebraic manipulation, we showed that this is equal to the middle expression: the total flow crossing from S to T, minus the total flow crossing from T back to S. This is upper bounded by the expression on the right (by using $x_{ij} \leq c_{ij}$ on every term of the first sum, and $x_{ij} \geq 0$ on every term of the middle sum), and the expression on the right is just the capacity of the cut (S, T).

Now let's think about what happens in the special case where S is the set of all vertices reachable from the source in the residual graph. This means that there can be no residual arc from any node $i \in S$ to any node $j \in T$.

There are two kinds of residual arcs.

- Forward residual arcs $i \to j$ corresponding to arcs (i, j) with $x_{ij} < c_{ij}$.

  If there are no such arcs from S to T, that means that for every $i \in S$ and $j \in T$, we have $x_{ij} = c_{ij}$.

- Backward residual arcs $i \leftarrow j$ corresponding to arcs (i, j) with $x_{ij} > 0$.

  If there are no such arcs from S to T, that means that for every $i \in T$ and $j \in S$, we have $x_{ij} = 0$.

As a result, for this cut, instead of the inequality we get above, we have the equation

$$\sum_{i\in S}\sum_{j\in T} x_{ij} - \sum_{i\in T}\sum_{j\in S} x_{ij} = \sum_{i\in S}\sum_{j\in T} c_{ij} - \sum_{i\in T}\sum_{j\in S} 0 = \sum_{i\in S}\sum_{j\in T} c_{ij}.$$

Therefore the value of the flow $\mathbf{x}$ is equal to the capacity of the cut (S, T).

The optimality of both of the flow and the cut follows. We know the capacity of the cut (S, T) is an upper bound on the value of any flow; since $\mathbf{x}$ achieves that upper bound, it is a maximum flow (no other flow can be better). Similarly, the value of $\mathbf{x}$ is a lower bound on the capacity of any cut: since (S, T) achieves that lower bound, it is a minimum cut.

3 The Ford–Fulkerson algorithm

This leads us to an algorithm for trying to find a maximum flow in a network without using linear programming.

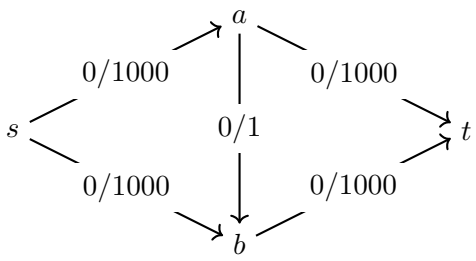
Start at a feasible flow: for example, the flow $\mathbf{x} = \mathbf{0}$. Then repeat the augmenting step we've developed:

1. Construct the residual graph for $\mathbf{x}$.
2. Attempt to find a path from s to t in the residual graph.
3. If a path exists, it gives us an augmenting path, which we use to improve $\mathbf{x}$ and go back to step 1.
4. If no path exists, we use Theorem 1.1 to obtain a minimum cut whose capacity is equal to the value of $\mathbf{x}$. We know $\mathbf{x}$ is optimal, and the cut gives us a certificate of optimality.

As with the simplex algorithm, there is one more thing left to worry about. Ford–Fulkerson lets us always keep getting a better flow, and only stops when we reach a maximum flow, but are we guaranteed to actually reach it?

Unfortunately, as stated so far, there is no such guarantee in general. We *can* say that we'll eventually be done in cases where all the capacities c_{ij} are integers. In this case, at every step, the value of the flow increases by at least 1, and so eventually it will reach $\sum_{j:(s,j)\in A} c_{sj}$, which is an upper bound on the value of the maximum flow. Similarly, if all the capacities are rational numbers, the flow always increases by at least $\frac{1}{d}$, where d is the greatest common denominator of all the capacities.

But this is a very bad upper bound. Consider the following example:



The maximum flow here has value 2000, which can be reached in just 2 steps. But if we have poor judgement, and alternate between the augmenting paths $s \to a \to b \to t$ and $s \to b \leftarrow a \to t$, we increase the flow by 1 at each step, and only finish is 2000 steps.

Things are even worse if some capacities are irrational numbers. Then there is an example[2] in which a poor choice of augmenting paths means we never even get close to the maximum flow.

Fortunately, there is a simple rule to avoid this situation. If we always pick the shortest augmenting path available at every step, then it can be shown that we'll always reach the maximum flow after at most $n \cdot m$ steps (in a network with n nodes and m arcs). This refinement is called the Edmonds–Karp algorithm.

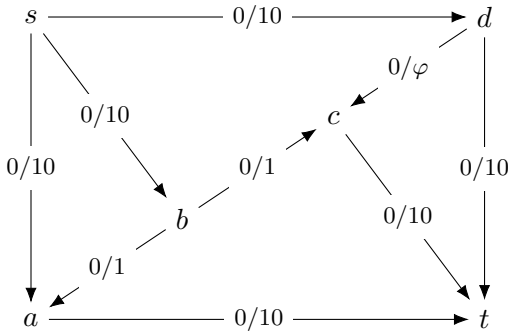
[2] See the extremely long optional section at the end of these notes.

Edmonds-Karp algorithm.

4 Infinite loops in Ford–Fulkerson (very optional)

I get this example by way of the notes at http://jeffe.cs.illinois.edu/teaching/algorithms/book/10-maxflow.pdf (pp. 335–336), which use an example from "The smallest networks on which the Ford–Fulkerson maximum flow procedure may fail to terminate" by Uri Zwick. Feel free to consult either of these sources, but I hope to develop the example in more detail here.

Consider the network

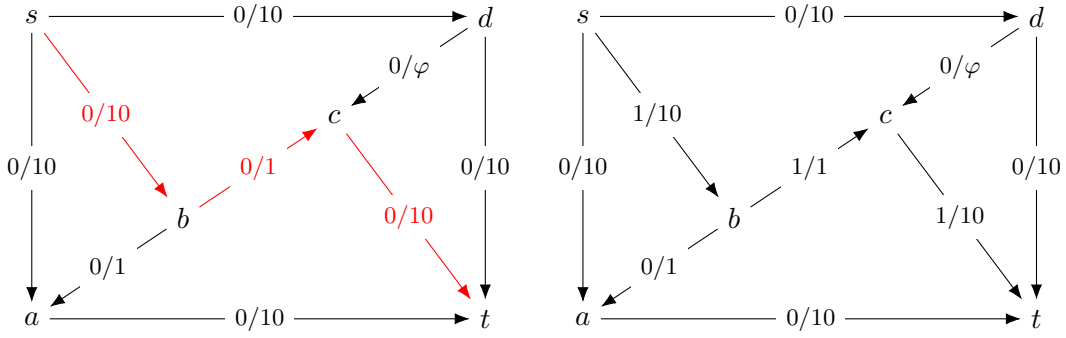


where the capacity φ is the real number $\frac{\sqrt{5}-1}{2} \approx 0.618$. This value is chosen to satisfy $\varphi^2 = 1 - \varphi$. The maximum flow in this network has value 21, but we'll never get there: in fact, the value of the flow will always be less than 5. So for the purposes of our augmenting paths, the arcs with capacity 10 might as well have infinite capacity; we'll never use it all up anyway.

The augmenting paths we use have to be very carefully (badly) chosen to end up in a situation where we keep repeating forever. Note, though, that we're not being *stupidly* self-sabotaging: whenever we choose an augmenting path, we do augment the flow along that path as much as we can.

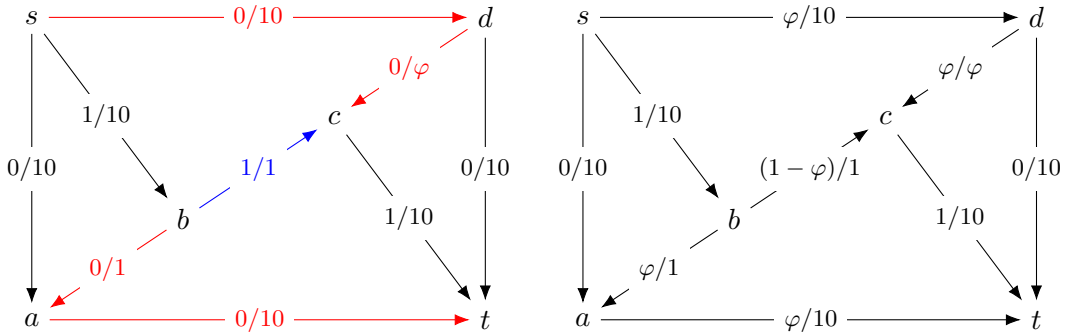
The sequence of augmenting paths has a repeating pattern. There is an initial augmenting step which is done once:

1. Augment along the path $s \to b \to c \to t$ (shown on the left), getting a flow with a value of 1 (shown on the right). The bottleneck is the arc $b \to c$, which has a residual capacity of 1.

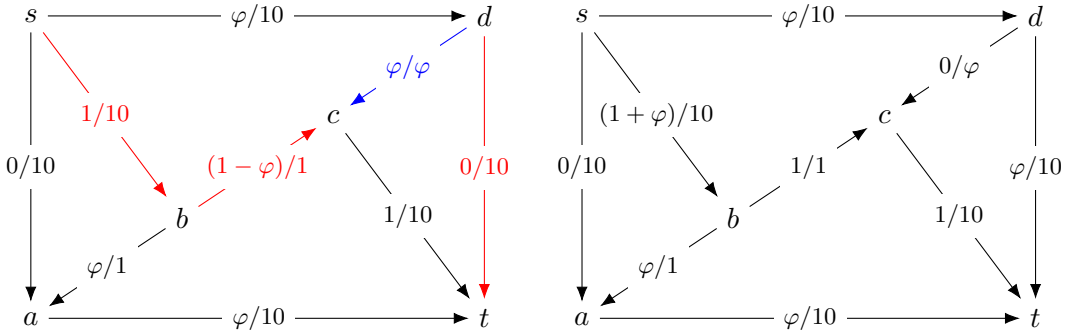


From here, there is a sequence of four augmenting paths that repeat forever (steps 2–5 below). That is, the paths we choose to augment by will repeat; the flow $\mathbf{x}$ will keep changing, and we'll augment by a smaller and smaller amount in each cycle.

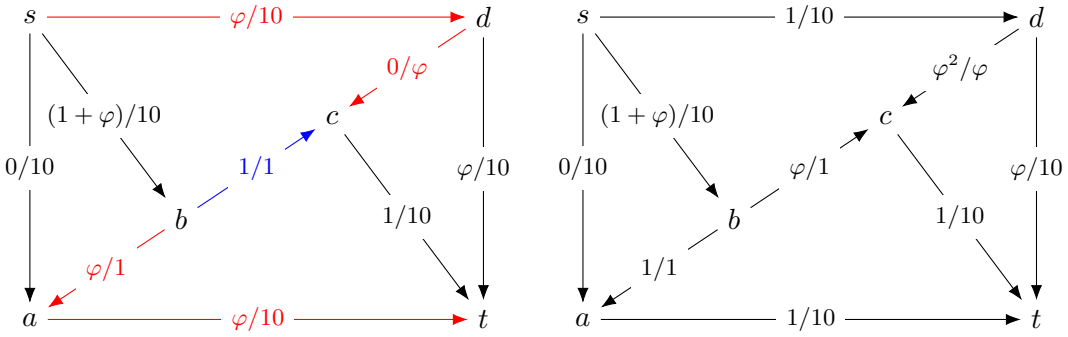
2. Augment along the path $s \to d \to c \leftarrow b \to a \to t$ (shown on the left), getting a flow with a value of $1 + \varphi$ (shown on the right). We are limited by the arc $d \to c$, which has a residual capacity of φ.



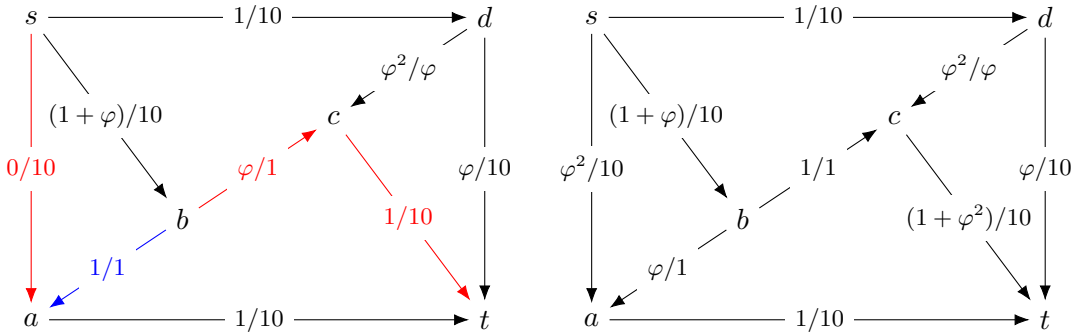
3. Augment along the path $s \to b \to c \leftarrow d \to t$ (shown on the left), getting a flow with a value of $1 + 2\varphi$ (shown on the right). We are limited by the forward arc $b \to c$ and the backward arc $d \to c$, both with a residual capacity of φ.



4. Augment along the path $s \to d \to c \leftarrow b \to a \to t$ (shown on the left), increasing the value by φ^2 (which is equal to $1 - \varphi$) and getting a flow with a value of $1 + 2\varphi + \varphi^2$ (shown on the right). We are limited by the arc $b \to a$, which has residual capacity $1 - \varphi$.



5. Augment along the path $s \to a \leftarrow b \to c \to t$ (shown on the left), increasing the value by φ^2 and getting a flow with a value of $1 + 2\varphi + 2\varphi^2 = 3$ (shown on the right). We are limited by the arc $b \to c$, which has residual capacity $1 - \varphi = \varphi^2$.



Let's compare the network at the end of step 1 and the network at the end of step 5.

- In the network at the end of step 1, arcs $b \to a$, $b \to c$, and $d \to c$ have residual capacities 1, 0, and φ, respectively.
- In the network at the end of step 5, arcs $b \to a$, $b \to c$, and $d \to c$ have residual capacities $1 - \varphi$, 0, and $\varphi - \varphi^2$, respectively.

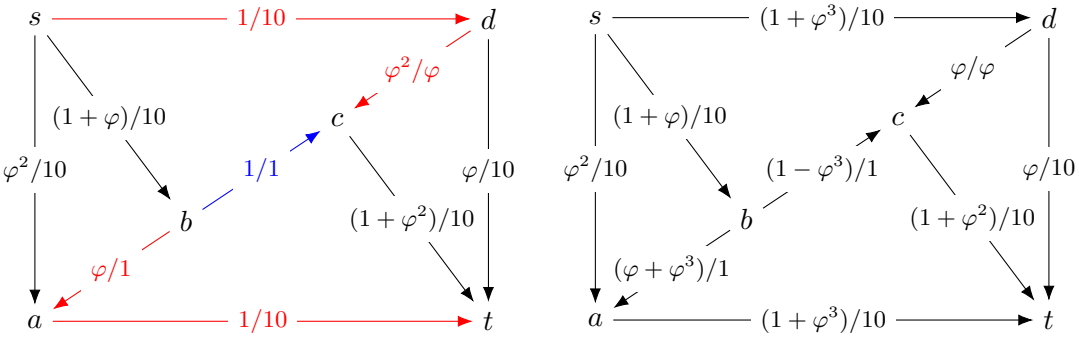
  But that's not the clearest way to write these residual capacities. Because $\varphi^2 = 1 - \varphi$, these residual capacities can be written as φ^2, 0, and φ^3, respectively.

We see that after steps 2–5, all three residual capacities have multiplied by φ^2.

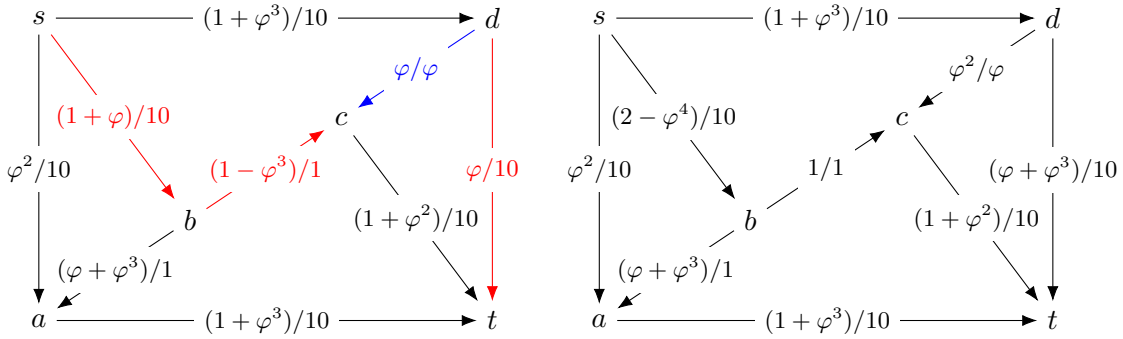
This is the pattern that will continue in the next four steps. Steps 6–9 will be the same as steps 2–5, except that in every augmenting step, the amount we augment by will be multiplied by φ^2. (That is, we augmented by $\phi, \phi, \phi^2, \phi^2$ in steps 2, 3, 4, 5 and we will augment by $\phi^3, \phi^3, \phi^4, \phi^4$ in step 6, 7, 8, 9.) In steps 10–13, the amount will be multiplied by φ^2 again: we will augment by $\phi^5, \phi^5, \phi^6, \phi^6$ in steps 10, 11, 12, 13. This pattern will continue forever.

I will repeat one more iteration (steps 6–9) to illustrate how the next infinitely many steps will go. (Feel free to skip this if you think you get it.)

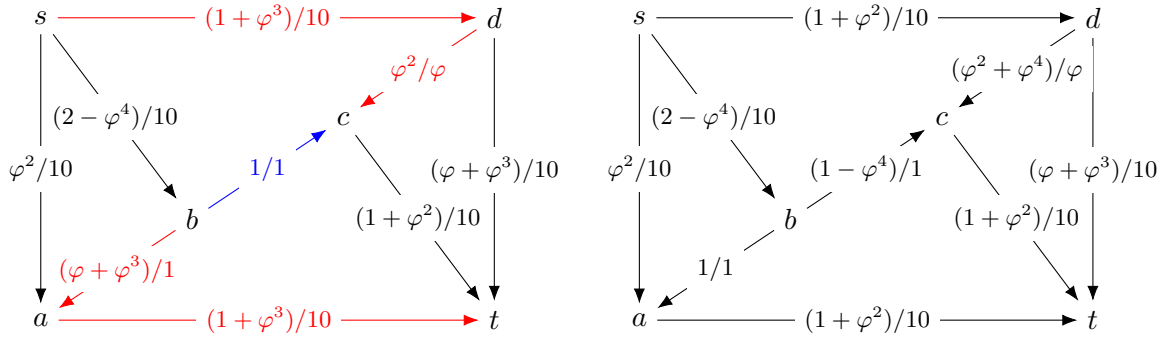
6. Augment along the path $s \to d \to c \leftarrow b \to a \to t$ (shown on the left), getting a flow with a value of $3 + \varphi^3$ (shown on the right). We are limited by the arc $d \to c$, which has a residual capacity of $\varphi - \varphi^2 = \varphi^3$.



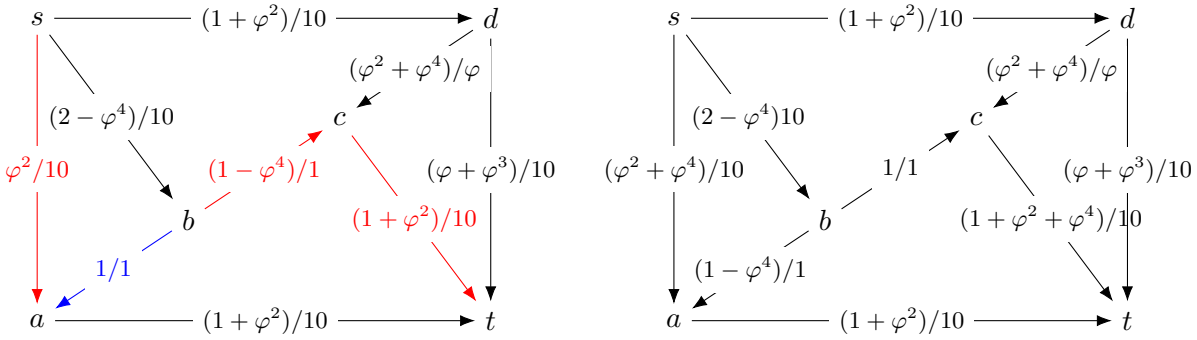
7. Augment along the path $s \to b \to c \leftarrow d \to t$ (shown on the left), getting a flow with a value of $3 + 2\varphi^3$ (shown on the right). We are limited by the arc $b \to c$, with residual capacity φ^3.



8. Augment along the path $s \to d \to c \leftarrow b \to a \to t$ (shown on the left), increasing the value by φ^4 and getting a flow with a value of $3 + 2\varphi^3 + \varphi^4$ (shown on the right). We are limited by the arc $b \to a$, which has residual capacity $1 - \varphi - \varphi^3 = \varphi^2 - \varphi^3 = \varphi^2(1 - \varphi) = \varphi^4$.



9. Augment along the path $s \to a \leftarrow b \to c \to t$ (shown on the left), increasing the value by φ^4 and getting a flow with a value of $3 + 2\varphi^3 + 2\varphi^4$ (shown on the right). We are limited by the arc $b \to c$, which has residual capacity φ^4.



Say that this pattern keeps going forever. What will we get after infinitely many steps like this? (How do we know that the value of the flow won't get arbitrarily large?)

Steps 2–5 increased the flow by $2\varphi^2 + 2\varphi^3$. Steps 6–9 increased it by $2\varphi^3 + 2\varphi^4$. Steps 10–13 will

increase it by $2\varphi^5 + 2\varphi^6$. Since $\varphi < 1$, these increments get smaller and smaller in a geometric progression.

We can compute the total using the formula for a geometric series: $a + ar + ar^2 + \cdots = \frac{a}{1-r}$. In this case, the total value is

$$1 + 2\varphi + 2\varphi^2 + 2\varphi^3 + 2\varphi^4 + \cdots = 1 + \frac{2\varphi}{1-\varphi} = 1 + \frac{2\phi}{\phi^2} = 2 + \sqrt{5} \approx 4.236.$$

This is the *limit* of the procedure; at any step in this infinite loop, the value of the flow will be slightly smaller than this.