CSE 421 Section 7

Max Flow / Min Cut

Administrivia

Announcements & Reminders

- Midterm Exam
  - If you think something was graded incorrectly, submit a regrade request!
  - If you have concerns about your overall grade in the course, schedule a 1-on-1 with Robbie to discuss privately

- HW6
  - Due yesterday, Wednesday 2/22

- HW7
  - Due Wednesday 3/1

Ford-Fulkerson Algorithm

Finding the Max-Flow / Min-Cut

We use the Ford-Fulkerson algorithm to find Max-Flow / Min-Cut.

Key Ideas:

- Keep searching through the residual graph to find a path from s to t that we can send more flow down.
- Keep updating the residual graph to track how much flow we can still push through and how much flow we can potentially reroute.
- When we can no longer reach t in the residual graph, we can't send any more flow, so the algorithm terminates!

Residual Graph

The residual graph indicates how much flow can still go along an edge, and how much flow we could potentially reroute back from an edge.

Key ideas:

- the sum of the residual edges between any two nodes should be equal to the value of the edge between them in the original graph
- The residual edge pointing in the direction of the original edge should have a value equal to the amount of flow that could still pass through that edge
- The residual edge pointing in the opposite direction of the original edge should have a value equal to the amount of flow you have currently sent down that edge

Ford-Fulkerson (formally)

While (flow is not maximum)

Run BFS in residual graph starting from s

Record predecessors to find an s,t-path

Iterate through path, finding c minimum residual capacity on path

Add c to every edge on path in flow

Update residual graph

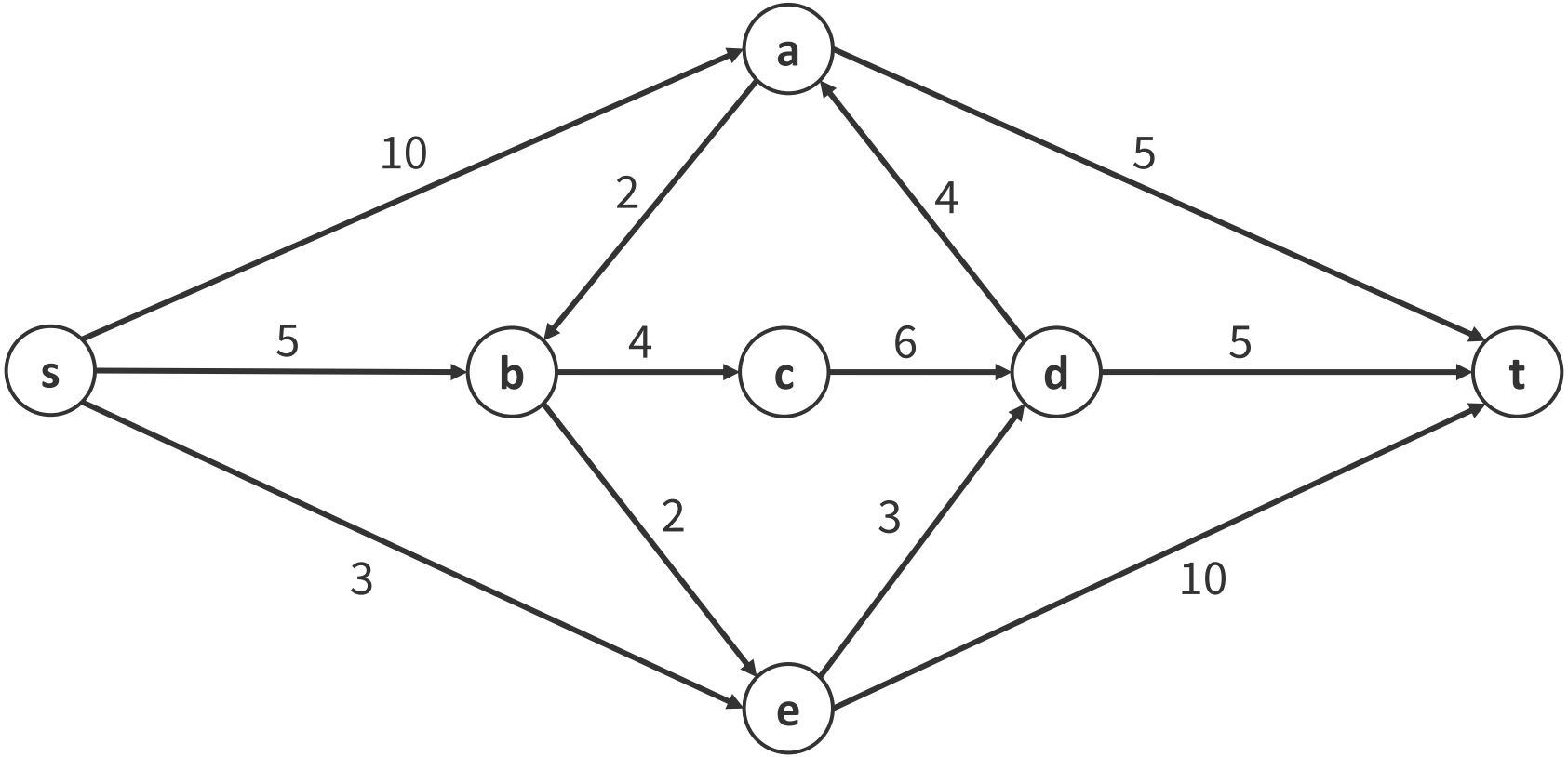
1. Go With the Flow

Problem 1 – Go With the Flow

Using Ford-Fulkerson, find the maximum $s - t$ flow in the graph G below, the corresponding residual graph, and list out the corresponding minimum cut.

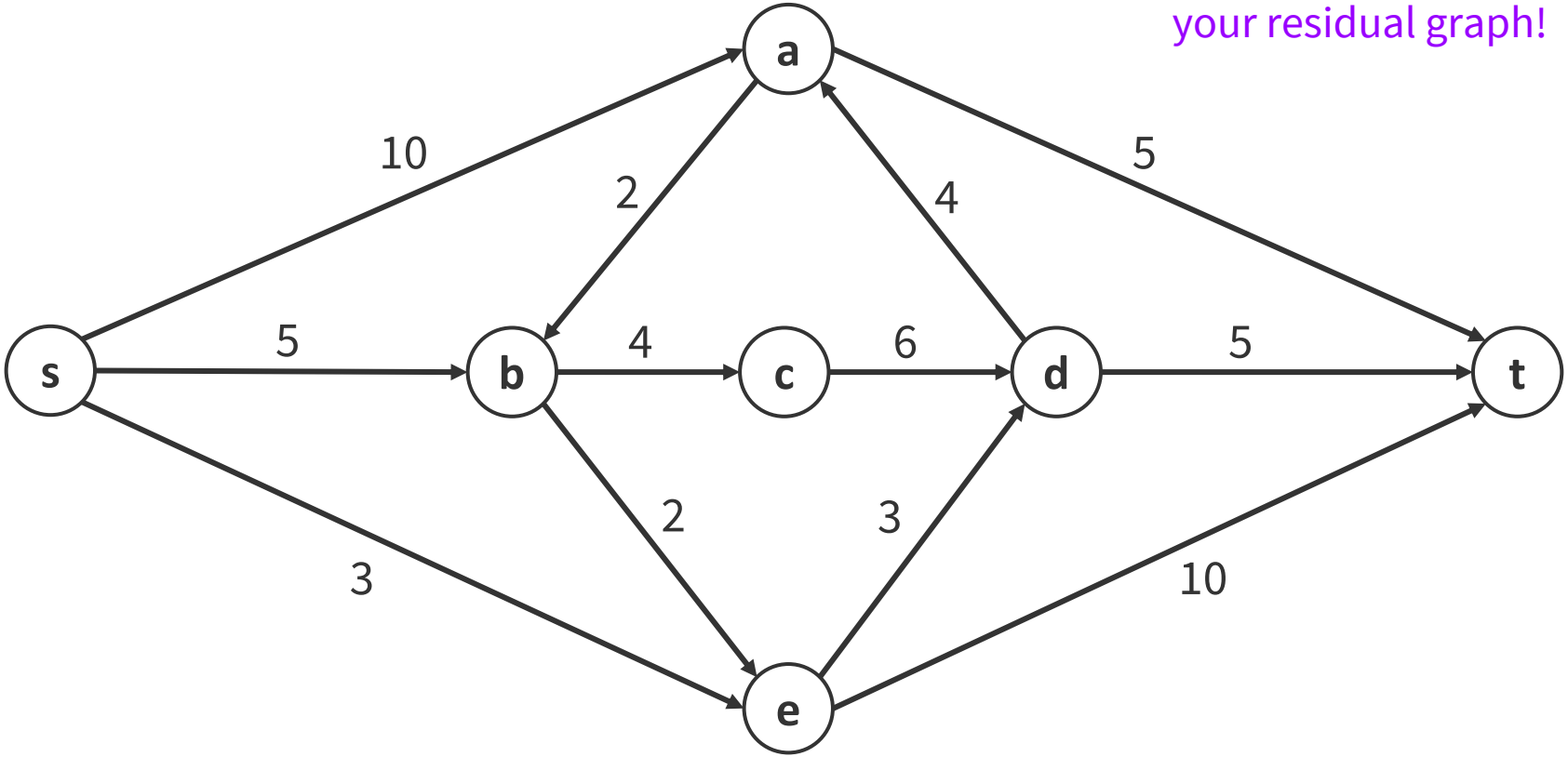
Work through this problem with the people around you, and then we'll go over it together!

Problem 1 – Go With the Flow



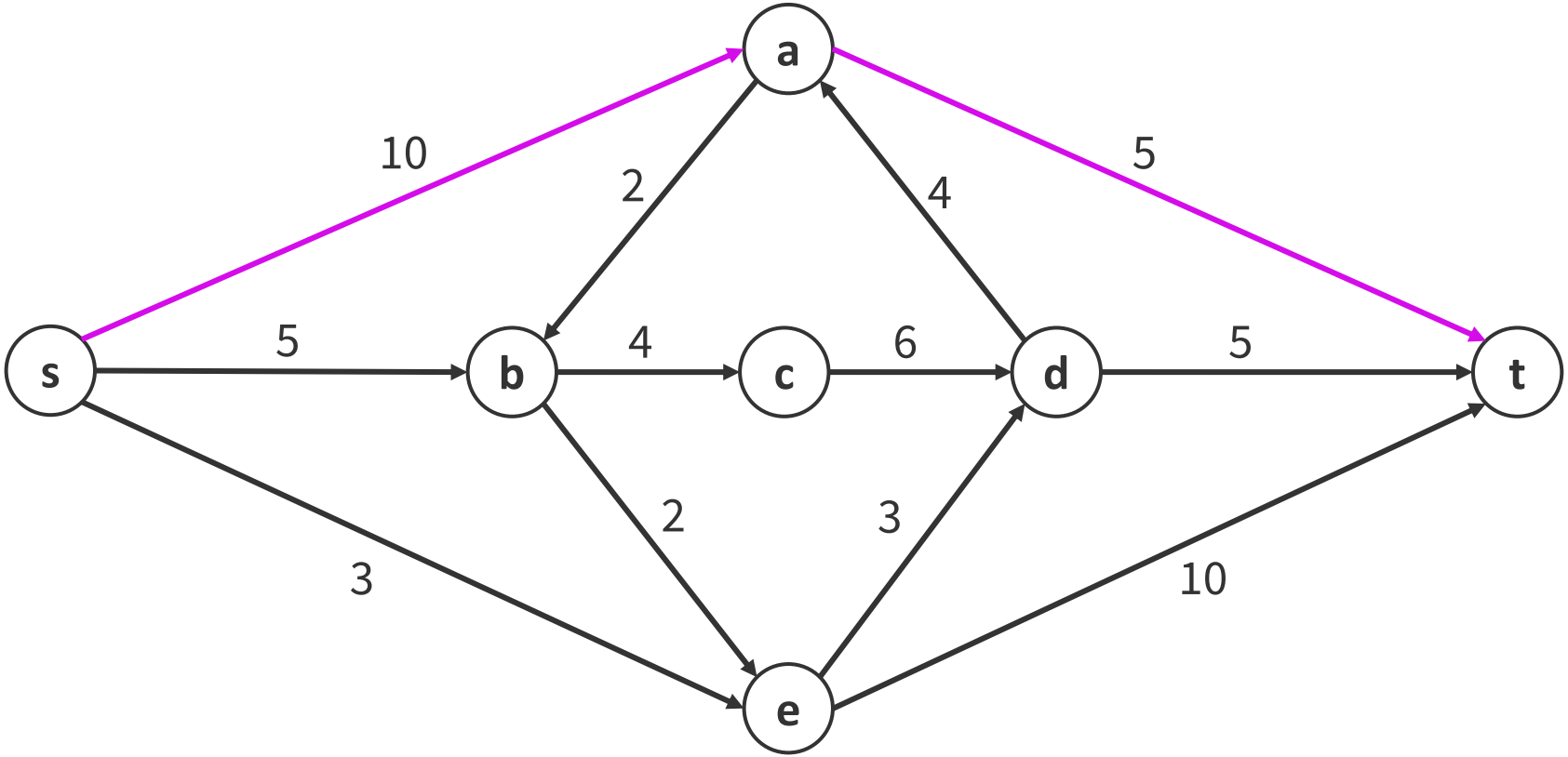
Problem 1 – Go With the Flow

Start by making a copy of the original graph to be your residual graph!



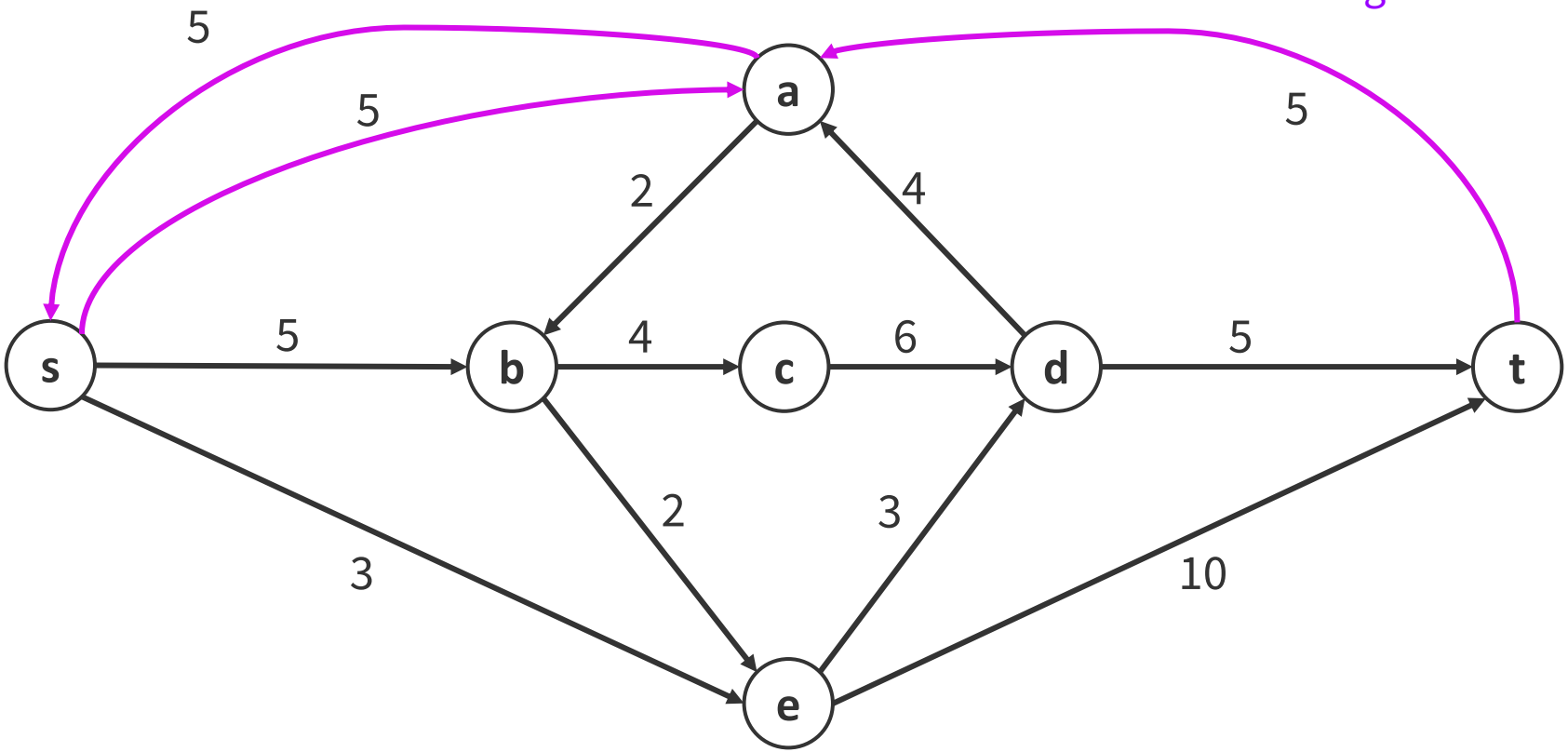
Problem 1 – Go With the Flow

Find a path from s to t

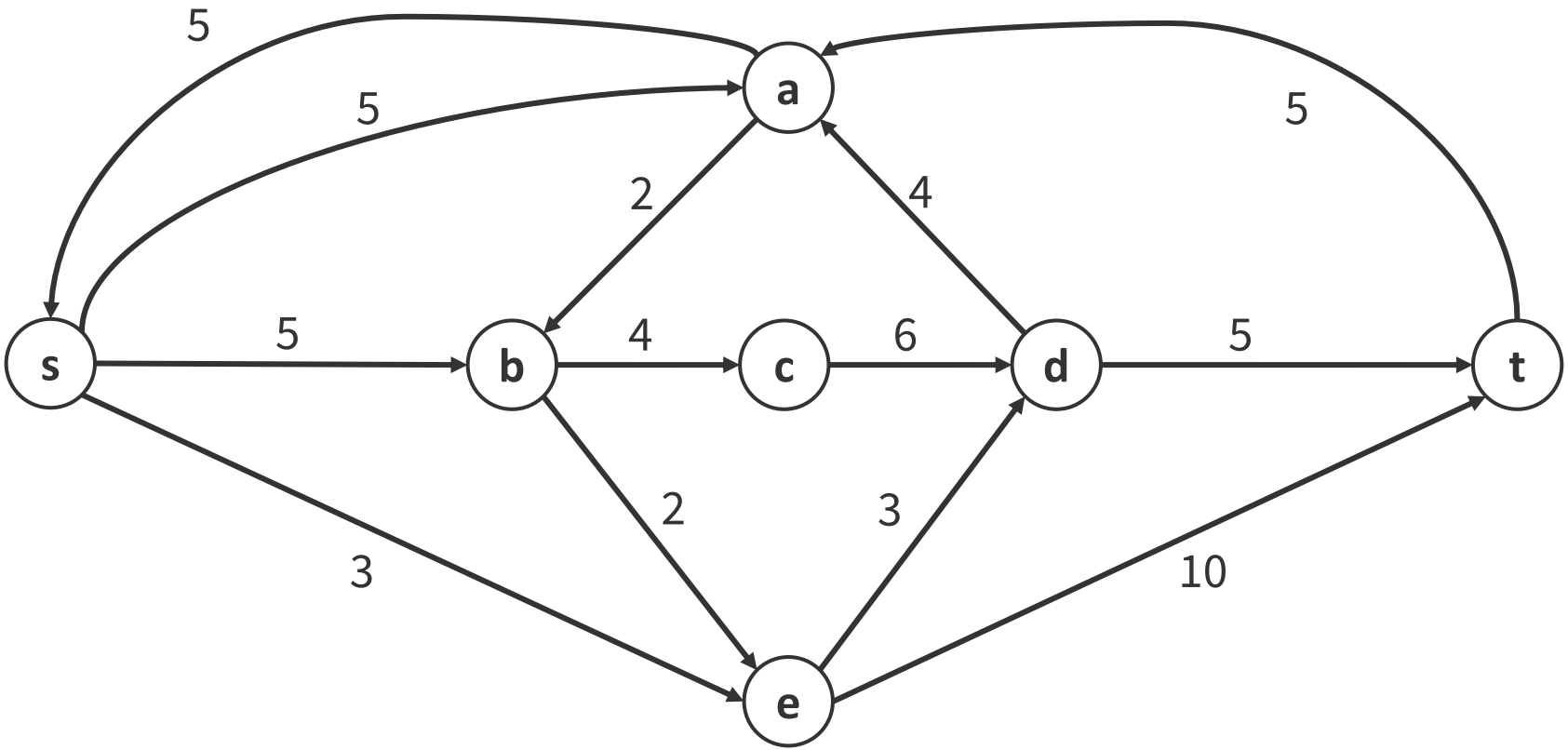


Problem 1 – Go With the Flow

Update the residual edges to push the maximum flow through

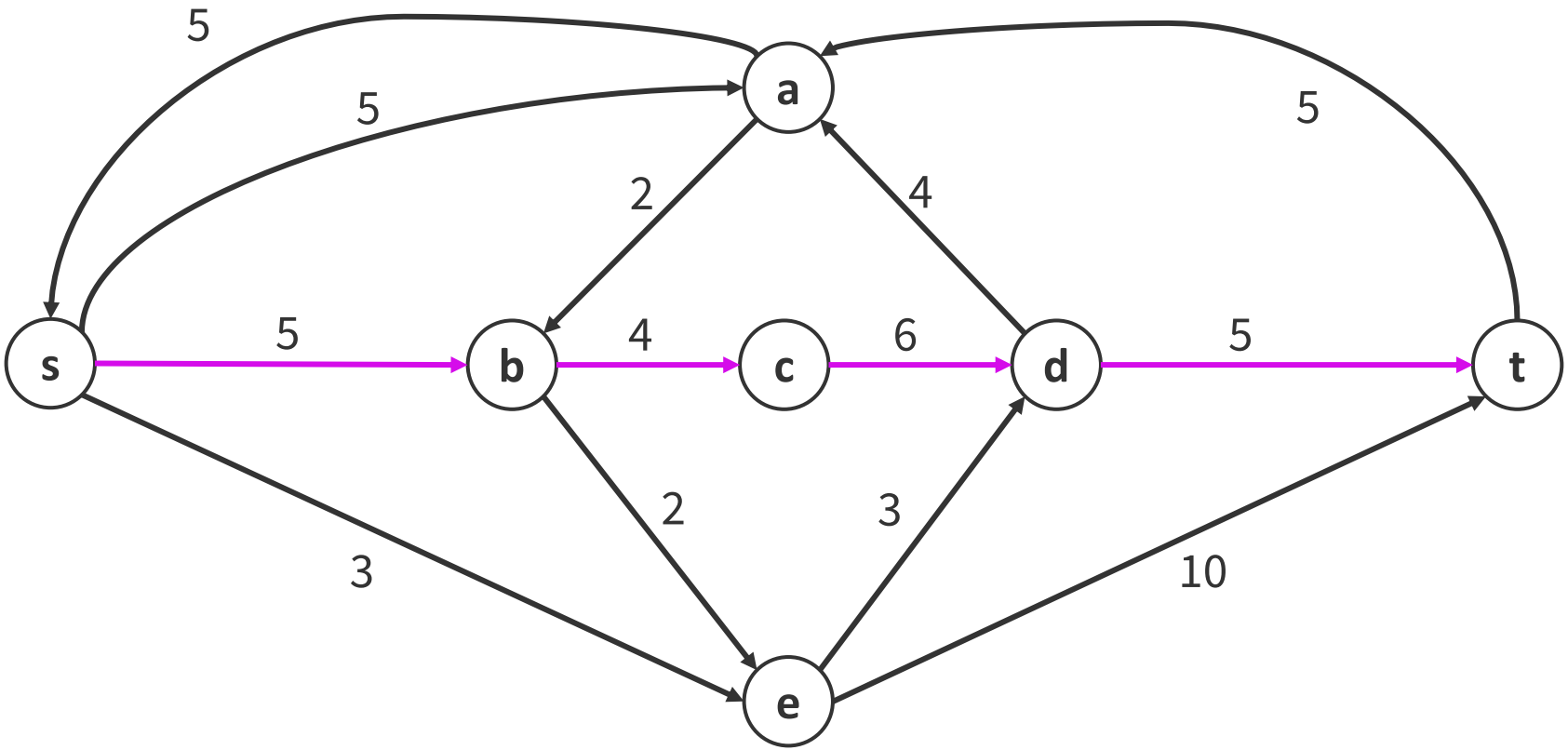


Problem 1 – Go With the Flow



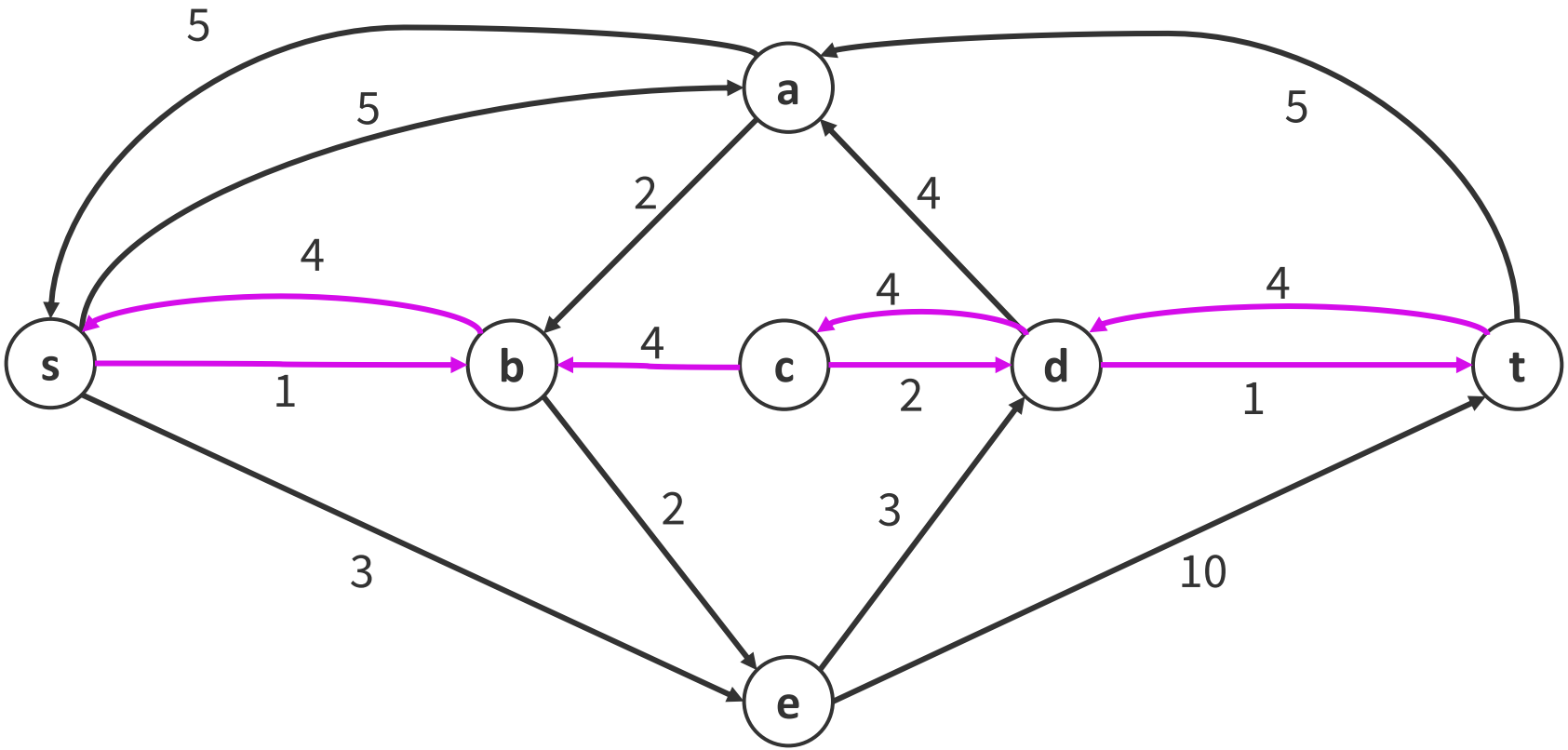
Problem 1 – Go With the Flow

Look to see if there is another path from s to t

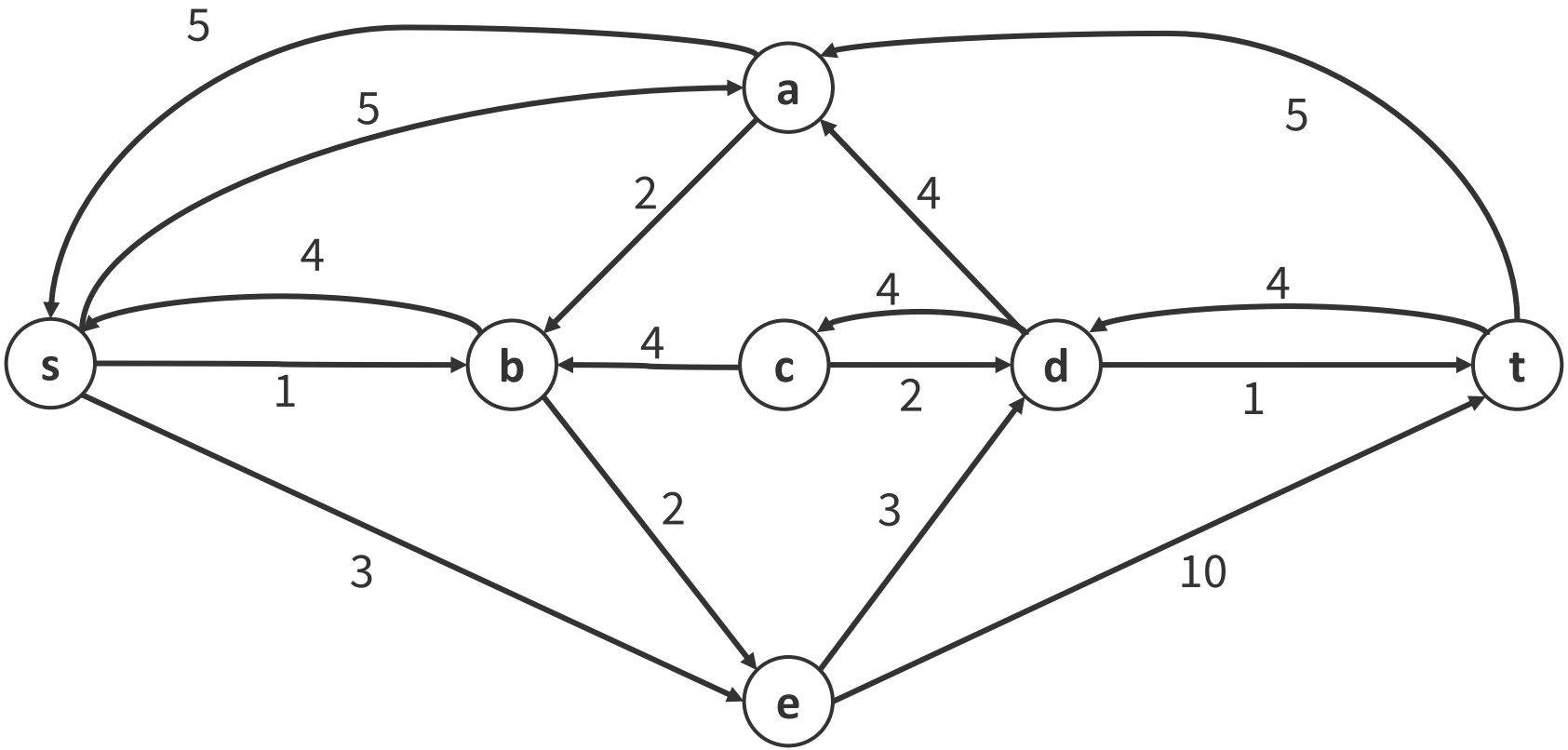


Problem 1 – Go With the Flow

Update the residual edges

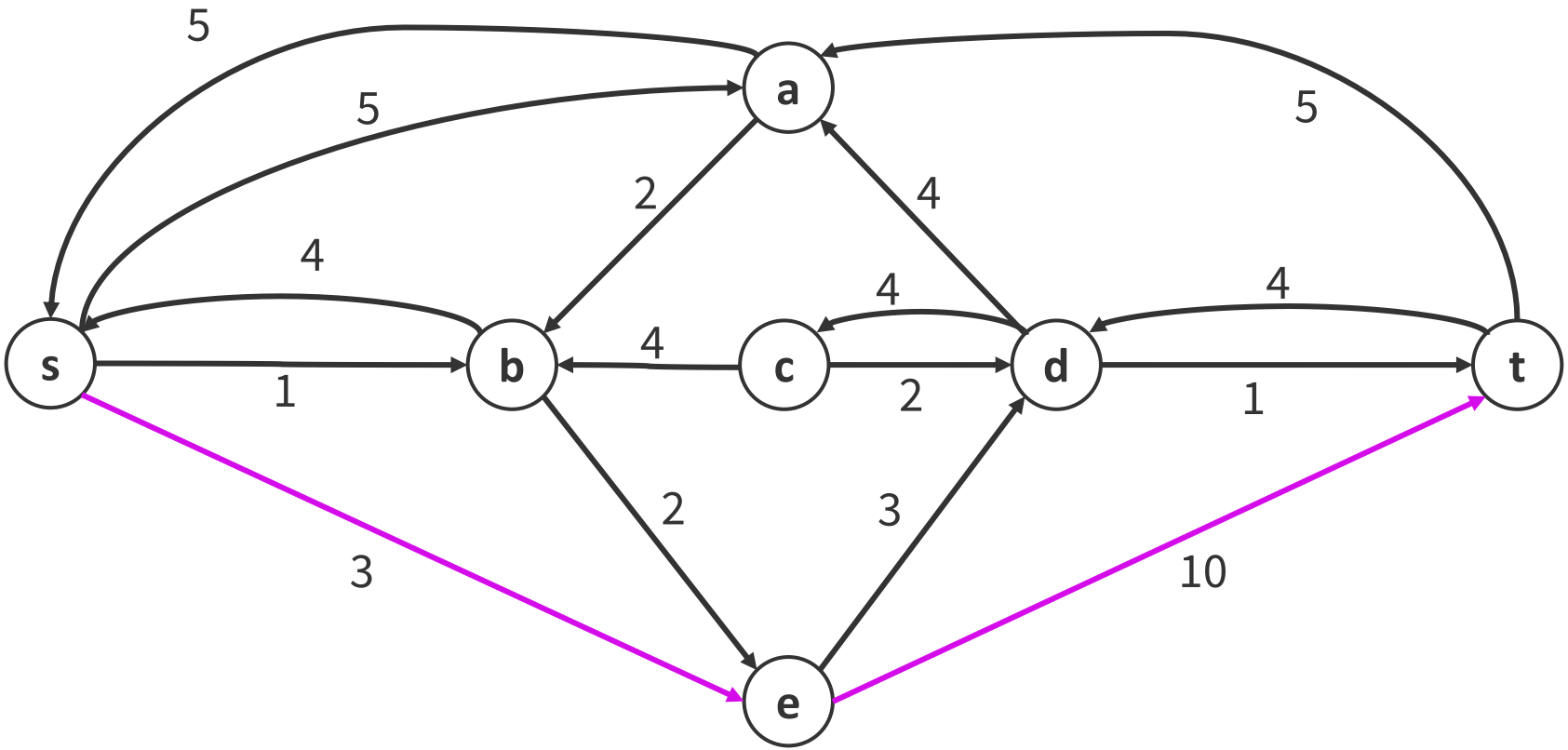


Problem 1 – Go With the Flow



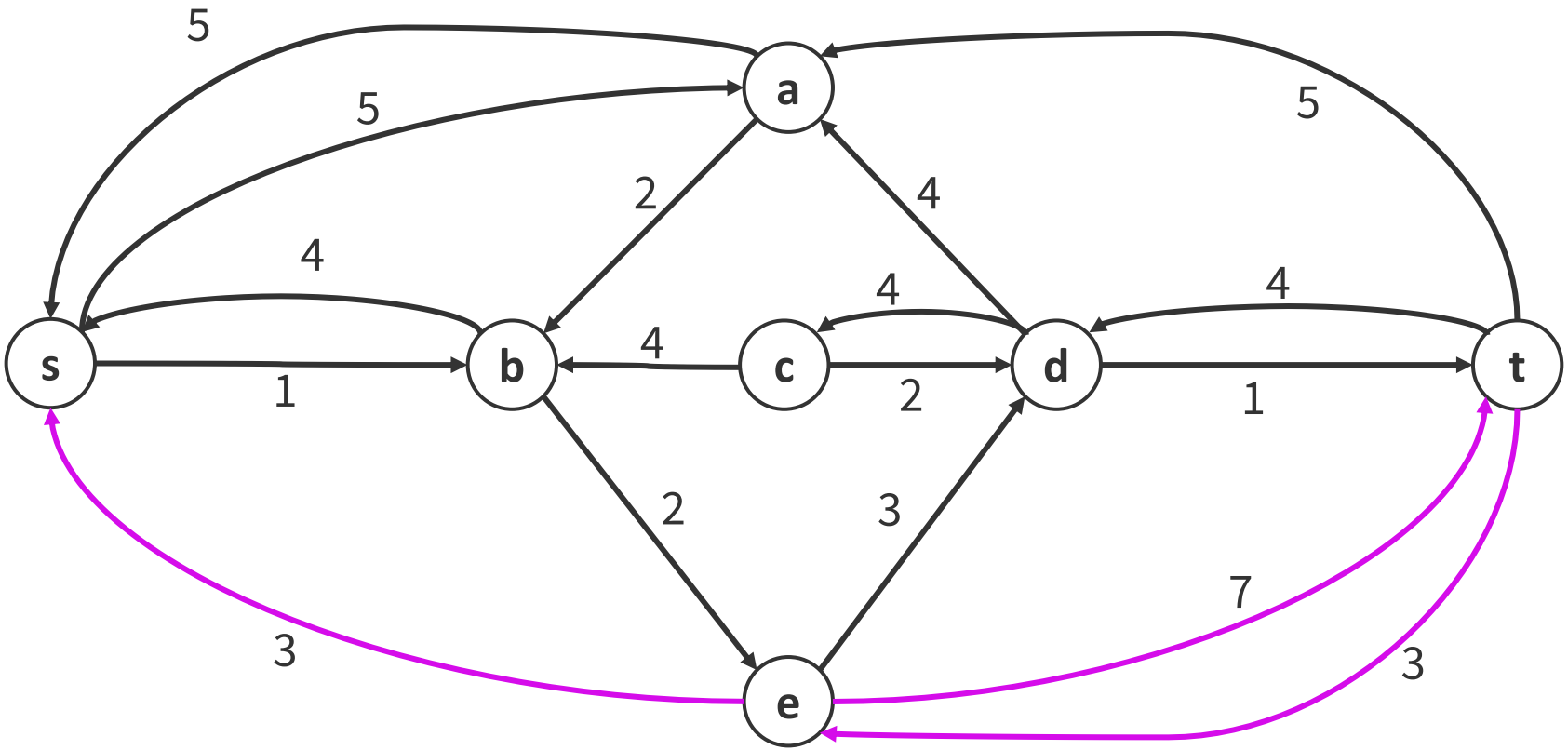
Problem 1 – Go With the Flow

Look to see if there is another path from s to t

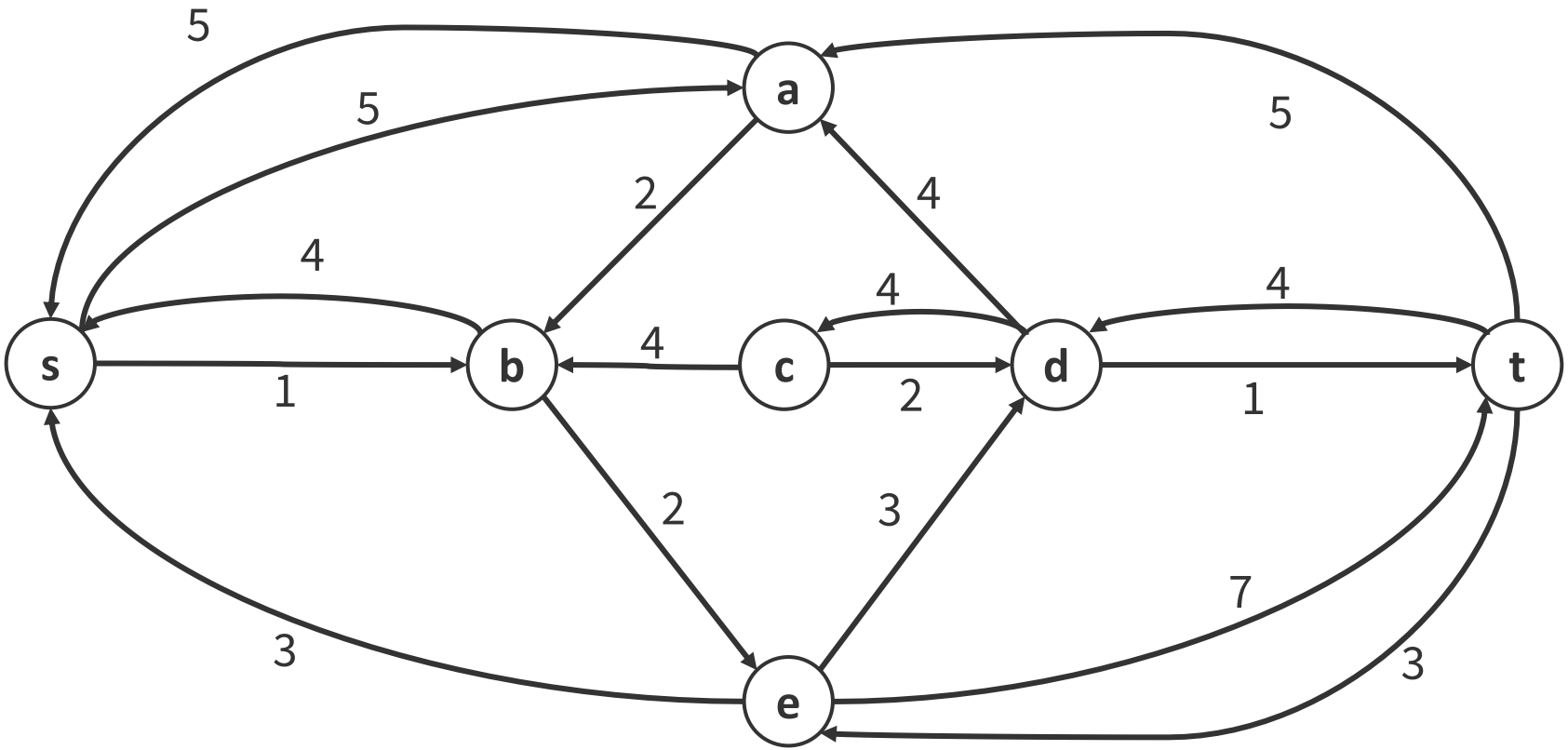


Problem 1 – Go With the Flow

Update the residual edges

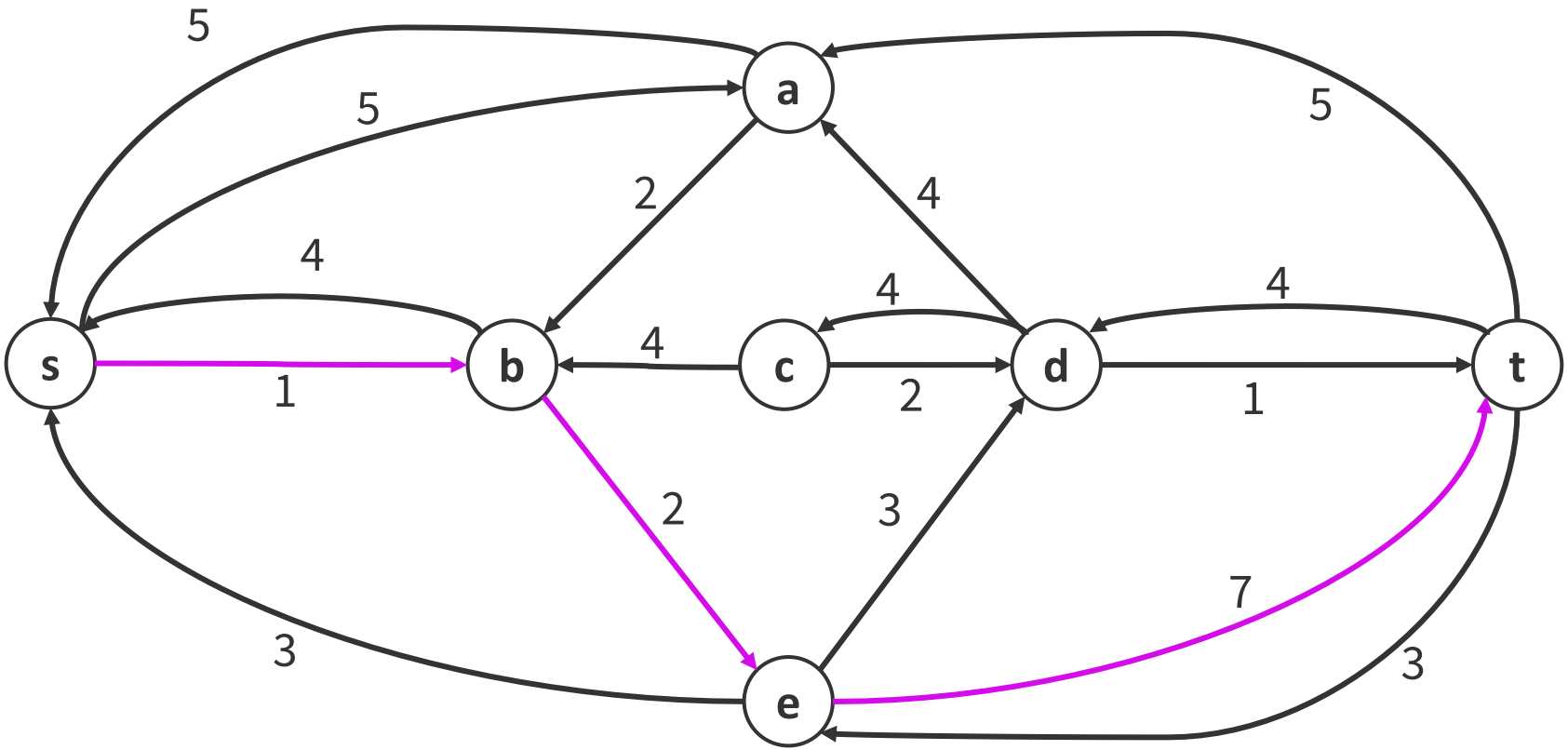


Problem 1 – Go With the Flow



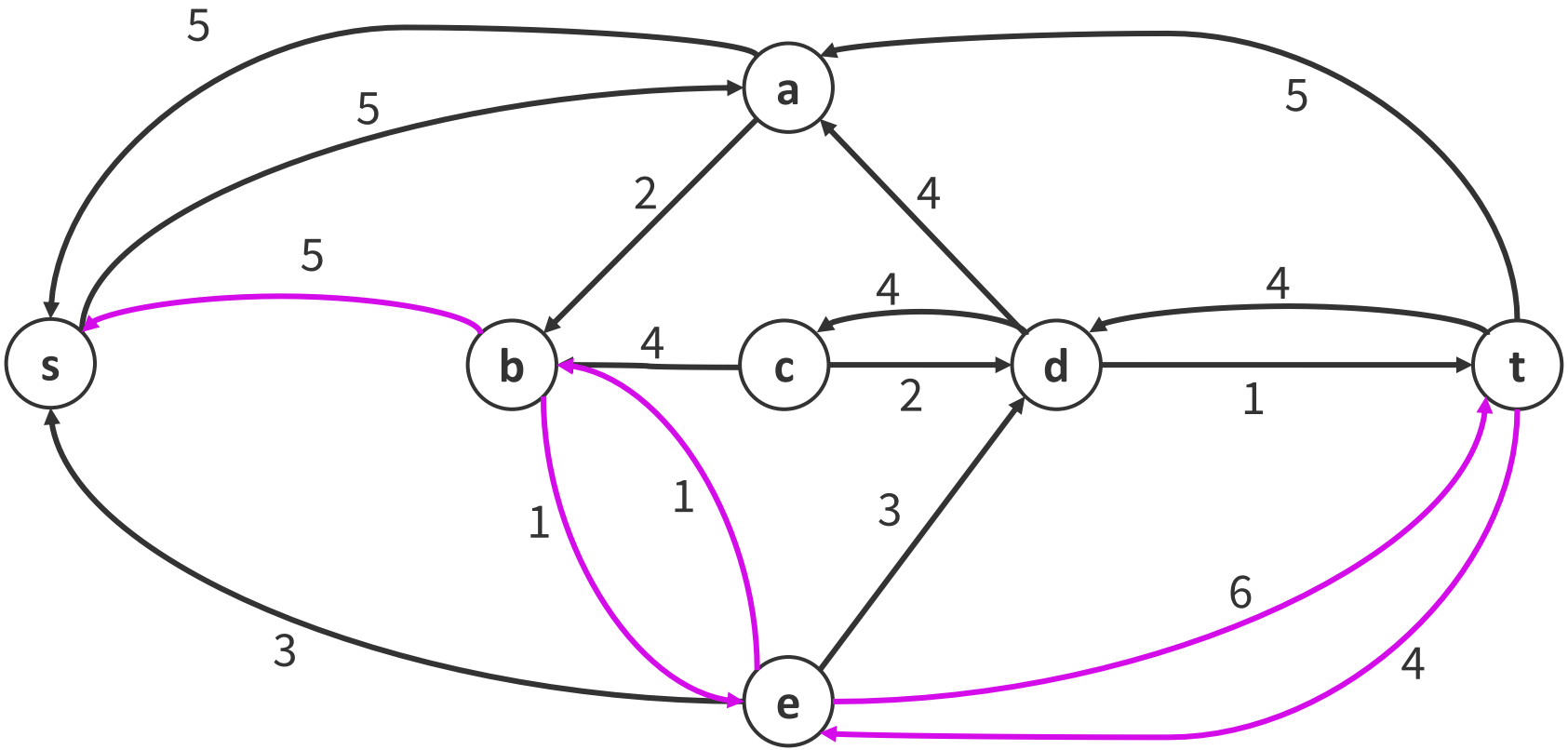
Problem 1 – Go With the Flow

Look to see if there is another path from s to t

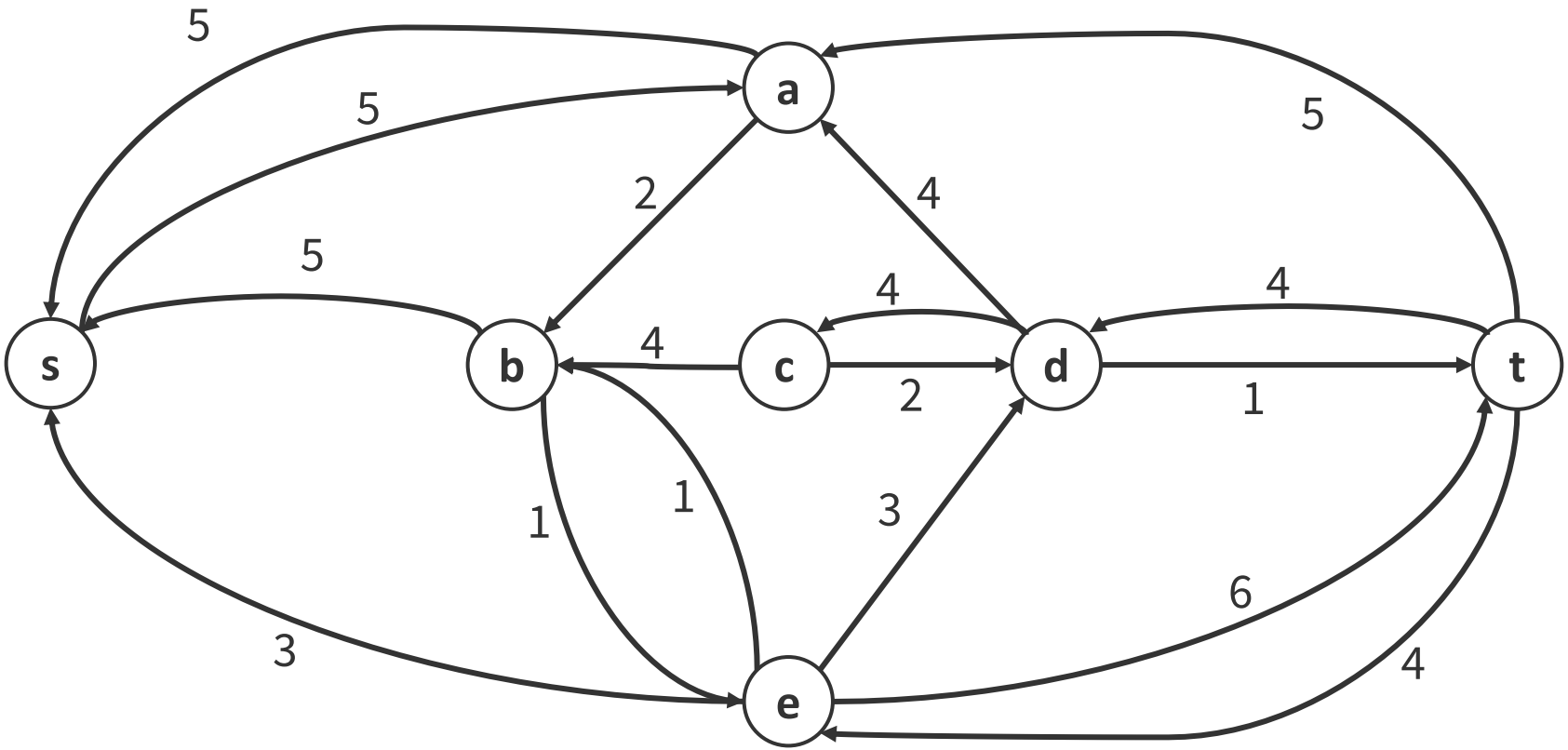


Problem 1 – Go With the Flow

Update the residual edges

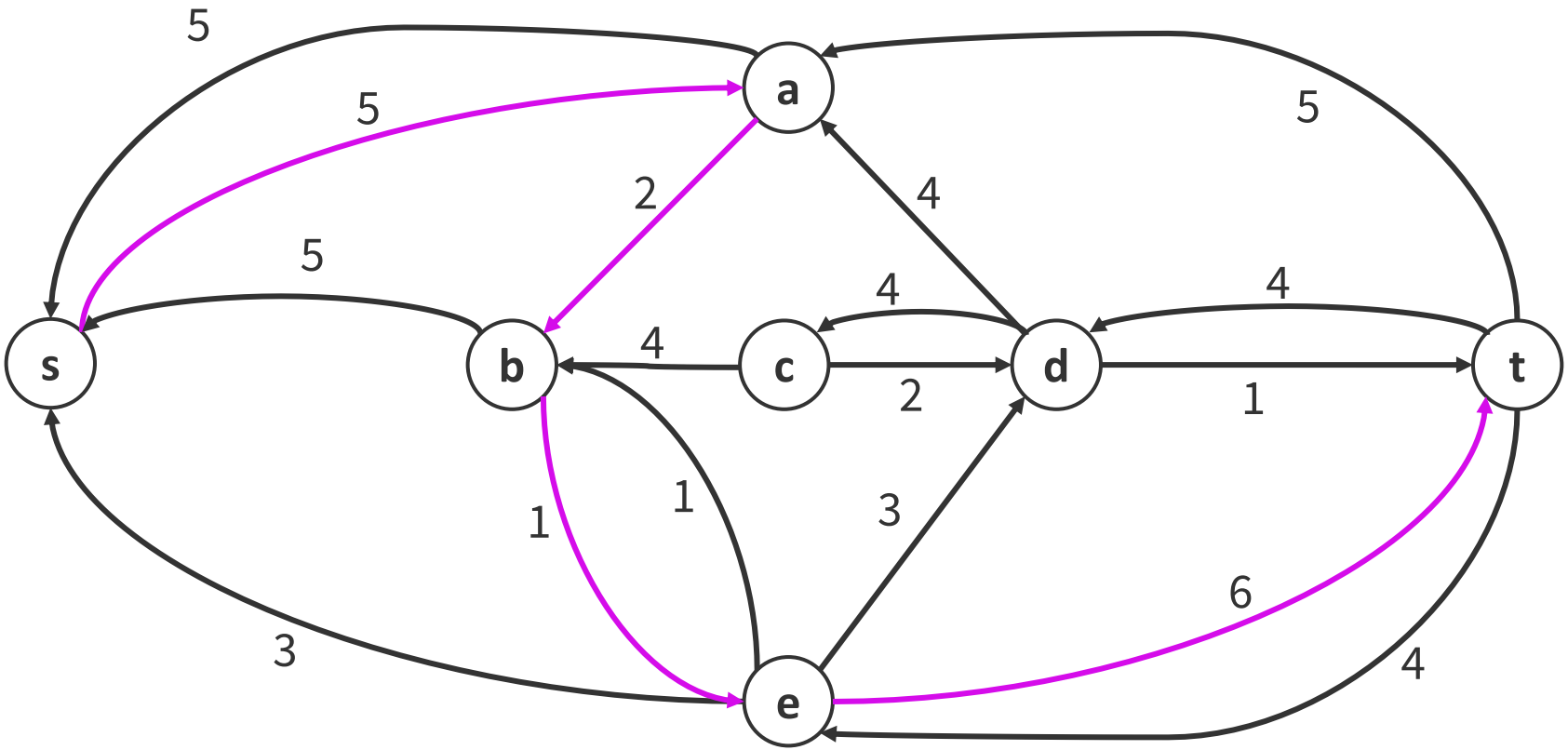


Problem 1 – Go With the Flow



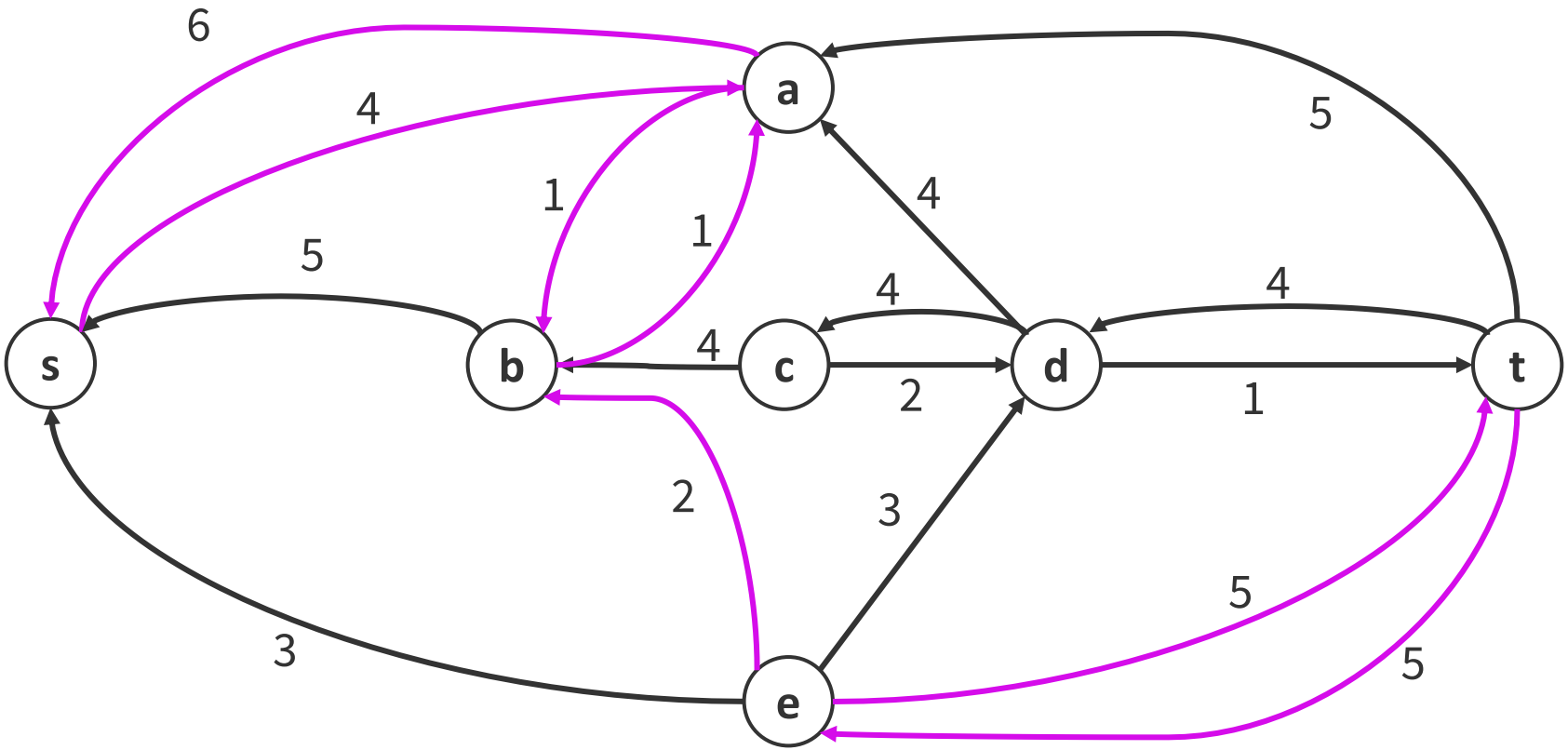
Problem 1 – Go With the Flow

Look to see if there is another path from s to t



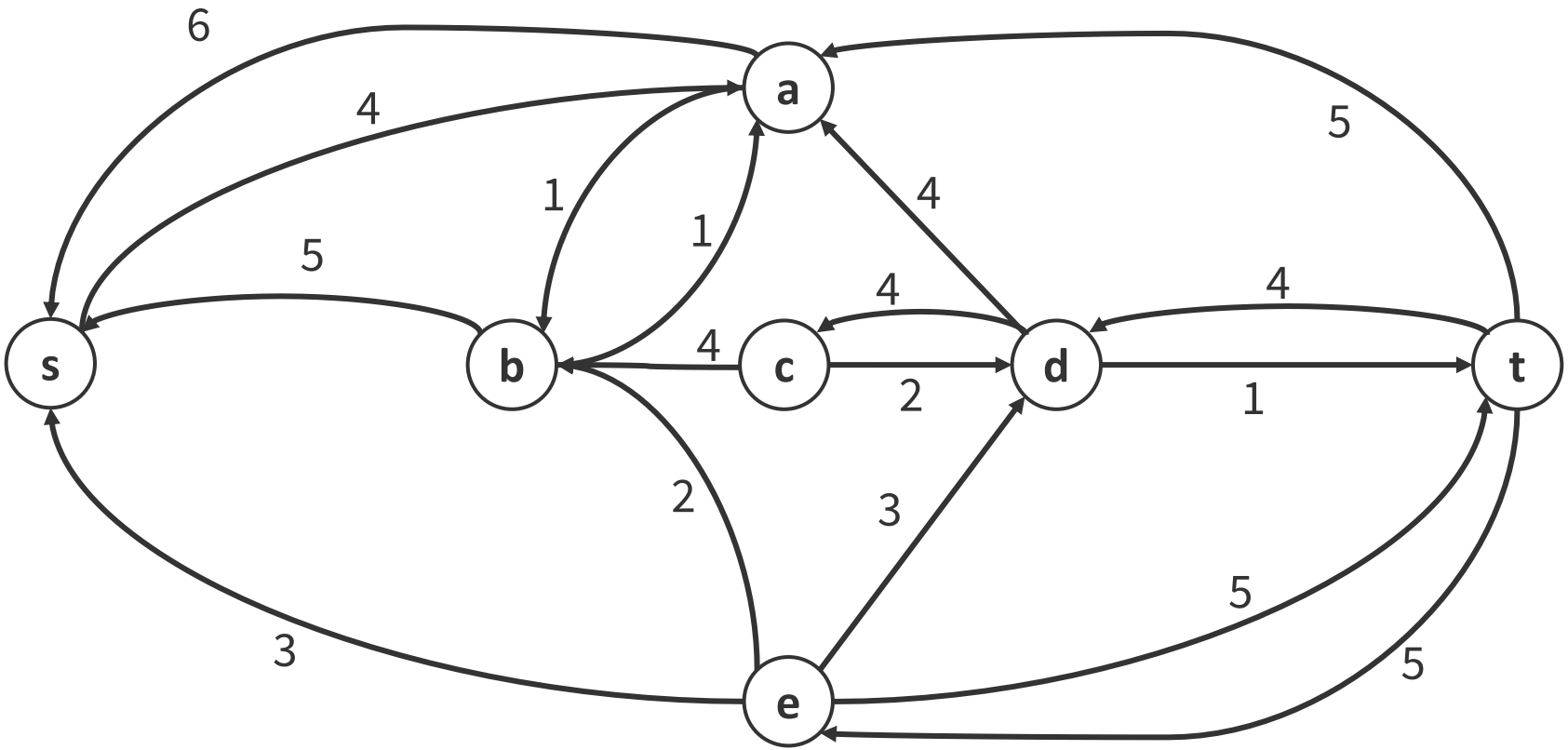
Problem 1 – Go With the Flow

Update the residual edges



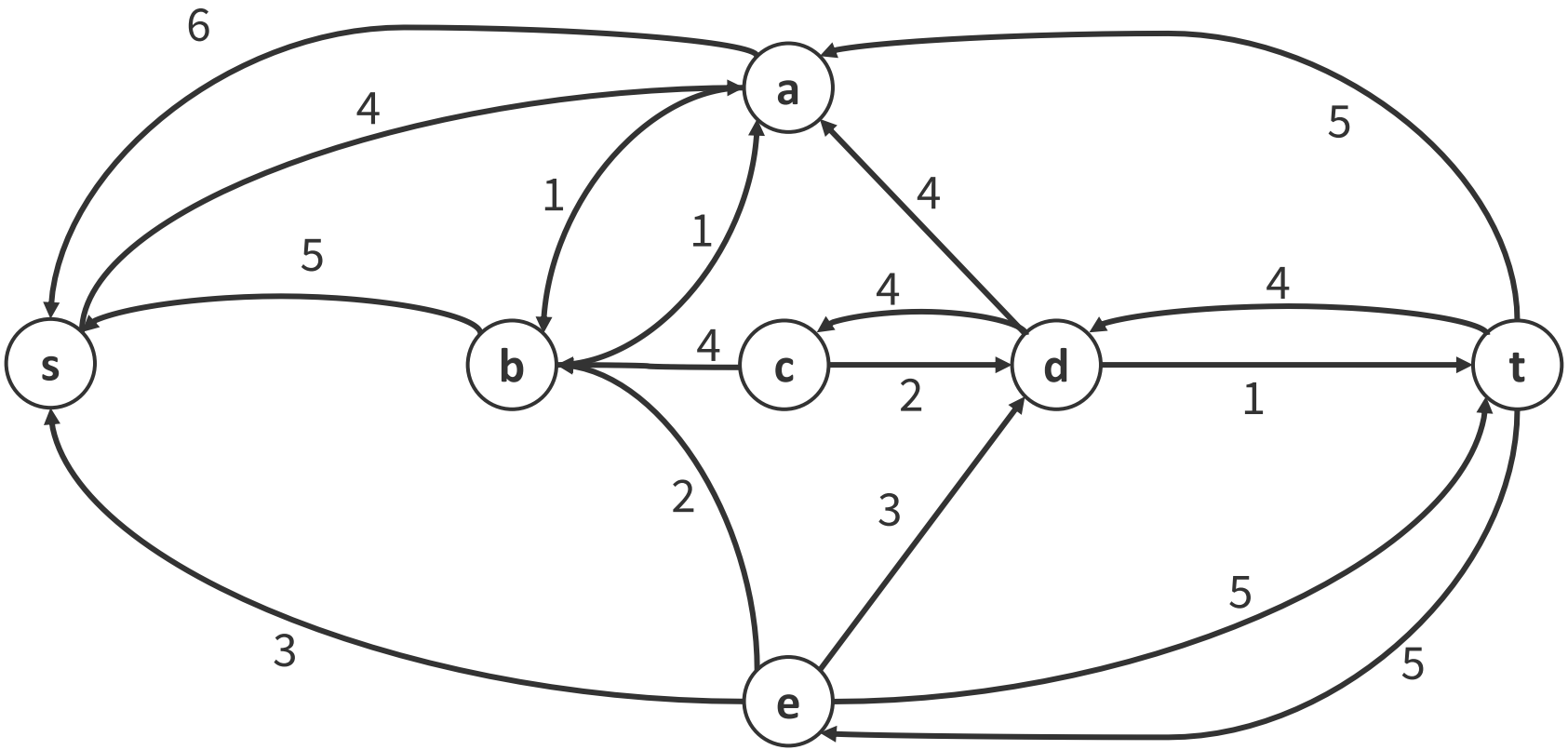
Problem 1 – Go With the Flow

Are there still any paths from s to t?



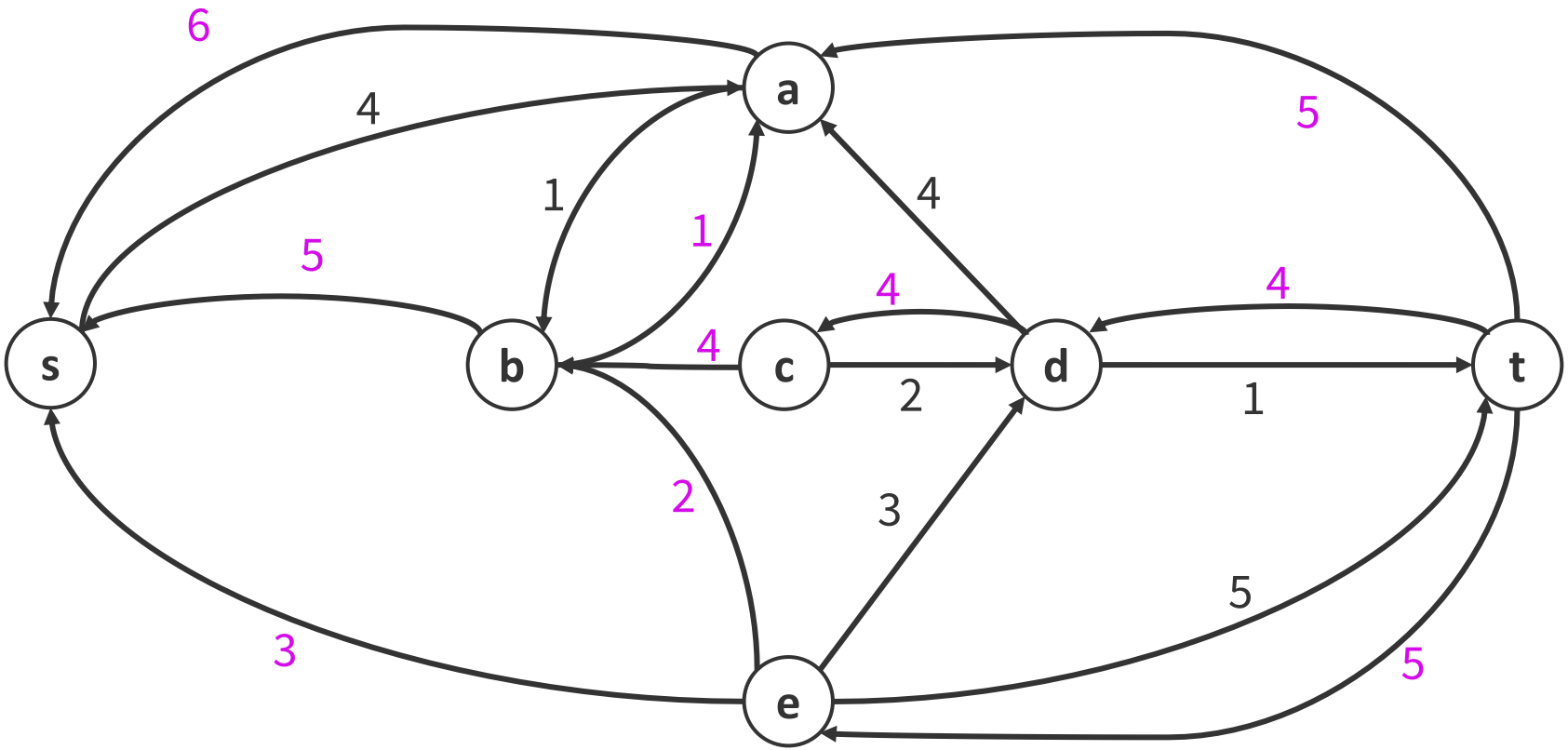
Problem 1 – Go With the Flow

No more paths, so we're done updating!



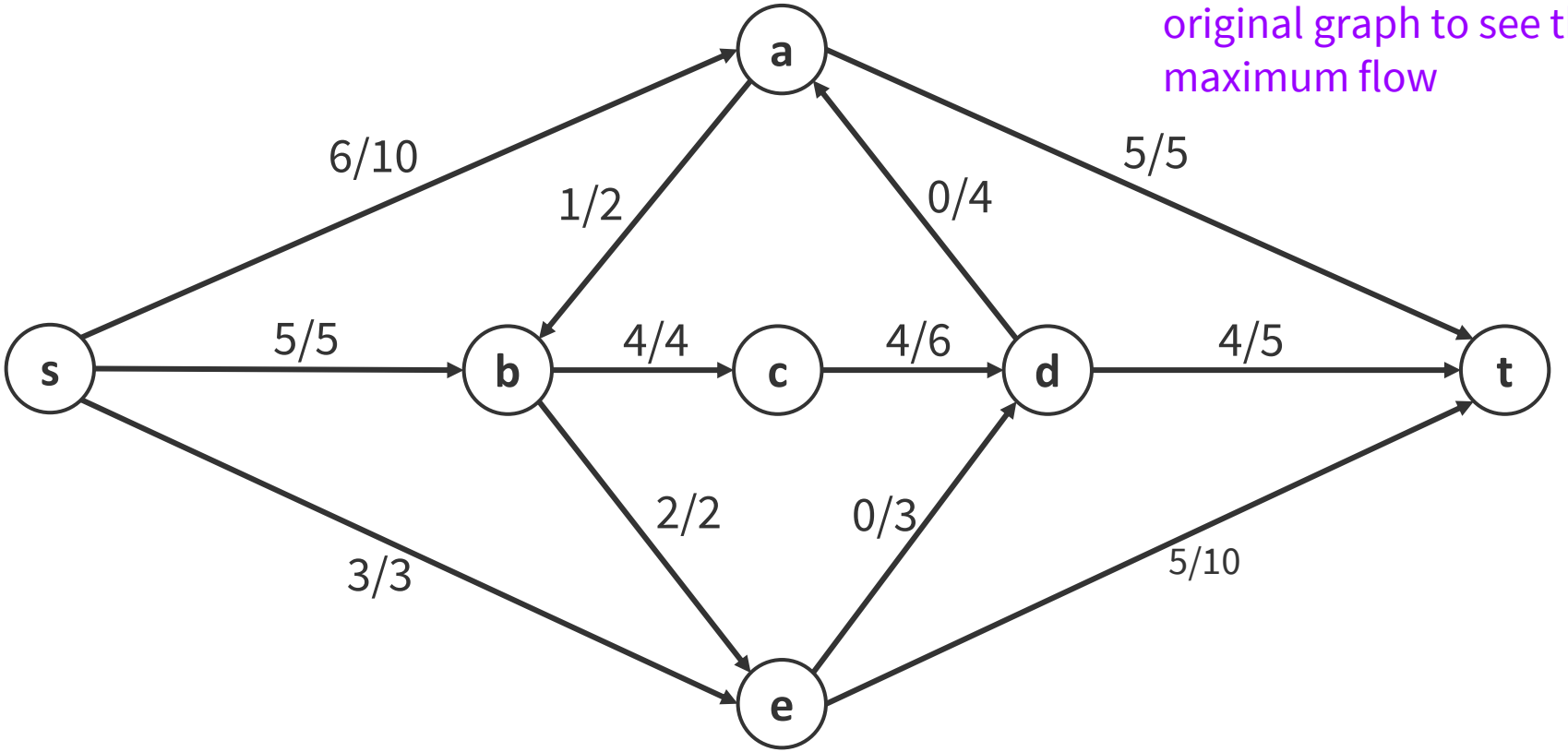
Problem 1 – Go With the Flow

All the residual edges going backwards show the flow we are sending down that path



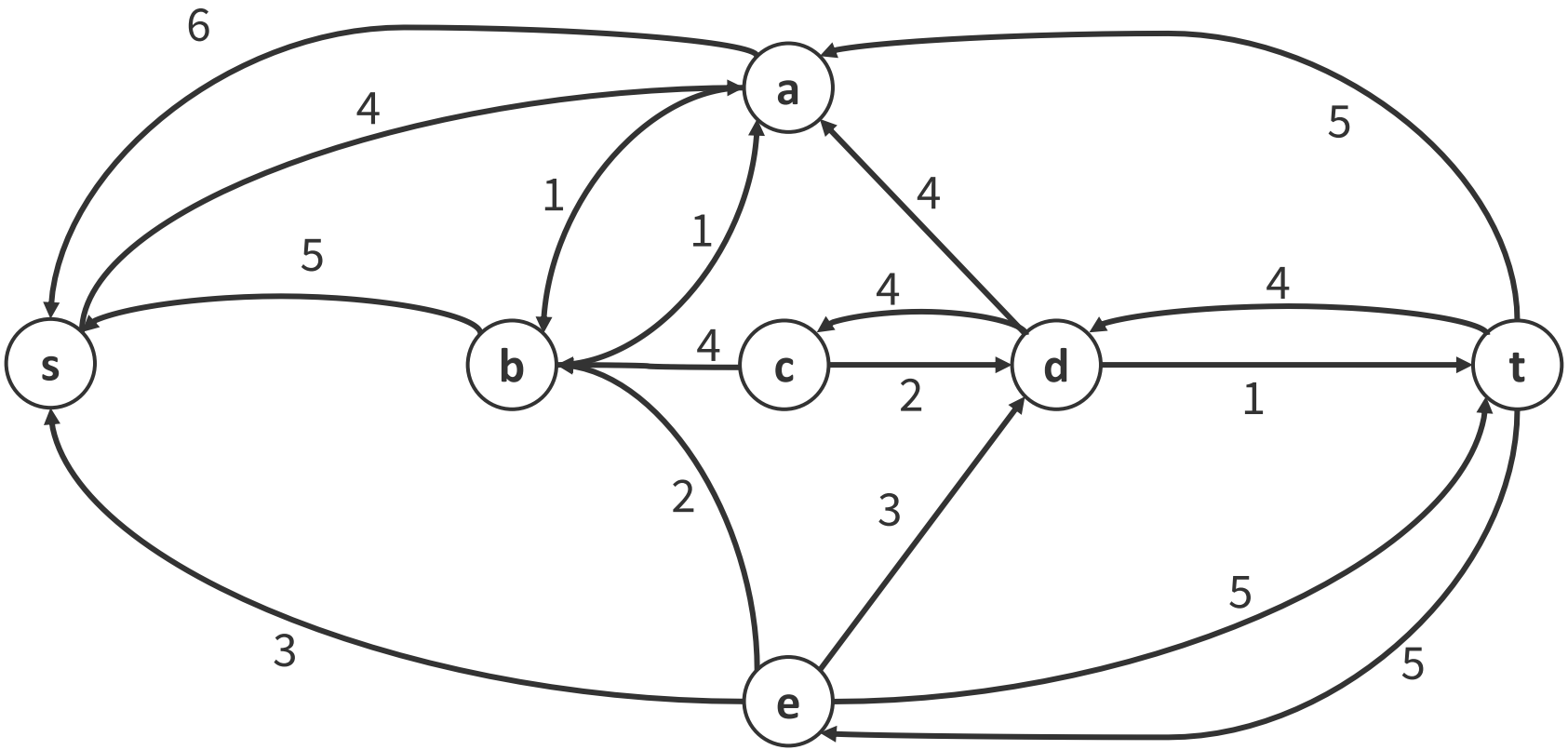
Problem 1 – Go With the Flow

We can put all these final flow values back into our original graph to see the maximum flow



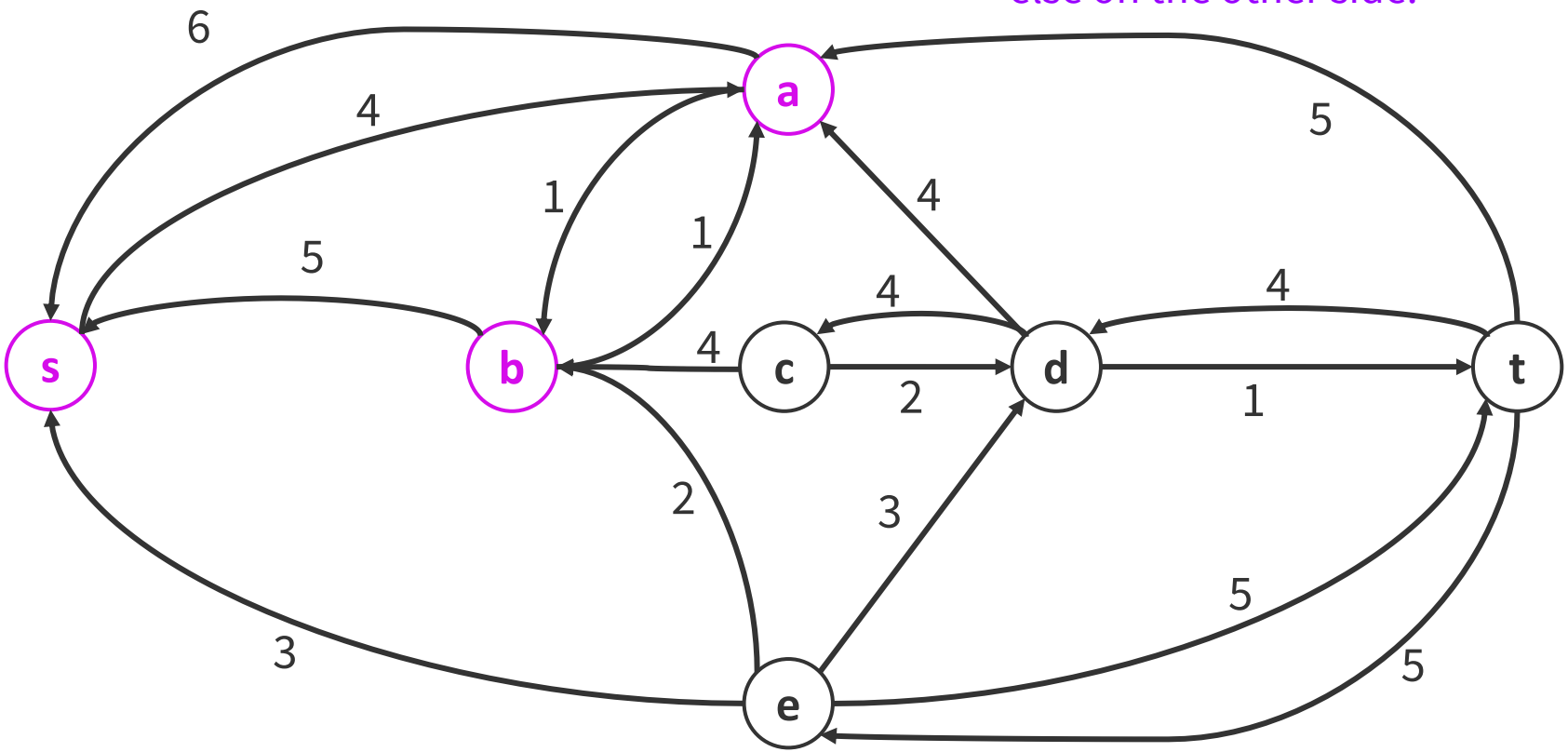
Problem 1 – Go With the Flow

Now let's find the min cut.



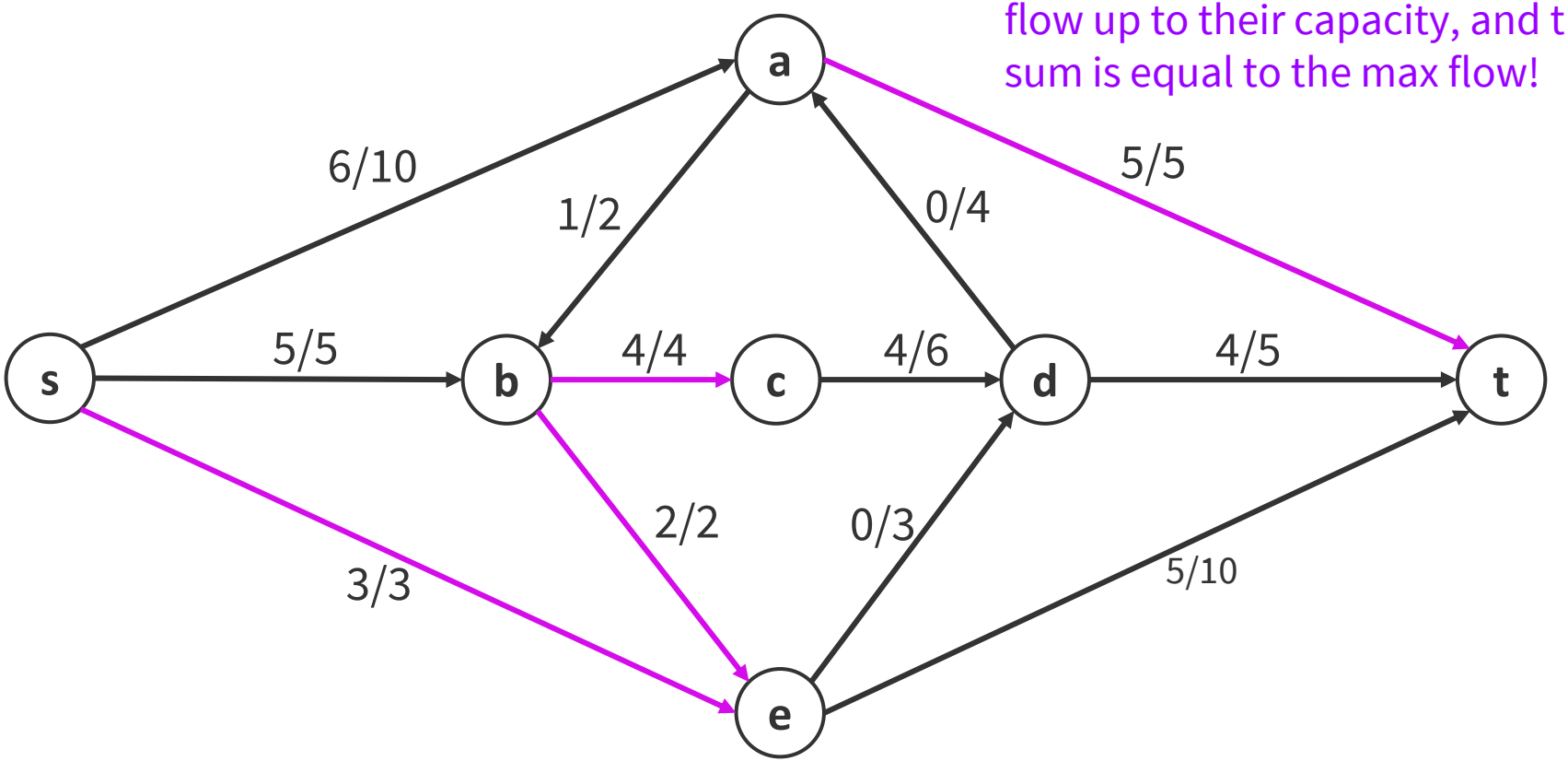
Problem 1 – Go With the Flow

The min cut is s and the vertices you can reach from it in the residual graph on one side, and everything else on the other side.



Problem 1 – Go With the Flow

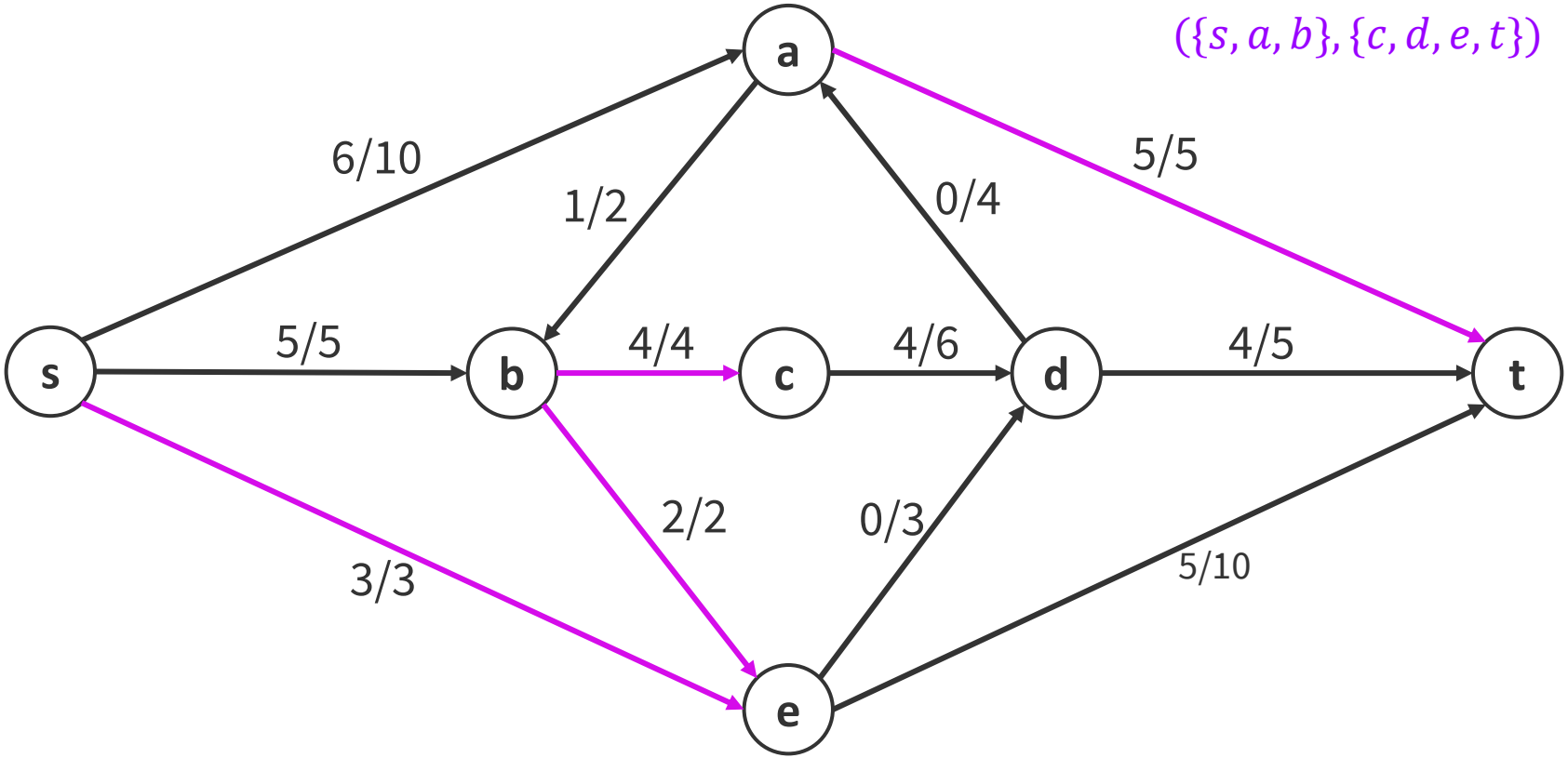
The edges going across the min cut from the s side to the t side all have flow up to their capacity, and the sum is equal to the max flow!



Problem 1 – Go With the Flow

The maximum flow is 14

The $s - t$ cut is $(\{s, a, b\}, \{c, d, e, t\})$



Max-Flow / Min-Cut Tricks

Max-Flow / Min-Cut

We can use the concepts of Max-Flow / Min-Cut and the Ford-Fulkerson algorithm to solve a wide variety of problems. Since we already have an algorithm, we can just call it like a library function.

Most of the difficulty comes in taking a problem and turning it into a good graph so that max-flow / min-cut gives us the solution we are actually looking for. So how can we do it?

The Strategy

1. Read the Problem Carefully
2. Make a Basic Model
3. Brainstorm: How can you fix the graph?
4. Correctness and Running Time

The Tricks

We have three tricks that can be really helpful in converting a problem into a good form for max-flow or min-cut. Sometimes you only need one, but sometimes you can use them in a combination. There are other things you might need to do in a given problem, but these are three very common tricks to try:

- Add “dummy vertices” for source or sink
- Split vertices to add vertex capacity
- Use infinite weight for edges that shouldn’t be considered for max-flow or min-cut

2. You’re Not a Dummy…

Problem 2 – You're Not a Dummy…

You have three overfilled reservoirs and two underfilled reservoirs. You want to (as quickly as possible) move a total of 10,000 gallons of water from the overfilled reservoirs to the underfilled ones. You do not care how much comes from each of the three individual reservoirs (as long as the total is 10,000 gallons) nor how much arrives at each of the underfilled ones (again, as long as the total is 10,000 gallons). You have a map of (one-way) pipes connecting the reservoirs (in the form of a directed graph); each pipe has a maximum capacity in gallons per minute. **You wish to find the way to route the water and the amount of time that will be required.**

Problem 2.1 – Read the Problem Carefully

Answer the usual quick-check questions:

- Are there any technical terms in the problem you don’t know? Are there any words that look like normal words, but are actually technical terms?

- What is the input type?

- What is the output type?

Work through this problem with the people around you, and then we’ll go over it together!

Problem 2.1 – Read the Problem Carefully

Answer the usual quick-check questions:

- Are there any technical terms in the problem you don’t know? Are there any words that look like normal words, but are actually technical terms?

- What is the input type?

- What is the output type?

Problem 2.1 – Read the Problem Carefully

Answer the usual quick-check questions:

- Are there any technical terms in the problem you don't know? Are there any words that look like normal words, but are actually technical terms?

One-way pipes: directed edges on our graph
Maximum capacity: the maximum amount of flow in gallons per minute for each pipe

- What is the input type?

- What is the output type?

Problem 2.1 – Read the Problem Carefully

Answer the usual quick-check questions:

- Are there any technical terms in the problem you don't know? Are there any words that look like normal words, but are actually technical terms?

One-way pipes: directed edges on our graph
Maximum capacity: the maximum amount of flow in gallons per minute for each pipe

- What is the input type?

Input: a graph with reservoirs as vertices and one-way pipes as edges.

- What is the output type?

Problem 2.1 – Read the Problem Carefully

Answer the usual quick-check questions:

- Are there any technical terms in the problem you don't know? Are there any words that look like normal words, but are actually technical terms?

One-way pipes: directed edges on our graph
Maximum capacity: the maximum amount of flow in gallons per minute for each pipe

- What is the input type?

Input: a graph with reservoirs as vertices and one-way pipes as edges.

- What is the output type?

Output: a flow network indicating the path the water should flow, with the max flow (fastest way to move the water from overfull reservoirs to underfull reservoirs)

Problem 2.2 – Make a Basic Model

This sounds like a flow problem. From what you know so far, what would the flow model be? What parts of the problem have you represented successfully? What is still missing?

Work through this problem with the people around you, and then we'll go over it together!

Problem 2.2 – Make a Basic Model

This sounds like a flow problem. From what you know so far, what would the flow model be? What parts of the problem have you represented successfully? What is still missing?

Problem 2.2 – Make a Basic Model

This sounds like a flow problem. From what you know so far, what would the flow model be? What parts of the problem have you represented successfully? What is still missing?

We use the map as our graph, with the pipes being directed edges and the capacities of the pipes being the edge capacities. But we don't have a single source or sink – we have three possible sources and two possible sinks.

We'll also have to convert our flow, which will be in gallons/minute to a time, but that will just be doing some division. The main problem is the multiple sources and sinks.

Problem 2.3 – Brainstorm: How can You Fix It?

Find a clever trick to represent the missing piece. The goal here is to do a reduction. By the end of this step, you should have a “standard” flow problem.

Work through this problem with the people around you, and then we’ll go over it together!

Problem 2.3 – Brainstorm: How can You Fix It?

Find a clever trick to represent the missing piece. The goal here is to do a reduction. By the end of this step, you should have a “standard” flow problem.

Problem 2.3 – Brainstorm: How can You Fix It?

Find a clever trick to represent the missing piece. The goal here is to do a reduction. By the end of this step, you should have a "standard" flow problem.

Since we don't care **which** reservoirs water is coming from or to, we can treat each of them as "one unit." Add an additional vertex to represent this "combined source" and add an infinite capacity edge from the new source to each of the three source reservoirs. Similarly add an additional vertex for the "combined sink" with infinite capacity edges from the target reservoirs to the new vertex. The combined vertices will be the source and sink for the max-flow.

These combined vertices are called "dummy" vertices. They don't represent anything "real" in the problem, the way the other vertices and edges do. But they let us make this different-looking problem into a standard flow problem. Dummy vertices are a common trick in flow and path-finding problems.

Our algorithm is then just to run any max-flow algorithm on our graph, and then calculate $\frac{10{,}000}{f}$ where f is the value of the maximum-flow.

Problem 2.4 – Correctness and Run Time

Explain why your algorithm is correct. For flow problems, the proof is usually just explaining how you've represented each part of the problem and relying on the correctness of the flow algorithm.

Work through this problem with the people around you, and then we'll go over it together!

Problem 2.4 – Correctness and Run Time

Explain why your algorithm is correct. For flow problems, the proof is usually just explaining how you've represented each part of the problem and relying on the correctness of the flow algorithm.

Problem 2.4 – Correctness and Run Time

Explain why your algorithm is correct. For flow problems, the proof is usually just explaining how you've represented each part of the problem and relying on the correctness of the flow algorithm.

We respect the capacities of the pipes, as they are encoded as edge capacities; each source and sink reservoir will behave as a source/sink via the added edge, and we don't limit the flow because the added edges are infinite capacity. Finally, we will get a maximum-flow by the correctness of the algorithm. We can move 10,000 gallons in $\frac{10{,}000}{f}$ minutes. And we cannot move that much water any faster, as some flow in that time would have greater movement than the maximum flow.

3. Split Personality

Problem 3 – Split Personality

You have been given a map of the water cleaning system for the city of Seattle. Water enters from a marked vertex and flows through pipes (directed edges with specified capacities), through processing facilities, and back out to nature (marked as a specified sink vertex). The processing facilities are vertices in your graph. As the facilities process the water, they **also** have maximum capacities, which may be less than the total capacity entering or leaving the vertex. You wish to **find the amount of water that can flow through this network** while **respecting both the facility and pipe capacities**.

Problem 3.1 – Read the Problem Carefully

Answer the usual quick-check questions:

- Are there any technical terms in the problem you don’t know? Are there any words that look like normal words, but are actually technical terms?

- What is the input type?

- What is the output type?

Work through this problem with the people around you, and then we’ll go over it together!

Problem 3.1 – Read the Problem Carefully

Answer the usual quick-check questions:

- Are there any technical terms in the problem you don’t know? Are there any words that look like normal words, but are actually technical terms?

- What is the input type?

- What is the output type?

Problem 3.1 – Read the Problem Carefully

Answer the usual quick-check questions:

- Are there any technical terms in the problem you don’t know? Are there any words that look like normal words, but are actually technical terms?

Maximum capacity of facilities: another capacity in addition to normal edge capacities

- What is the input type?

- What is the output type?

Problem 3.1 – Read the Problem Carefully

Answer the usual quick-check questions:

- Are there any technical terms in the problem you don’t know? Are there any words that look like normal words, but are actually technical terms?

Maximum capacity of facilities: another capacity in addition to normal edge capacities

- What is the input type?

Input: a graph with reservoirs as vertices and one-way pipes as edges.

- What is the output type?

Problem 3.1 – Read the Problem Carefully

Answer the usual quick-check questions:

- Are there any technical terms in the problem you don’t know? Are there any words that look like normal words, but are actually technical terms?

Maximum capacity of facilities: another capacity in addition to normal edge capacities

- What is the input type?

Input: a graph with cleaning facilities as vertices and pipes as edges

- What is the output type?

Output: the maximum amount of water that can flow through the cleaning network

Problem 3.2 – Make a Basic Model

This sounds like a flow problem. From what you know so far, what would the flow model be? What parts of the problem have you represented successfully? What is still missing?

Work through this problem with the people around you, and then we'll go over it together!

Problem 3.2 – Make a Basic Model

This sounds like a flow problem. From what you know so far, what would the flow model be? What parts of the problem have you represented successfully? What is still missing?

Problem 3.2 – Make a Basic Model

This sounds like a flow problem. From what you know so far, what would the flow model be? What parts of the problem have you represented successfully? What is still missing?

Take the map as our graph, with the source, sink, edges and capacities as marked. Each facility will start as a vertex.

We have so far failed to represent the flow limits in each processing plant (i.e., the vertices).

Problem 3.3 – Brainstorm: How can You Fix It?

Find a clever trick to represent the missing piece. The goal here is to do a reduction. By the end of this step, you should have a “standard” flow problem.

Work through this problem with the people around you, and then we’ll go over it together!

Problem 3.3 – Brainstorm: How can You Fix It?

Find a clever trick to represent the missing piece. The goal here is to do a reduction. By the end of this step, you should have a “standard” flow problem.

Problem 3.3 – Brainstorm: How can You Fix It?

Find a clever trick to represent the missing piece. The goal here is to do a reduction. By the end of this step, you should have a “standard” flow problem.

We can only put capacities on edges, not vertices... but we could add an edge! We want to make sure there is only so much water flowing through a vertex, so make an edge to represent that. For each vertex v, split it into two vertices v_{in} and v_{out}. Every edge (u, v) is replaced by (u, v_{in}) and every edge (v, w) is replaced by (v_{out}, w). Finally, we add an edge $(v_{\text{in}}, v_{\text{out}})$ with capacity equal to the capacity given in the problem for that vertex.

Problem 3.4 – Correctness and Run Time

Explain why your algorithm is correct. For flow problems, the proof is usually just explaining how you've represented each part of the problem and relying on the correctness of the flow algorithm.

Work through this problem with the people around you, and then we'll go over it together!

Problem 3.4 – Correctness and Run Time

Explain why your algorithm is correct. For flow problems, the proof is usually just explaining how you've represented each part of the problem and relying on the correctness of the flow algorithm.

Problem 3.4 – Correctness and Run Time

Explain why your algorithm is correct. For flow problems, the proof is usually just explaining how you've represented each part of the problem and relying on the correctness of the flow algorithm.

Our algorithm will find a maximum-flow in the altered graph. Note that a flow in our altered graph will be valid; we still encode all edge capacities, we interpret the flow on $(v_{\text{in}}, v_{\text{out}})$ as the flow going through facility v. Since v_{in} has only one outgoing edge and v_{out} has only one incoming edge, this accurately represents the flow on that edge and thus represents the total amount flowing through v at any point, and the capacity is enforced.

That's All, Folks!

Thanks for coming to section this week!
Any questions?