IN4344 Advanced Algorithms

Lecture 7 – Dual Simplex, Complementary Slackness, Total unimodularity, Shortest paths, Max flow

Yuki Murakami

Delft University of Technology

September 27, 2023

The Dual Simplex Method

Very useful when **re-optimizing** an LP after adding a constraint

This will be done when we solve (M)ILPs

Example

Suppose we have solved the following problem to optimality:

$$\begin{array}{lrcrcrcl} \min & z & = & 2x_1 & + & x_2 & & \\ \text{s.t.} & & & x_1 & + & x_2 & \geq & 1 \\ & & & 3x_1 & + & 2x_2 & \leq & 6 \\ & & & & & x_1 \geq 0, & x_2 \geq 0\,. & \end{array}$$

The optimal Simplex-tableau is:

basis	$\bar{b}$	x_1	x_2	s_1	s_2
x_2	1	1	1	-1	
s_2	4	1		2	1
$-z$	-1	1		1	

Suppose we now add an extra constraint $x_2 \leq 1/2$.

Example

basis	$\bar{b}$	x_1	x_2	s_1	s_2
x_2	1	1	1	-1	
s_2	4	1		2	1
$-z$	-1	1		1	

Suppose we now add an extra constraint $x_2 \leq 1/2$.
Add a slack variable: $x_2 + s_3 = 1/2$, and express x_2 in non-basic variables:

$$x_1 + x_2 - s_1 = 1 \Rightarrow \quad x_2 = 1 - x_1 + s_1 \,.$$

The new constraint is:

$$-x_1 + s_1 + s_3 = -1/2 \,.$$

Add it to the Simplex tableau:

Example

basis	$\bar{b}$	x_1	x_2	s_1	s_2
x_2	1	1	1	-1	
s_2	4	1		2	1
$-z$	-1	1		1	

The new constraint is:

$$-x_1 + s_1 + s_3 = -1/2\,.$$

Add it to the Simplex tableau:

basis	$\bar{b}$	x_1	x_2	s_1	s_2	s_3
x_2	1	1	1	-1		
s_2	4	1		2	1	
s_3	$-1/2$	-1		1		1
$-z$	-1	1		1		

Example

basis	$\bar{b}$	x_1	x_2	s_1	s_2	s_3
x_2	1	1	1	-1		
s_2	4	1		2	1	
s_3	$-1/2$	-1		1		1
$-z$	-1	1		1		

This solution is infeasible in the primal problem!

But, all $\bar{c}_j \geq 0$, which we can interpret as dual feasibility.

We will pivot to **maintain dual feasibility**, and **obtain primal feasibility**.

We choose a basic variable i' with $\bar{b}_i < 0$ as a leaving basic variable. Here, s_3.

If $\bar{a}_{i'j} \geq 0$ for all j, then no feasible solution exists since $\sum_j \bar{a}_{i'j} x_j \geq 0$ in all feasible solutions, and $\bar{b}_{i'} < 0$.

Example

basis	$\bar{b}$	x_1	x_2	s_1	s_2	s_3
x_2	1	1	1	-1		
s_2	4	1		2	1	
s_3	$-1/2$	-1		1		1
$-z$	-1	1		1		

We choose a basic variable i' with $\bar{b}_i < 0$ as a leaving basic variable. Here, s_3.

If $\bar{a}_{i'j} \geq 0$ for all j, then no feasible solution exists since $\sum_j \bar{a}_{i'j} x_j \geq 0$ in all feasible solutions, and $\bar{b}_{i'} < 0$.

So, we **pivot only on elements $\bar{a}_{i'j} < 0$!** Here we only have one choice: $\bar{a}_{31} = -1$. x_1 **becomes entering basis variable**.

Example

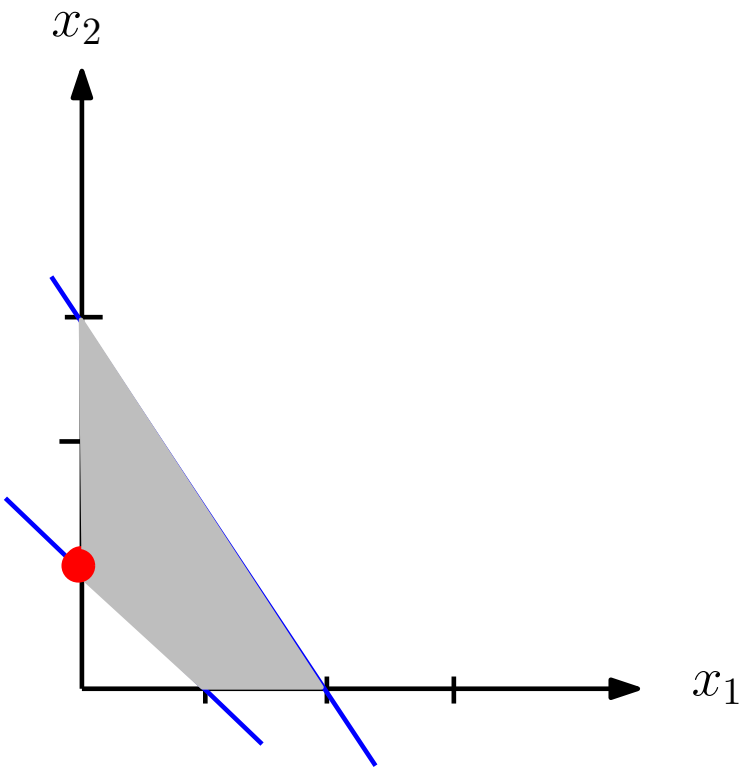
basis	$\bar{b}$	x_1	x_2	s_1	s_2	s_3
x_2	1	1	1	-1		
s_2	4	1		2	1	
s_3	$-1/2$	-1		1		1
$-z$	-1	1		1		

basis	$\bar{b}$	x_1	x_2	s_1	s_2	s_3	
x_2	$1/2$		1			1	$r_1 + r_3$
s_2	$7/2$			3	1	1	$r_2 + r_3$
x_1	$1/2$	1		-1		-1	$-1 \cdot r_3$
$-z$	$-3/2$			2		1	$r_0 + r_3$

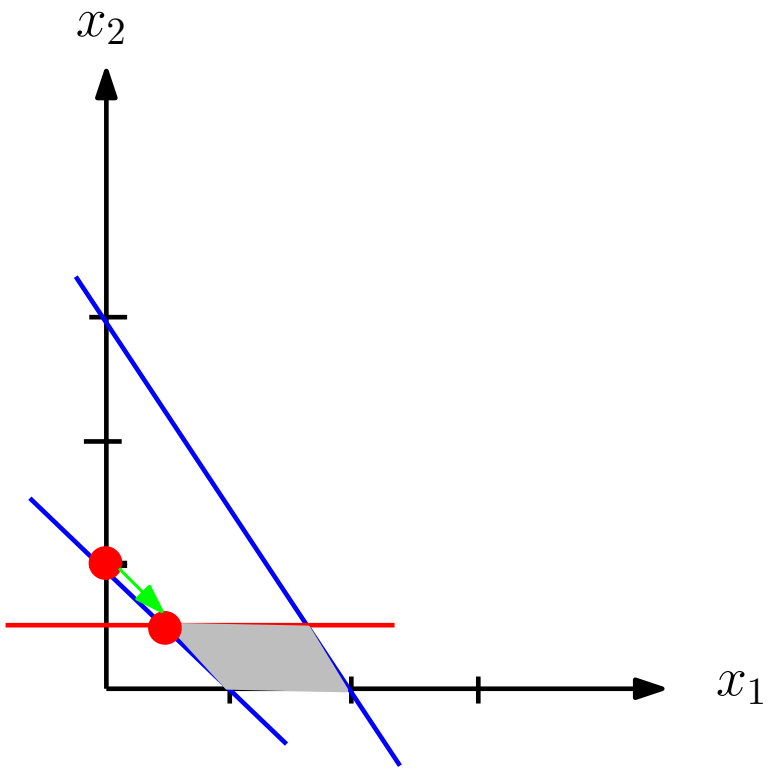
Now, $\bar{b}_i \geq 0$ for all i, and $\bar{c}_j \geq 0$ for all j, so we have a new optimal solution!

$$\begin{pmatrix} x_1^* \\ x_2^* \end{pmatrix} = \begin{pmatrix} 1/2 \\ 1/2 \end{pmatrix}, \quad \text{with } z^* = 3/2\,.$$

Graphically



Original problem



Problem after adding $x_2 \leq 1/2$

- **(Primal) Simplex method**
  - Go from **primal feasible** basic solution (bfs) to a next (not worse) bfs.
  - Stop when $\bar{c}_j \geq 0$ for all j (for a min problem), so when a solution is found that is **dual feasible**.

- **Dual Simplex method**
  - Go from **dual feasible** basic solution (a solution with correct values of the $\bar{c}_j$s) to a next (not worse) dual feasible basic solution.
  - Stop when $\bar{b}_i \geq 0$ for all i, so when a solution is found that is **primal feasible**.
  - You use the **primal** tableau.

Dual Simplex Algorithm for **min**imization problems:

Given is a basic solution, not necessary feasible, with $\bar{c}_j \geq 0$ for all j.

1. If $\bar{b}_i \geq 0$ for all i then the current solution is **feasible** and **optimal**. Stop!
2. Choose **leaving** basic variable corresponding to a **row** i' **with** $\bar{b}_{i'} < 0$.
3. If $\bar{a}_{i'j} \geq 0$ for all j then there is **no feasible solution**. Stop!
4. Choose **entering** variable $x_{j'}$ such that

$$\frac{\bar{c}_{j'}}{\bar{a}_{i'j'}} = \min_j \left\{ \left| \frac{\bar{c}_j}{\bar{a}_{i'j}} \right| \mid \bar{a}_{i'j} < 0 \right\}.$$

Why do we divide the objective function instead of the b column?

5. Apply elementary row operations such that column j' gets a 1 in row i' and 0s elsewhere. Go to (1).

Dual Simplex Algorithm for **max**imization problems:

Given is a basic solution, not necessary feasible, with $\bar{c}_j \leq 0$ for all j.

1. If $\bar{b}_i \geq 0$ for all i then the current solution is **feasible** and **optimal**. Stop!
2. Choose **leaving** basic variable corresponding to a **row** i' **with** $\bar{b}_{i'} < 0$.
3. If $\bar{a}_{i'j} \geq 0$ for all j then there is **no feasible solution**. Stop!
4. Choose **entering** variable $x_{j'}$ such that

$$\frac{\bar{c}_{j'}}{\bar{a}_{i'j'}} = \min_j \left\{ \left| \frac{\bar{c}_j}{\bar{a}_{i'j}} \right| \quad | \quad \bar{a}_{i'j} < 0 \right\}.$$

5. Apply elementary row operations such that column j' gets a 1 in row i' and 0s elsewhere. Go to (1).

Complementary Slackness

The **primal** LP (P):

$$\begin{aligned} \textbf{min} \quad & c^\mathsf{T} x & & \\ \text{s.t.} \quad & a_i x = b_i & i \in M \\ & a_i x \geq b_i & i \in \bar{M} \\ & x_j \geq 0 & j \in N \\ & x_j \in \mathbb{R} & j \in \bar{N} \end{aligned}$$

The **dual** LP (D):

$$\begin{aligned} \textbf{max} \quad & b^\mathsf{T} \pi & & \\ \text{s.t.} \quad & \pi_i \in \mathbb{R} & i \in M \\ & \pi_i \geq 0 & i \in \bar{M} \\ & \pi^\mathsf{T} A_j \leq c_j & j \in N \\ & \pi^\mathsf{T} A_j = c_j & j \in \bar{N} \end{aligned}$$

Theorem (Complementary Slackness)

Let $\hat{x}$ be a feasible solution of (P) and $\hat{\pi}$ a feasible solution of (D). Solutions $\hat{x}$ and $\hat{\pi}$ are both **optimal if and only if**:

$$\hat{\pi}_i(a_i \hat{x} - b_i) = 0 \qquad i = 1, \ldots, m \qquad (1)$$

$$\hat{x}_j(c_j - \hat{\pi}^\mathsf{T} A_j) = 0 \qquad j = 1, \ldots, n \qquad (2)$$

Proof

Follows from strong duality. See pages 71-72 in lecture notes.

Example (1. Diet Problem)

Potato chips, muesli, sausage, carbs, protein, fat.

$$
\begin{array}{lllll}
\text{(P): min} & 3x_1 + 3x_2 + 4x_3 & & \text{(D): max} & 3\pi_1 + 2\pi_2 + 9\pi_3 \\
\text{s.t.} & 2x_1 + x_2 \geq 3 & & \text{s.t.} & 2\pi_1 + 4\pi_3 \leq 3 \\
& x_2 + 4x_3 \geq 2 & & & \pi_1 + \pi_2 \leq 3 \\
& 4x_1 + 8x_3 \geq 9 & & & 4\pi_2 + 8\pi_3 \leq 4 \\
& x_1, x_2, x_3 \geq 0 & & & \pi_1, \pi_2, \pi_3 \geq 0
\end{array}
$$

Suppose we know an optimal solution $\pi^* = (1\frac{1}{2}, 1, 0)$ of the dual (D).
Use CS to find an optimal solution of the primal (P).

Formulate the CS conditions:

$$
\begin{array}{llllllll}
\pi_1^*(2x_1^* & + & x_2^* & & & -3) & = & 0 \quad (1) \\
\pi_2^*(& & x_2^* & + & 4x_3^* & -2) & = & 0 \quad (2) \\
\pi_3^*(4x_1^* & & & + & 8x_3^* & -9) & = & 0 \quad (3)
\end{array}
$$

$$
\begin{array}{llllllll}
x_1^*(3 & - & 2\pi_1^* & & & - & 4\pi_3^*) & = & 0 \quad (4) \\
x_2^*(3 & - & \pi_1^* & - & \pi_2^* & &) & = & 0 \quad (5) \\
x_3^*(4 & & & - & 4\pi_2^* & - & 8\pi_3^*) & = & 0 \quad (6)
\end{array}
$$

Suppose we know an optimal solution $\pi^* = (1\frac{1}{2}, 1, 0)$ of the dual (D).
Use CS to find an optimal solution of the primal (P).
Formulate the CS conditions:

$$\begin{array}{llllllll}
\pi_1^*(2x_1^* & + & x_2^* & & & -3) & = & 0 \quad (1) \\
\pi_2^*(& & x_2^* & + & 4x_3^* & -2) & = & 0 \quad (2) \\
\pi_3^*(4x_1^* & & & + & 8x_3^* & -9) & = & 0 \quad (3)
\end{array}$$

$$\begin{array}{llllllll}
x_1^*(3 & - & 2\pi_1^* & & & - & 4\pi_3^*) & = & 0 \quad (4) \\
x_2^*(3 & - & \pi_1^* & - & \pi_2^* & &) & = & 0 \quad (5) \\
x_3^*(4 & & & - & 4\pi_2^* & - & 8\pi_3^*) & = & 0 \quad (6)
\end{array}$$

Since we know that $\pi_1^*, \pi_2^* \neq 0$, we know from (1) & (2) that

$$\begin{array}{llllll}
2x_1^* & + & x_2^* & & = & 3 \quad (7) \\
 & & x_2^* & + 4x_3^* & = & 2 \quad (8)
\end{array}$$

Insert the dual solution in the dual constraints. In (5), we see that
$3 - 1.5 - 1 = 0.5 \neq 0$, so $x_2^* = 0$.
Set $x_2^* = 0$ in (7) & (8), and we obtain $x_1^* = 3/2,\ x_3^* = 1/2$.

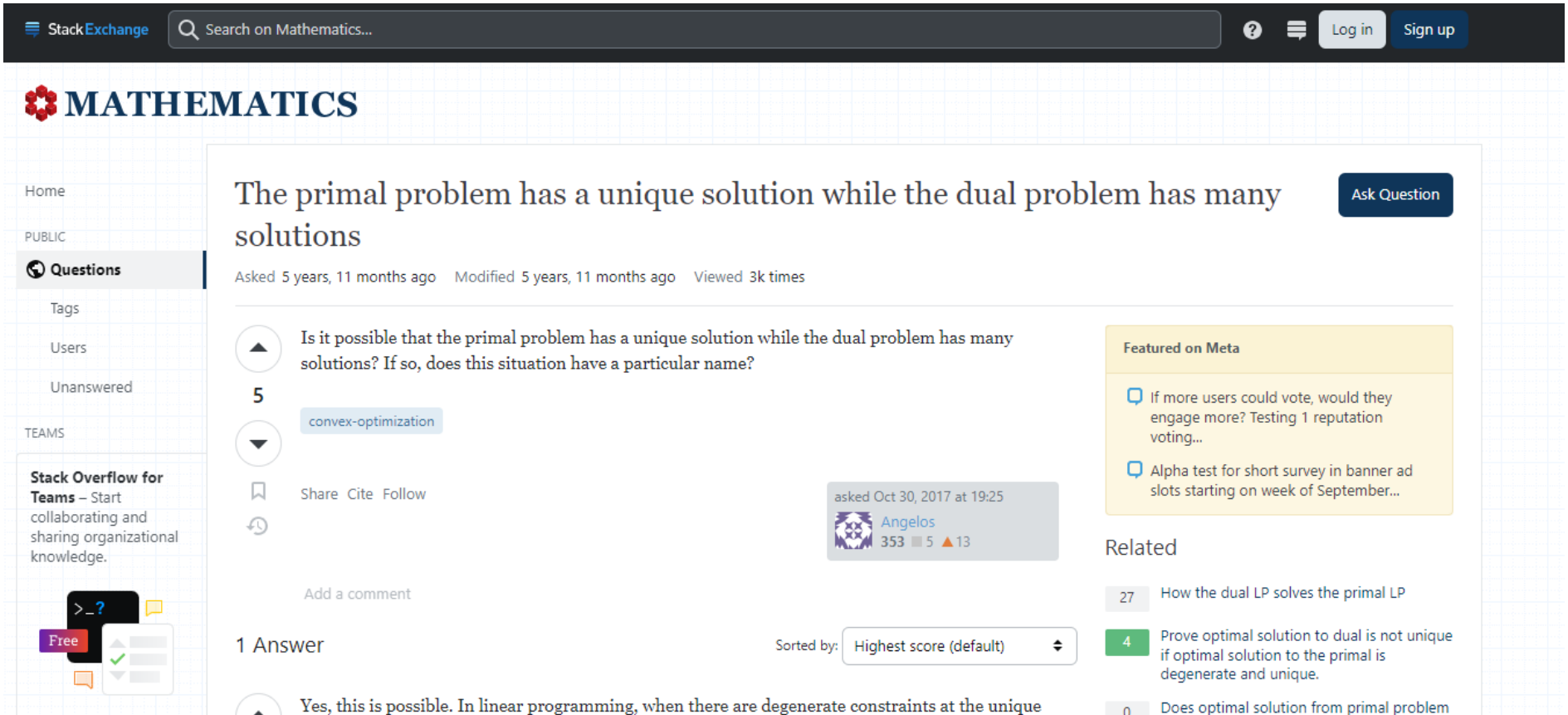
Check the objective values

$$
\begin{aligned}
\text{(P): min} \quad & z = 3x_1 + 3x_2 + 4x_3 \\
\text{s.t.} \quad & 2x_1 + x_2 \geq 3 \\
& x_2 + 4x_3 \geq 2 \\
& 4x_1 + 8x_3 \geq 9 \\
& x_1, x_2, x_3 \geq 0
\end{aligned}
\qquad
\begin{aligned}
\text{(D): max} \quad & w = 3\pi_1 + 2\pi_2 + 9\pi_3 \\
\text{s.t.} \quad & s.t.\ 2\pi_1 + 4\pi_3 \leq 3 \\
& \pi_1 + \pi_2 \leq 3 \\
& 4\pi_2 + 8\pi_3 \leq 4 \\
& \pi_1, \pi_2, \pi_3 \geq 0
\end{aligned}
$$

$$
\begin{pmatrix} x_1^* \\ x_2^* \\ x_3^* \end{pmatrix} = \begin{pmatrix} 3/2 \\ 0 \\ 1/2 \end{pmatrix}, \quad z^* = 6.5\,, \qquad \begin{pmatrix} \pi_1^* \\ \pi_2^* \\ \pi_3^* \end{pmatrix} = \begin{pmatrix} 3/2 \\ 1 \\ 0 \end{pmatrix}, \quad w^* = 6.5\,.
$$

Final word

- Adding new constraints to the primal problem corresponds to adding variables to the dual. This means that a feasible solution to the dual prior to adding the constraint remains feasible.
- Given an optimal solution to a primal, we can find an optimal solution to the dual (by CS).
- In general, one can have many optimal solutions for the dual



This lecture

Unimodular and totally unimodular matrices

Primal-Dual algorithms for:

- Shortest Path
  - Dijkstra
- Max Flow
  - Ford-Fulkerson

Unimodularity

Which ILPs can be solved by solving the LP-relaxation?

Definition

A square matrix B of integers is **unimodular (UM)** if $\det(B) = 1$ or $\det(B) = -1$.

Definition

A matrix A is **totally unimodular (TUM)** if all subdeterminants are equal to 0, $+1$ or -1. So, each square non-singular submatrix of A is unimodular.

As a consequence, a TUM matrix contains only elements from $\{0, -1, 1\}$.

Unimodularity

Which ILPs can be solved by solving the LP-relaxation?

Definition

A square matrix B of integers is **unimodular (UM)** if $\det(B) = 1$ or $\det(B) = -1$.

Definition

A matrix A is **totally unimodular (TUM)** if all subdeterminants are equal to 0, $+1$ or -1. So, each square non-singular submatrix of A is unimodular.

As a consequence, a TUM matrix contains only elements from $\{0, -1, 1\}$.

Question

Is $\begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ -1 & 1 & -1 \end{pmatrix}$ TUM? NO!

Unimodularity

Which ILPs can be solved by solving the LP-relaxation?

Definition

A square matrix B of integers is **unimodular (UM)** if $\det(B) = 1$ or $\det(B) = -1$.

Definition

A matrix A is **totally unimodular (TUM)** if all subdeterminants are equal to 0, $+1$ or -1. So, each square non-singular submatrix of A is unimodular.

As a consequence, a TUM matrix contains only elements from $\{0, -1, 1\}$.

Question

Is $\begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ -1 & 1 & -1 \end{pmatrix}$ TUM? NO!

Some results related to totally unimodular matrices

Theorem (12.1)

If an $m \times n$ matrix A is **totally unimodular** and $b \in \mathbb{Z}^m$, then all extreme points of the polyhedron

$$\{x \in \mathbb{R}^n \mid Ax = b, x \geq 0\}$$

are integral.

"is integral" means that all extreme points are integral.

Theorem (12.2)

If an $m \times n$ matrix A is **totally unimodular** and $b \in \mathbb{Z}^m$, then all extreme points of the polyhedron

$$\{x \in \mathbb{R}^n \mid Ax \leq b, x \geq 0\}$$

are integral.

So, if we have an ILP with feasible set $S = \{x \in \mathbb{Z}^n_{\geq 0} \mid Ax \leq b\}$ with A totally unimodular and $b \in \mathbb{Z}^m$, then we can solve this problem as an LP, since each extreme point of the LP-relaxation will be integral!

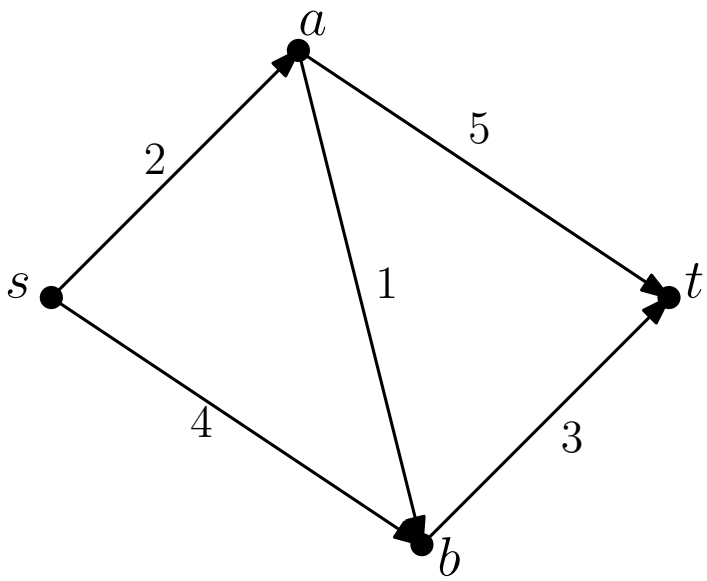
Problem

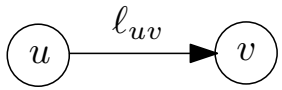
Shortest Path

Given: directed graph $D = (V, A)$ with length $\ell_a \geq 0$ for each arc $a \in A$ and two distinct vertices $s, t \in V$.

Find: a shortest path from s to t.

Formulate as an optimization problem





$$
\begin{array}{lrcrcrcrcrcr}
\min z = & 2f_{sa} & + & 4f_{sb} & + & f_{ab} & + & 5f_{at} & + & 3f_{bt} & & \\
\text{s.t.} & f_{sa} & + & f_{sb} & & & & & & & = & 1 \\
 & -f_{sa} & & & + & f_{ab} & + & f_{at} & & & = & 0 \\
 & & - & f_{sb} & - & f_{ab} & & & + & f_{bt} & = & 0 \\
 & & & & & & - & f_{at} & - & f_{bt} & = & -1
\end{array}
$$

$$f_a \in \{0, 1\}, \text{ for all } a \in A$$

Problem

Shortest Path

Given: directed graph $D = (V, A)$ with length $\ell_a \geq 0$ for each arc $a \in A$ and two distinct vertices $s, t \in V$.

Find: a shortest path from s to t.

The **node-arc incidence matrix** M of D is the $|V| \times |A|$ matrix with:

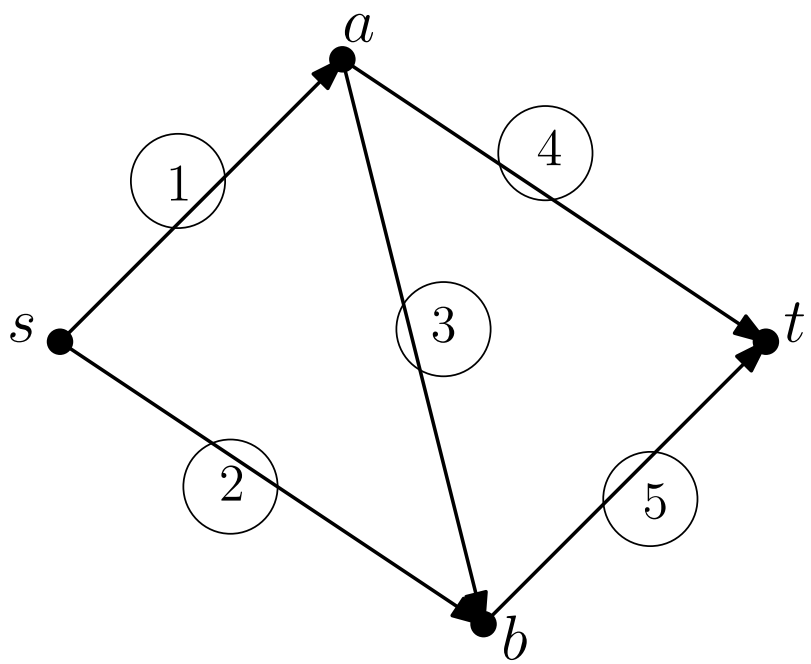
$$M_{va} = \begin{cases} 1 & \text{if vertex } v \text{ is the tail of arc } a \\ -1 & \text{if vertex } v \text{ is the head of arc } a \\ 0 & \text{otherwise.} \end{cases}$$

Decision variables:

$$f_a = \begin{cases} 1 & \text{if the path uses arc } a \\ 0 & \text{otherwise} \end{cases}$$

Example

The number within the circles that you see on the arcs just symbolize the arc number.



$$M = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ -1 & 0 & 1 & 1 & 0 \\ 0 & -1 & -1 & 0 & 1 \\ 0 & 0 & 0 & -1 & -1 \end{pmatrix}$$

Notice: **Each column has precisely one +1 and one -1.**

Formulate as an optimization problem

The **constraint matrix**, i.e., the node-arc incidence matrix of the graph, in **totally unimodular**.

Theorem (12.3)

A matrix A with $A_{ij} \in \{0, -1, 1\}$ is TUM **if**

- each column contains at most two non-zero elements; and
- the rows of A can be partitioned into two sets I_1 and I_2 s.t.:
  1. if a column contains two non-zero elements of the same sign, then the corresponding rows belong to different sets and
  2. if a column contains two non-zero elements of different signs, then the corresponding rows belong to the same set.

In case of the node-arc incidence matrix M, we let I_1 be the set of all rows, and $I_2 = \emptyset$.

Formulate as an optimization problem

Since the constraint matrix is TUM, and the right-hand side vector in integral, we can replace

$$f_a \in \{0, 1\}, \text{ for all } a \in A$$

by

$$0 \leq f_a \leq 1, \text{ for all } a \in A\,,$$

and view the shortest path problem as an LP!

If, in addition, all arc lengths ℓ_a are nonnegative, we know that all $f_a \leq 1$ in an optimal solution, so we can drop the constraints

$$f_a \leq 1, \text{ for all } a \in A\,.$$

Formulate the dual

$$
\begin{array}{llrcrcrcrcrcr}
\text{(P)} & \min z = & 2f_{sa} & + & 4f_{sb} & + & f_{ab} & + & 5f_{at} & + & 3f_{bt} & & \\
& \text{s.t.} & f_{sa} & + & f_{sb} & & & & & & & = & 1 \\
& & -f_{sa} & & & + & f_{ab} & + & f_{at} & & & = & 0 \\
& & & - & f_{sb} & - & f_{ab} & & & + & f_{bt} & = & 0 \\
& & & & & & & - & f_{at} & - & f_{bt} & = & -1 \\
& & & & & & f_a \geq 0, & \text{for all } & a \in A & & & &
\end{array}
$$

The primal has precisely one $+1$ and one -1 in each column, so the dual will have precisely one $+1$ and one -1 in each row!

$$
\begin{array}{lrcrcrcrcr}
\max w = & \pi_s & & & & & - & \pi_t & & \\
\text{s.t.} & \pi_s & - & \pi_a & & & & & \leq & 2 \\
& \pi_s & & & - & \pi_b & & & \leq & 4 \\
& & & \pi_a & - & \pi_b & & & \leq & 1 \\
& & & \pi_a & & & - & \pi_t & \leq & 5 \\
& & & & & \pi_b & - & \pi_t & \leq & 3 \\
& & & & \pi_v \in \mathbb{R}, & \text{for all } & v \in V & & &
\end{array}
$$

Primal:

$$\begin{aligned} \min \quad & \sum_{a \in A} \ell_a f_a \\ \text{s.t.} \quad & Mf = \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \\ -1 \end{pmatrix} \\ & f \geq 0 \end{aligned}$$

Dual:

$$\begin{aligned} \max \quad & \pi_s - \pi_t \\ \text{s.t.} \quad & M^{\mathsf{T}} \pi \leq \ell^T \\ & \pi \in \mathbb{R}^n \end{aligned}$$

Equivalently:

$$\begin{aligned} \max \quad & \pi_s - \pi_t \\ \text{s.t.} \quad & \pi_u - \pi_v \leq \ell_{uv} \quad \forall (u,v) \in A \\ & \pi \in \mathbb{R}^n \end{aligned}$$

Complementary Slackness:

$$\begin{aligned} f_{uv} > 0 \quad &\Rightarrow \quad \pi_u - \pi_v = \ell_{uv} \\ \pi_u - \pi_v < \ell_{uv} \quad &\Rightarrow \quad f_{uv} = 0 \end{aligned}$$

with notation: if $a = (u, v)$ then $f_a = f_{uv}$ and $\ell_a = \ell_{uv}$.
We can use Complementary slackness to verify correctness of Dijkstra's algorithm!

Dijkstra's Algorithm for finding a shortest path from s to t in a directed graph $D = (V, A)$ with length $\ell_{uv} \geq 0$ for each arc $(u, v) \in A$ and with $s, t \in V$.

Overview:

- Start with $W := \{s\}$ and add vertices, **one by one**, to W until $W = V$.
- Determine for each vertex v the value $\rho(v)$: the length of a shortest path from s to v that only uses vertices from $W \cup \{v\}$.
- Each iteration, add a $u \in V \setminus W$ with minimal $\rho(u)$ to W.

Observation

$\pi = -\rho$ is a feasible solution of the dual.
The value $\pi_s - \pi_t$ is equal to the length of the shortest path.

Dijkstra's Algorithm for finding a shortest path from s to t in a directed graph $D = (V, A)$ with length $\ell_{uv} \geq 0$ for each arc $(u, v) \in A$ and with $s, t \in V$.

Algorithm

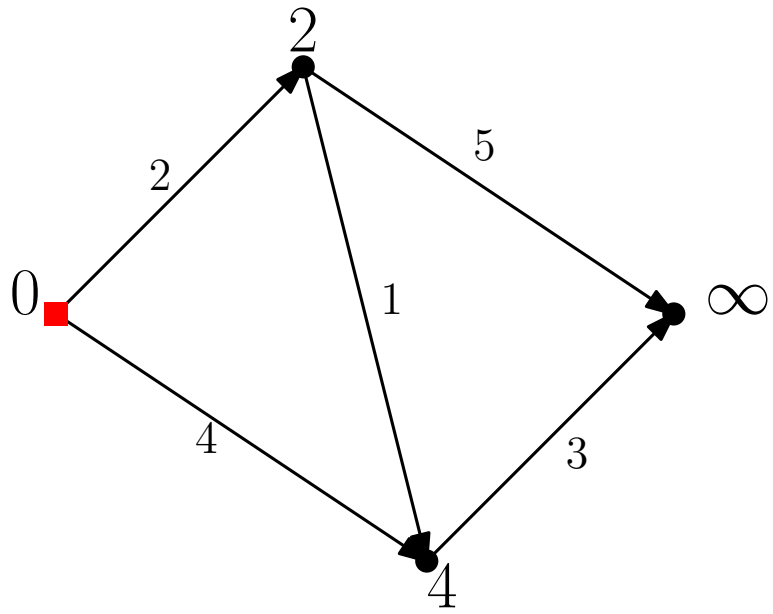
$W := \{s\}$
$\rho(s) := 0$
$\rho(v) := \ell_{sv}$ for each $v \in V \setminus \{s\}$
with $\ell_{sv} = \infty$ if $(s, v) \notin A$

Repeat until $W = V$:

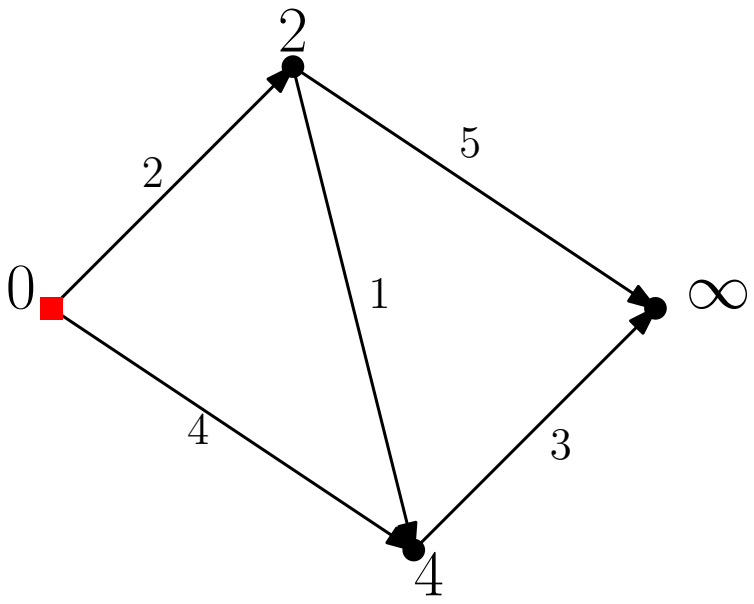
1. Find $u \in V \setminus W$ with minimal $\rho(u)$
2. $W := W \cup \{u\}$
3. For each $v \in V \setminus W$ with $(u, v) \in A$
   - $\rho(v) := \min\{\rho(v), \rho(u) + \ell_{uv}\}$

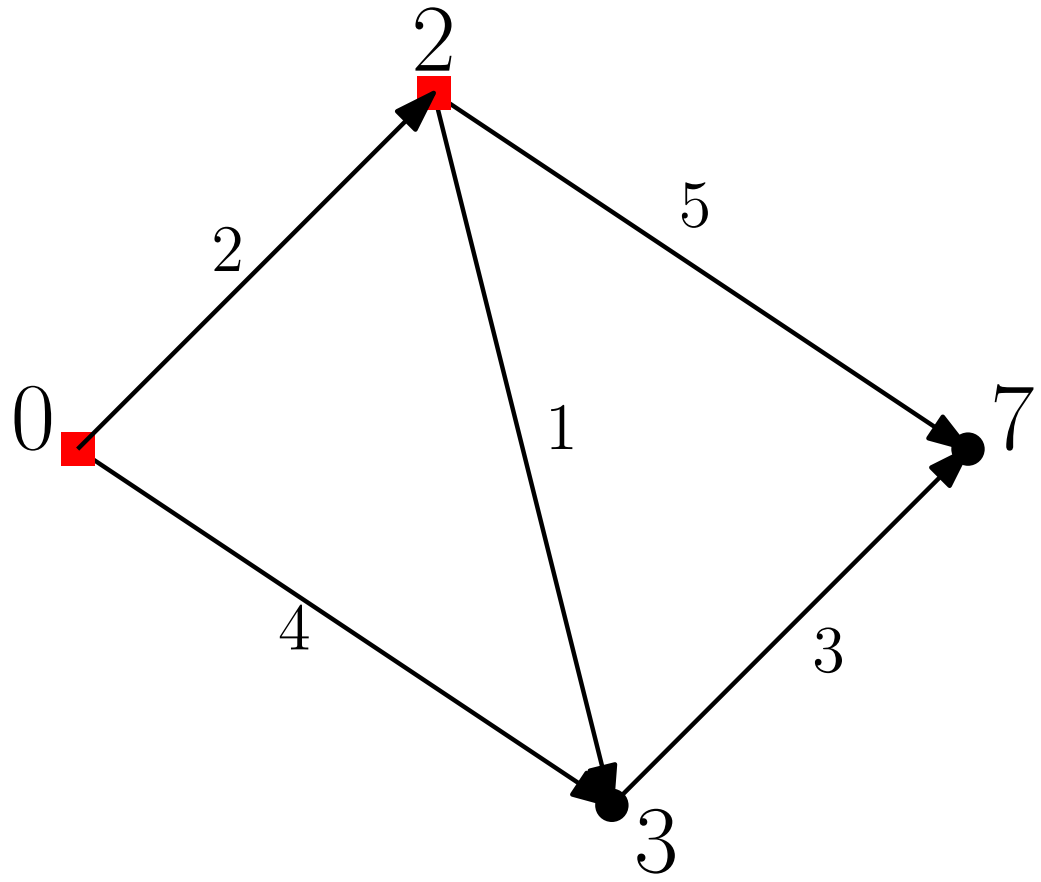
Example

After initialization, $W = \{s\}$ and the vertices have the following labels:



Now, we find the vertex $v \in V \setminus W$ with the smallest vertex label $\rho(v)$.
That is vertex a, so we let $W := W \cup \{a\}$.
For all $v \in V \setminus W$, we update the $\rho(v)$s.

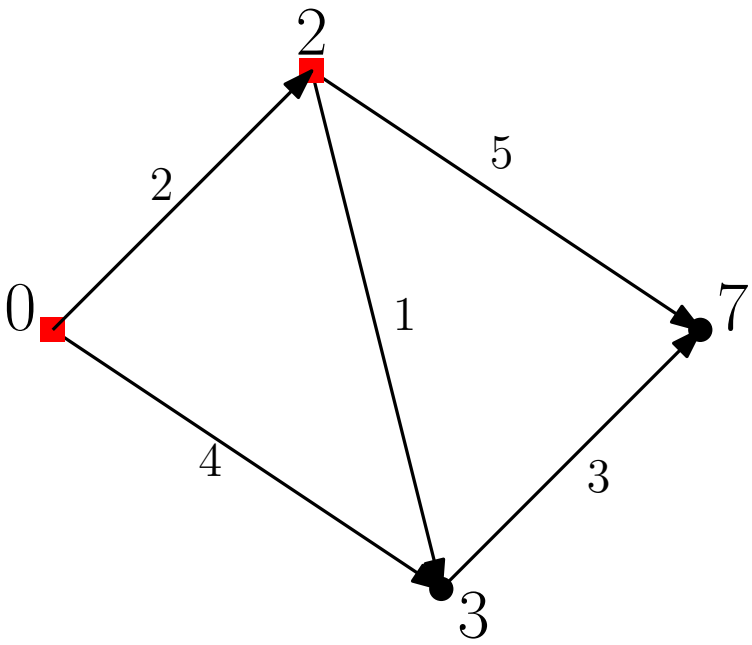


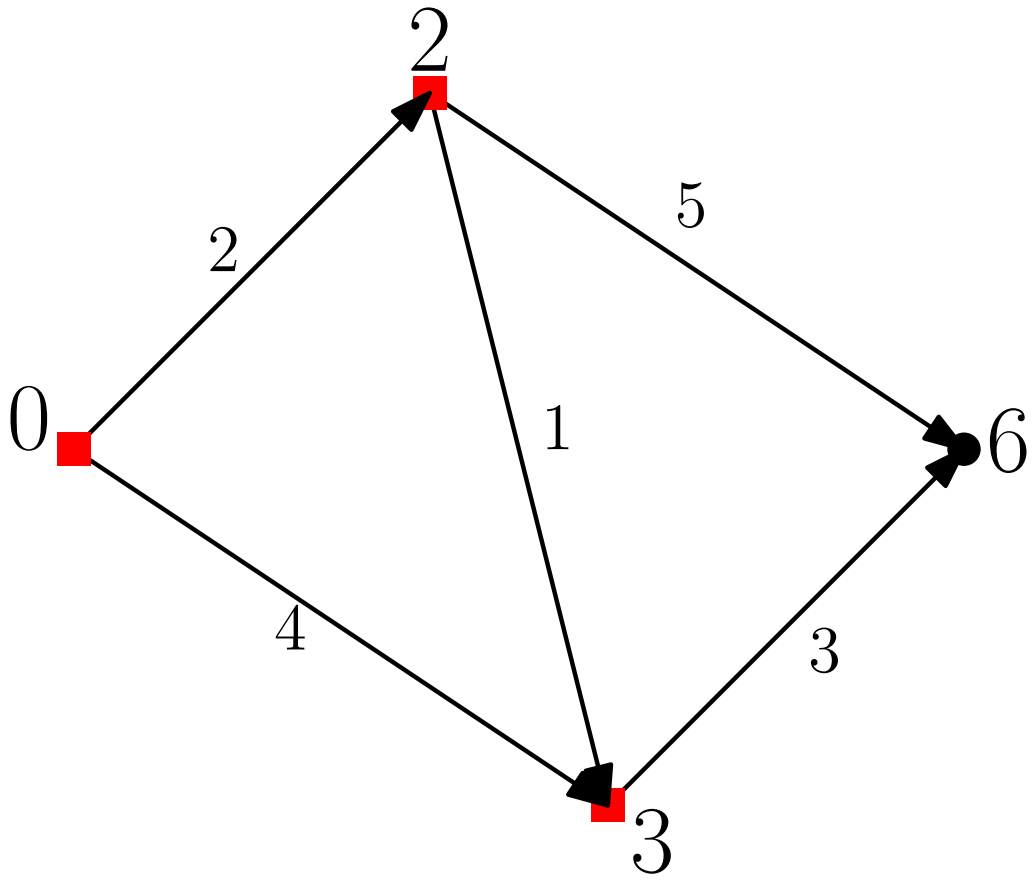


We can update both $\rho(b)$ and $\rho(t)$

$$\begin{aligned} \rho(b) &= \min\{4,\ \rho(a) + \ell_{(a,b)}\} = \min\{4,\ 2+1\} = 3 \\ \rho(t) &= \min\{\infty,\ \rho(a) + \ell_{(a,t)}\} = \min\{\infty,\ 2+5\} = 7\,. \end{aligned}$$

Since now vertex $b \in V \setminus W$ is the vertex outside W with the smallest vertex label, we add b to W, $W := W \cup \{b\}$.



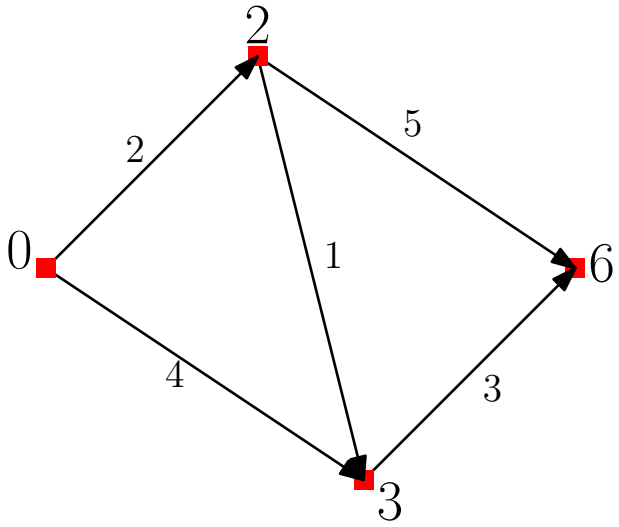


Now can update $\rho(t)$

$$\rho(t) = \min\{7,\ \rho(b) + \ell_{(b,t)}\} = \min\{7,\ 3+3\} = 6\,.$$

Since now vertex $t \in V \setminus W$ is the vertex outside W with the smallest vertex label, we add t to W, $W := W \cup \{t\}$.
Since $W = V$ we can stop! The shortest path from s to t is 6.

- The final value of $\rho(t)$ is the length of a shortest path from s to t.
- For a given s, the algorithm finds this distance for **all** $v \in V$.
- If you only want to find a shortest path from s to t, you can change the algorithm to **Repeat until** $t \in W$.
- What you probably have seen before:
  A shortest s-t path can be found by **backtracking** from t:
  - find an incoming arc (v, t) of t with $\rho(t) = \rho(v) + \ell_{vt}$
  - do the same for v
  - continue until you reach s.
- Now we can use **Complementary slackness** to argue a shortest path s-t:
  - arcs (u, v) with $\pi_u - \pi_v = \rho(v) - \rho(u) < \ell_{uv}$ cannot be on the path
  - find any path from s to t using only arcs with $\rho(v) - \rho(u) = \ell_{uv}$
  - this path has length $\rho(t) - \rho(s) = \rho(t)$ and is therefore optimal.



Verify at home that the dual solution $\pi(v) = -\rho(v)$ is feasible.
To find the actual path that gives length 6, we find the set of arcs that satisfies

$$\pi(u) - \pi(v) = \ell_{(u,v)}\,.$$

For these arcs, we *may* set $f_{(u,v)} = 1$.
pause In our case, the set of arcs for which this holds is precisely

$$\{(s,a),\ (a,b),\ (b,t)\}\,,$$

and since there is only one s-t path using these arcs, we set $f_{(u,v)} = 1$ for these arcs, and obtain the unique shortest path, namely *s-a-b-t*.

Run time of Dijkstra

1. How many loops do we have?

   We add precisely one vertex to the set W in each loop and stop when $W = V$. **So, we have $|V|$ loops**

2. What happens in each loop?

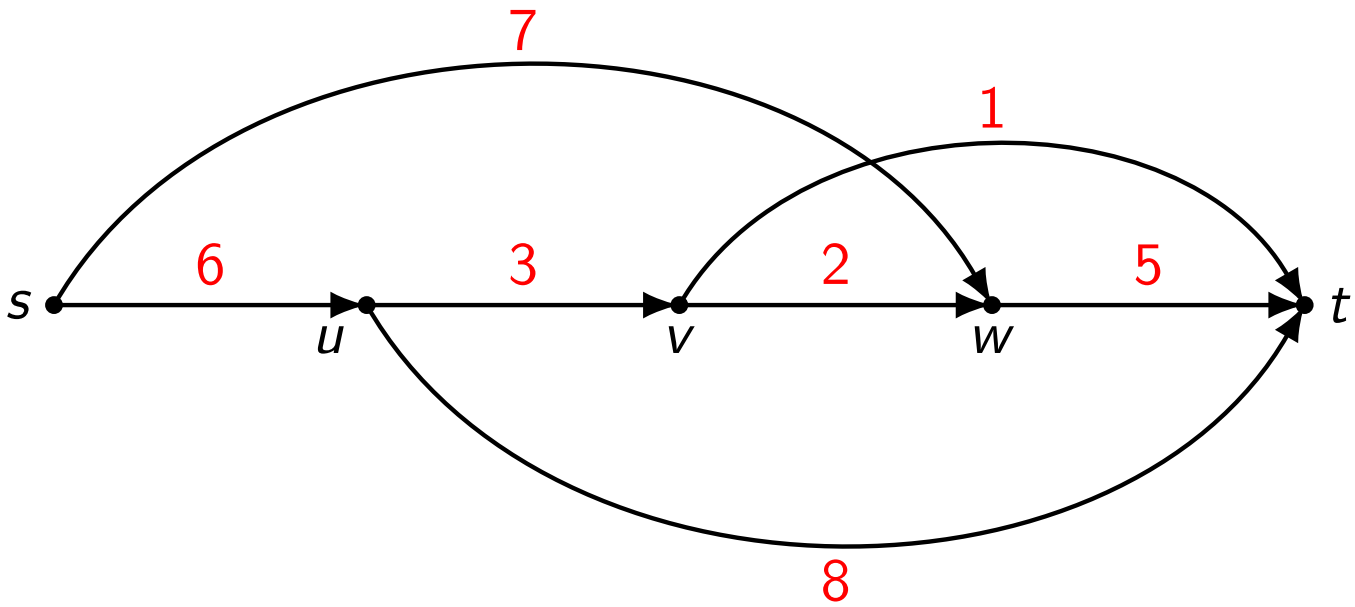
   We search for a vertex $u \in V \setminus W$ with minimal $\rho(u)$.
   There are at most $|V|$ elements to compare.
   Once we found u, we need to update the ρ-values, which costs $O(|V|)$ time.

Conclusion. The run time is $O(|V|^2)$, so polynomial.

Using clever data structures (Fibonacci heap), we can get the run time down to $O(|A| + |V| \log |V|)$

Max Flow

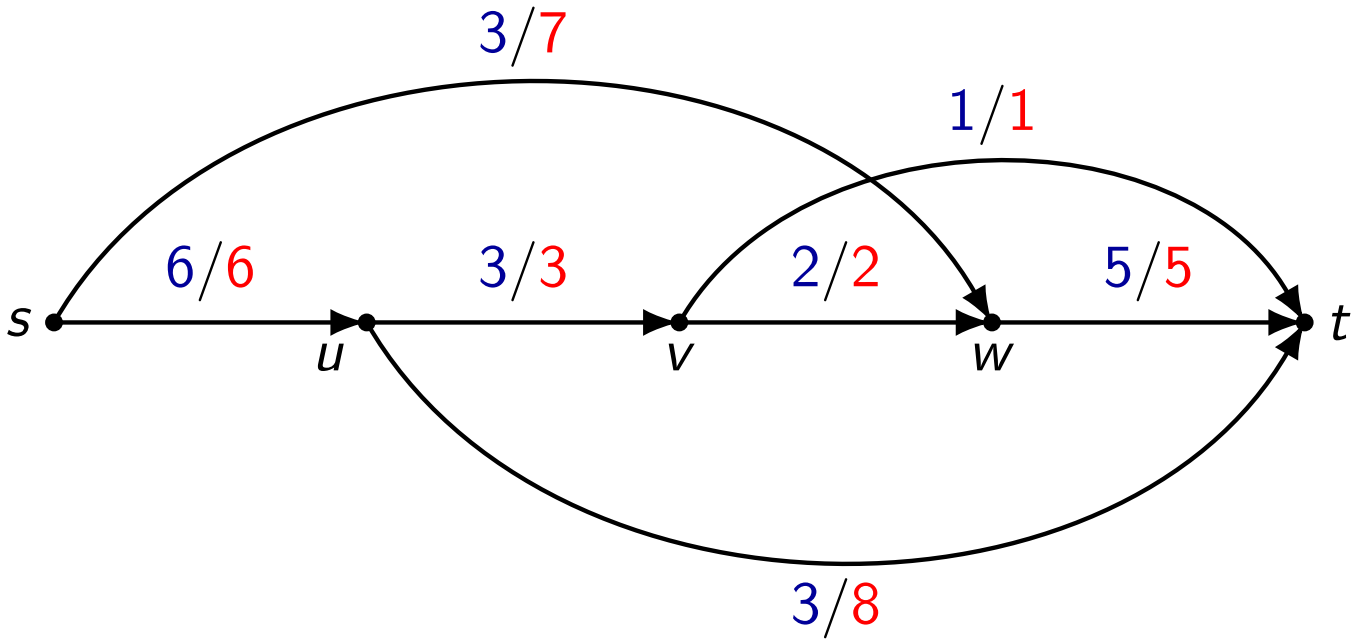
- Given is a directed graph $D = (V, A)$ with distinct vertices s and t.
- Each arc $(u, v) \in A$ has a certain **capacity** $b_{uv} \geq 0$.
- We want to find a largest possible **flow** from s to t,
- such that the **capacities** of the arcs are not exceeded,
- and there is **conservation of flow** at each vertex except s and t.



Directed graph with **capacities**.

Max Flow

- Given is a network with distinct vertices s and t.
- Each arc has a certain **capacity** $b_{uv} \geq 0$.
- We want to find a largest possible **flow** from s to t.
- Such that the **capacities** of the arcs are not exceeded
- and there is **conservation of flow** at each vertex except s and t.



Network with **flow** and **capacities**.

Mathematical Formulation

Problem

MAX FLOW

Given: directed graph $D = (V, A)$, vertices $s, t \in V$ and a capacity $b_{uv} \geq 0$, for each arc $(u, v) \in A$.

Find: a **value** $f_{uv} \geq 0$, for each arc $(u, v) \in A$ such that:

(i) there is **conservation of flow** in each vertex except s and t:

$$\sum_{(u,v)\in A} f_{uv} = \sum_{(v,w)\in A} f_{vw}, \quad \forall v \in V \setminus \{s, t\}$$

(ii) the flow does not exceed the **capacities** of the arcs:

$$f_{uv} \leq b_{uv}, \qquad \forall (u, v) \in A$$

(iii) the **net outflow** at s is **maximized**:

$$\sum_{(s,v)\in A} f_{sv} - \sum_{(u,s)\in A} f_{us} \qquad \text{is maximized.}$$

Observation

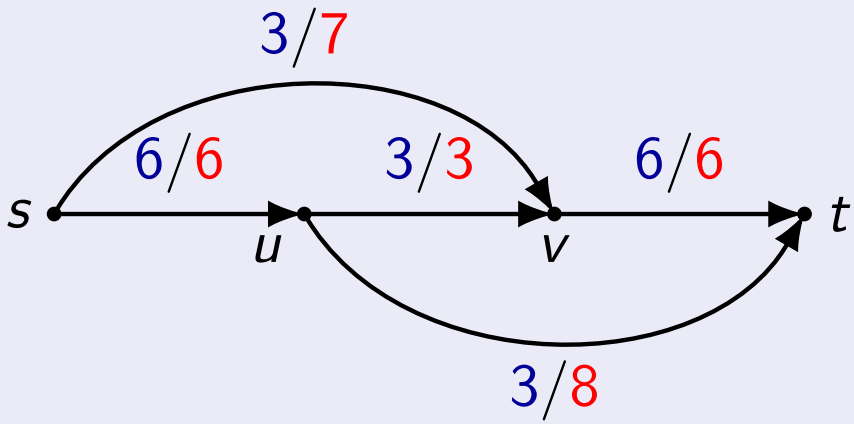
The **net outflow at s** is equal to the **net inflow at t**:

$$\sum_{(s,v)\in A} f_{sv} - \sum_{(u,s)\in A} f_{us} = \sum_{(u,t)\in A} f_{ut} - \sum_{(t,v)\in A} f_{tv}$$

This is the **value** of the flow.

Question (1)

What is the value of the flow indicated in blue in the network below?



Given is a directed graph $D = (V, a)$. Let M be the node-arc incidence matrix, and let b_{uv} be the capacity of arc $(u, v) \in A$.

Decision variables:

- f_{uv}: the flow on arc (u, v)
- w: the value of the flow from s to t

$$
\begin{array}{lrcl}
\max & & & w \\
\text{s.t} & Mf + \begin{pmatrix} -1 \\ 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix} w & = & 0 \\
& f & \leq & b \\
& f,\ w \in \mathbb{Z}_{\geq 0} & &
\end{array}
$$

We can view w as adding an extra arc on which the flow goes back from t to s.

The constraint matrix

Notice, the constraint matrix for the max flow problem is as follows:

$$\begin{pmatrix} M' \\ I \end{pmatrix},$$

where M' is the matrix M with the extra column

$$\begin{pmatrix} -1 \\ 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix},$$

which is still a node-arc incidence matrix, and hence TUM.

The constraint matrix

Theorem (12.4)

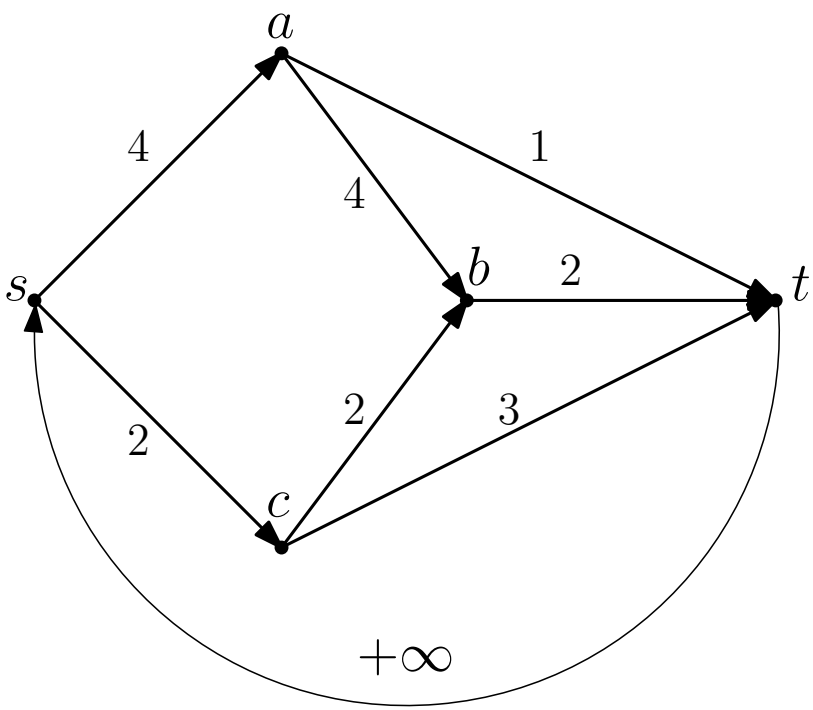
If A is TUM, then the matrix

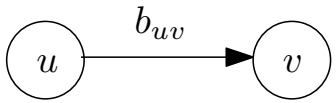
$$\begin{pmatrix} A \\ I \end{pmatrix}$$

is TUM.

So, the constraint matrix for Max flow is TUM and if the capacities are integral, then we can drop the integrality restrictions, and solve Max flow as an LP!

Example





$$
\begin{array}{lrcrcrcrcrcrcrcrcl}
\max & & & & & & & & & & & & & w & & \\
\text{s.t.} & f_{sa} & + & f_{sc} & & & & & & & & & - & w & = & 0 \\
& -f_{sa} & & & + & f_{ab} & & & f_{at} & & & & & & = & 0 \\
& & & & - & f_{ab} & - & f_{cb} & & + & f_{bt} & & & & = & 0 \\
& & - & f_{sc} & & & + & f_{cb} & + & & & + & f_{ct} & & = & 0 \\
& & & & & & & & - & f_{at} & - & f_{bt} & - & f_{ct} & + & w & = & 0 \\
& f_{sa} & & & & & & & & & & & & & \leq & 4 \\
& & & f_{sc} & & & & & & & & & & & \leq & 2 \\
& & & & & f_{ab} & & & & & & & & & \leq & 4 \\
& & & & & & & f_{cb} & & & & & & & \leq & 2 \\
& & & & & & & & & f_{at} & & & & & \leq & 1 \\
& & & & & & & & & & & f_{bt} & & & \leq & 2 \\
& & & & & & & & & & & & & f_{st} & \leq & 3
\end{array}
$$

$$
f_{uv} \geq 0, \text{ for all } (u, v) \in A, w \geq 0
$$

Primal:

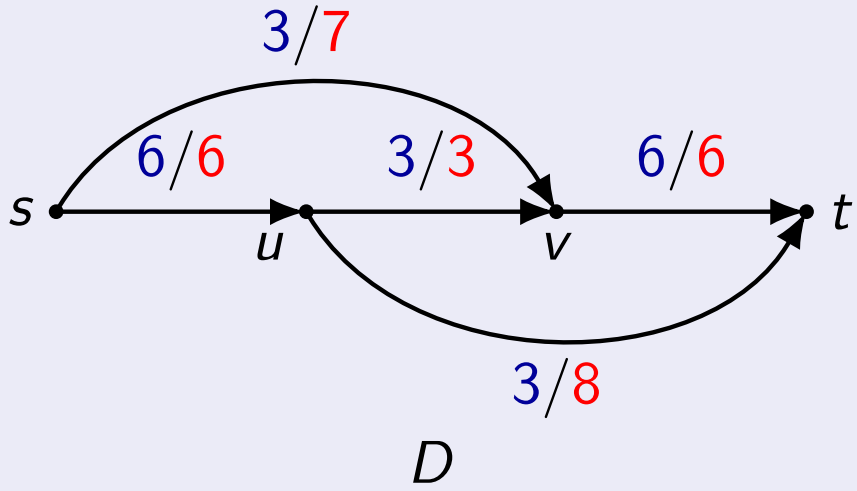
$$
\begin{aligned}
\max \quad & w \\
\text{s.t.} \quad & Mf + \begin{pmatrix} -1 \\ 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix} w = 0 \\
& f \leq b \\
& f, w \geq 0
\end{aligned}
$$

Dual:

$$
\begin{aligned}
\min \quad & \sum_{(u,v) \in A} b_{uv} \gamma_{uv} & \\
\text{s.t.} \quad & \pi_u - \pi_v + \gamma_{uv} \geq 0 & \forall (u,v) \in A \\
& -\pi_s + \pi_t \geq 1 & \\
& \pi_v \in \mathbb{R} & \forall v \in V \\
& \gamma_{uv} \geq 0 & \forall (u,v) \in A
\end{aligned}
$$

Question (2)

Prove that there does **not** exist a flow with **value 15** or more in the network below.

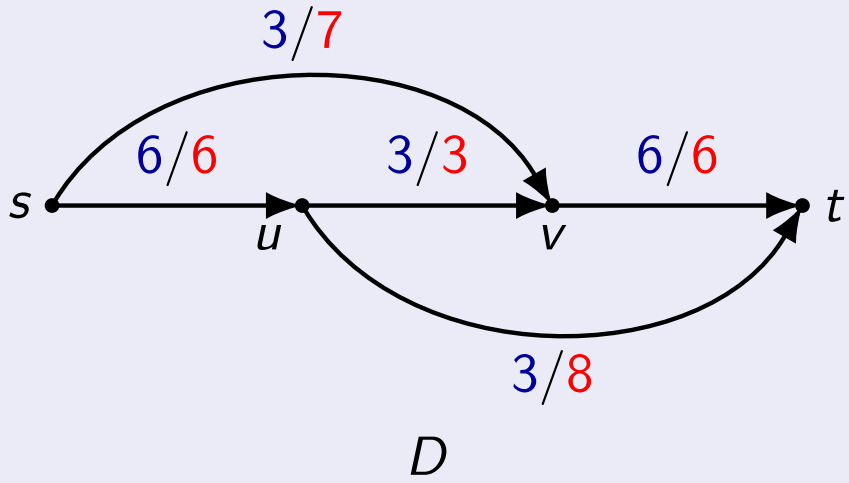


Answer

The total capacity on the incoming arcs to t is 6+8=14.

Question (3)

Prove that there does **not** exist a flow with **value 14** or more in the network below.



D

Answer

The total capacity on the arcs leaving s is 7+6=13.

Both cases are examples of a so-called "s-t cut" in the graph, i.e., a partition of the vertex set such that s belongs to one set and t to the other set.

Definition

- An s-t **cut** in a directed graph $D = (V, A)$ is a partition $(W, V \setminus W)$ of the vertices such that $s \in W$ and $t \in V \setminus W = \overline{W}$.
- The **capacity** of the cut $(W, \overline{W})$ is

$$\text{capacity}(W, \overline{W}) = \sum_{(u,v) \in A \mid u \in W,\, v \in \overline{W}} b_{uv}$$

Notice: The capacity of a cut is the sum of the capacity on the "forward" arcs of the cut, i.e., arcs that have a tail in W and a head in $\overline{W}$.

Lemma

An s-t cut $(W, \overline{W})$ determines a feasible solution to the dual with objective function value equal to capacity$(W, \overline{W})$.

Set:
$\pi_v = 0$ for all $v \in W$,
$\pi_v = 1$ for all $v \in \overline{W}$,
$\gamma_{uv} = 1$ if $u \in W,\ v \in \overline{W}$,
$\gamma_{uv} = 0$ for all other $(u, v) \in A$
Find a minimum $s - t$ cut.

Dual:

$$\begin{aligned} \min \quad & \sum_{(u,v)\in A} b_{uv}\gamma_{uv} & \\ \text{s.t.} \quad & \pi_u - \pi_v + \gamma_{uv} \geq 0 & \forall (u, v) \in A \\ & -\pi_s + \pi_t \geq 1 & \\ & \pi_v \in \mathbb{R} & \forall v \in V \end{aligned}$$

Theorem (Max Flow Min Cut)

(i) For each feasible s-t flow f and for each s-t cut $(W, \overline{W})$ holds that value$(f) \leq$ capacity$(W, \overline{W})$. (**Weak duality!**)

(ii) The **maximum** value of an s-t **flow** is equal to the **minimum** capacity of an s-t **cut**. (**Strong duality!**)

Theorem (Complementary Slackness for Max Flow)

A flow f and a cut $(W, \overline{W})$ are both optimal if and only if:

- $f_{uv} = b_{uv}$ for all $(u, v) \in A$ with $u \in W$ and $v \in \overline{W}$ ("forward arcs");
- $f_{uv} = 0$ for all $(u, v) \in A$ with $u \in \overline{W}$ and $v \in W$ ("backward arcs").

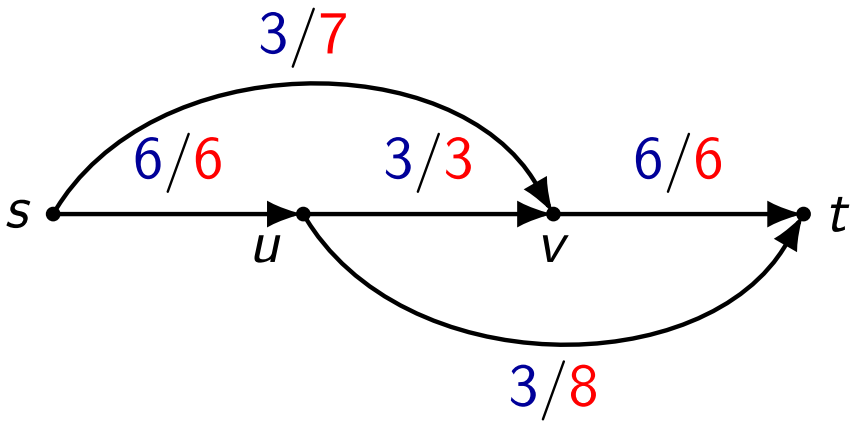
Algorithm of Ford-Fulkerson (1962)

- Algorithm for the MAX FLOW problem
- The idea is as follows:
  - **Start** with **0** flow on each arc.
  - **Augment** the flow over a **path** from s to t.
  - If the path goes **"against the flow"** on some arcs, then **decrease** the flow on those arcs.
  - Repeat until there is no **augmenting path** anymore.
  - To find the augmenting paths, we use an **auxiliary directed graph**.

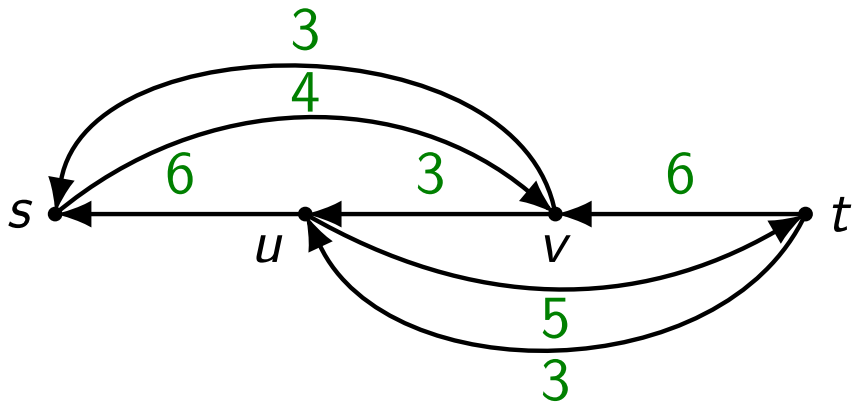
Auxiliary directed graph

For a given flow f, the auxiliary directed graph D_f has the same vertices as D and the following arcs:

- for each arc (u, v) of D with $f_{uv} < b_{uv}$
  - D_f gets an arc (u, v) with "capacity" label $b_{uv} - f_{uv}$
  - because we can increase the flow by at most $b_{uv} - f_{uv}$
- for each arc (u, v) of D with $f_{uv} > 0$
  - D_f gets an arc (v, u) with "capacity" label f_{uv}
  - because we can decrease the flow by at most f_{uv}.



D



D_f

Ford-Fulkerson in pseudo code

1. $f_{uv} := 0$ for all $(u, v) \in A$
2. Construct D_f with the same vertices as D and
   for each arc (u, v) of D:
   if $f_{uv} < b_{uv}$ then D_f gets an arc (u, v) with label $\bar{b}_{uv} = b_{uv} - f_{uv}$
   if $f_{uv} > 0$ then D_f gets an arc (v, u) with label $\bar{b}_{uv} = f_{uv}$
3. **Case 1:** there is a directed **path** P **from** s **to** t **in** D_f.

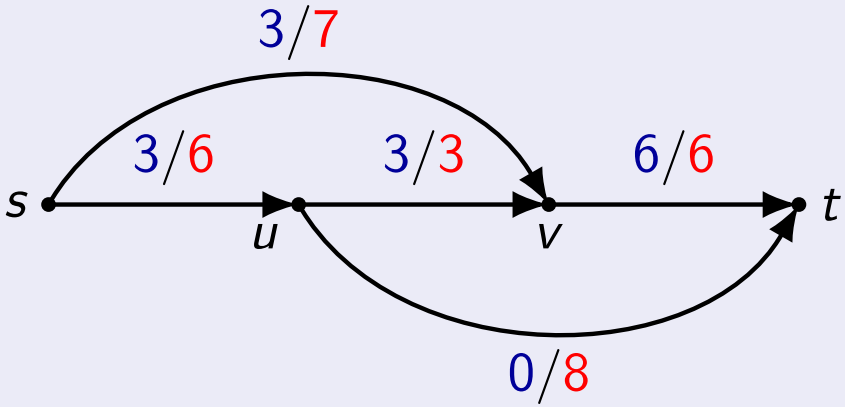
   $$\alpha := \min\{\bar{b}_{uv} \mid (u, v) \text{ lies on } P\}$$

   Augment the flow f **by** α **along** P
   Go to (2).
4. **Case 2:** there is **no path** from s to t in D_f. Define:

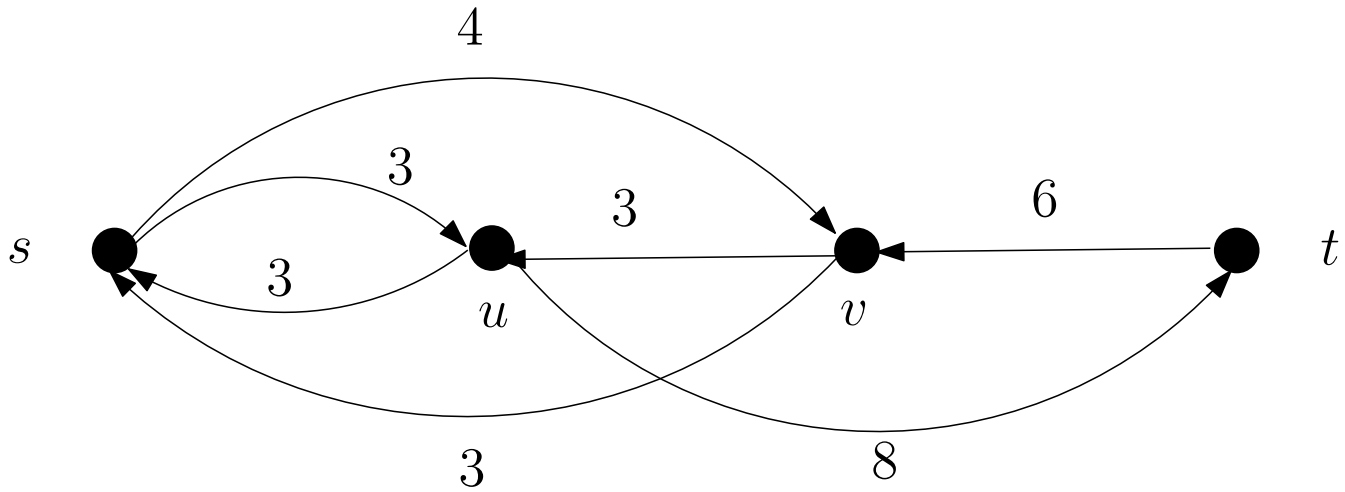
   $W := \{u \in V \mid$ there is a path from s to u in $D_f\}$
   $(W, \bar{W})$ is an s-t **cut** with **capacity** equal to
   the **value** of f.

Example

Suppose that, after a few iterations, we have found the following flow. Find a maximum flow by continuing the Ford-Fulkerson algorithm.

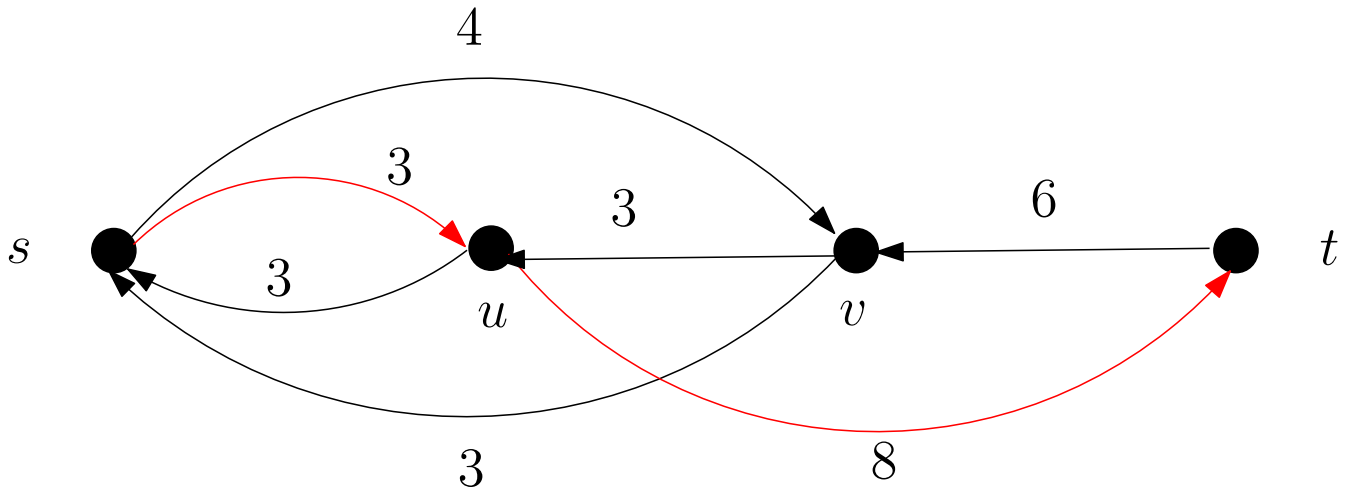


The current flow from s to t is 6.

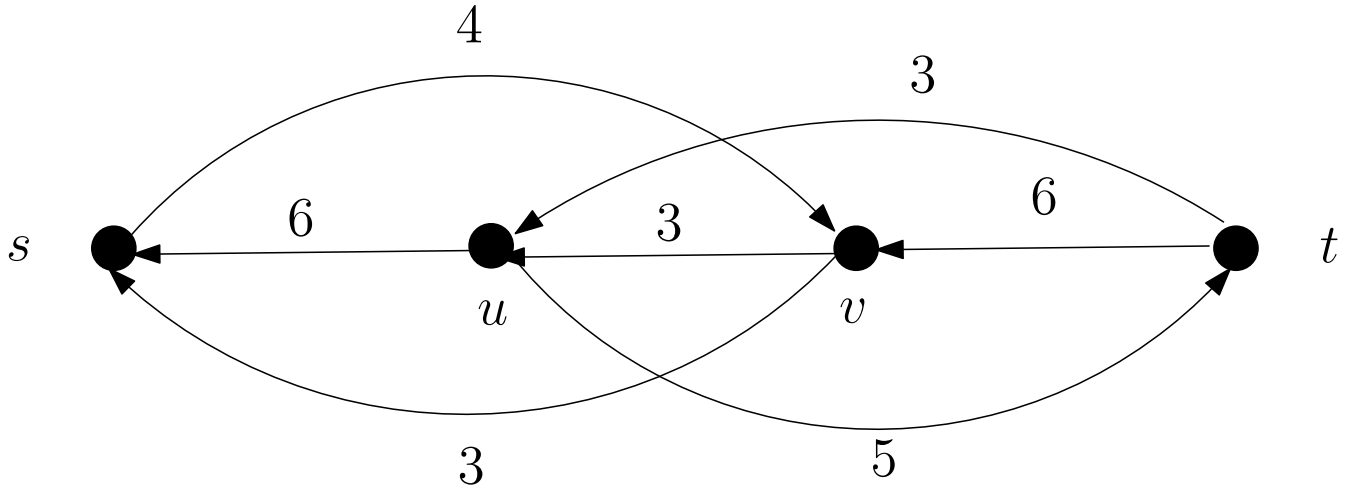


Auxiliary graph

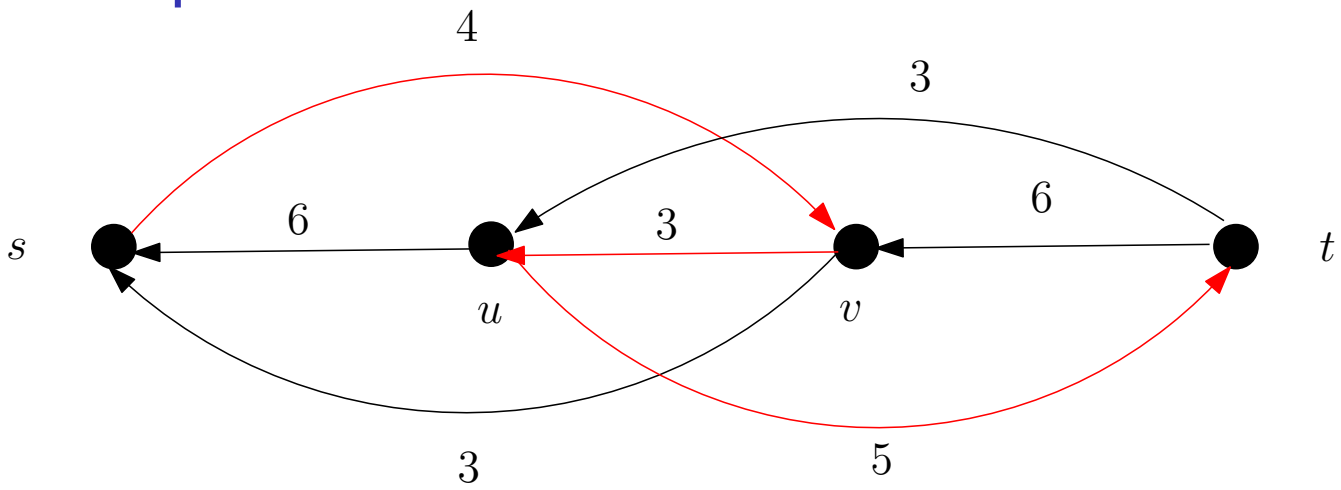
Example



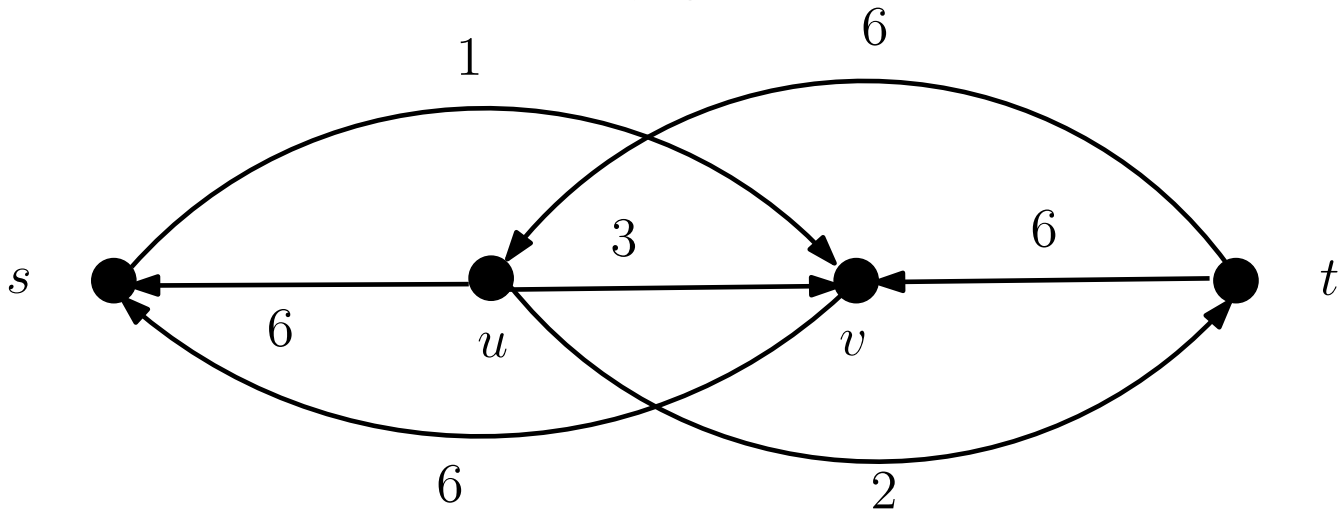
There is an augmenting path s–u–t with capacity $\min\{3, 8\} = 3$.
Augment the flow by 3 along that path.
The current flow from s to t is 6+3=9.
We obtain a new auxiliary graph:



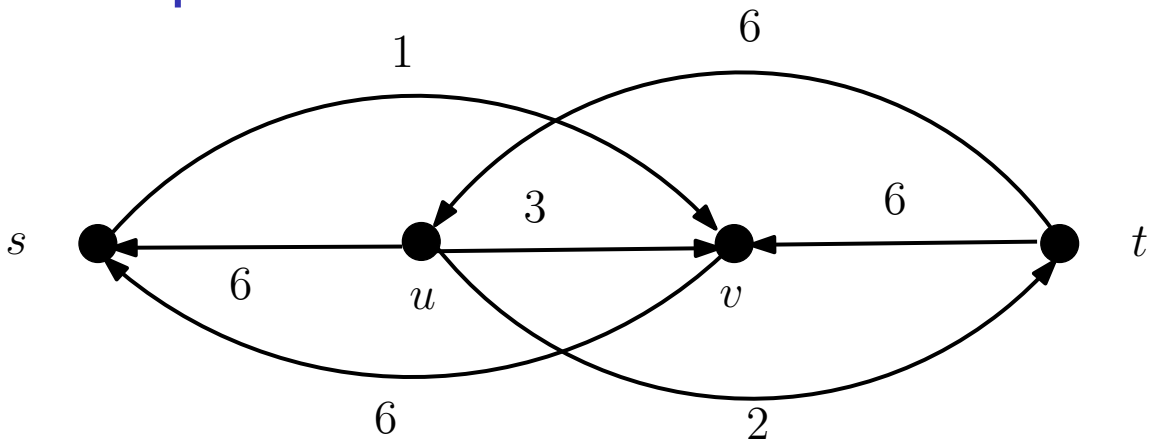
Example



There is an augmenting path s–v–u–t with capacity $\min\{4, 3, 5\} = 3$.
Augment the flow by 3 along that path.
The current flow from s to t is $9+3 = 12$.
We obtain a new auxiliary graph:

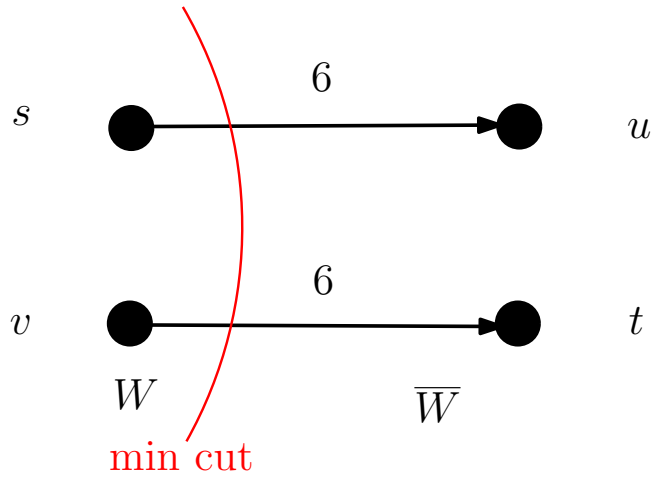


Example

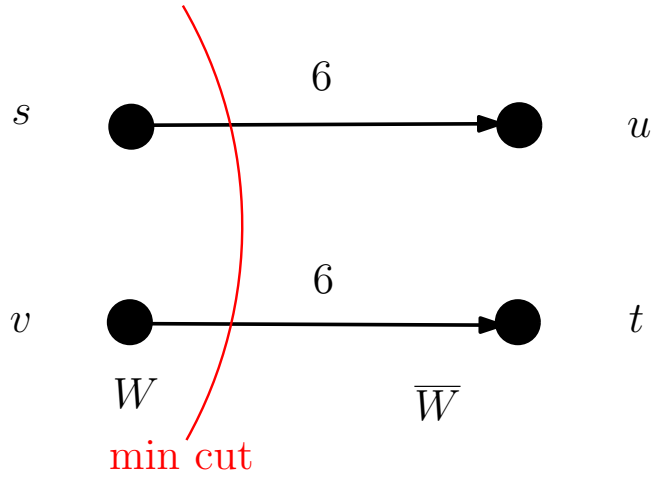


We can only reach vertex v from s. So $W = \{s, v\}$ and $\bar{W} = \{u, t\}$. Look in the **original graph** and calculate the capacity of all arcs that go in the forward direction from W to $\bar{W}$.

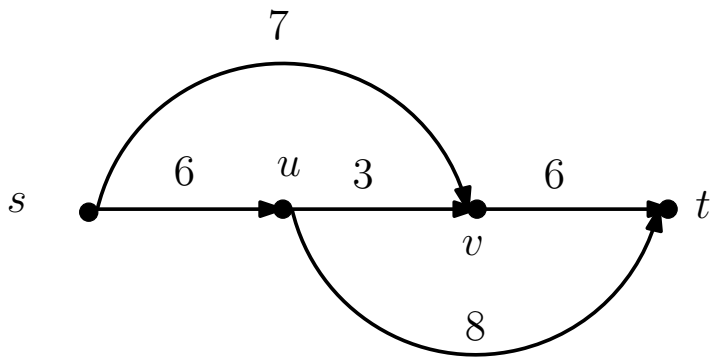
To make it easier to see, I have redrawn the graph such that W, $\bar{W}$ are easier to see.

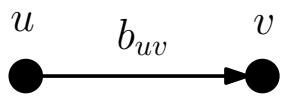


Optimal solution

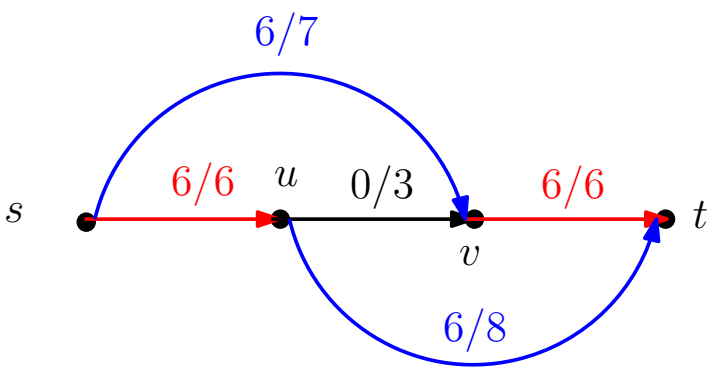


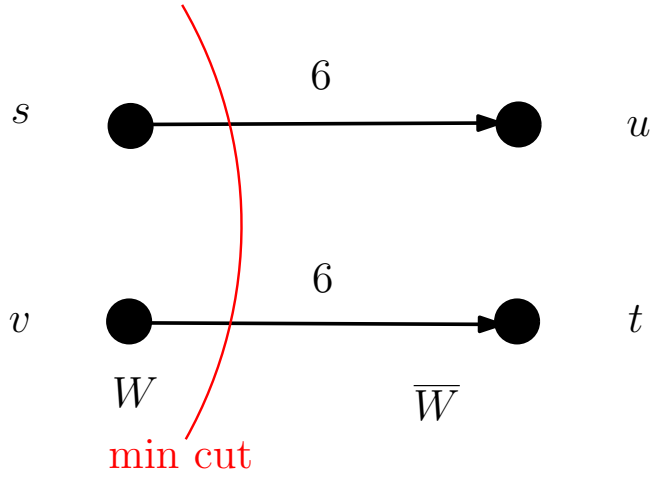
Since the value of the s-t cut is 12, and we have an s-t flow of value 12, the flow has to be maximum, and the s-t cut minimum!





Optimal solution



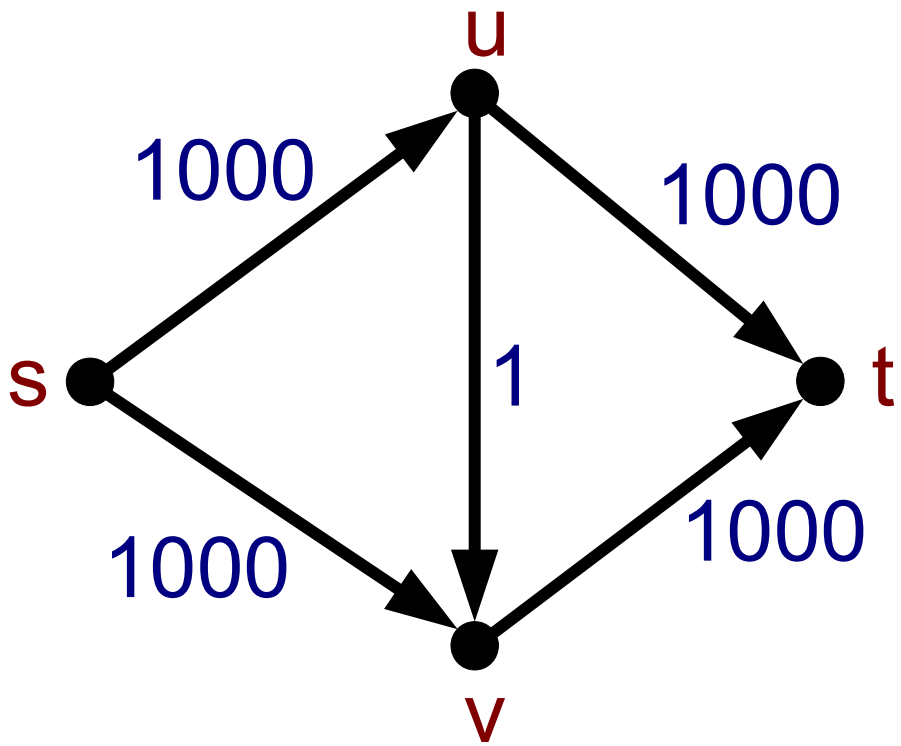


We can now verify the complementary slackness conditions:

- $f_{uv} = b_{uv}$ for all $(u, v) \in A$ with $u \in W$ and $v \in \overline{W}$ ("forward arcs");
- $f_{uv} = 0$ for all $(u, v) \in A$ with $u \in \overline{W}$ and $v \in W$ ("backward arcs"). Here (u, v) is a backward arc, which has zero flow.

Example

The number of iterations of the **Ford-Fulkerson** algorithm can be exponential in the input size.

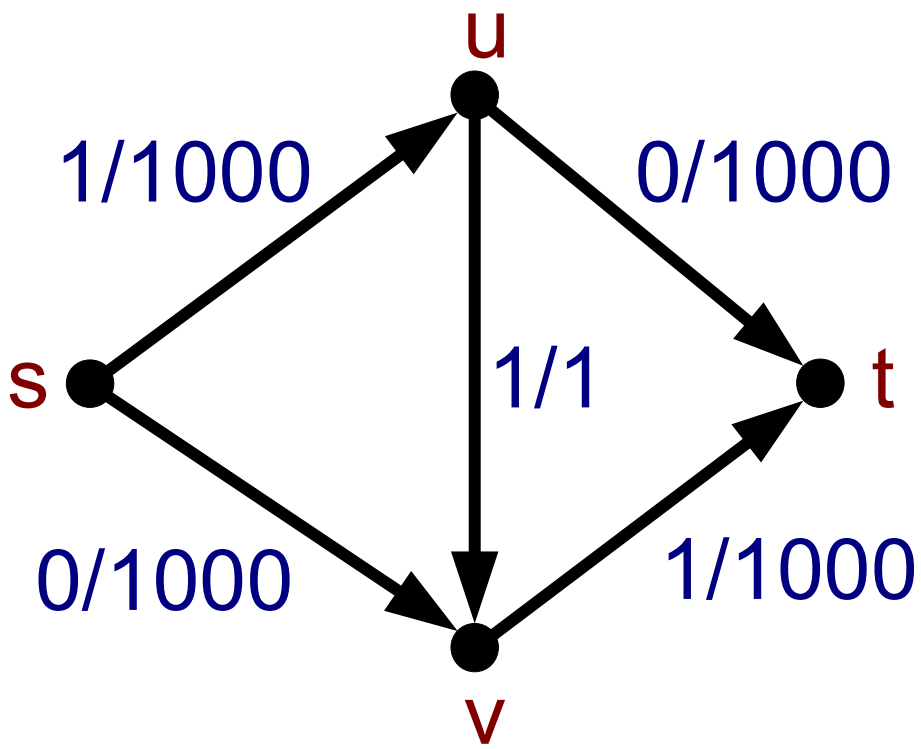


First iteration: augment the flow over the path (s, u, v, t).

Example

The number of iterations of the **Ford-Fulkerson** algorithm can be exponential in the input size.

This gives a flow with value 1:

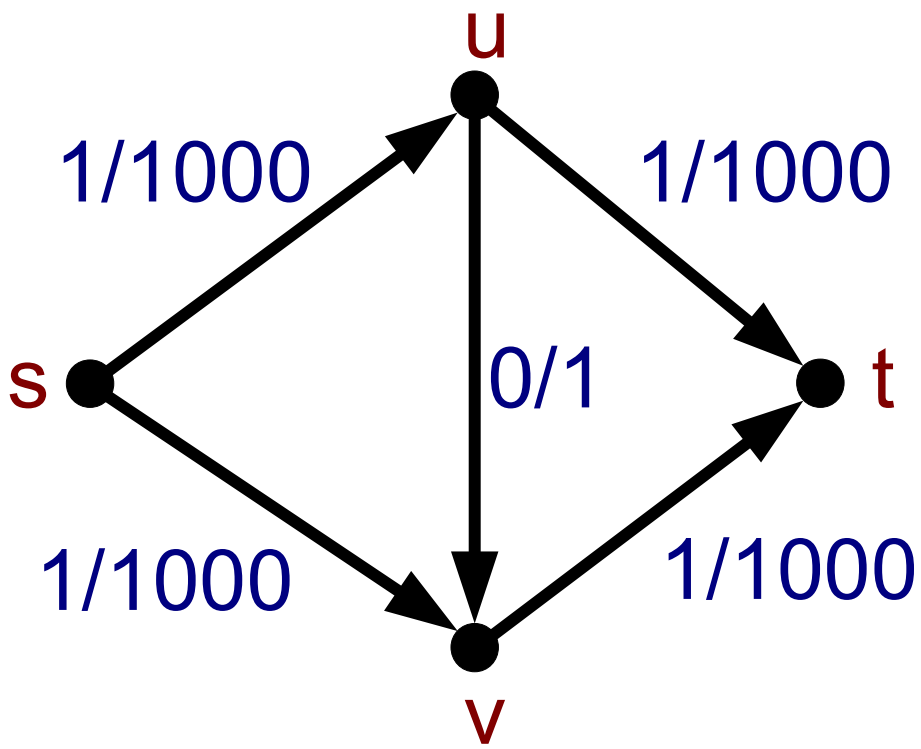


Second iteration: augment the flow over the path (s, v, u, t).

Example

The number of iterations of the **Ford-Fulkerson** algorithm can be exponential in the input size.

This gives a flow with value 2:

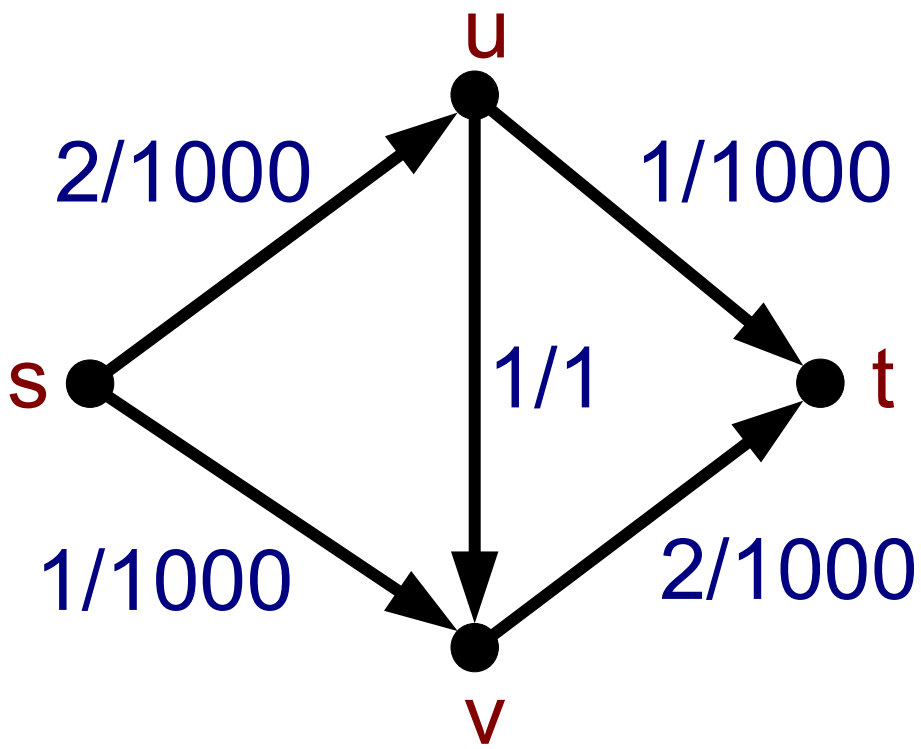


Third iteration: augment the flow over the path (s, u, v, t).

Example

The number of iterations of the **Ford-Fulkerson** algorithm can be exponential in the input size.

This gives a flow with value 3:

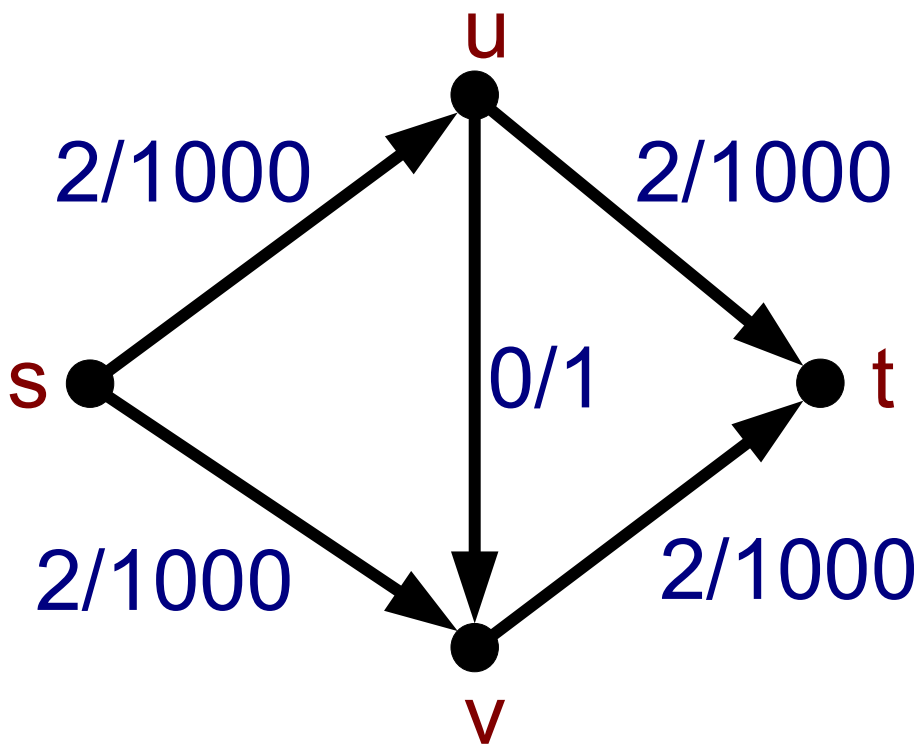


Fourth iteration: augment the flow over the path (s, v, u, t).

Example

The number of iterations of the **Ford-Fulkerson** algorithm can be exponential in the input size.

This gives a flow with value 4:

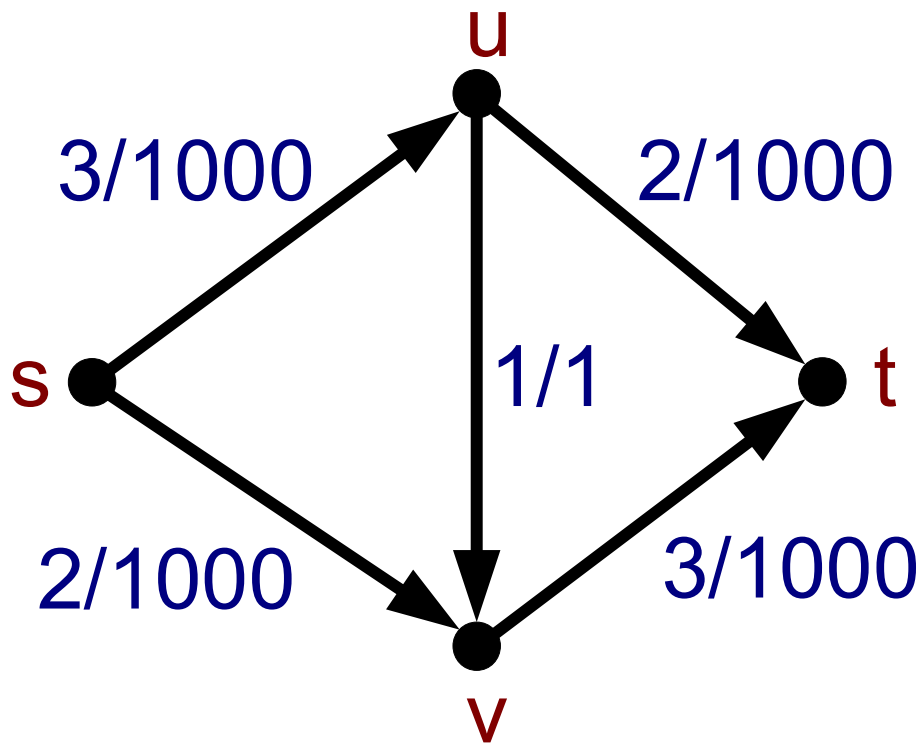


Fifth iteration: augment the flow over the path (s, u, v, t).

Example

The number of iterations of the **Ford-Fulkerson** algorithm can be exponential in the input size.

This gives a flow with value 5:

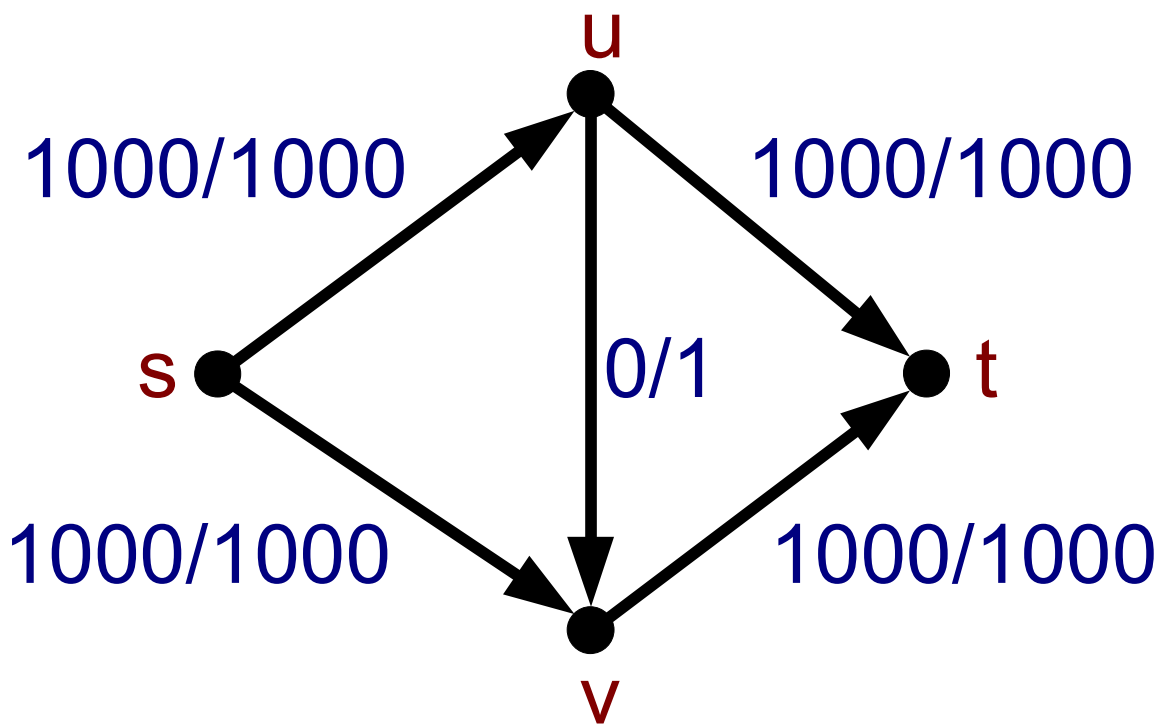


Continue like this. . .

Example

The number of iterations of the **Ford-Fulkerson** algorithm can be exponential in the input size.

After **2000 iterations** we finally find an optimal flow with value 2000:



If we replace 1000 by 2^N
then the input size is $O(\log(2^N)) = O(N)$
and we need 2^{N+1} iterations!

So the **Ford-Fulkerson** algorithm is **not polynomial**.

Question

How can we improve the algorithm?

Theorem (Dinits and Edmonds-Karp)

The Ford-Fulkerson algorithm has **polynomial running time** if we choose a **shortest** flow-augmenting path in each iteration.

Here, the length on each arc is 1, so we count "shortest" in terms of the **number of arcs** in the path.

To Do: Exercises 10.1, 9.3, 9.4

In 10.1, verify that $\pi = -\rho$ is a feasible solution of the dual, and that the value $\pi_s - \pi_t$ is equal to the length of the shortest path. Determine the shortest path using Complementary slackness.

In 9.3, also write down the dual and Complementary slackness conditions, and verify that the max flow is equal to the min cut.