Algorithms and Data Structures

Degeneracy, Geometry, and Duality

Historic Note

The Simplex Method was invented by George Dantzig in 1947.

It is still being used today in most of the LP-solvers.

Historic Note

The Simplex Method was invented by George Dantzig in 1947.

It is still being used today in most of the LP-solvers.

> The origins of the simplex method go back to one of two famous unsolved problems in mathematical statistics proposed by Jerzy Neyman, which I mistakenly solved as a homework problem; it later

Dantzig. Origins of the Simplex Method. In *A History of Scientific Computing*, 1990.

What if we have this dictionary?

Maximise $\zeta = 5 + x_3 - x_1$

subject to

$$
\begin{aligned}
x_2 &= 5 + 2x_3 - 3x_1 \\
x_4 &= 7 - 4x_1 \\
x_5 &= x_1
\end{aligned}
$$

$$x_1, x_2, x_3, x_4, x_5 \geq 0$$

What if we have this dictionary?

Maximise $\zeta = 5 + x_3 - x_1$

subject to

$$
\begin{aligned}
x_2 &= 5 + 2x_3 - 3x_1 \\
x_4 &= 7 - 4x_1 \\
x_5 &= x_1
\end{aligned}
$$

$$x_1, x_2, x_3, x_4, x_5 \geq 0$$

Entering variable: Any variable with positive coefficient in the objective function. If none exists, `break`;

What if we have this dictionary?

$$
\begin{array}{llrrrrrr}
\textbf{Maximise} & \zeta = & 5 & + & x_3 & - & x_1 \\
\textbf{subject to} & x_2 = & 5 & +2 & x_3 & -3 & x_1 \\
 & x_4 = & 7 & & & -4 & x_1 \\
 & x_5 = & & & & & x_1 \\
\end{array}
$$

$$x_1, x_2, x_3, x_4, x_5 \geq 0$$

Entering variable: Any variable with positive coefficient in the objective function. If none exists, `break`;

Leaving variable: The variable with the smallest ratio $\hat{b}_i / \hat{a}_{ik}$ (for the constraint $\hat{b}_i - \hat{a}_{ik} x_k \geq 0$).

What if we have this dictionary?

entering variable

$$\begin{array}{llllll}
\textbf{Maximise} & \zeta = & 5 & + \; x_3 & - & x_1 \\
\textbf{subject to} & x_2 = & 5 & +2 \; x_3 & -3 & x_1 \\
 & x_4 = & 7 & & -4 & x_1 \\
 & x_5 = & & & & x_1
\end{array}$$

$$x_1, x_2, x_3, x_4, x_5 \geq 0$$

Entering variable: Any variable with positive coefficient in the objective function. If none exists, `break`;

Leaving variable: The variable with the smallest ratio $\hat{b}_i / \hat{a}_{ik}$ (for the constraint $\hat{b}_i - \hat{a}_{ik} x_k \geq 0$).

What if we have this dictionary?

entering variable

Maximise $\zeta = 5 + x_3 - x_1$

subject to

$$
\begin{aligned}
x_2 &= 5 + 2x_3 - 3x_1 \\
x_4 &= 7 - 4x_1 \\
x_5 &= x_1
\end{aligned}
$$

$$x_1, x_2, x_3, x_4, x_5 \geq 0$$

What if we have this dictionary?

entering variable

Maximise $\zeta = 5 + x_3 - x_1$

subject to

$$\begin{aligned} x_2 &= 5 + 2x_3 - 3x_1 \\ x_4 &= 7 - 4x_1 \\ x_5 &= x_1 \end{aligned}$$

$$x_1, x_2, x_3, x_4, x_5 \geq 0$$

We can increase the value of some nonbasic variable, here x_3

What if we have this dictionary?

entering variable

Maximise $\zeta = 5 + x_3 - x_1$

subject to

$$\begin{aligned} x_2 &= 5 + 2x_3 - 3x_1 \\ x_4 &= 7 - 4x_1 \\ x_5 &= x_1 \end{aligned}$$

$$x_1, x_2, x_3, x_4, x_5 \geq 0$$

We can increase the value of some nonbasic variable, here x_3

We should not violate any constraints though!

What if we have this dictionary?

entering variable

$$\textbf{Maximise} \quad \zeta = 5 + x_3 - x_1$$

$$\textbf{subject to} \quad \begin{aligned} x_2 &= 5 + 2x_3 - 3x_1 \\ x_4 &= 7 \quad\quad\;\; - 4x_1 \\ x_5 &= \quad\quad\quad\quad\quad\;\; x_1 \end{aligned}$$

$$x_1, x_2, x_3, x_4, x_5 \geq 0$$

We can increase the value of some nonbasic variable, here x_3

We should not violate any constraints though!

We don't want any of the slack variables to become negative.

What if we have this dictionary?

entering variable

Maximise $\zeta = 5 + x_3 - x_1$

subject to

$$\begin{aligned} x_2 &= 5 + 2x_3 - 3x_1 \\ x_4 &= 7 - 4x_1 \\ x_5 &= x_1 \end{aligned}$$

$$x_1, x_2, x_3, x_4, x_5 \geq 0$$

We can increase the value of some nonbasic variable, here x_3

We should not violate any constraints though!

We don't want any of the slack variables to become negative.

What if we have this dictionary?

entering variable

Maximise $\zeta = 5 + x_3 - x_1$

subject to

$$
\begin{aligned}
x_2 &= 5 + 2x_3 - 3x_1 \\
x_4 &= 7 - 4x_1 \\
x_5 &= x_1
\end{aligned}
$$

$$x_1, x_2, x_3, x_4, x_5 \geq 0$$

We can increase the value of some nonbasic variable, here x_3

We should not violate any constraints though!

We don't want any of the slack variables to become negative.

This does not happen regardless of how much we increase x_3.

What if we have this dictionary?

entering variable

Maximise $\zeta = 5 + x_3 - x_1$

subject to

$$
\begin{aligned}
x_2 &= 5 + 2x_3 - 3x_1 \\
x_4 &= 7 - 4x_1 \\
x_5 &= x_1
\end{aligned}
$$

$$x_1, x_2, x_3, x_4, x_5 \geq 0$$

The LP is unbounded!

We can increase the value of some nonbasic variable, here x_3

We should not violate any constraints though!

We don't want any of the slack variables to become negative.

This does not happen regardless of how much we increase x_3.

What about this dictionary?

Maximise $\zeta = 3 - 0.5x_1 + 2x_2 - 1.5w_1$

subject to

$$x_3 = 1 - 0.5x_1 - 0.5w_1$$
$$w_2 = x_1 - x_2 + w_1$$

$$x_1, x_2, x_3, w_1, w_2 \geq 0$$

What about this dictionary?

Maximise $\zeta = 3 - 0.5\,x_1 + 2\,x_2 - 1.5\,w_1$

subject to

$$
\begin{aligned}
x_3 &= 1 - 0.5\,x_1 - 0.5\,w_1 \\
w_2 &= x_1 - x_2 + w_1
\end{aligned}
$$

entering variable

$$x_1, x_2, x_3, w_1, w_2 \geq 0$$

What about this dictionary?

Maximise $\zeta = 3 - 0.5\,x_1 + 2\,x_2 - 1.5\,w_1$

subject to

$$x_3 = 1 - 0.5\,x_1 - 0.5\,w_1$$

leaving variable $w_2 = x_1 - x_2 + w_1$ entering variable

$$x_1, x_2, x_3, w_1, w_2 \geq 0$$

What about this dictionary?

$$
\begin{array}{llrrrr}
\textbf{Maximise} & \zeta = & 3 & -0.5\,x_1 & +2\;x_2 & -1.5\,w_1 \\
\textbf{subject to} & x_3 = & 1 & -0.5\,x_1 & & -0.5\,w_1 \\
 & w_2 = & & x_1 & -\;x_2 & +\;w_1 \\
 & & & & & x_1, x_2, x_3, w_1, w_2 \geq 0
\end{array}
$$

entering variable

leaving variable

We can increase the value of some nonbasic variable, here x_2

We should not violate any constraints though!

We don't want any of the slack variables to become negative.

What about this dictionary?

Maximise $\zeta = 3 - 0.5\,x_1 + 2\,x_2 - 1.5\,w_1$

subject to

$$x_3 = 1 - 0.5\,x_1 - 0.5\,w_1$$

$$w_2 = x_1 - x_2 + w_1$$

leaving variable

entering variable

$$x_1, x_2, x_3, w_1, w_2 \geq 0$$

We can increase the value of some nonbasic variable, here x_2

We should not violate any constraints though!

We don't want any of the slack variables to become negative.

x_2 cannot be increased! Are we stuck?

What about this dictionary?

Maximise $\zeta = 3 - 0.5\, x_1 + 2\, x_2 - 1.5\, w_1$

subject to $x_3 = 1 - 0.5\, x_1 - 0.5\, w_1$

leaving variable $w_2 = x_1 - x_2 + w_1$ entering variable

$x_1, x_2, x_3, w_1, w_2 \geq 0$

We can increase the value of some nonbasic variable, here x_2

We should not violate any constraints though!

We don't want any of the slack variables to become negative.

x_2 cannot be increased! Are we stuck? **Degeneracy!**

Degeneracy

Degeneracy

Degenerate dictionary: A dictionary in which one of the b_i variables becomes zero.

Degeneracy

Degenerate dictionary: A dictionary in which one of the b_i variables becomes zero.

Equivalently: In a basic feasible solution, one of the basic variables is 0.

Degeneracy not necessarily and issue

entering variable

Maximise $\zeta = 5 + x_3 - x_1$

subject to

$$
\begin{aligned}
x_2 &= 5 + 2x_3 - 3x_1 \\
x_4 &= 7 \qquad\quad - 4x_1 \\
x_5 &= \qquad\qquad\quad x_1
\end{aligned}
$$

$$x_1, x_2, x_3, x_4, x_5 \geq 0$$

The LP is unbounded!

We can increase the value of some nonbasic variable, here x_3

We should not violate any constraints though!

We don't want any of the slack variables to become negative.

This does not happen regardless of how much we increase x_3.

Degeneracy

Degenerate dictionary: A dictionary in which one of the b_i variables becomes zero.

Equivalently: In a basic feasible solution, one of the basic variables is 0.

Degeneracy

Degenerate dictionary: A dictionary in which one of the b_i variables becomes zero.

Equivalently: In a basic feasible solution, one of the basic variables is 0.

Degenerate Pivot: The entering variable stays at 0 without increasing.

What about this dictionary?

$$
\begin{array}{llllllll}
\textbf{Maximise} & \zeta = & 3 & -0.5\,x_1 & +2 & x_2 & -1.5\,w_1 \\
\textbf{subject to} & x_3 = & 1 & -0.5\,x_1 & & & -0.5\,w_1 \\
 & w_2 = & & x_1 & - & x_2 & + & w_1
\end{array}
$$

leaving variable

entering variable

$$x_1, x_2, x_3, w_1, w_2 \geq 0$$

We can increase the value of some nonbasic variable, here x_2

We should not violate any constraints though!

We don't want any of the slack variables to become negative.

x_2 cannot be increased! Are we stuck?

Degeneracy!

Degeneracy

Degenerate dictionary: A dictionary in which one of the b_i variables becomes zero.

Equivalently: In a basic feasible solution, one of the basic variables is 0.

Degeneracy

Degenerate dictionary: A dictionary in which one of the b_i variables becomes zero.

Equivalently: In a basic feasible solution, one of the basic variables is 0.

Degenerate Pivot: The entering variable stays at 0 without increasing.

Degeneracy

Degenerate dictionary: A dictionary in which one of the b_i variables becomes zero.

Equivalently: In a basic feasible solution, one of the basic variables is 0.

Degenerate Pivot: The entering variable stays at 0 without increasing.

"Degenerate pivots are quite common and usually harmless."

Let's not give up

Maximise $\zeta = 3 - 0.5\,x_1 + 2\,x_2 - 1.5\,w_1$

subject to

$$
\begin{aligned}
x_3 &= 1 - 0.5\,x_1 - 0.5\,w_1 \\
w_2 &= x_1 - x_2 + w_1
\end{aligned}
$$

$x_1, x_2, x_3, w_1, w_2 \geq 0$

leaving variable

entering variable

We can increase the value of some nonbasic variable, here x_2

We should not violate any constraints though!

We don't want any of the slack variables to become negative.

x_2 cannot be increased! Are we stuck?

Let's not give up

Maximise $\zeta = 3 - 0.5x_1 + 2x_2 - 1.5w_1$

subject to

$$x_3 = 1 - 0.5x_1 - 0.5w_1$$

entering variable

leaving variable

$$w_2 = x_1 - x_2 + w_1$$

$$x_1, x_2, x_3, w_1, w_2 \geq 0$$

We can increase the value of some nonbasic variable, here x_2

We should not violate any constraints though!

We don't want any of the slack variables to become negative.

Let's not give up

Maximise $\zeta = 3 - 0.5\, x_1 + 2\, x_2 - 1.5\, w_1$

subject to

$$
\begin{aligned}
x_3 &= 1 - 0.5\, x_1 - 0.5\, w_1 \\
w_2 &= x_1 - x_2 + w_1
\end{aligned}
$$

leaving variable

entering variable

$$x_1, x_2, x_3, w_1, w_2 \geq 0$$

We can increase the value of some nonbasic variable, here x_2

We should not violate any constraints though!

We don't want any of the slack variables to become negative.

Increase the variable as much as we can (as before): here 0 increase

Let's not give up

Maximise $\zeta = 3 - 0.5\,x_1 + 2\,x_2 - 1.5\,w_1$

subject to

$$
\begin{aligned}
x_3 &= 1 - 0.5\,x_1 - 0.5\,w_1 \\
w_2 &= x_1 - x_2 + w_1
\end{aligned}
$$

leaving variable

entering variable

$x_1, x_2, x_3, w_1, w_2 \geq 0$

Actually pivot!

We can increase the value of some nonbasic variable, here x_2

We should not violate any constraints though!

We don't want any of the slack variables to become negative.

Increase the variable as much as we can (as before): here 0 increase

Let's not give up

Maximise $\zeta = 3 - 0.5x_1 + 2x_2 - 1.5w_1$

subject to

$$x_3 = 1 - 0.5x_1 - 0.5w_1$$
$$w_2 = x_1 - x_2 + w_1$$

leaving variable

entering variable

$$x_1, x_2, x_3, w_1, w_2 \geq 0$$

Actually pivot!

$(x_1, x_2, x_3, w_1, w_2) = (0,0,1,0,0)$

We can increase the value of some nonbasic variable, here x_2

We should not violate any constraints though!

We don't want any of the slack variables to become negative.

Increase the variable as much as we can (as before): here 0 increase

The new dictionary

Maximise $\zeta = 3 + 1.5\, x_1 + 2\, w_2 + 0.5\, w_1$

subject to

$$
\begin{aligned}
x_3 &= 1 - 0.5\, x_1 - 0.5\, w_1 \\
x_2 &= x_1 - w_2 + w_1
\end{aligned}
$$

$$x_1, x_2, x_3, w_1, w_2 \geq 0$$

The new dictionary

Maximise $\zeta = 3 + 1.5\,x_1 + 2\,w_2 + 0.5\,w_1$

subject to

$$\begin{aligned} x_3 &= 1 - 0.5\,x_1 - 0.5\,w_1 \\ x_2 &= x_1 - w_2 + w_1 \end{aligned}$$

$$x_1, x_2, x_3, w_1, w_2 \geq 0$$

$(x_1, x_2, x_3, w_1, w_2) = (0,0,1,0,0)$

The new dictionary

Maximise $\zeta = 3 + 1.5\,x_1 + 2\,w_2 + 0.5\,w_1$

entering variable

subject to

$$
\begin{aligned}
x_3 &= 1 - 0.5\,x_1 - 0.5\,w_1 \\
x_2 &= x_1 - w_2 + w_1
\end{aligned}
$$

$$x_1, x_2, x_3, w_1, w_2 \geq 0$$

$(x_1, x_2, x_3, w_1, w_2) = (0,0,1,0,0)$

The new dictionary

Maximise $\zeta = 3 + 1.5\,x_1 + 2\,w_2 + 0.5\,w_1$

entering variable

subject to $x_3 = 1 - 0.5\,x_1 - 0.5\,w_1$

leaving variable

$x_2 = x_1 - w_2 + w_1$

$x_1, x_2, x_3, w_1, w_2 \geq 0$

$(x_1, x_2, x_3, w_1, w_2) = (0,0,1,0,0)$

The new dictionary

Maximise $\zeta = 3 + 1.5\, x_1 + 2\, w_2 + 0.5\, w_1$

entering variable

subject to

$$x_3 = 1 - 0.5\, x_1 - 0.5\, w_1$$

leaving variable

$$x_2 = x_1 - w_2 + w_1$$

$$x_1, x_2, x_3, w_1, w_2 \geq 0$$

$(x_1, x_2, x_3, w_1, w_2) = (0,0,1,0,0)$

We can now increase x_1 to $x_1 = 2$

The new dictionary

$$
\begin{array}{lrcrrrr}
\textbf{Maximise} & \zeta = & 3 & +1.5\,x_1 & +2\,w_2 & +0.5\,w_1 \\
\textbf{subject to} & x_3 = & 1 & -0.5\,x_1 & & -0.5\,w_1 \\
 & x_2 = & & x_1 & -\,w_2 & +\,w_1 \\
\end{array}
$$

$$x_1, x_2, x_3, w_1, w_2 \geq 0$$

entering variable

leaving variable

$(x_1, x_2, x_3, w_1, w_2) = (0,0,1,0,0)$

We can now increase x_1 to $x_1 = 2$

The pivot is not degenerate!

The new dictionary

$$
\begin{array}{llllll}
\textbf{Maximise} & \zeta = & 3 & +1.5\,x_1 & +2\,w_2 & +0.5\,w_1 \\
\textbf{subject to} & x_3 = & 1 & -0.5\,x_1 & & -0.5\,w_1 \\
 & x_2 = & & x_1 & -\,w_2 & +\,w_1
\end{array}
$$

entering variable

leaving variable

$$x_1, x_2, x_3, w_1, w_2 \geq 0$$

$$(x_1, x_2, x_3, w_1, w_2) = (0,0,1,0,0)$$

We can now increase x_1 to $x_1 = 2$

The pivot is not degenerate!

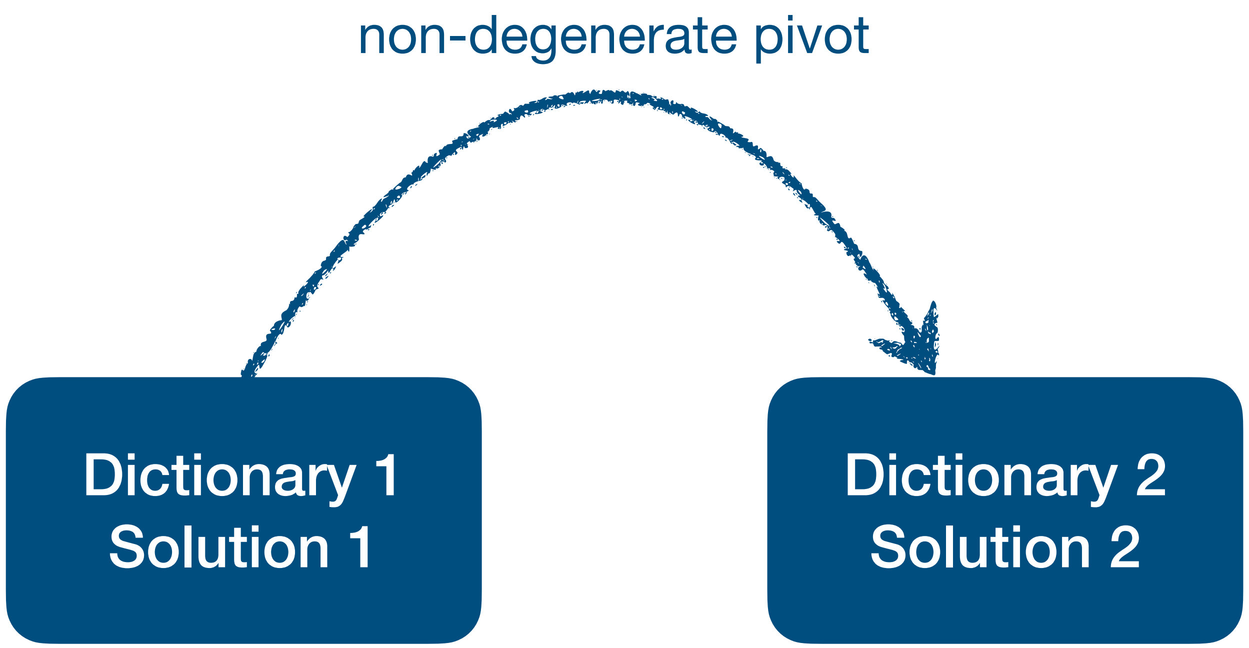
It will actually lead to a final dictionary, and an optimal solution.

Pictorially

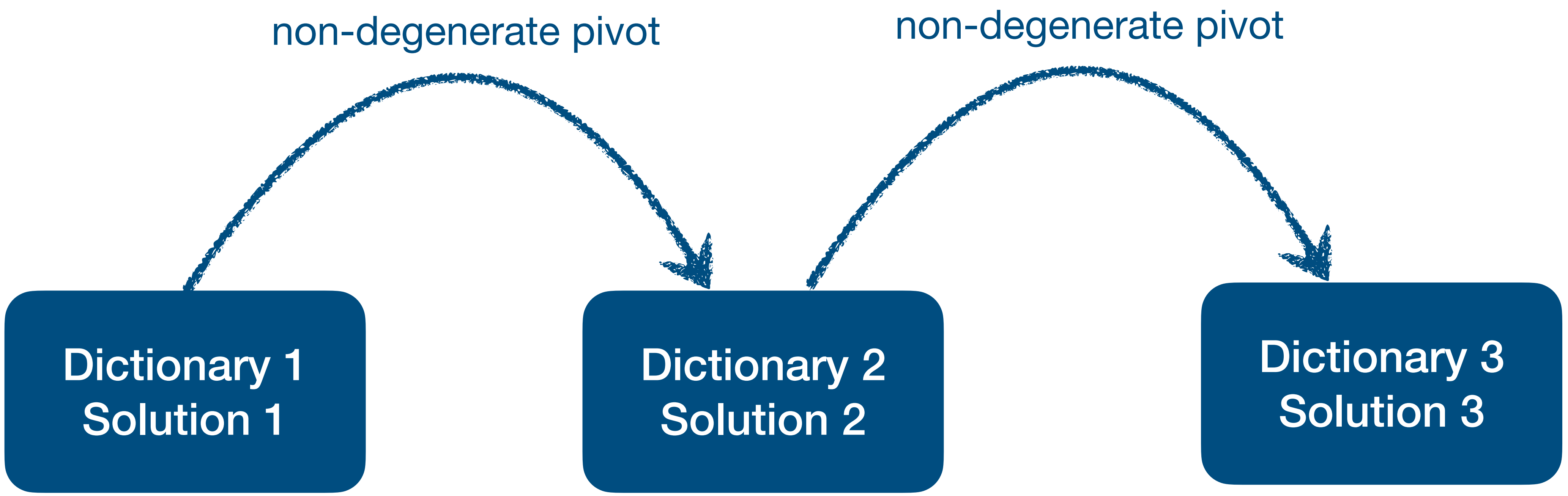
Pictorially

Dictionary 1
Solution 1

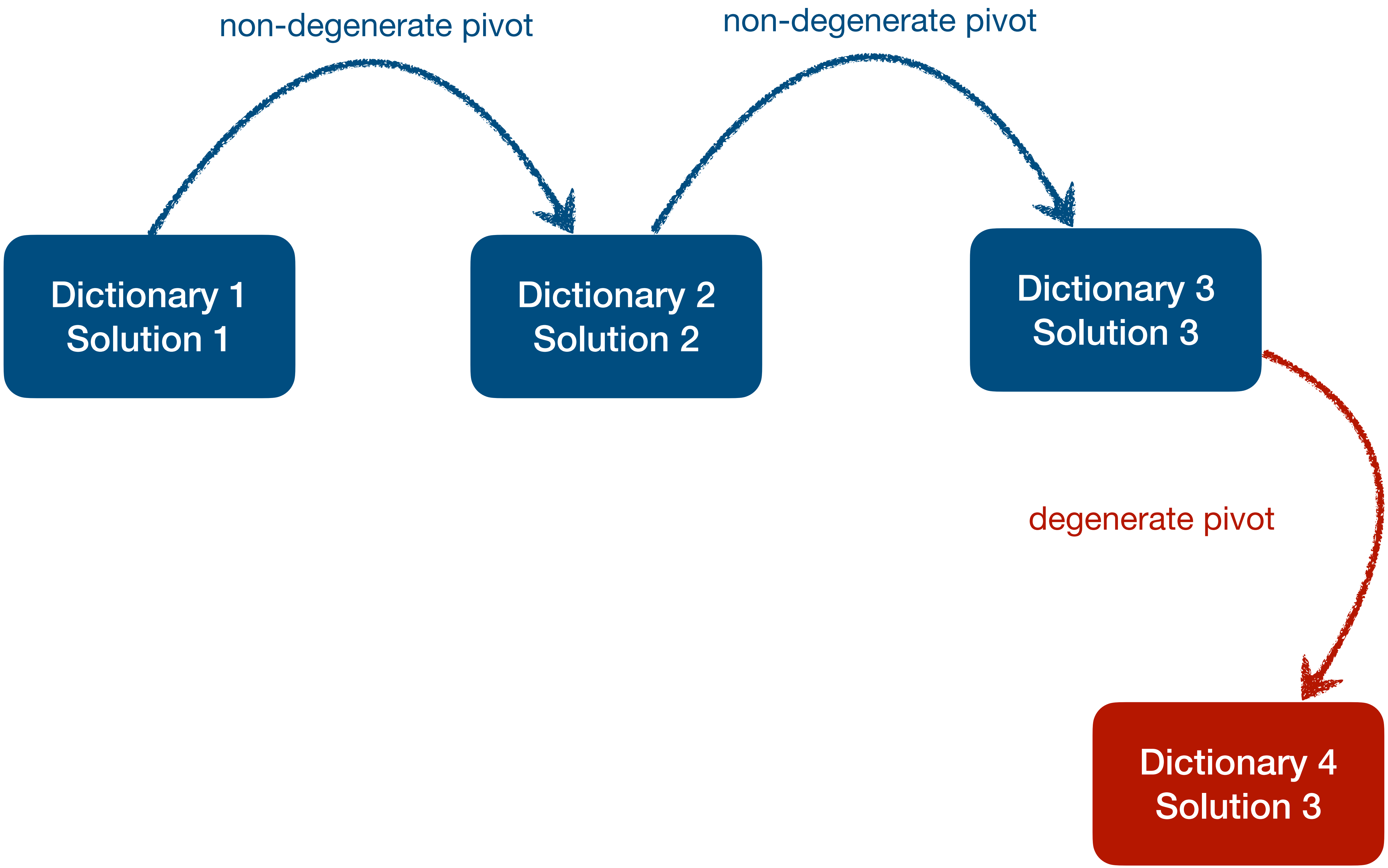
Pictorially



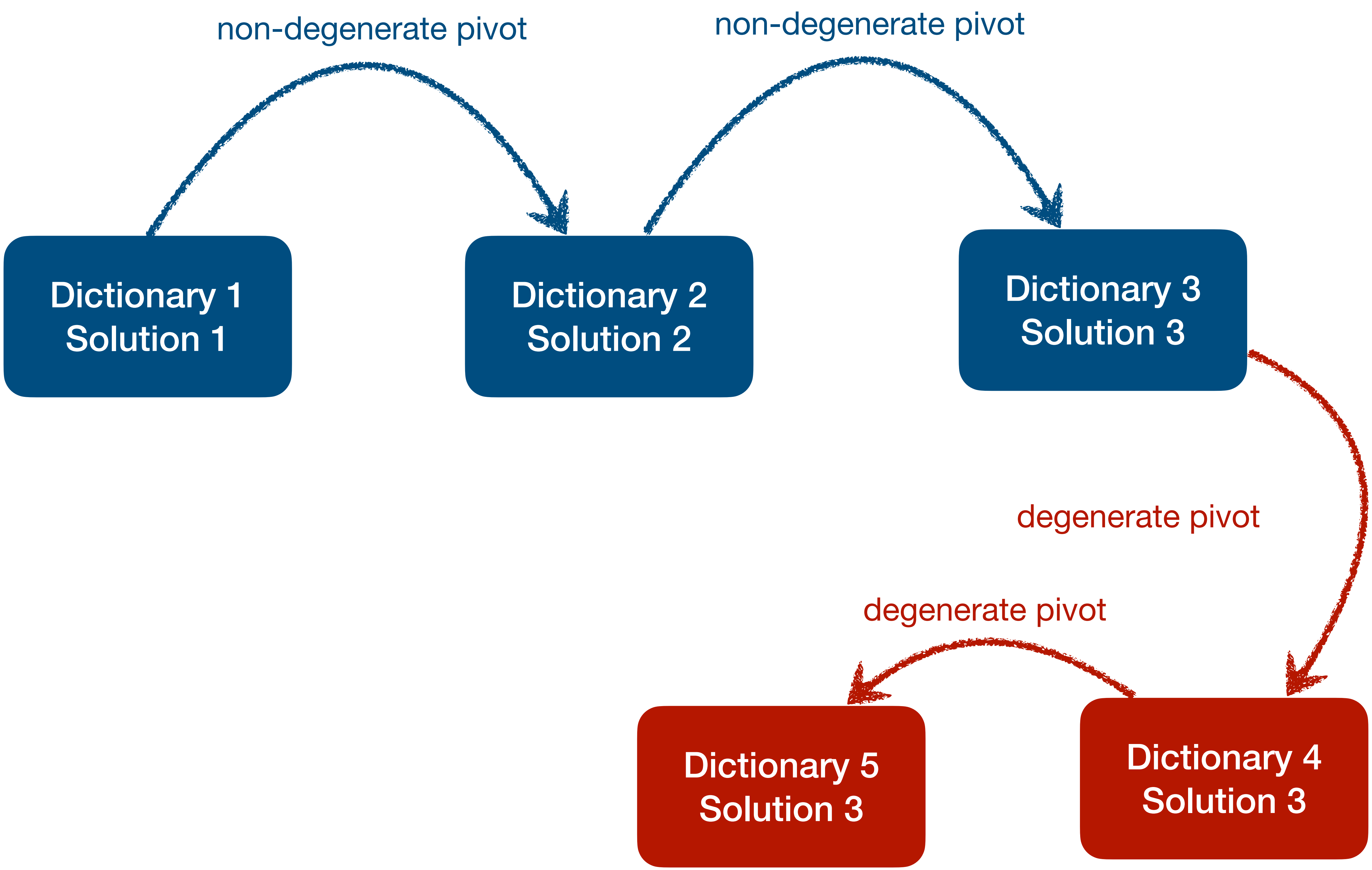
Pictorially



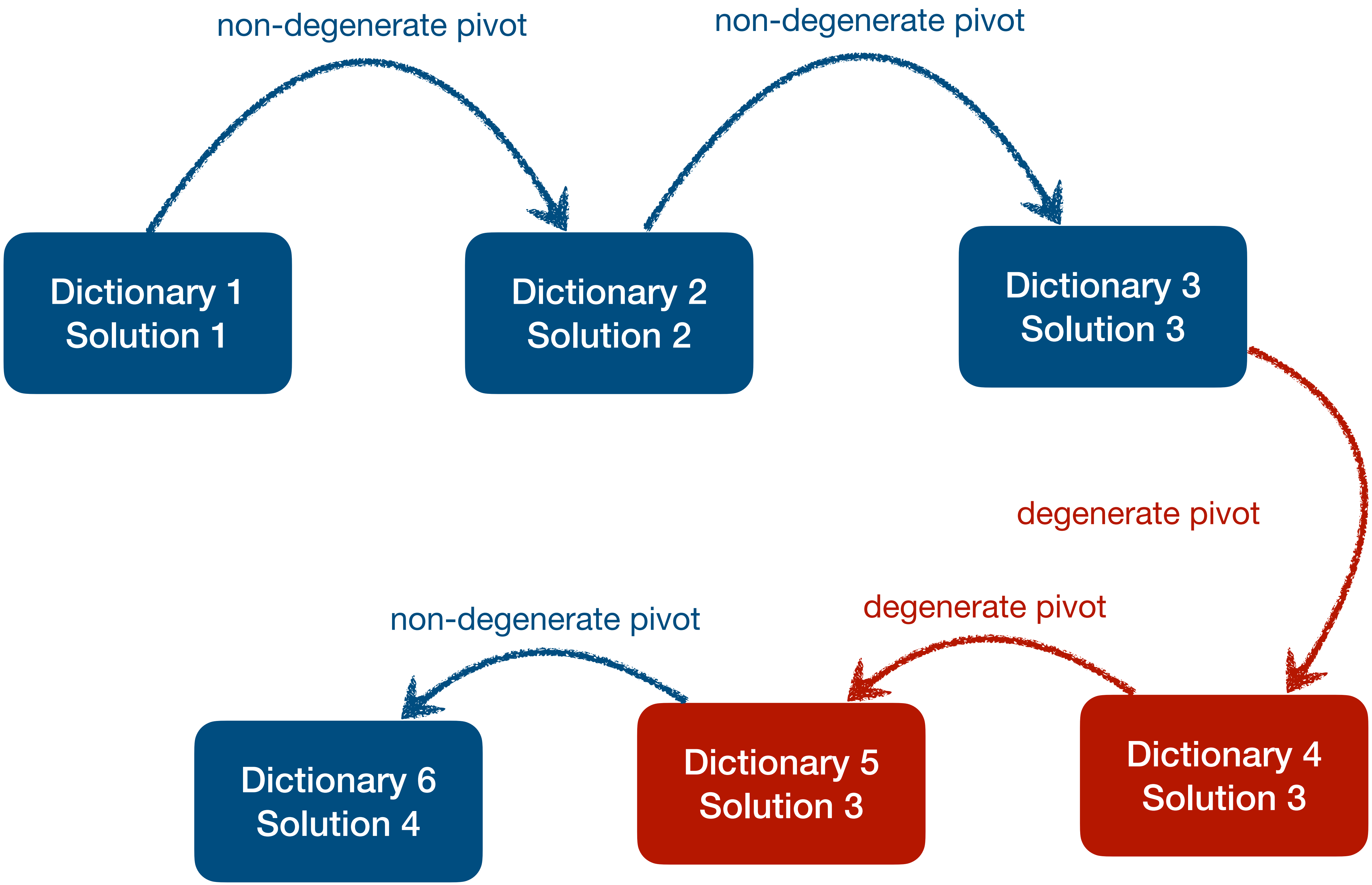
Pictorially



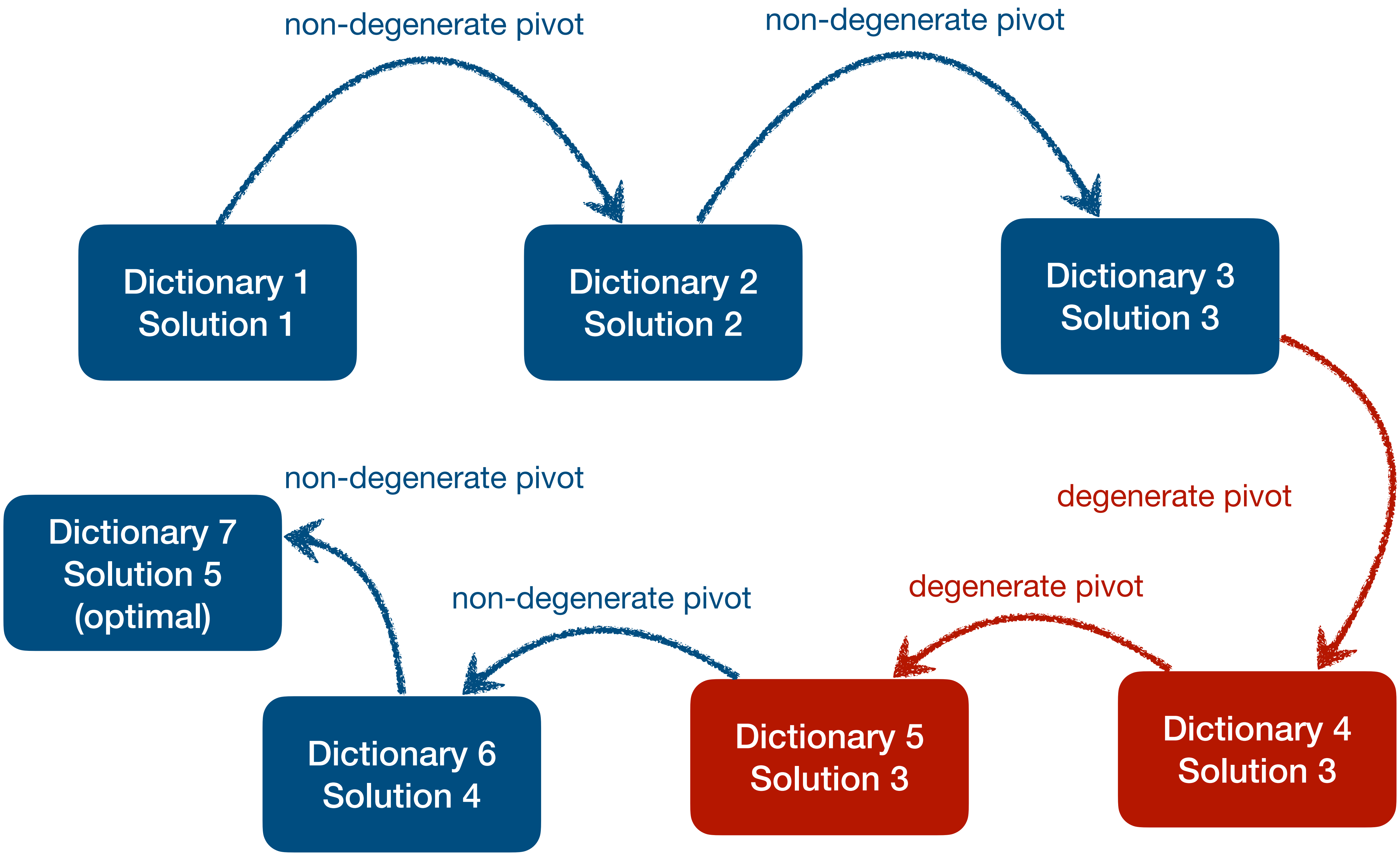
Pictorially



Pictorially



Pictorially

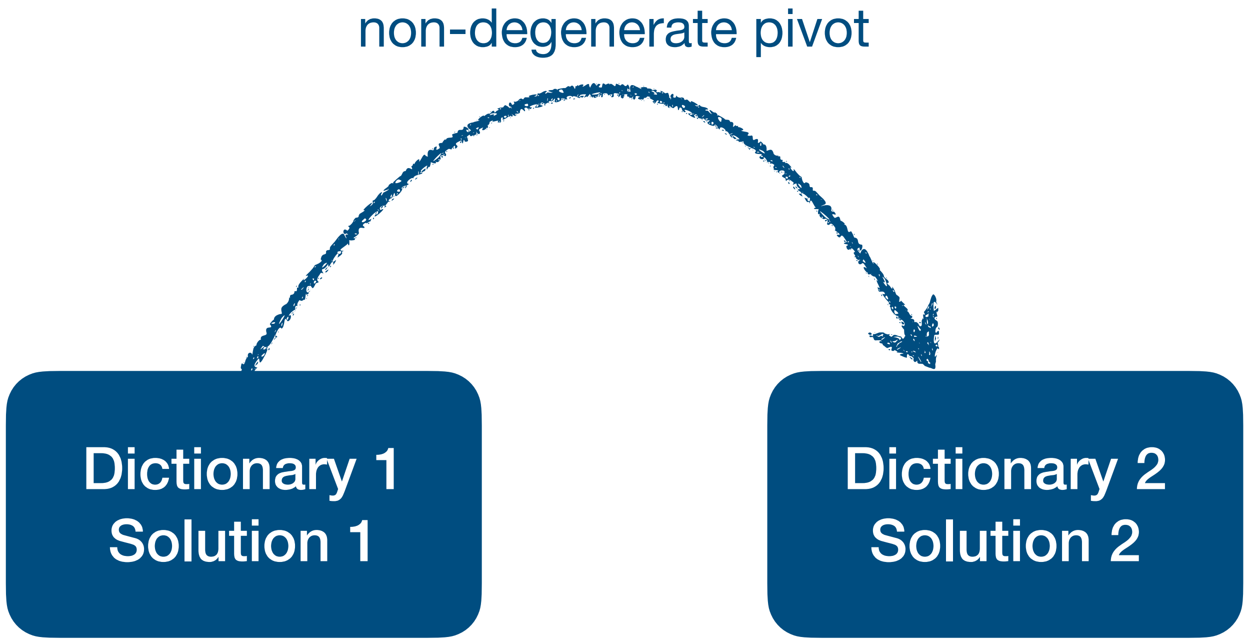


Could this happen though?

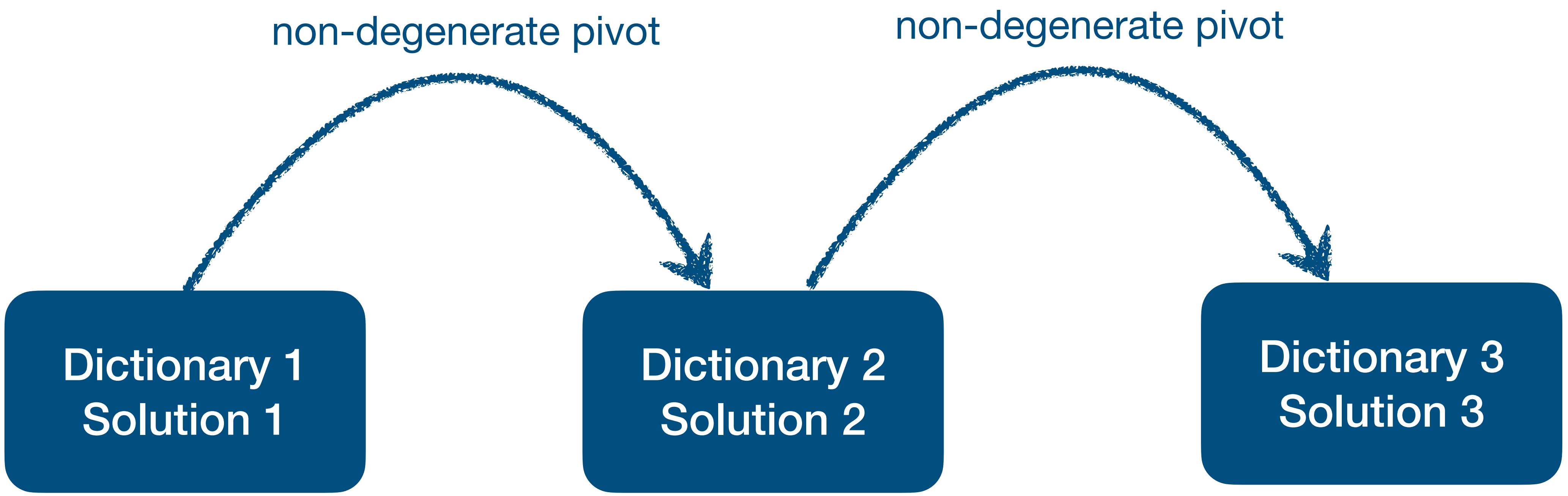
Could this happen though?

Dictionary 1
Solution 1

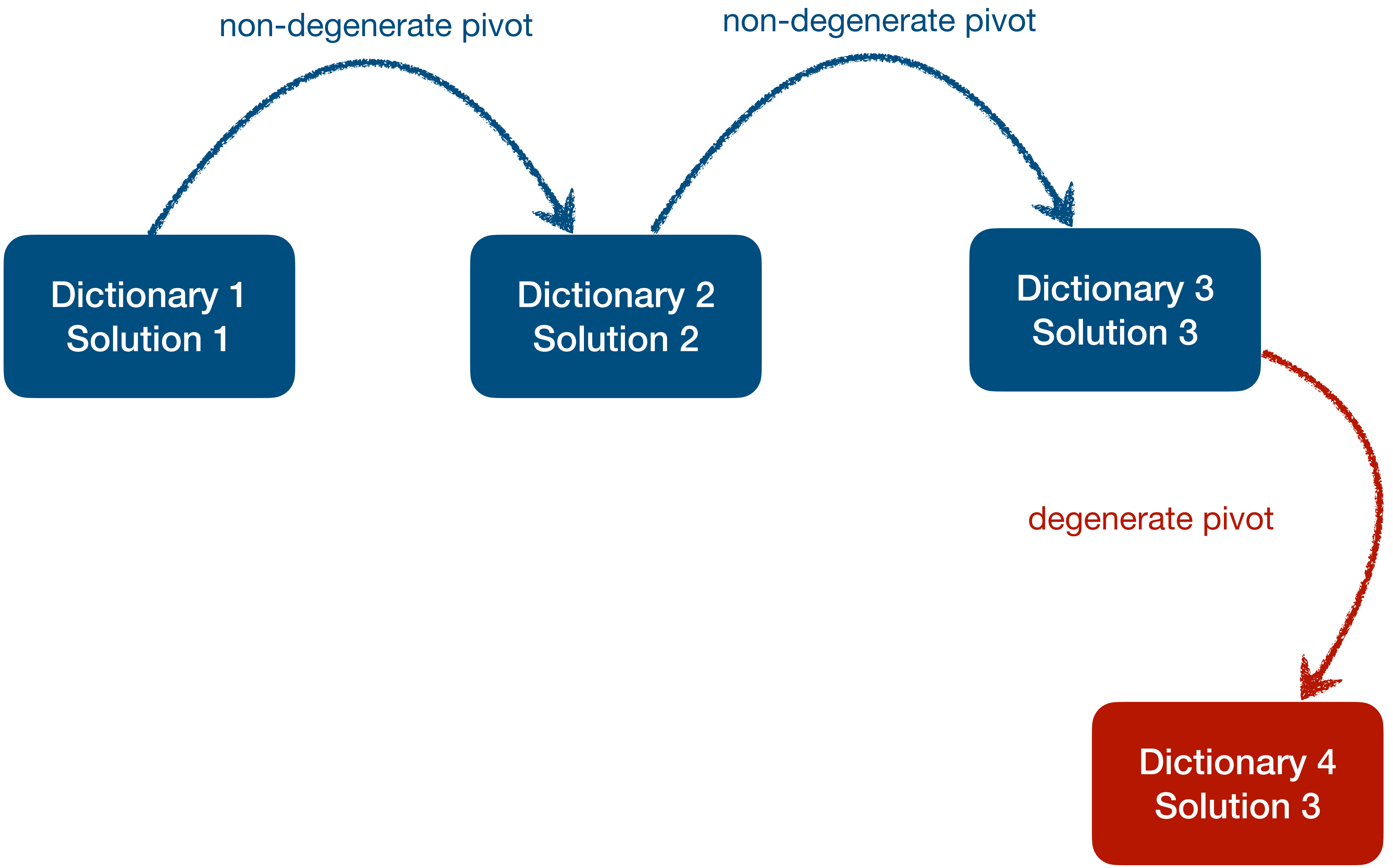
Could this happen though?



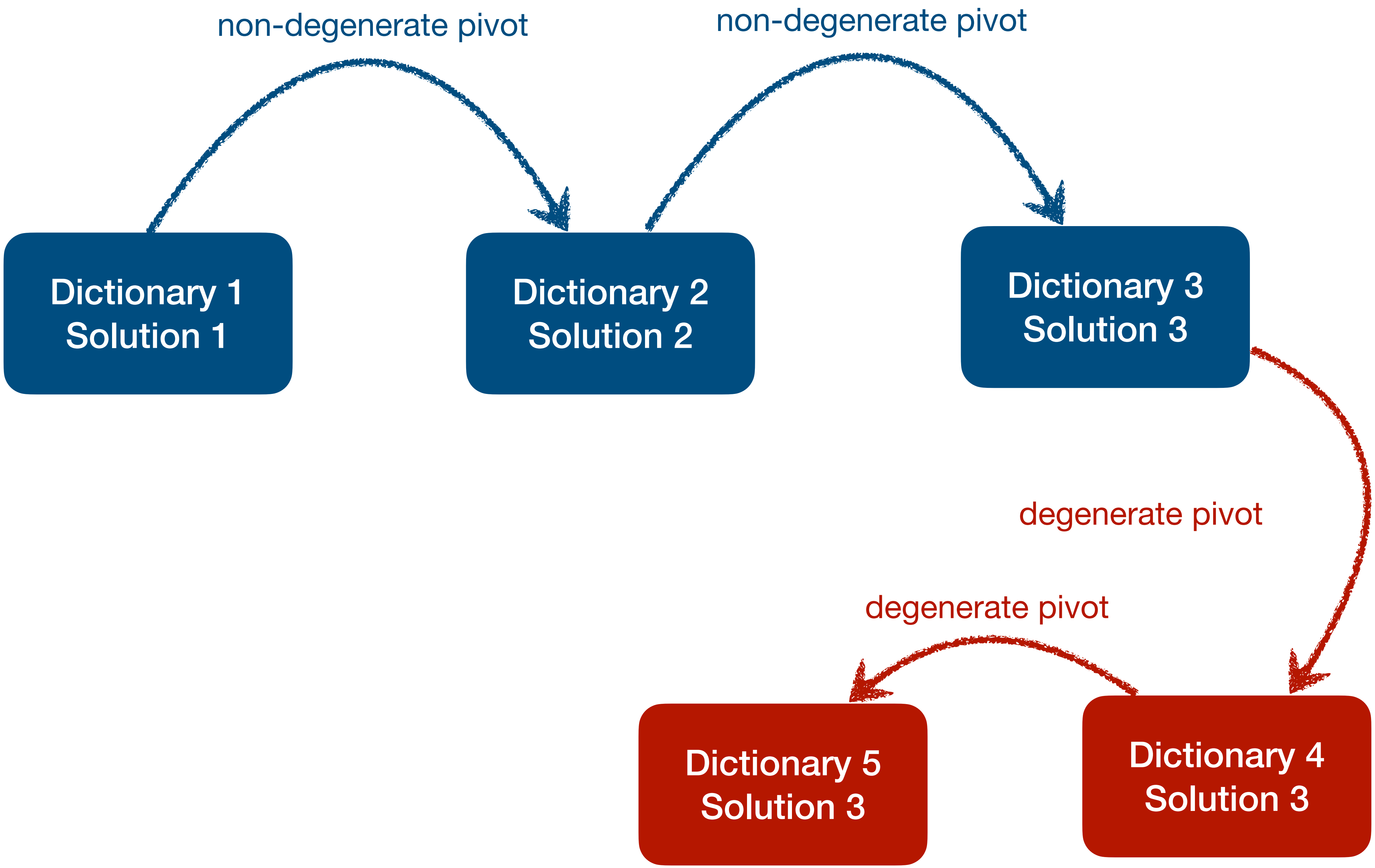
Could this happen though?



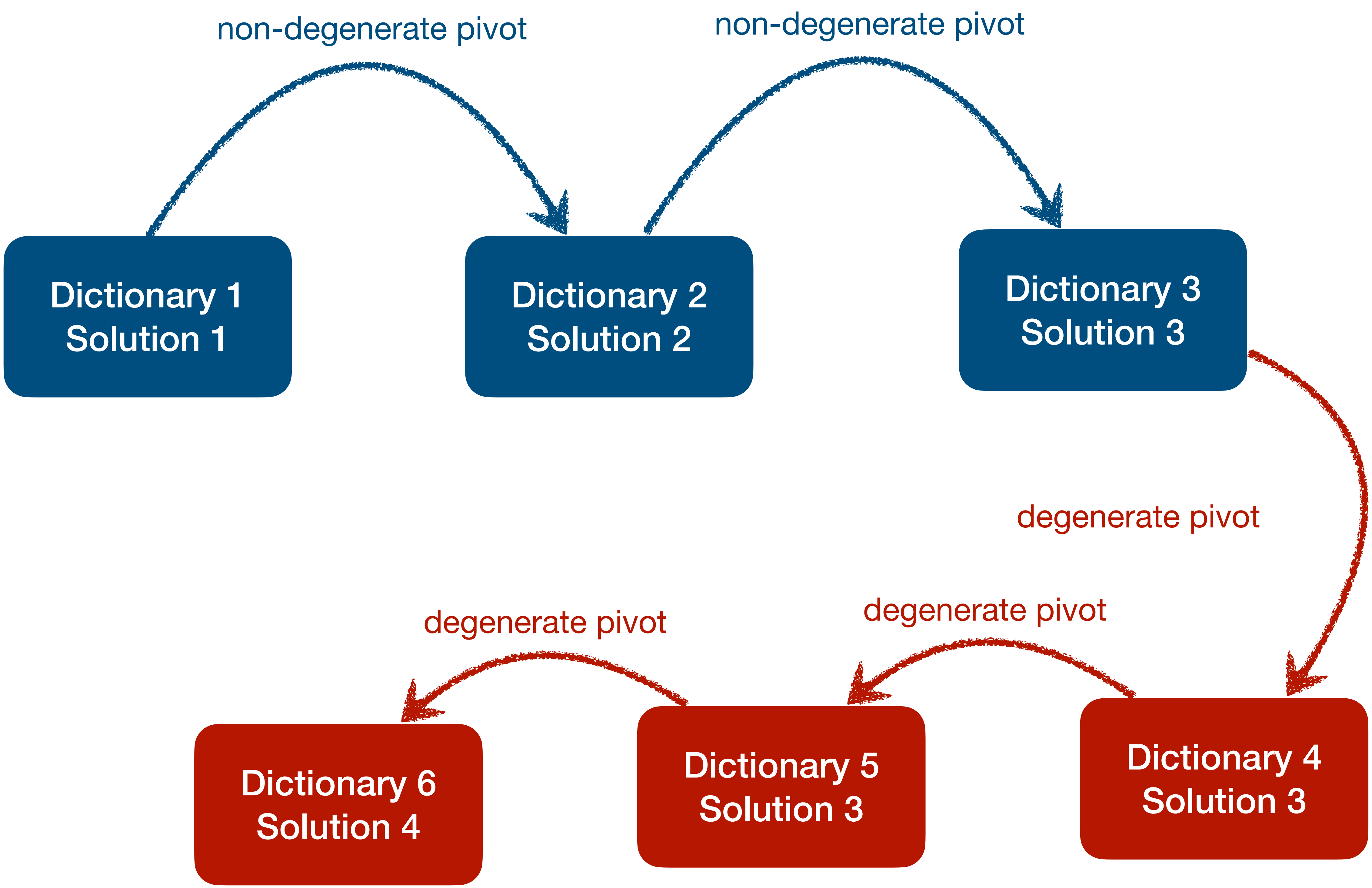
Could this happen though?



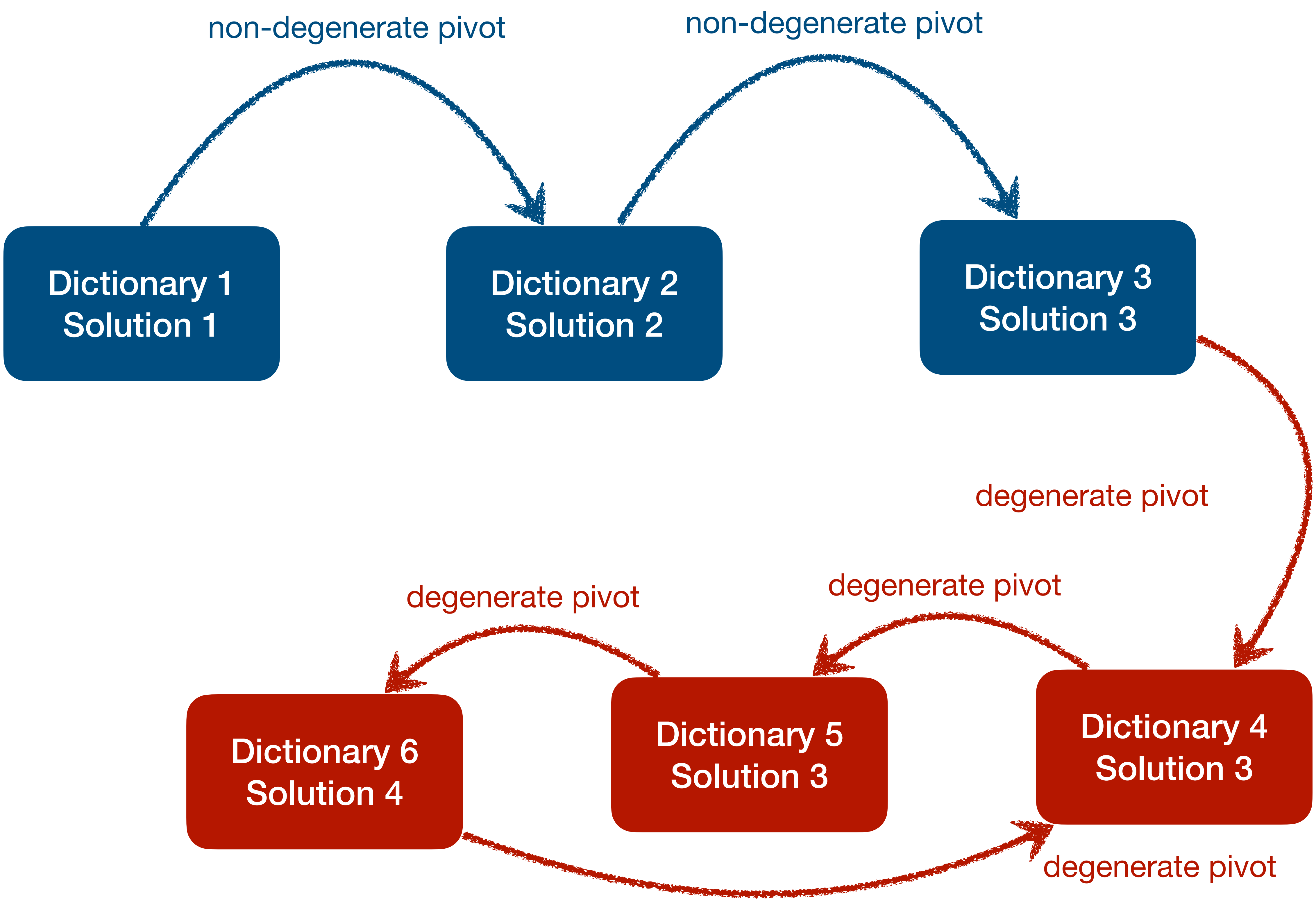
Could this happen though?



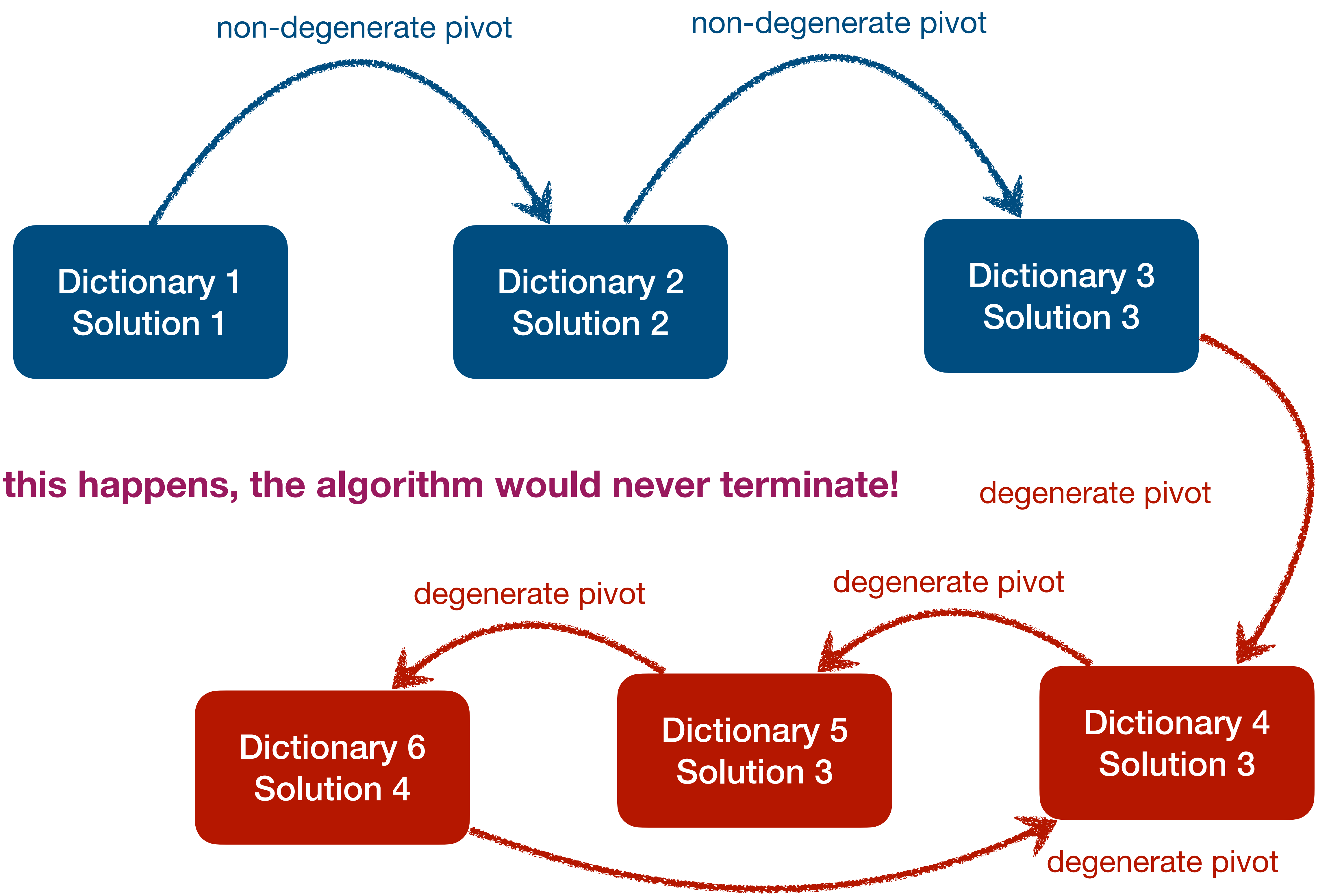
Could this happen though?



Could this happen though?



Could this happen though?



If this happens, the algorithm would never terminate!

Cycling

Cycling

In theory: Cycling can happen.

Cycling

In theory: Cycling can happen.

In practice: Cycling rarely happens.

Cycling

In theory: Cycling can happen.

In practice: Cycling rarely happens.

But non-degenerate pivots are quite common.

Cycling

In theory: Cycling can happen.

In practice: Cycling rarely happens.

But non-degenerate pivots are quite common.

Can we avoid cycling in theory too?

Cycling

In theory: Cycling can happen.

In practice: Cycling rarely happens.

But non-degenerate pivots are quite common.

Can we avoid cycling in theory too?

Bland's rule: For both the entering variable and the leaving variable, choose the one with the smallest index.

Termination

Termination

Theorem: If the simplex method does not cycle, it terminates.

Termination

Theorem: If the simplex method does not cycle, it terminates.

Proof: A dictionary is determined by which variables are basic and which are non-basic.

Termination

Theorem: If the simplex method does not cycle, it terminates.

Proof: A dictionary is determined by which variables are basic and which are non-basic.

There only $\binom{n+m}{m} = \frac{(n+m)!}{n!m!}$ possibilities.

Termination

Theorem: If the simplex method does not cycle, it terminates.

Proof: A dictionary is determined by which variables are basic and which are non-basic.

There only $\binom{n+m}{m} = \frac{(n+m)!}{n!m!}$ possibilities.

Geometry

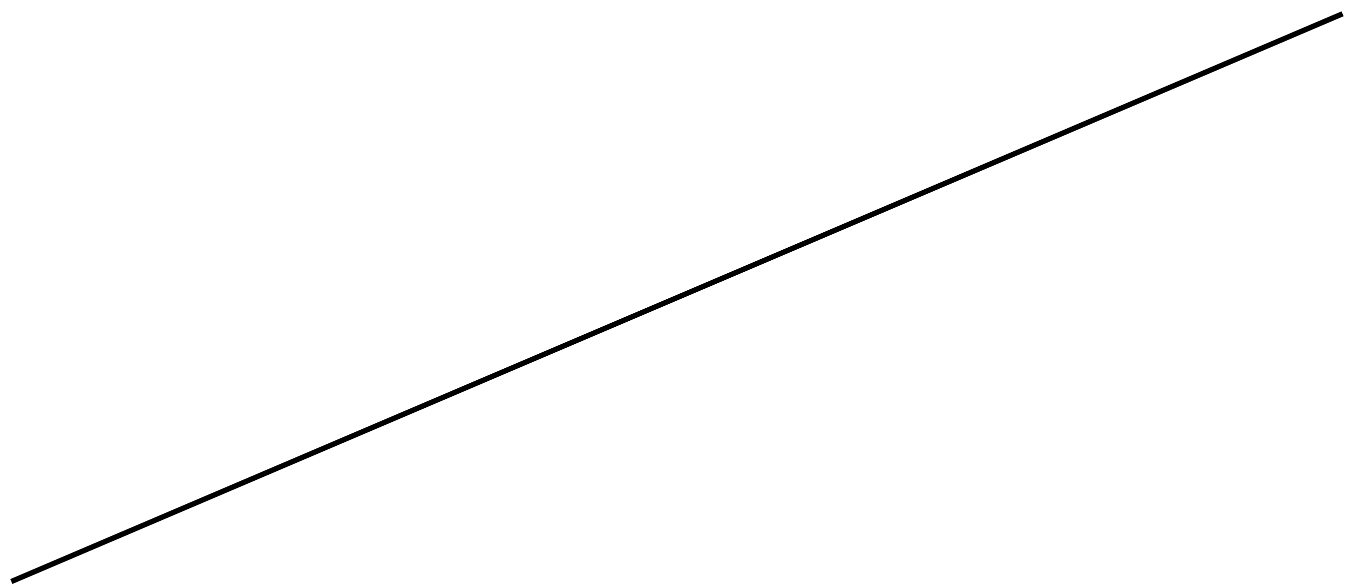
Every constraint corresponds to a hyperplane, which defines a halfspace.

The intersection of those halfspaces is the feasible region, which is a polyhedron (or polytope).

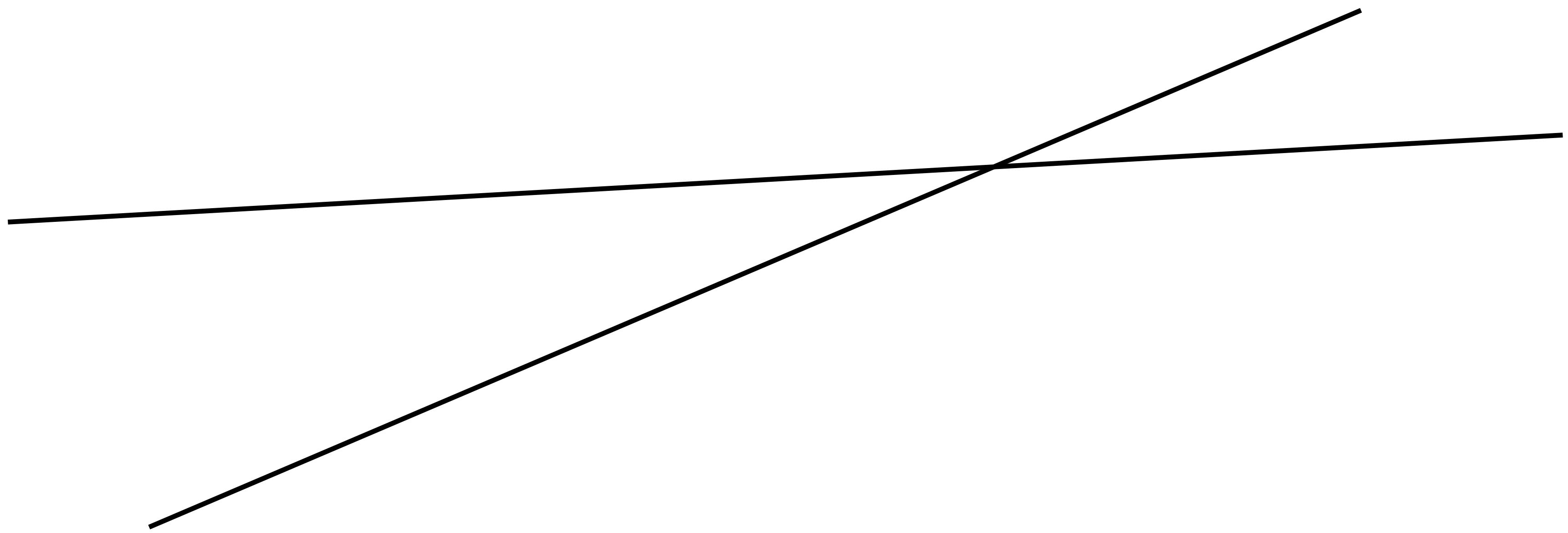
The part of each hyperplane that intersects with the feasible region is called a facet.

Geometry

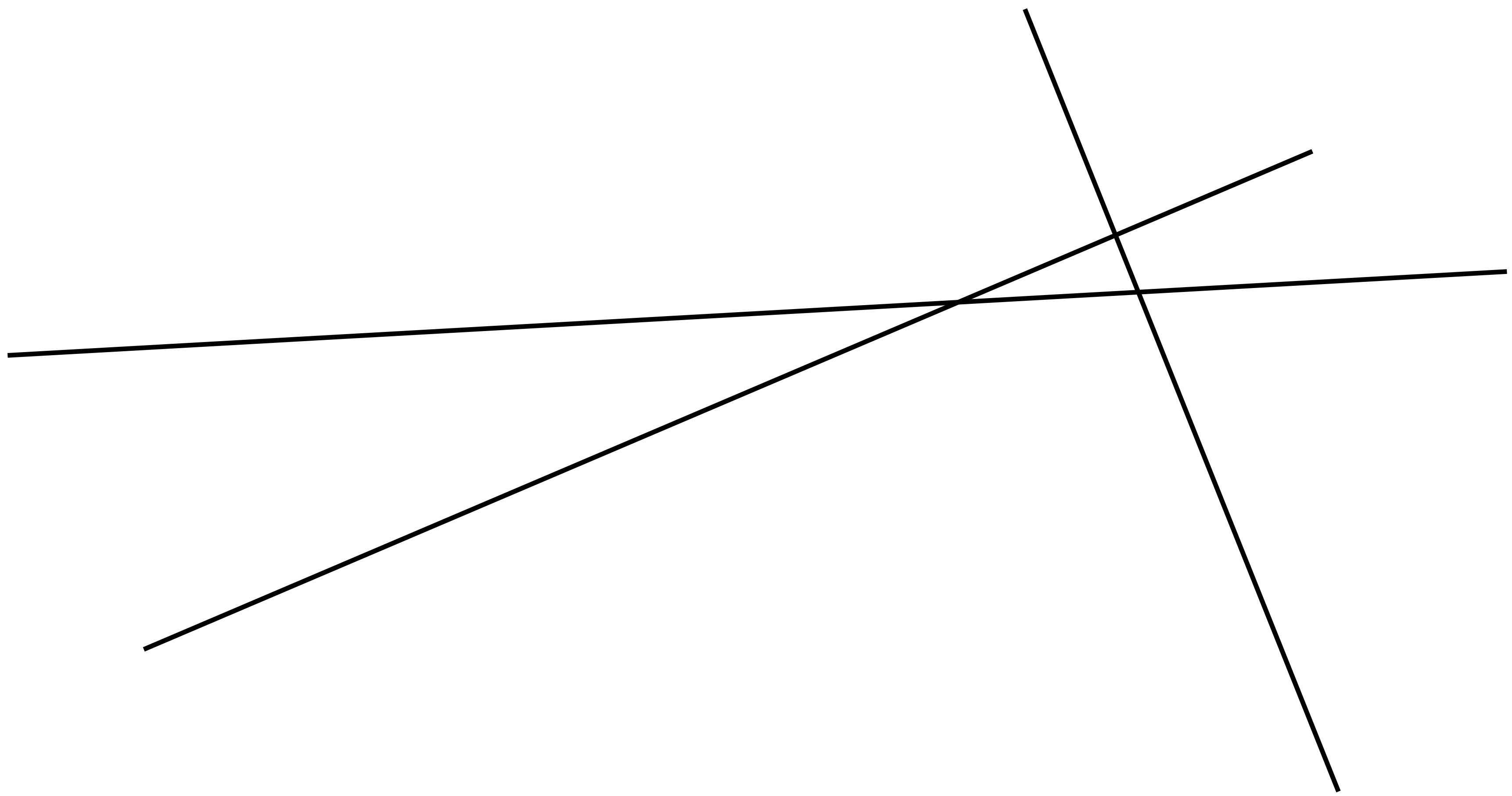
Geometry



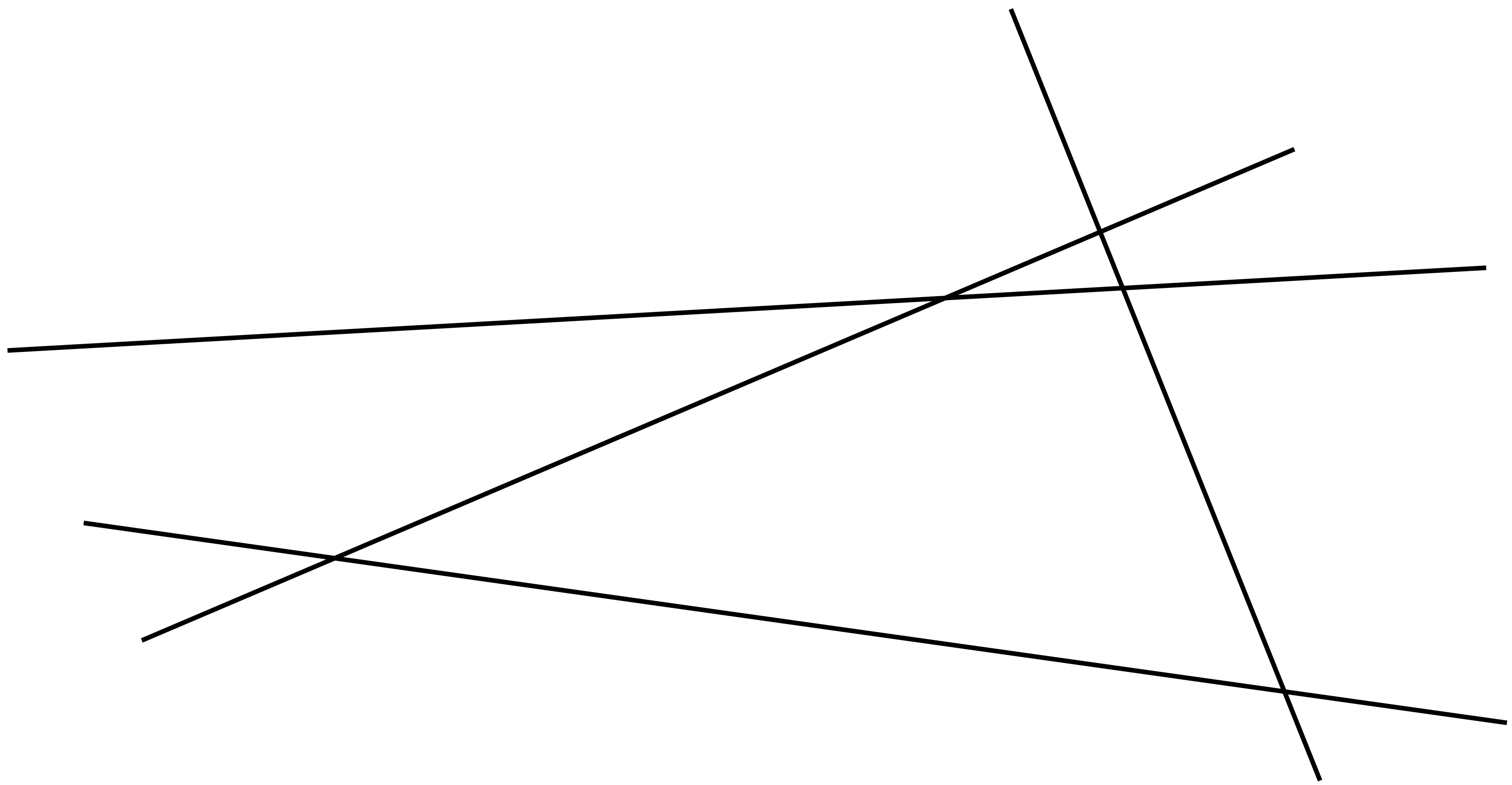
Geometry



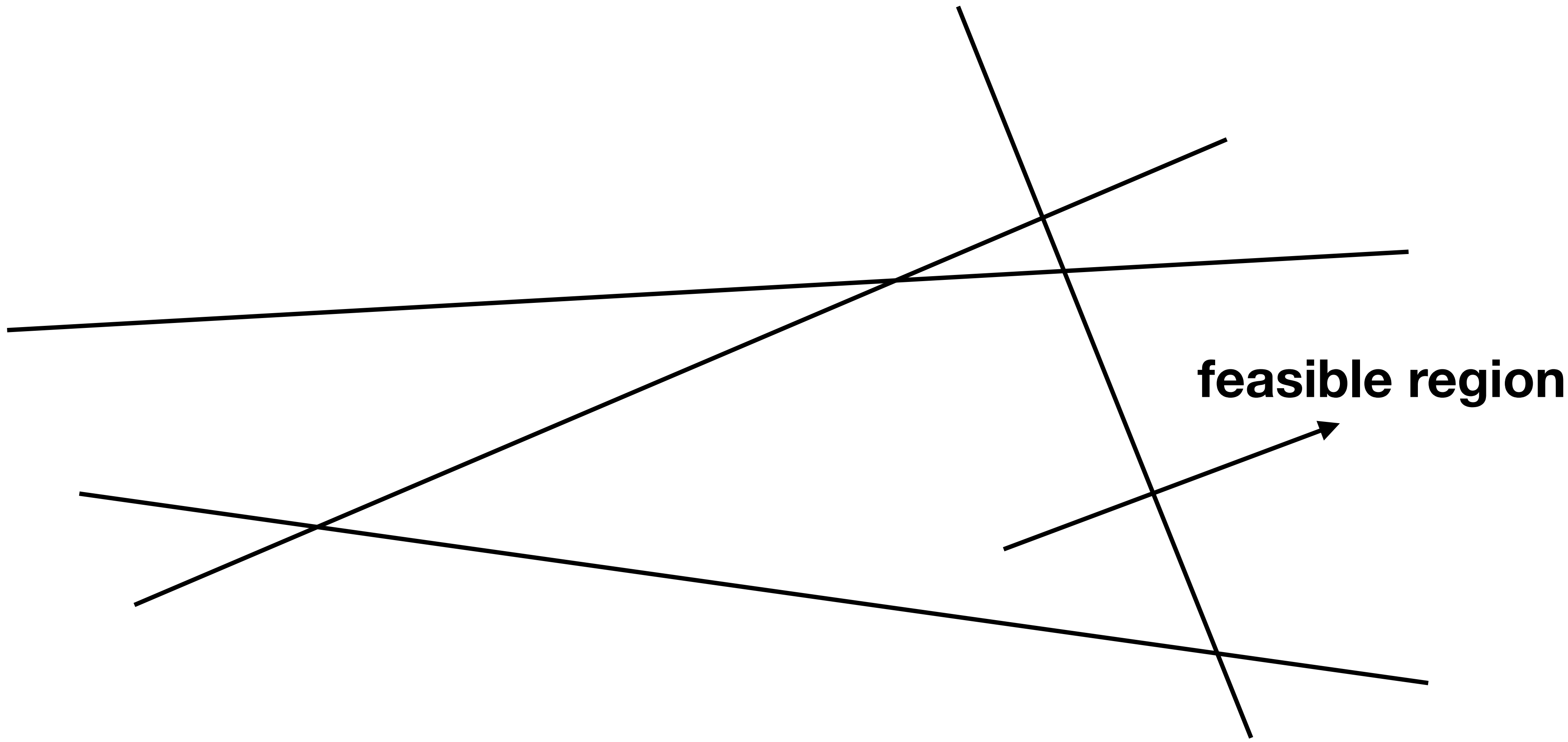
Geometry



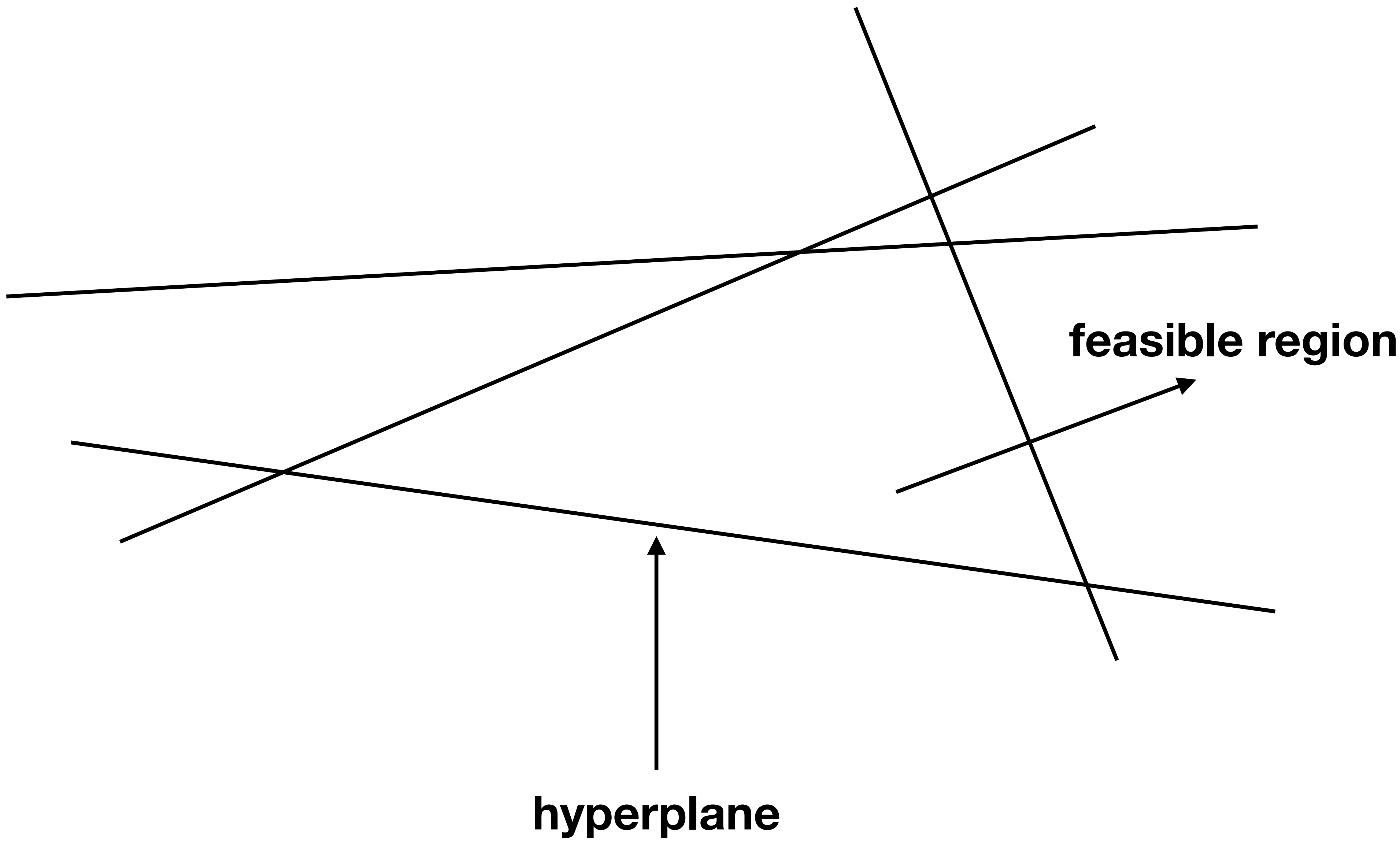
Geometry



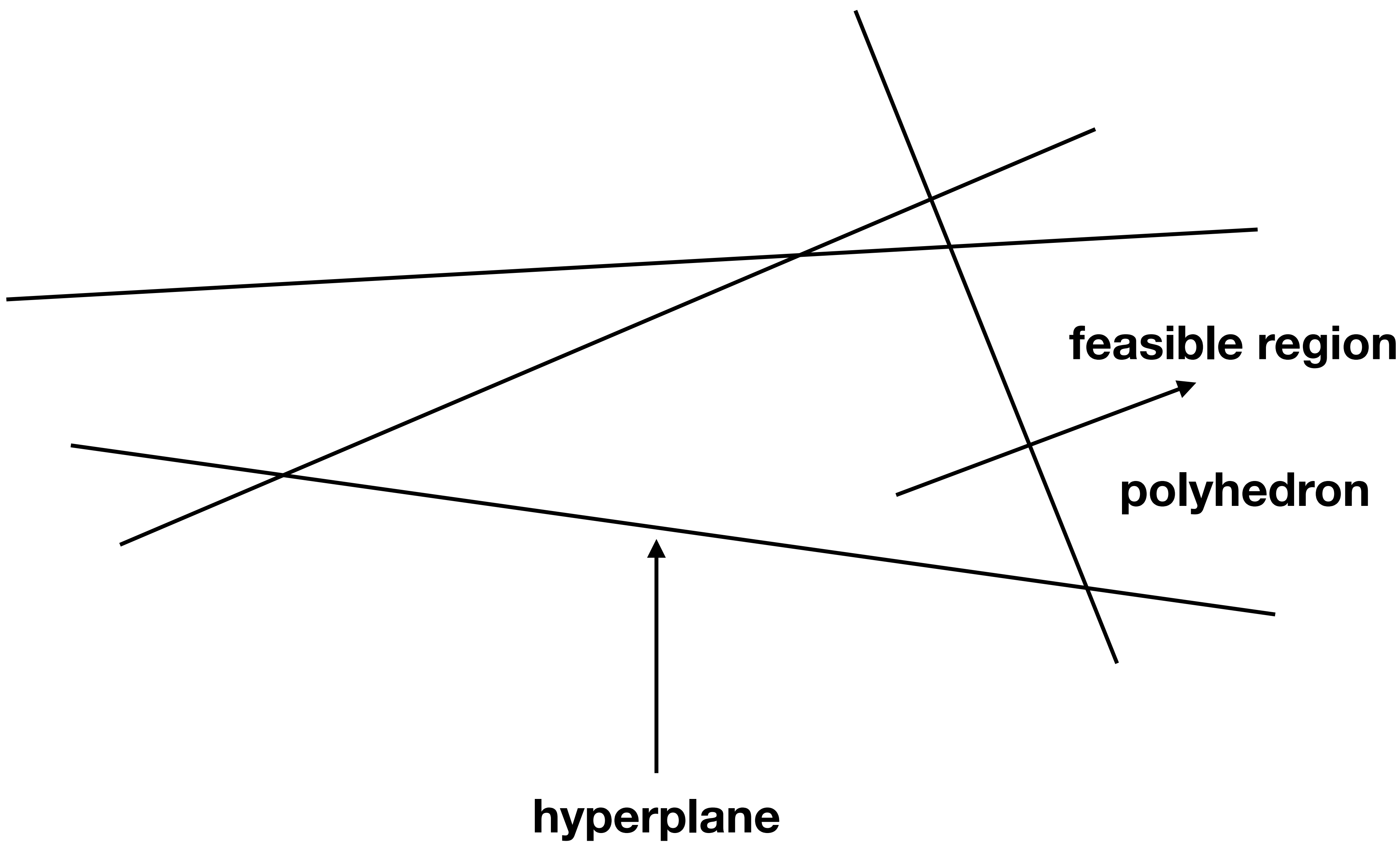
Geometry



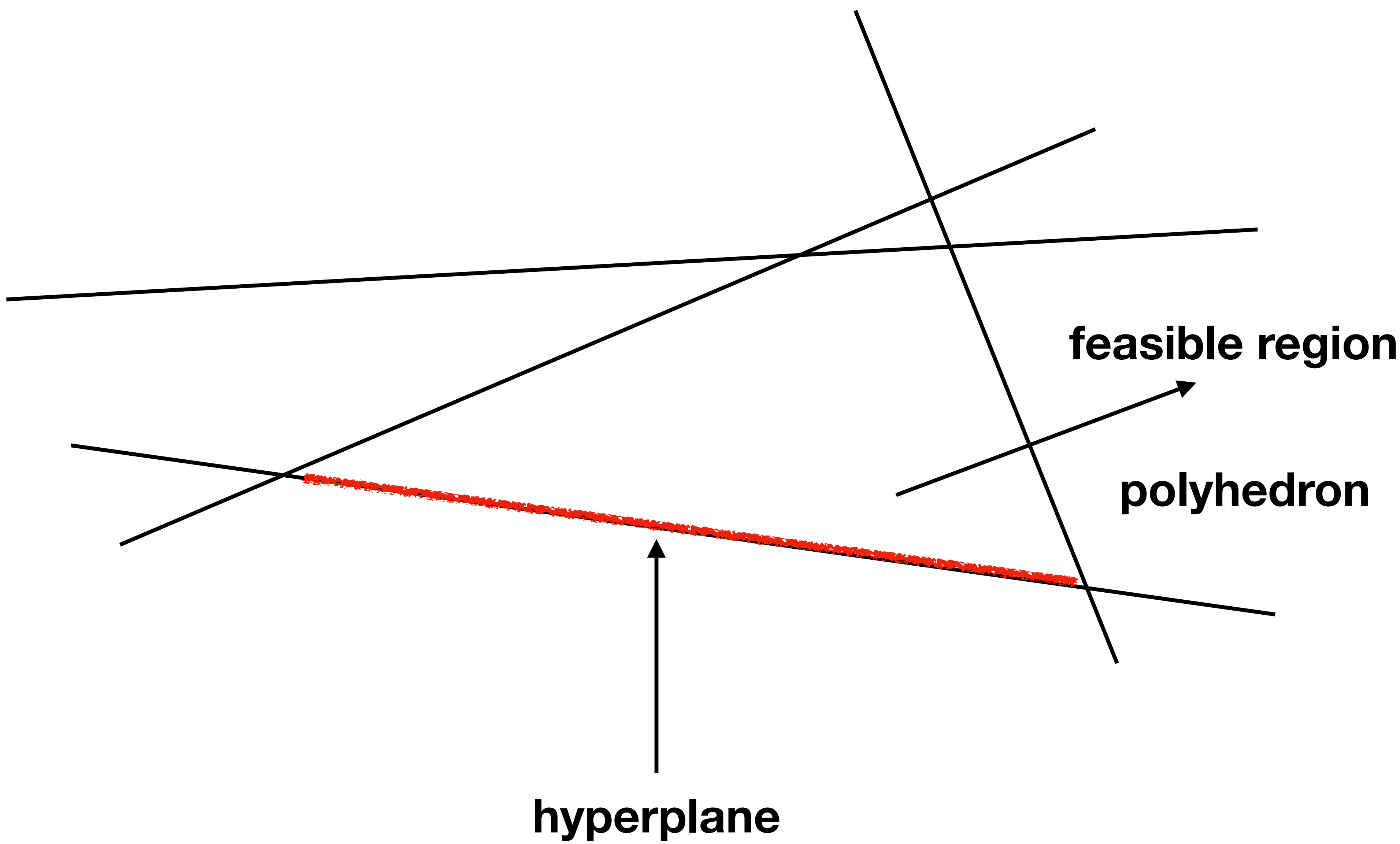
Geometry



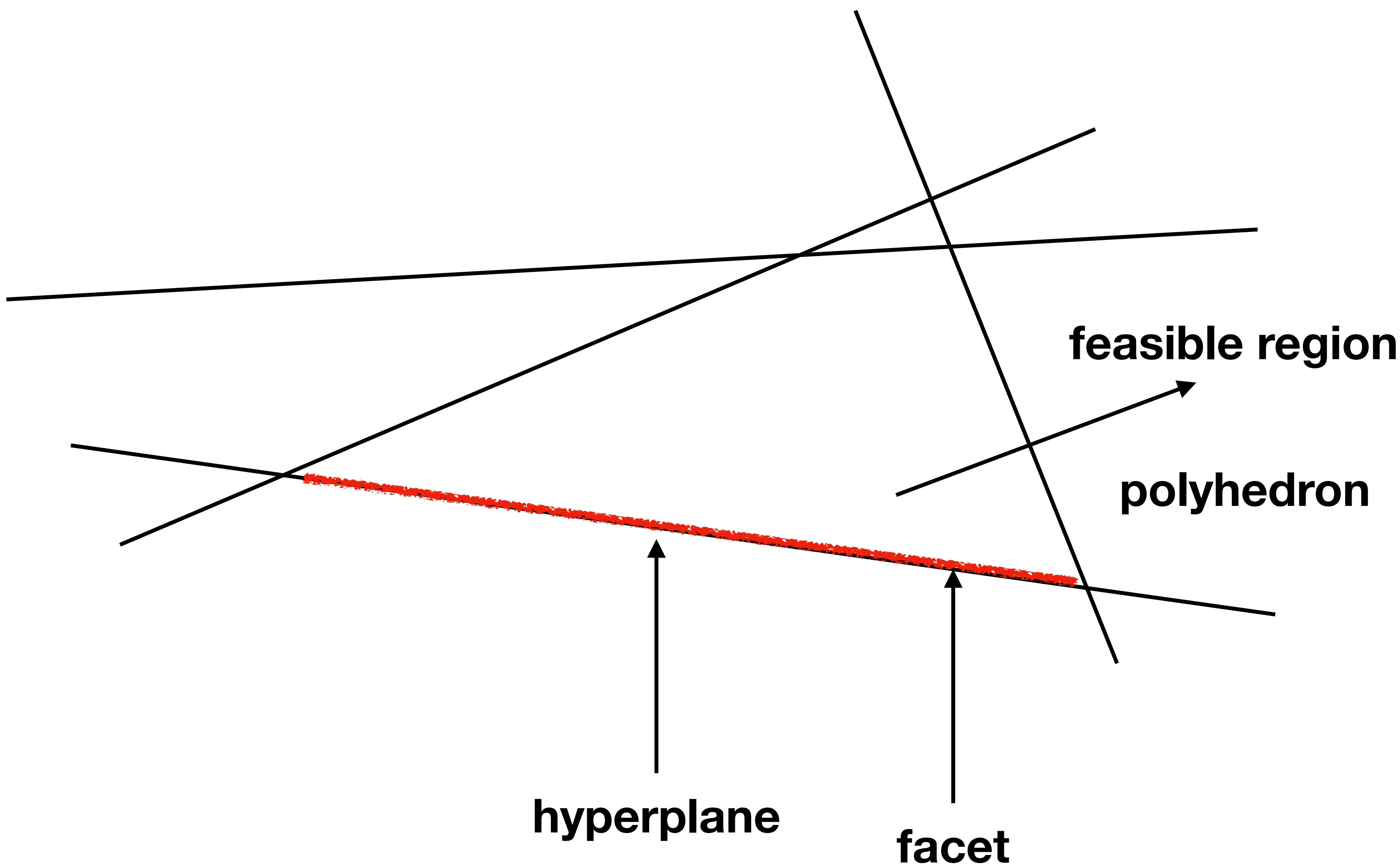
Geometry



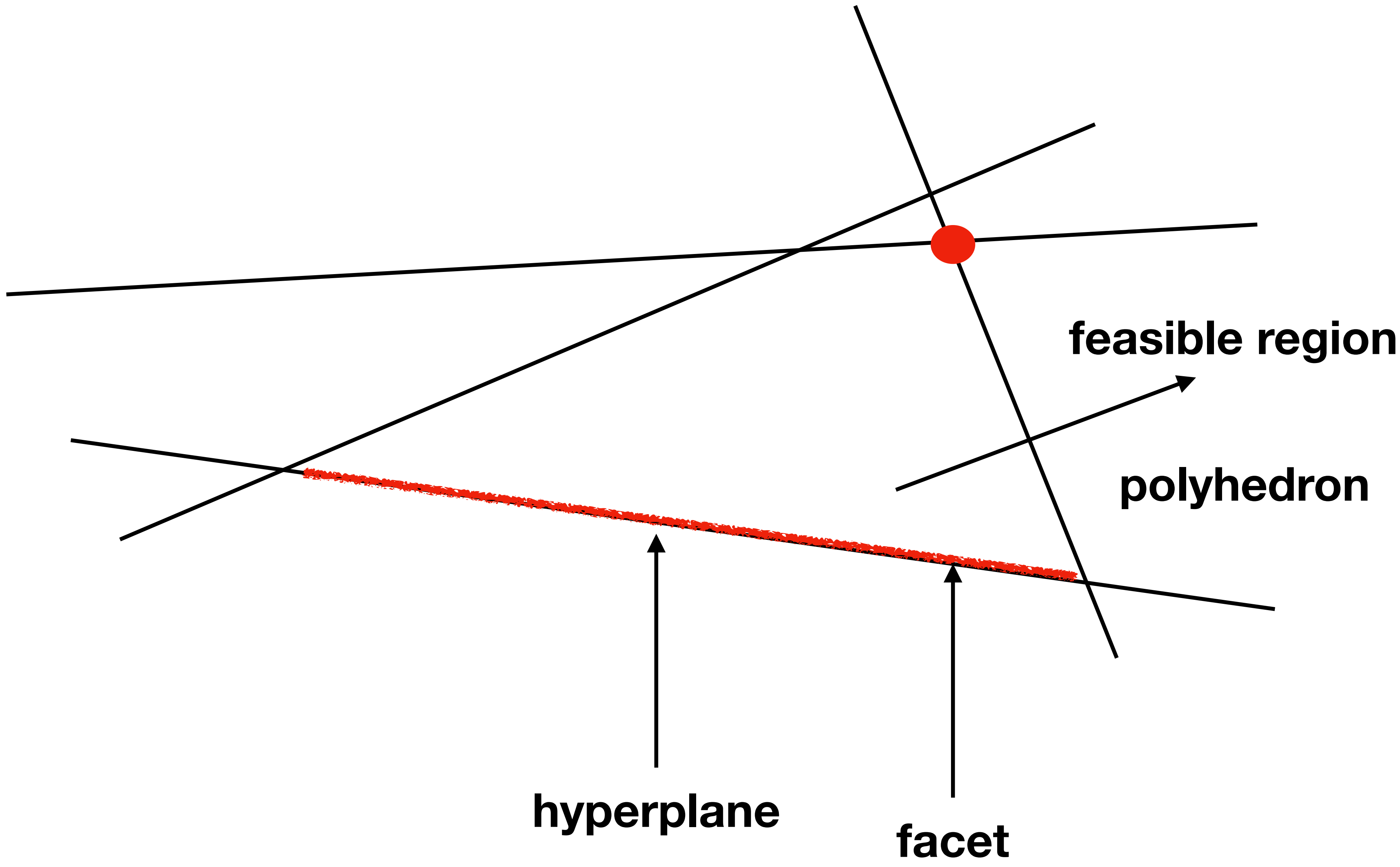
Geometry



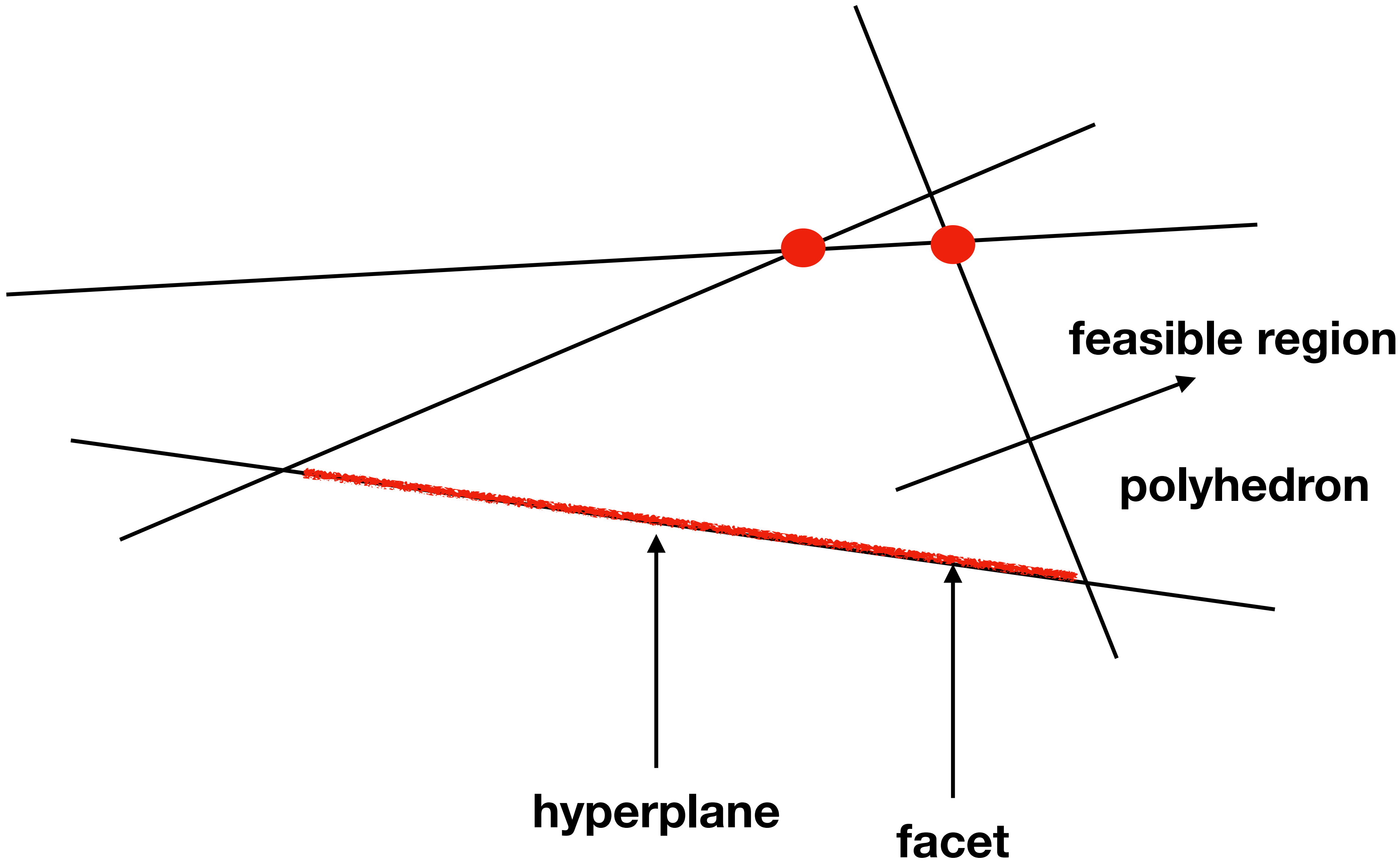
Geometry



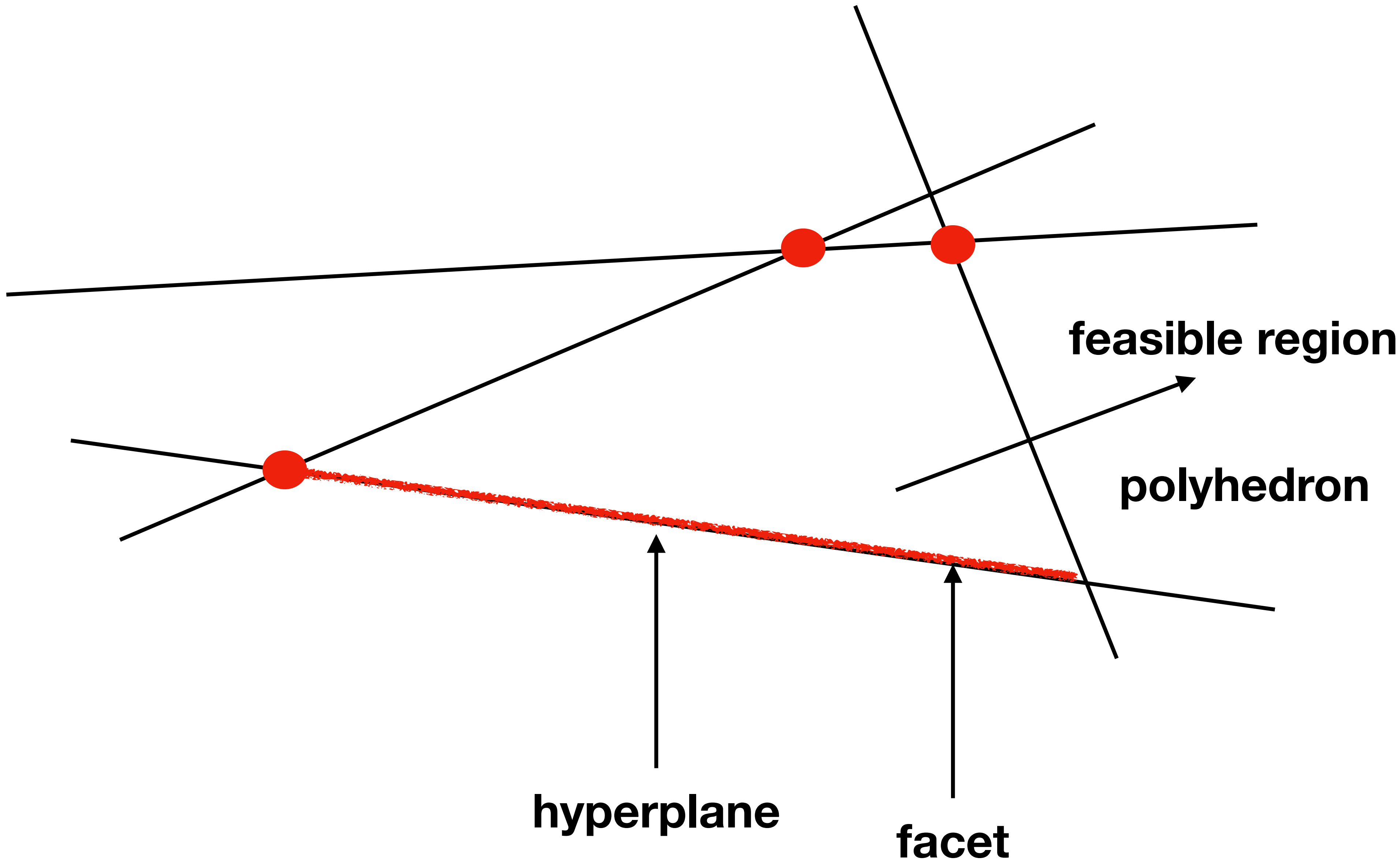
Geometry



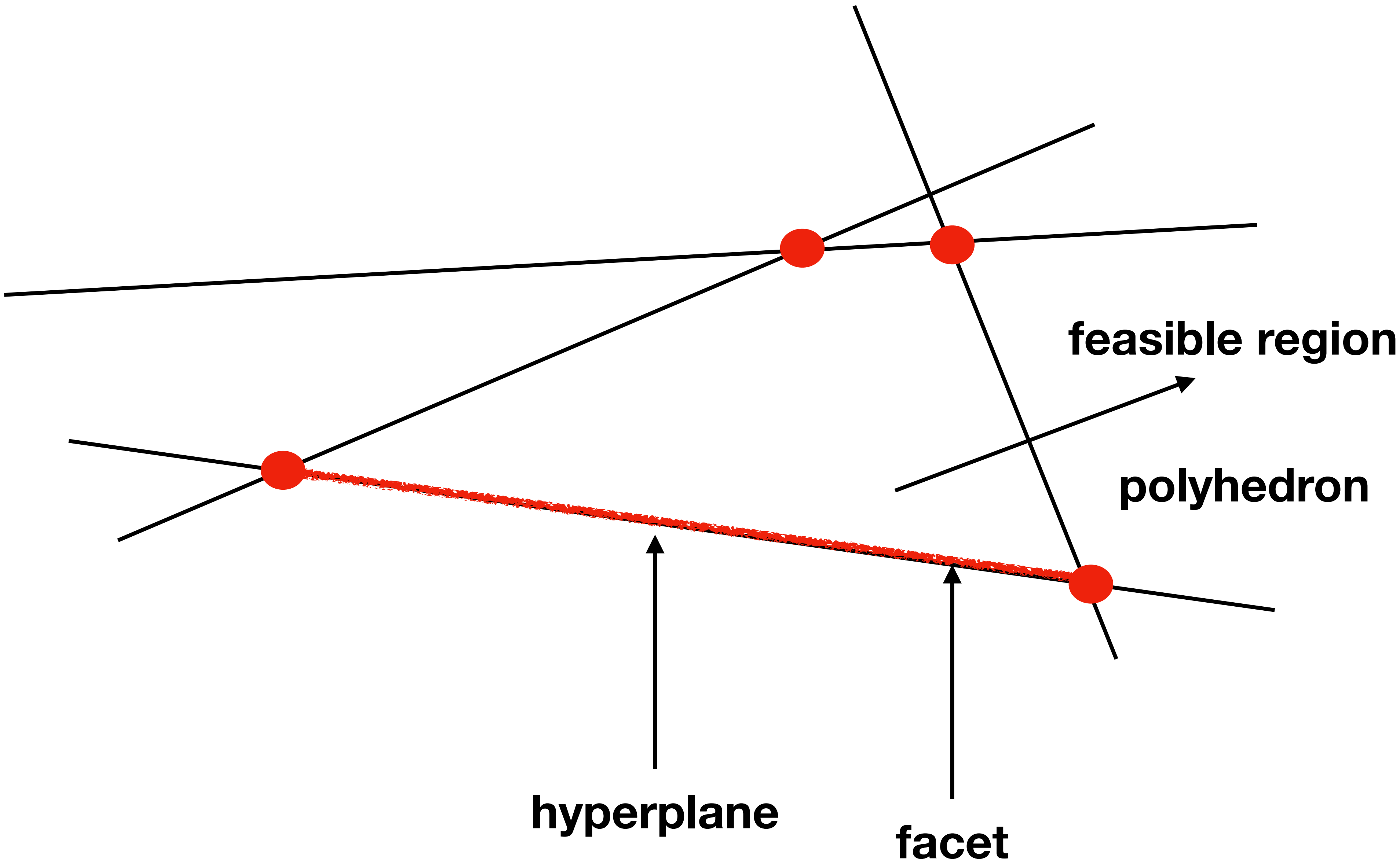
Geometry



Geometry



Geometry



Geometry

Every constraint corresponds to a hyperplane, which defines a halfspace.

The intersection of those halfspaces is the feasible region, which is a polyhedron (or polytope).

The part of each hyperplane that intersects with the feasible region is called a facet.

Geometry

Every constraint corresponds to a hyperplane, which defines a halfspace.

The intersection of those halfspaces is the feasible region, which is a polyhedron (or polytope).

The part of each hyperplane that intersects with the feasible region is called a facet.

A facet corresponds to a constraint satisfied with equality, e.g., $x_1 + 2x_3 = 3$

Geometry

Every constraint corresponds to a hyperplane, which defines a halfspace.

The intersection of those halfspaces is the feasible region, which is a polyhedron (or polytope).

The part of each hyperplane that intersects with the feasible region is called a facet.

A facet corresponds to a constraint satisfied with equality, e.g., $x_1 + 2x_3 = 3$

In terms of the dictionary, a facet corresponds to the corresponding variable (original or slack) being 0.

Geometry

Consider an LP with three variables x_1, x_2, x_3.

In terms of the dictionary, a facet corresponds to the corresponding variable (original or slack) being 0.

An edge corresponds to two variables being 0.

A vertex corresponds to three variables being 0.

Geometry

Maximise $5x_1 + 4x_2 + 3x_3$

subject to

$$\left.\begin{array}{l} 2x_1 + 3x_2 + x_3 \leq 5 \\ 4x_1 + x_2 + 2x + 3 \leq 11 \\ 3x_1 + 4x_2 + 2x_3 \leq 8 \end{array}\right\} \begin{array}{c} w_1 = w_2 = w_3 = 0 \\ \text{corresponds to the intersection of} \\ \text{these three hyperplanes.} \end{array}$$

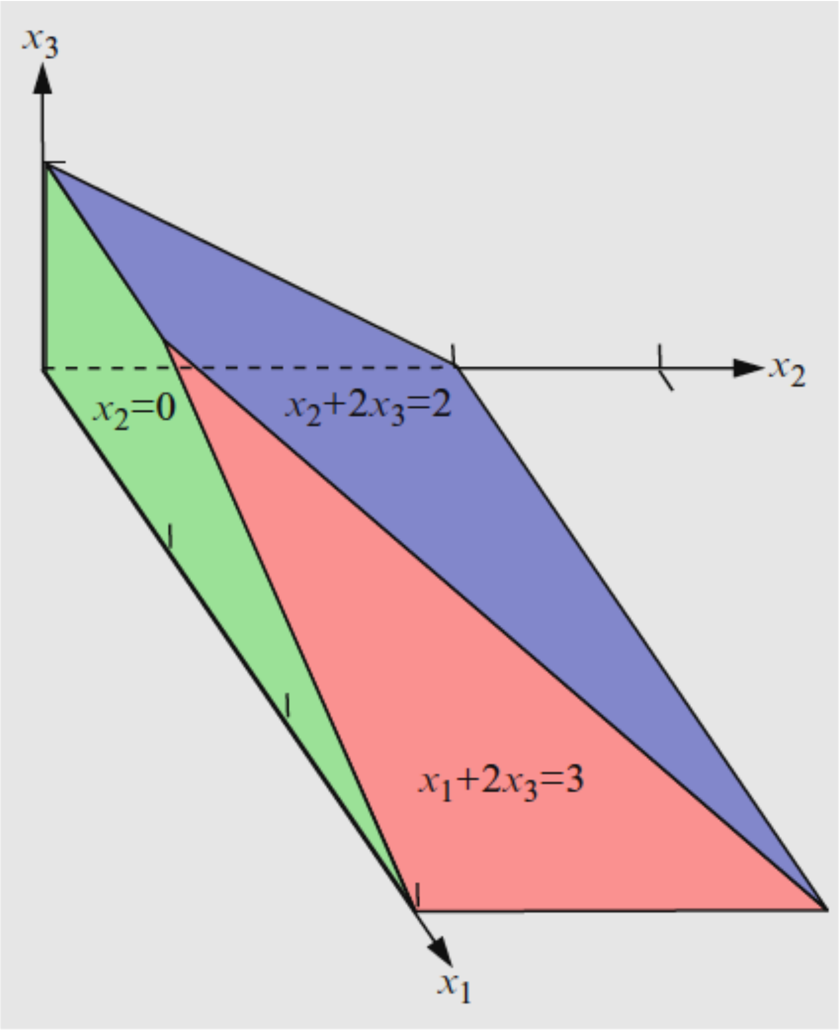
$$x_1, x_2, x_3 \geq 0$$

Example

Maximise $x_1 + 2x_2 + 3x_3$

subject to

$$x_1 + 2x_3 \leq 3$$
$$x_2 + 2x_3 \leq 2$$
$$x_1, x_2, x_3 \geq 0$$

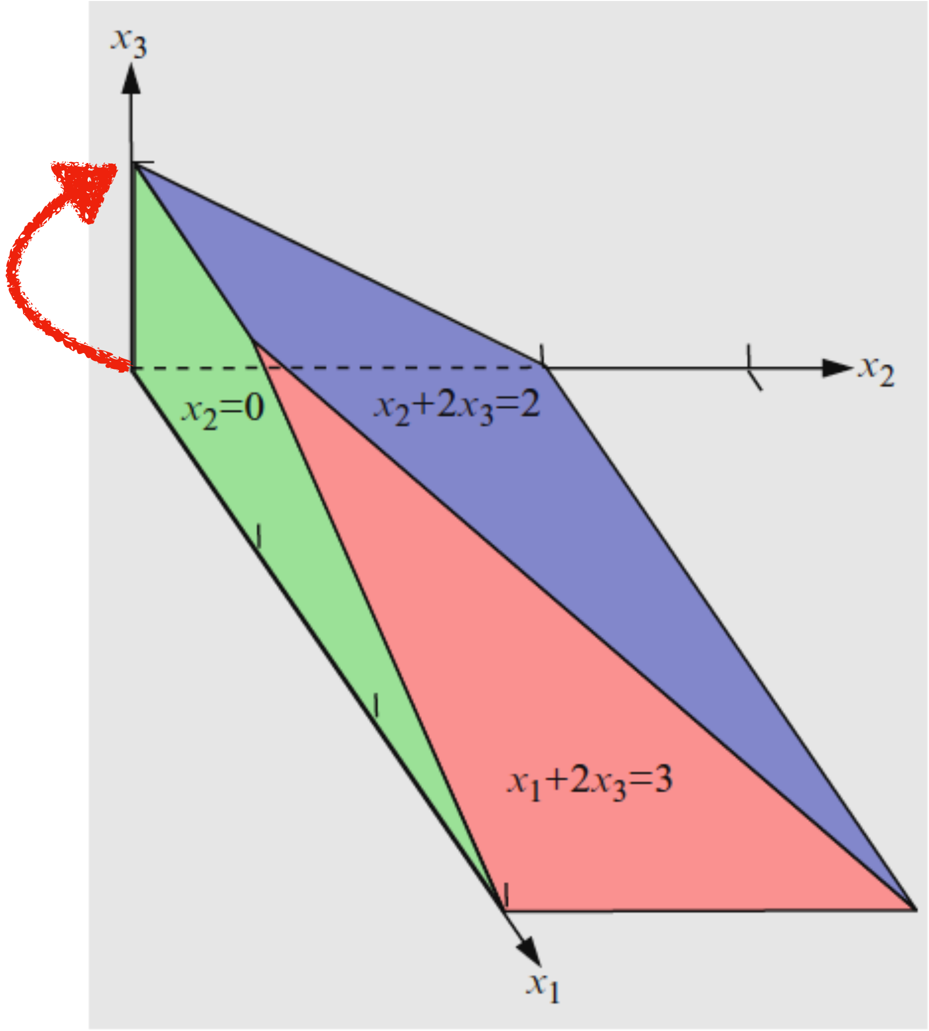


Example

Maximise $x_1 + 2x_2 + 3x_3$

subject to

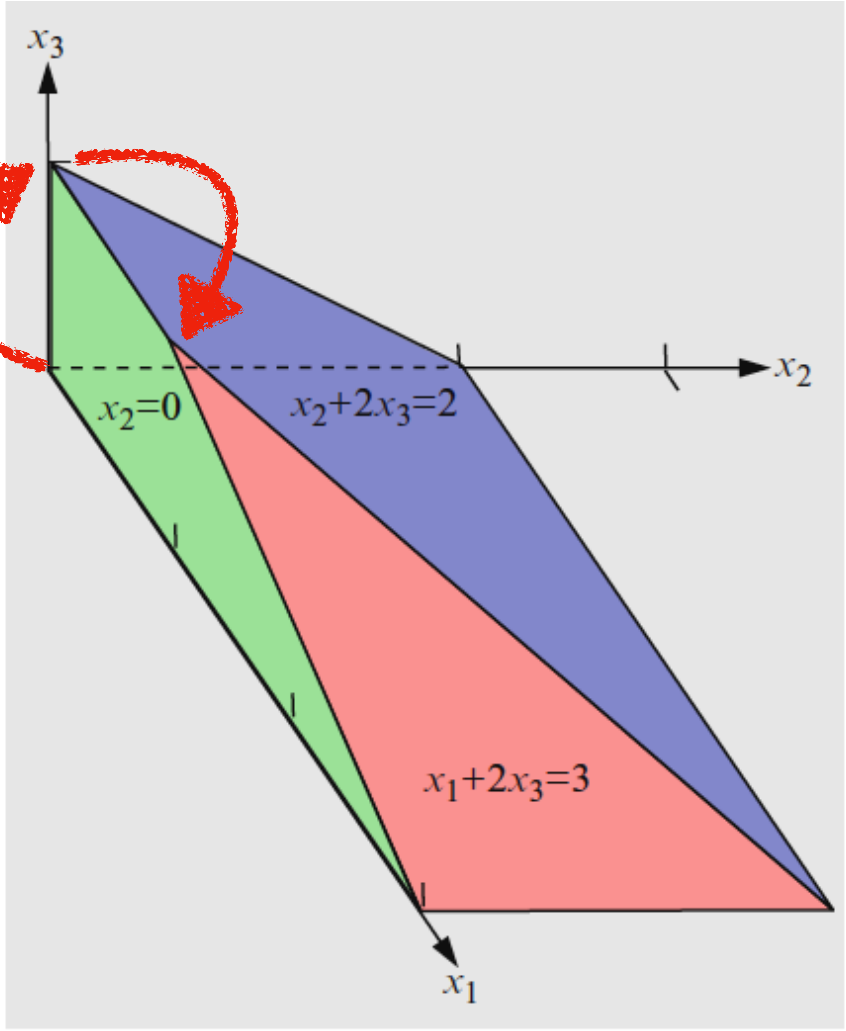
$$x_1 + 2x_3 \leq 3$$
$$x_2 + 2x_3 \leq 2$$
$$x_1, x_2, x_3 \geq 0$$



Example

Maximise $$x_1 + 2x_2 + 3x_3$$

subject to
$$x_1 + 2x_3 \leq 3$$
$$x_2 + 2x_3 \leq 2$$
$$x_1, x_2, x_3 \geq 0$$

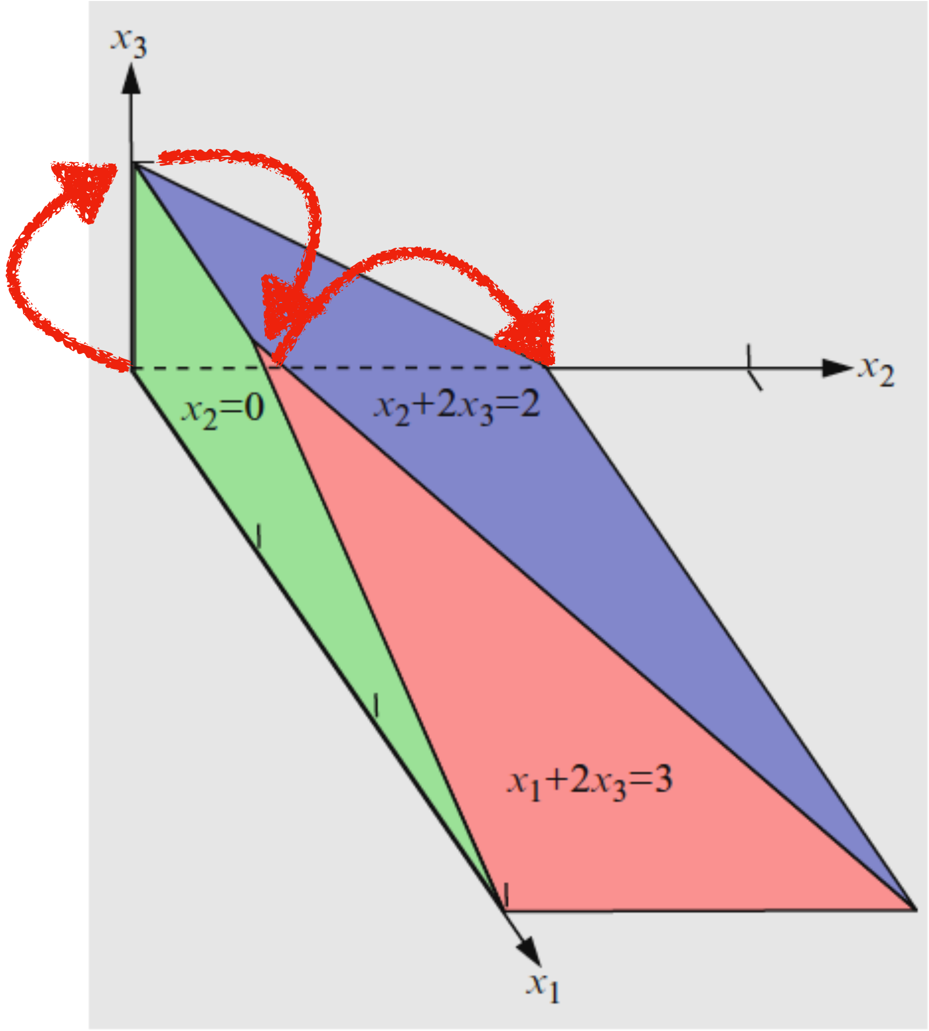


Example

Maximise $x_1 + 2x_2 + 3x_3$

subject to

$$x_1 + 2x_3 \leq 3$$
$$x_2 + 2x_3 \leq 2$$
$$x_1, x_2, x_3 \geq 0$$

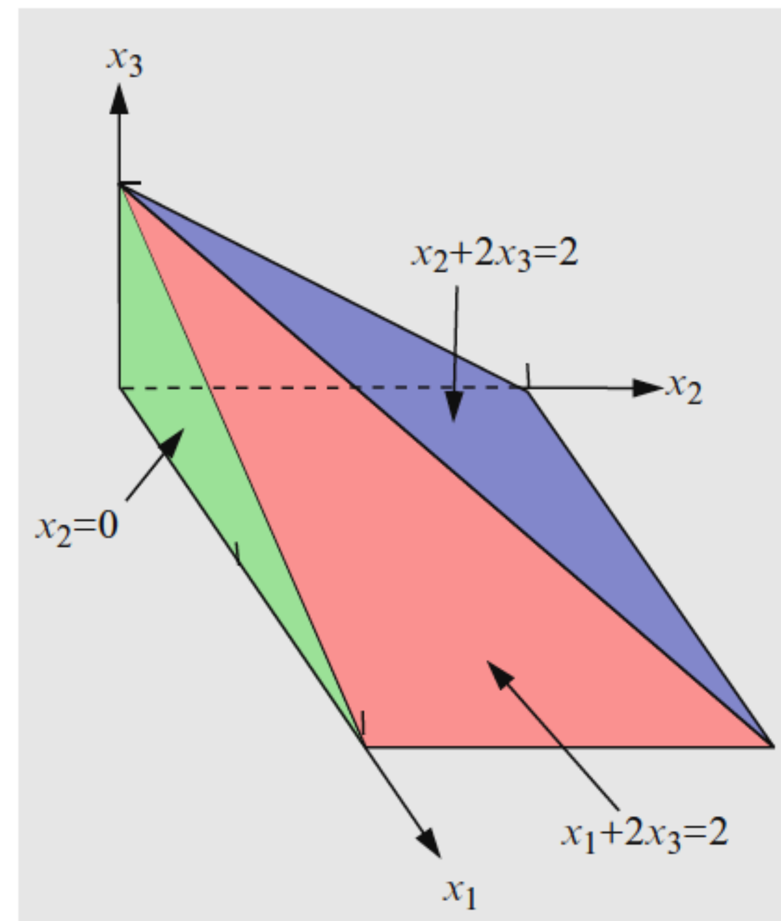


Example

Maximise $x_1 + 2x_2 + 3x_3$

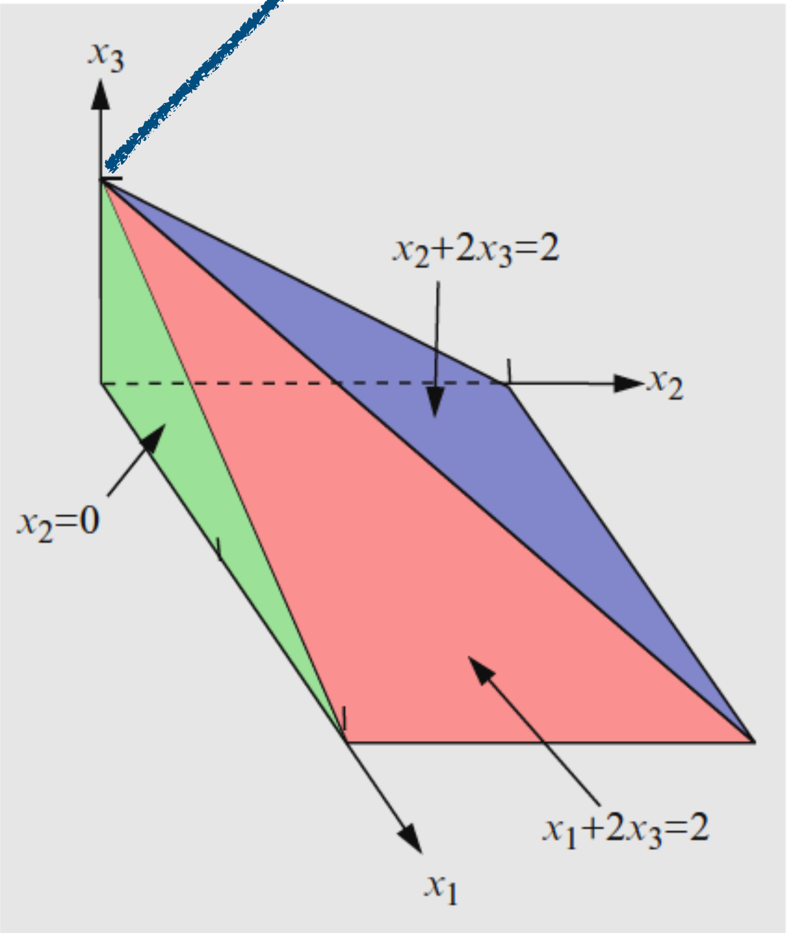
subject to

$$x_1 + 2x_3 \le 2$$
$$x_2 + 2x_3 \le 2$$
$$x_1, x_2, x_3 \ge 0$$



Example

intersection of four facets



Maximise $\quad x_1 + 2x_2 + 3x_3$

subject to $\quad x_1 + 2x_3 \leq 2$

$$x_2 + 2x_3 \leq 2$$

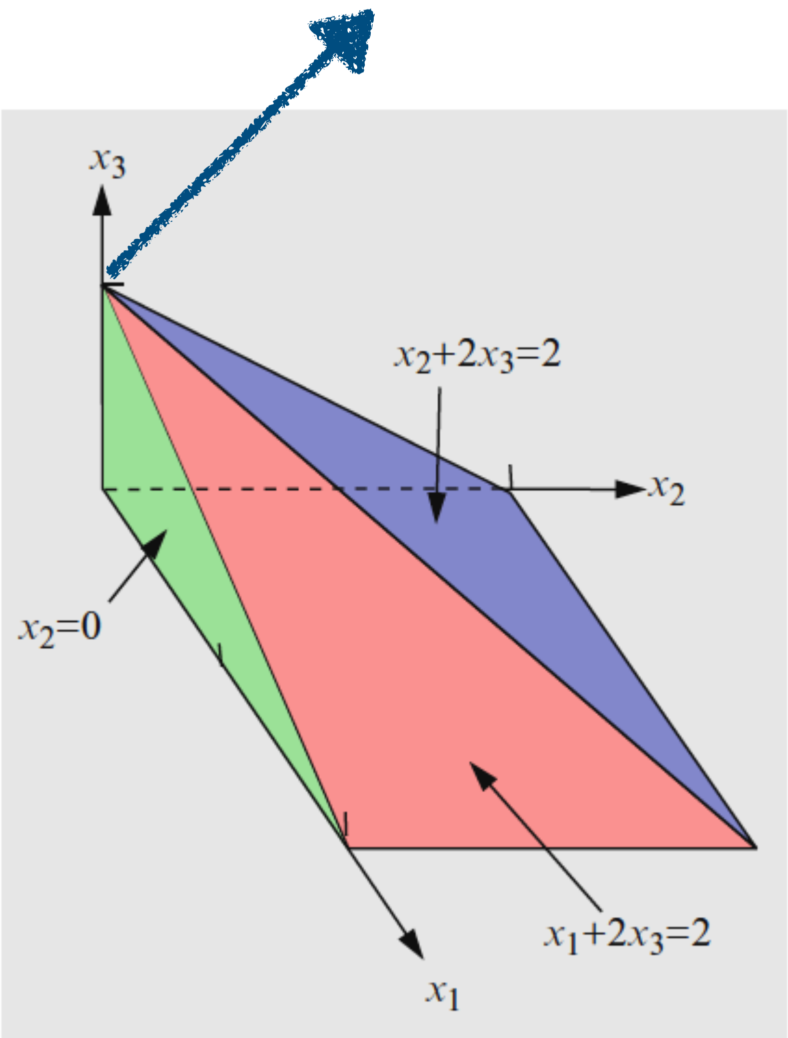
$$x_1, x_2, x_3 \geq 0$$

Example

intersection of four facets

Maximise $x_1 + 2x_2 + 3x_3$

subject to
$$x_1 + 2x_3 \leq 2$$
$$x_2 + 2x_3 \leq 2$$
$$x_1, x_2, x_3 \geq 0$$



The intersection point corresponds to the same solution of the LP.

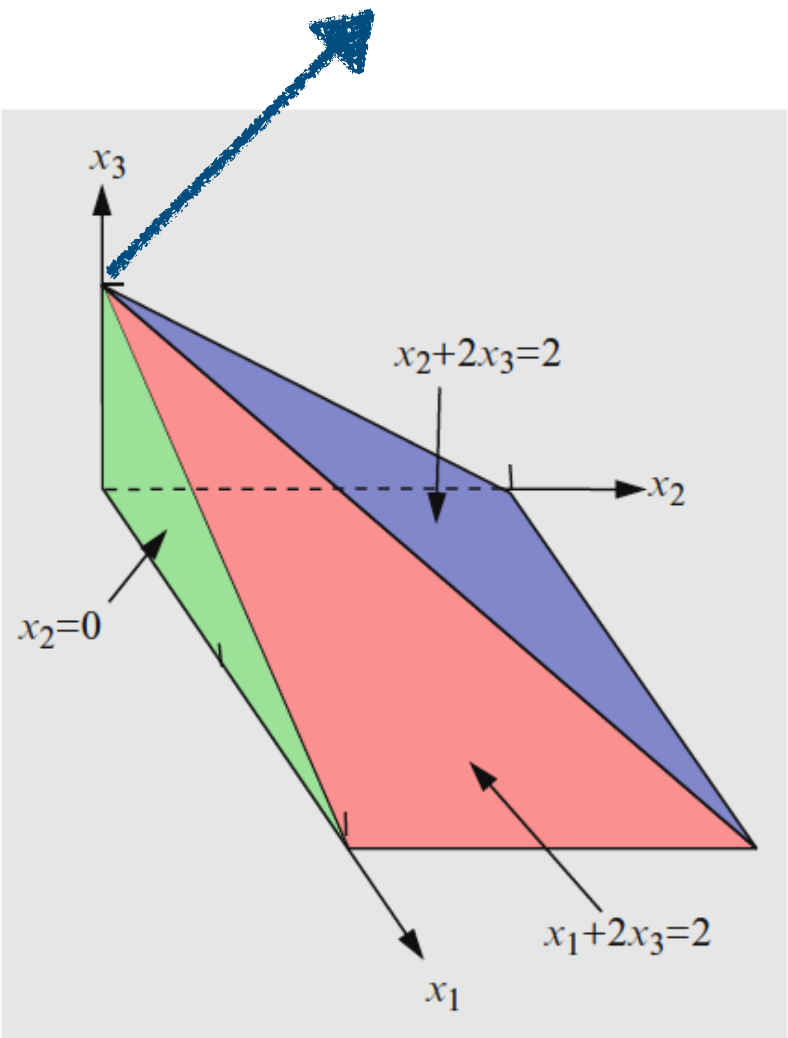
Example

Maximise $x_1 + 2x_2 + 3x_3$

subject to

$$x_1 + 2x_3 \leq 2$$
$$x_2 + 2x_3 \leq 2$$
$$x_1, x_2, x_3 \geq 0$$

intersection of four facets



The intersection point corresponds to the same solution of the LP.

But it corresponds to four different basic feasible solutions/dictionaries.

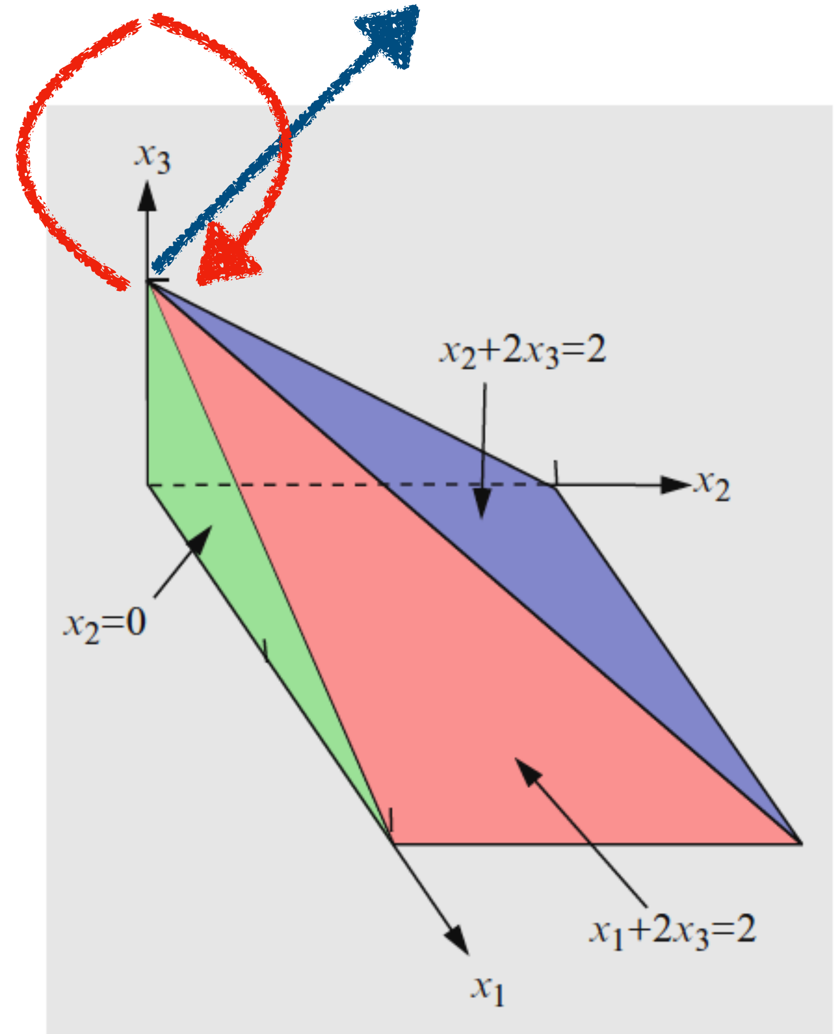
Example

Maximise $x_1 + 2x_2 + 3x_3$

subject to

$$x_1 + 2x_3 \leq 2$$
$$x_2 + 2x_3 \leq 2$$
$$x_1, x_2, x_3 \geq 0$$

intersection of four facets



The intersection point corresponds to the same solution of the LP.

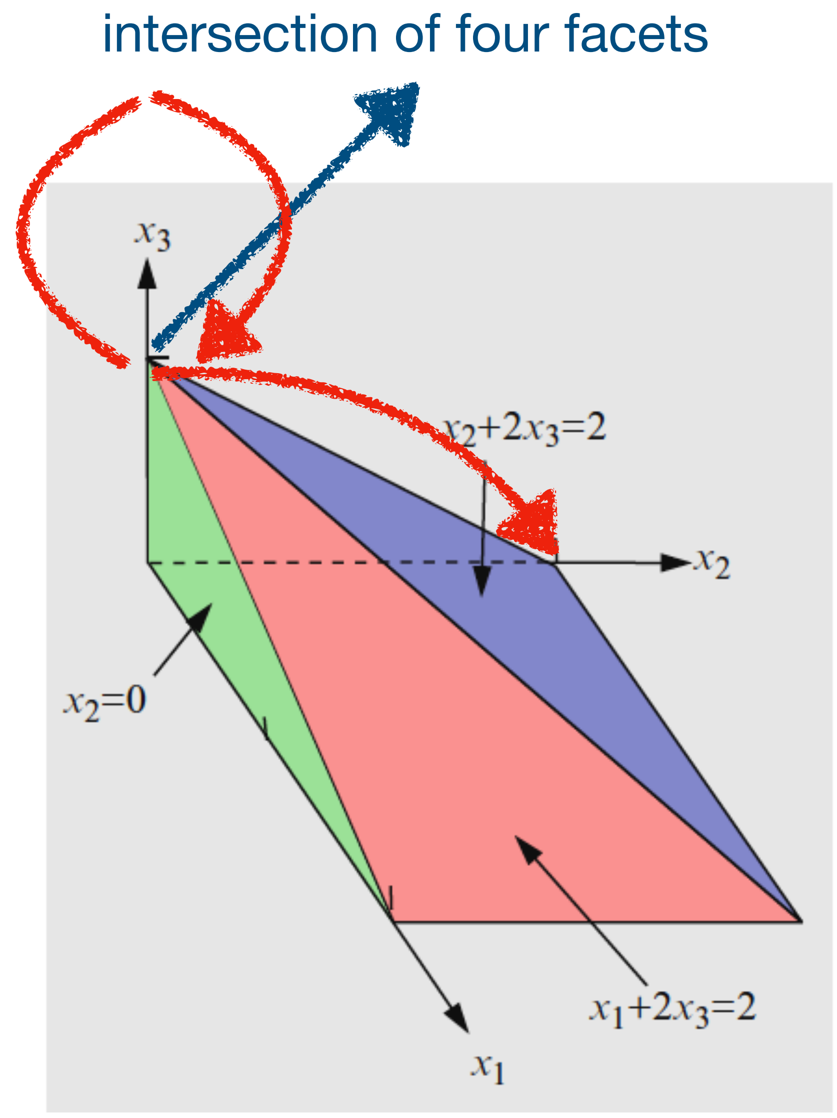
But it corresponds to four different basic feasible solutions/dictionaries.

Example

Maximise $x_1 + 2x_2 + 3x_3$

subject to

$$x_1 + 2x_3 \le 2$$
$$x_2 + 2x_3 \le 2$$
$$x_1, x_2, x_3 \ge 0$$



The intersection point corresponds to the same solution of the LP.

But it corresponds to four different basic feasible solutions/dictionaries.

Termination

Theorem: If the simplex method does not cycle, it terminates.

Proof: A dictionary is determined by which variables are basic and which are non-basic.

There only $\binom{n+m}{m} = \frac{(n+m)!}{n!m!}$ possibilities.

Simplex Running Time

Simplex Running Time

First: Simplex is a method, not an algorithm, parameterised by the pivoting rule.

Simplex Running Time

First: Simplex is a method, not an algorithm, parameterised by the pivoting rule.

Bad news: None of the known pivoting rules are known to result in a polynomial-time algorithm.

Simplex Running Time

First: Simplex is a method, not an algorithm, parameterised by the pivoting rule.

Bad news: None of the known pivoting rules are known to result in a polynomial-time algorithm.

More bad news: Most of the known pivoting rules have been shown to result in exponential running time.

Simplex Running Time

First: Simplex is a method, not an algorithm, parameterised by the pivoting rule.

Bad news: None of the known pivoting rules are known to result in a polynomial-time algorithm.

More bad news: Most of the known pivoting rules have been shown to result in exponential running time.

Good news: In practice the algorithm is quite efficient/fast.

Simplex Running Time

First: Simplex is a method, not an algorithm, parameterised by the pivoting rule.

Bad news: None of the known pivoting rules are known to result in a polynomial-time algorithm.

More bad news: Most of the known pivoting rules have been shown to result in exponential running time.

Good news: In practice the algorithm is quite efficient/fast.

More good news: "*Beyond the worst-case analysis*" shows that the algorithm is also efficient in theory.

Simplex Running Time

First: Simplex is a method, not an algorithm, parameterised by the pivoting rule.

Bad news: None of the known pivoting rules are known to result in a polynomial-time algorithm.

More bad news: Most of the known pivoting rules have been shown to result in exponential running time.

Good news: In practice the algorithm is quite efficient/fast.

More good news: "*Beyond the worst-case analysis*" shows that the algorithm is also efficient in theory.

Even more good news: We have other algorithms that run in worst-case polynomial running time (Ellipsoid Method, Interior Point Methods).

Duality

Suppose that we have a linear program, which we will refer to as *the primal*.

Duality

Suppose that we have a linear program, which we will refer to as *the primal*.

We will construct another linear program, which we will refer to as *the dual*.

Duality

Suppose that we have a linear program, which we will refer to as *the primal*.

We will construct another linear program, which we will refer to as *the dual*.

The *variables* of the primal become the *constraints* of the dual and vice-versa.

Duality

Suppose that we have a linear program, which we will refer to as *the primal*.

We will construct another linear program, which we will refer to as *the dual*.

The *variables* of the primal become the *constraints* of the dual and vice-versa.

Maximisation becomes minimisation.

Duality

Suppose that we have a linear program, which we will refer to as *the primal*.

We will construct another linear program, which we will refer to as *the dual*.

The *variables* of the primal become the *constraints* of the dual and vice-versa.

Maximisation becomes minimisation.

The two linear programs will have a very important connection.

Duality

The Primal

$$\text{maximise} \quad \sum_{j=1}^{n} c_j x_j$$

$$\text{subject to} \quad \sum_{j=1\in}^{n} \alpha_{ij} x_j \leq b_i, \quad i = 1, ..., m$$

$$x_j \geq 0, \quad j = 1, ..., n$$

The Dual

$$\text{minimize} \quad \sum_{i=1}^{m} b_i y_i$$

$$\text{subject to} \quad \sum_{i=1}^{m} \alpha_{ij} y_i \geq c_j, \quad i = 1, ..., n$$

$$y_j \geq 0, \qquad j = 1, ..., m$$

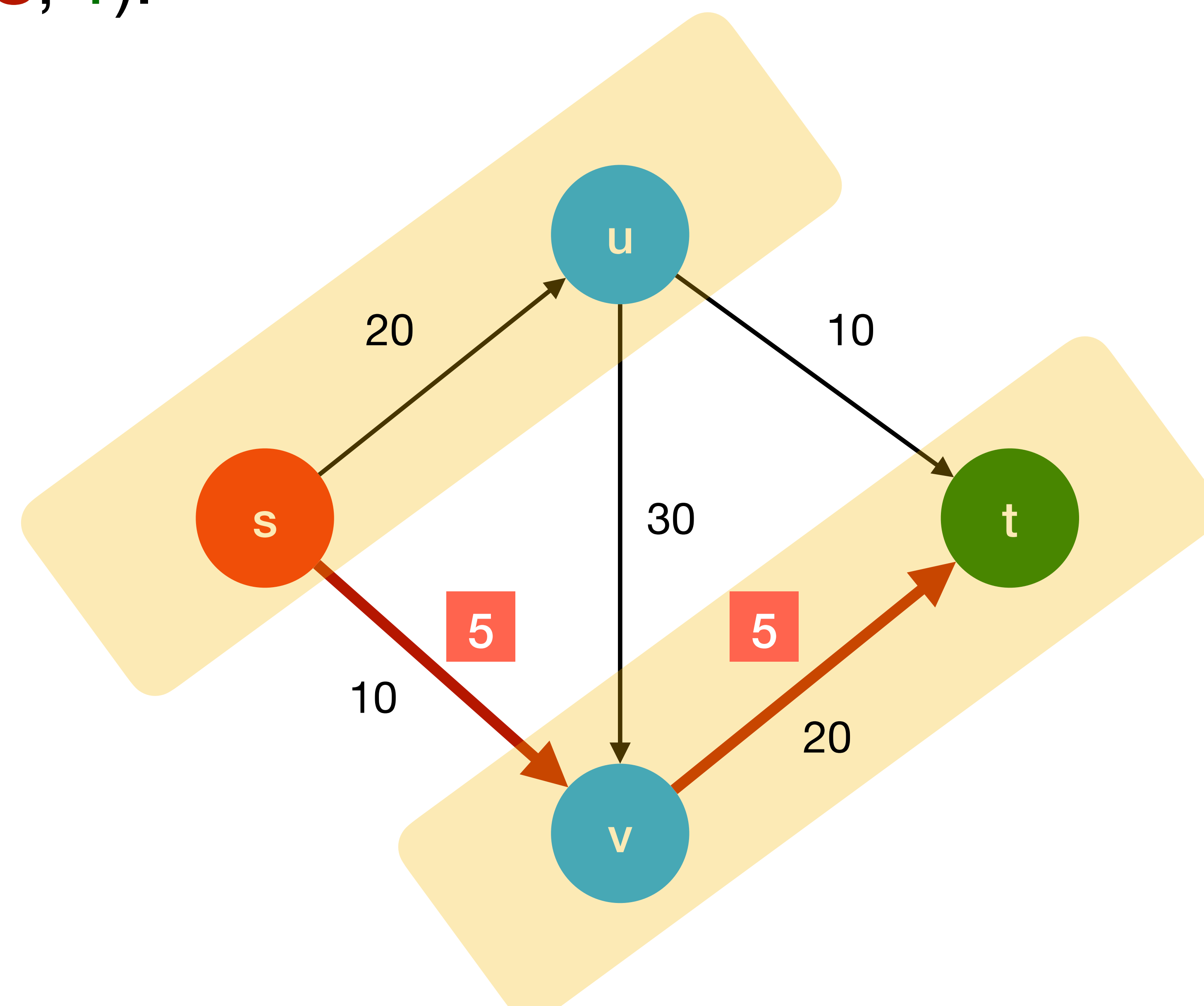
Weak Duality

Let x be any feasible solution to the Primal and let y be any feasible solution to the Dual. Then we have that

$$\text{value}(x) \leq \text{value}(y)$$

Weak Duality

Fact 3: Let f by any (s-t) flow and (S, T) be any (s-t) cut. Then v(f) ≤ c(S, T).



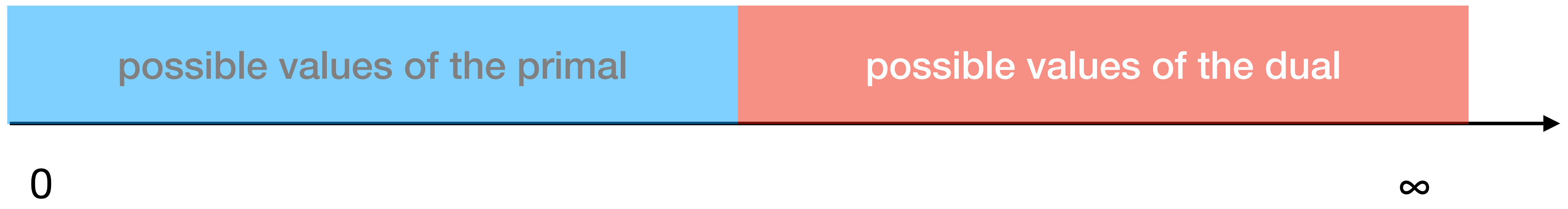
Strong Duality

Let x be any feasible solution to the Primal and let y be any feasible solution to the Dual. If

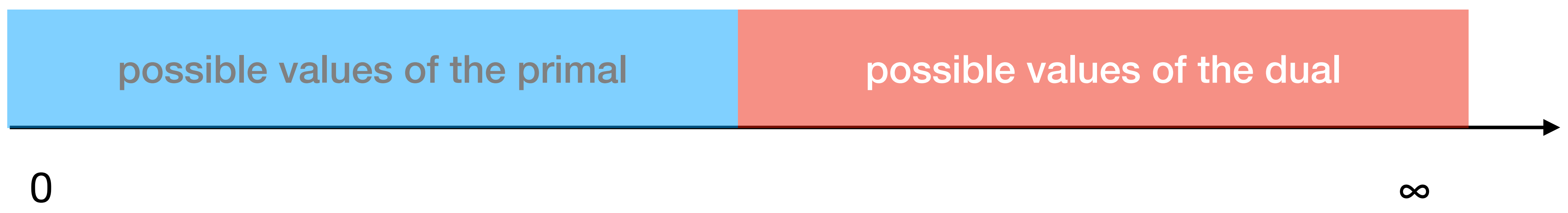
$$\text{value}(x) = \text{value}(y)$$

then x and y are both optimal solutions.

Pictorially

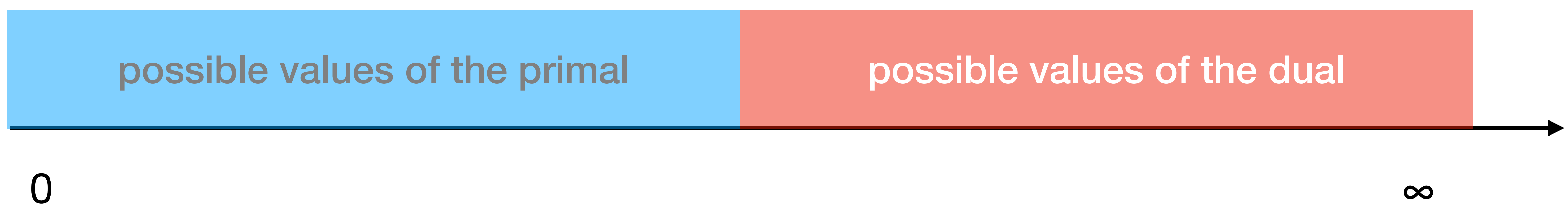


Pictorially



How can we prove that a solution x to the primal is maximum?

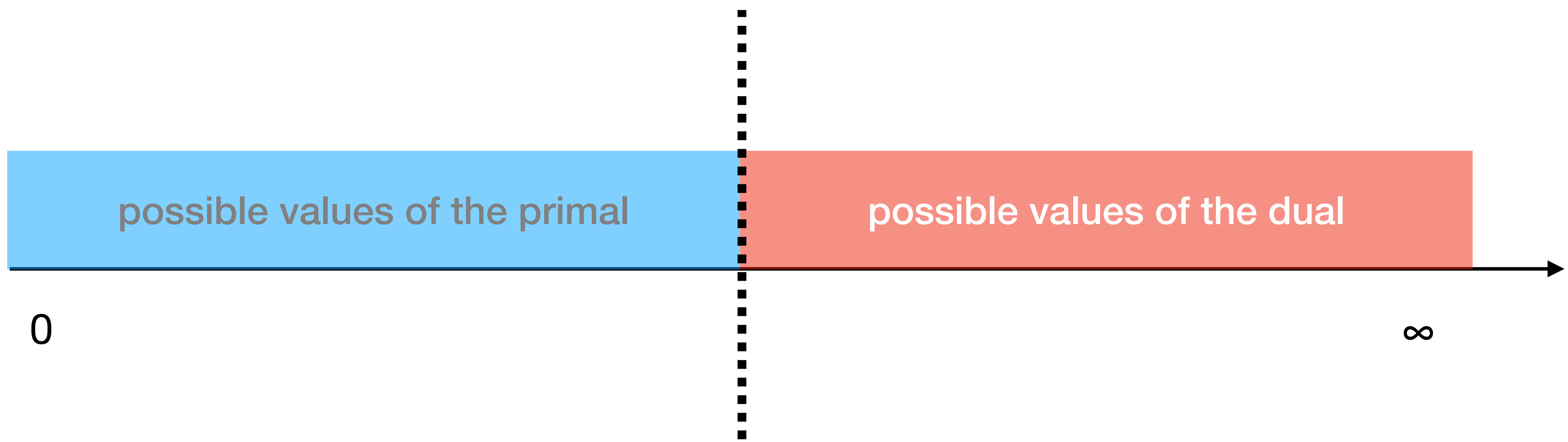
Pictorially



How can we prove that a solution x to the primal is maximum?

Find a solution y to the dual with value(y) = value(x)

Pictorially



How can we prove that a solution x to the primal is maximum?

Find a solution y to the dual with value(y) = value(x)

The Max-Flow Min-Cut Theorem

Theorem: In every flow network, the value of the maximum flow is *equal* to the capacity of the minimum cut.

This is a consequence of the *strong duality theorem* for linear programs!

As a matter of fact…

As a matter of fact…

The Simplex method starts from some feasible solution (possibly the point (0, …. , 0)).

As a matter of fact…

The Simplex method starts from some feasible solution (possibly the point (*0*, …. , *0*)).

It improves the solution in every step (via *pivoting*) until it can no longer be improved.

As a matter of fact…

The Simplex method starts from some feasible solution (possibly the point (*0*, …. , *0*)).

It improves the solution in every step (via *pivoting*) until it can no longer be improved.

To prove that the solution is optimal, we use duality.

As a matter of fact…

The Simplex method starts from some feasible solution (possibly the point (*0*, …. , *0*)).

It improves the solution in every step (via *pivoting*) until it can no longer be improved.

To prove that the solution is optimal, we use duality.

Do you know of any other algorithms for which the same principle applies?

Maximum Flow as an LP

We can write the maximum flow problem as a linear problem.

"Maximise the flow, subject to capacity and flow conservation constraints".

$$
\begin{array}{llll}
\text{maximise} & \displaystyle\sum_{v\in V} f_{sv} - \sum_{v\in V} f_{vs} & & \\
\text{subject to} & f_{uv} \le c_{uv}, & \text{for each} & u, v \in V \\
& \displaystyle\sum_{v\in V} f_{vu} = \sum_{v\in V} f_{uv}, & \text{for each } u \in V - \{s, t\} & \\
& f_{uv} \ge 0, & \text{for each} & u, v \in V
\end{array}
$$

Maximum Flow as an LP

We can write the maximum flow problem as a linear problem.

"Maximise the flow, subject to capacity and flow conservation constraints".

$$
\begin{array}{lrll}
\text{maximise} & \boxed{\displaystyle\sum_{v \in V} f_{sv}} - \displaystyle\sum_{v \in V} f_{vs} & & \\
\text{subject to} & f_{uv} \leq c_{uv}, & \text{for each} & u, v \in V \\
& \displaystyle\sum_{v \in V} f_{vu} = \sum_{v \in V} f_{uv}, & \text{for each } u \in V - \{s, t\} & \\
& f_{uv} \geq 0, & \text{for each} & u, v \in V
\end{array}
$$

Maximum Flow as an LP

We can write the maximum flow problem as a linear problem.

"Maximise the flow, subject to capacity and flow conservation constraints".

total flow out of s

$$
\begin{array}{lrll}
\text{maximise} & \boxed{\displaystyle\sum_{v \in V} f_{sv}} - \displaystyle\sum_{v \in V} f_{vs} & & \\
\text{subject to} & f_{uv} \leq c_{uv}, & \text{for each} & u, v \in V \\
& \displaystyle\sum_{v \in V} f_{vu} = \sum_{v \in V} f_{uv}, & \text{for each } u \in V - \{s, t\} & \\
& f_{uv} \geq 0, & \text{for each} & u, v \in V
\end{array}
$$

Maximum Flow as an LP

We can write the maximum flow problem as a linear problem.

"Maximise the flow, subject to capacity and flow conservation constraints".

total flow out of s

$$
\begin{aligned}
\text{maximise} \quad & \boxed{\sum_{v \in V} f_{sv}} - \boxed{\sum_{v \in V} f_{vs}} \\
\text{subject to} \quad & f_{uv} \leq c_{uv}, \quad \text{for each} \quad u, v \in V \\
& \sum_{v \in V} f_{vu} = \sum_{v \in V} f_{uv}, \quad \text{for each } u \in V - \{s, t\} \\
& f_{uv} \geq 0, \quad \text{for each} \quad u, v \in V
\end{aligned}
$$

Maximum Flow as an LP

We can write the maximum flow problem as a linear problem.

"Maximise the flow, subject to capacity and flow conservation constraints".

$$\begin{aligned}
&\text{maximise} && \underbrace{\boxed{\sum_{v\in V} f_{sv}}}_{\text{total flow out of s}} - \underbrace{\boxed{\sum_{v\in V} f_{vs}}}_{\text{total flow into s}} \\
&\text{subject to} && f_{uv} \le c_{uv}, \quad \text{for each } u, v \in V \\
& && \sum_{v\in V} f_{vu} = \sum_{v\in V} f_{uv}, \quad \text{for each } u \in V - \{s,t\} \\
& && f_{uv} \ge 0, \quad \text{for each } u, v \in V
\end{aligned}$$

Maximum Flow as an LP

We can write the maximum flow problem as a linear problem.

"Maximise the flow, subject to capacity and flow conservation constraints".

total flow out of s

total flow into s

$$
\begin{array}{lll}
\text{maximise} & \boxed{\sum_{v\in V} f_{sv}} - \boxed{\sum_{v\in V} f_{vs}} & \\
\text{subject to} & \boxed{f_{uv} \leq c_{uv}, \quad \text{for each} \quad u, v \in V} & \\
 & \sum_{v\in V} f_{vu} = \sum_{v\in V} f_{uv}, \quad \text{for each } u \in V - \{s,t\} & \\
 & f_{uv} \geq 0, \quad \text{for each} \quad u, v \in V &
\end{array}
$$

Maximum Flow as an LP

We can write the maximum flow problem as a linear problem.

"Maximise the flow, subject to capacity and flow conservation constraints".

total flow out of s total flow into s

$$\text{maximise} \quad \boxed{\sum_{v\in V} f_{sv}} - \boxed{\sum_{v\in V} f_{vs}}$$

capacity constraint

$$\begin{aligned}
\text{subject to} \quad & \boxed{f_{uv} \le c_{uv}, \quad \text{for each} \quad u,v \in V} \\
& \sum_{v\in V} f_{vu} = \sum_{v\in V} f_{uv}, \quad \text{for each } u \in V - \{s,t\} \\
& f_{uv} \ge 0, \quad \text{for each} \quad u,v \in V
\end{aligned}$$

Maximum Flow as an LP

We can write the maximum flow problem as a linear problem.

"Maximise the flow, subject to capacity and flow conservation constraints".

total flow out of s

total flow into s

capacity constraint

$$
\begin{aligned}
\text{maximise} \quad & \boxed{\sum_{v \in V} f_{sv}} - \boxed{\sum_{v \in V} f_{vs}} \\
\text{subject to} \quad & \boxed{f_{uv} \leq c_{uv}, \quad \text{for each } u, v \in V} \\
& \boxed{\sum_{v \in V} f_{vu} = \sum_{v \in V} f_{uv}, \quad \text{for each } u \in V - \{s, t\}} \\
& f_{uv} \geq 0, \quad \text{for each } u, v \in V
\end{aligned}
$$

Maximum Flow as an LP

We can write the maximum flow problem as a linear problem.

"Maximise the flow, subject to capacity and flow conservation constraints".

total flow out of s

total flow into s

capacity constraint

flow conservation

$$
\begin{array}{lll}
\text{maximise} & \boxed{\sum_{v \in V} f_{sv}} - \boxed{\sum_{v \in V} f_{vs}} & \\
\text{subject to} & f_{uv} \leq c_{uv}, & \text{for each } u, v \in V \\
 & \sum_{v \in V} f_{vu} = \sum_{v \in V} f_{uv}, & \text{for each } u \in V - \{s, t\} \\
 & f_{uv} \geq 0, & \text{for each } u, v \in V
\end{array}
$$

Maximum Flow as an LP

We can write the maximum flow problem as a linear problem.

"Maximise the flow, subject to capacity and flow conservation constraints".

total flow out of s

total flow into s

capacity constraint

flow conservation

$$\begin{aligned}
\text{maximise} \quad & \boxed{\sum_{v \in V} f_{sv}} - \boxed{\sum_{v \in V} f_{vs}} \\
\text{subject to} \quad & \boxed{f_{uv} \leq c_{uv}, \quad \text{for each} \quad u, v \in V} \\
& \boxed{\sum_{v \in V} f_{vu} = \sum_{v \in V} f_{uv}, \quad \text{for each } u \in V - \{s, t\}} \\
& \boxed{f_{uv} \geq 0, \quad \text{for each} \quad u, v \in V}
\end{aligned}$$

Maximum Flow as an LP

We can write the maximum flow problem as a linear problem.

"Maximise the flow, subject to capacity and flow conservation constraints".

total flow out of s

total flow into s

capacity constraint

flow conservation

non-negative flow

$$
\begin{array}{lll}
\text{maximise} & \boxed{\sum_{v \in V} f_{sv}} - \boxed{\sum_{v \in V} f_{vs}} & \\
\text{subject to} & \boxed{f_{uv} \leq c_{uv}, \quad \text{for each} \quad u, v \in V} & \\
 & \boxed{\sum_{v \in V} f_{vu} = \sum_{v \in V} f_{uv}, \quad \text{for each } u \in V - \{s, t\}} & \\
 & \boxed{f_{uv} \geq 0, \quad \text{for each} \quad u, v \in V} &
\end{array}
$$

Constructing the dual

$$
\begin{array}{llll}
\text{minimise} & \displaystyle\sum_{(u,v)\in E} c_{uv} d_{uv} & & \\
\text{subject to} & d_{uv} - z_u + z_v \geq 0 & \text{for each} & (u,v) \in E, u \neq s, v \neq t \\
 & d_{su} + z_v \geq 1 & \text{for each} & (s,u) \in E \\
 & d_{ut} - z_u \geq 0 & \text{for each} & (u,t) \in E \\
 & d_{uv} \geq 0, & \text{for each} & (u,v) \in E \\
 & z_u \geq 0 & \text{for each } u \in V - \{t,s\} &
\end{array}
$$

Constructing the dual

This is *1* if u is in S and v is in T
and *0* otherwise.

$$
\begin{aligned}
&\text{minimise} && \sum_{(u,v)\in E} c_{uv} \boxed{d_{uv}} \\
&\text{subject to} && d_{uv} - z_u + z_v \geq 0 \quad \text{for each } (u,v) \in E, u \neq s, v \neq t \\
& && d_{su} + z_v \geq 1 \quad \text{for each } (s,u) \in E \\
& && d_{ut} - z_u \geq 0 \quad \text{for each } (u,t) \in E \\
& && d_{uv} \geq 0, \quad \text{for each } (u,v) \in E \\
& && z_u \geq 0 \quad \text{for each } u \in V - \{t,s\}
\end{aligned}
$$

Constructing the dual

This is *1* if u is in S and v is in T
and *0* otherwise.

$$
\begin{array}{lrll}
\text{minimise} & \displaystyle\sum_{(u,v)\in E} c_{uv}\boxed{d_{uv}} & & \\
\text{subject to} & d_{uv} - z_u + z_v \geq 0 & \text{for each } (u,v) \in E, u \neq s, v \neq t \\
& d_{su} + z_v \geq 1 & \text{for each} & (s,u) \in E \\
& d_{ut} - z_u \geq 0 & \text{for each} & (u,t) \in E \\
& d_{uv} \geq 0, & \text{for each} & (u,v) \in E \\
& z_u \geq 0 & \text{for each } u \in V - \{t,s\} & \\
& d_{uv} \in \{0,1\}, & \text{for each } (u,v) \in E &
\end{array}
$$

Constructing the dual

This is *1* if u is in S and v is in T and *0* otherwise.

This is *1* if u is in S and *0* otherwise.

$$
\begin{array}{lll}
\text{minimise} & \displaystyle\sum_{(u,v)\in E} c_{uv}\boxed{d_{uv}} & \\
\text{subject to} & d_{uv} - \boxed{z_u} + z_v \geq 0 & \text{for each } (u,v) \in E, u \neq s, v \neq t \\
& d_{su} + z_v \geq 1 & \text{for each } (s,u) \in E \\
& d_{ut} - z_u \geq 0 & \text{for each } (u,t) \in E \\
& d_{uv} \geq 0, & \text{for each } (u,v) \in E \\
& z_u \geq 0 & \text{for each } u \in V - \{t,s\} \\
& d_{uv} \in \{0,1\}, & \text{for each } (u,v) \in E
\end{array}
$$

Constructing the dual

This is *1* if u is in S and v is in T
and *0* otherwise.

This is *1* if u is in S and *0* otherwise.

$$
\begin{array}{lll}
\text{minimise} & \displaystyle\sum_{(u,v)\in E} c_{uv}\boxed{d_{uv}} & \\
\text{subject to} & d_{uv} - \boxed{z_u} + z_v \geq 0 & \text{for each } (u,v) \in E, u \neq s, v \neq t \\
& d_{su} + z_v \geq 1 & \text{for each } (s,u) \in E \\
& d_{ut} - z_u \geq 0 & \text{for each } (u,t) \in E \\
& d_{uv} \geq 0, & \text{for each } (u,v) \in E \\
& z_u \geq 0 & \text{for each } u \in V - \{t,s\} \\
& d_{uv} \in \{0,1\}, & \text{for each } (u,v) \in E \\
& z_u \in \{0,1\}, & \text{for each } u \in V - \{s,t\}
\end{array}
$$

Constructing the dual

This is *1* if u is in S and v is in T
and *0* otherwise.

This is *1* if u is in S and *0* otherwise.

$$\begin{aligned}
&\text{minimise} && \sum_{(u,v)\in E} c_{uv} \boxed{d_{uv}} \\
&\text{subject to} && d_{uv} - \boxed{z_u} + z_v \geq 0 \quad \text{for each } (u,v)\in E, u\neq s, v\neq t \\
& && d_{su} + z_v \geq 1 \quad \text{for each } (s,u)\in E \\
& && d_{ut} - z_u \geq 0 \quad \text{for each } (u,t)\in E \\
& && d_{uv} \geq 0, \quad \text{for each } (u,v)\in E \\
& && z_u \geq 0 \quad \text{for each } u \in V - \{t,s\} \\
& && d_{uv} \in \{0,1\}, \quad \text{for each } (u,v)\in E \\
& && z_u \in \{0,1\}, \quad \text{for each } u \in V - \{s,t\}
\end{aligned}$$

If u is in S and v is
in T, then d_{uv} must be *1*.

Constructing the dual

This is *1* if u is in S and v is in T
and *0* otherwise.

This is *1* if u is in S and *0* otherwise.

$$
\begin{aligned}
\text{minimise} \quad & \sum_{(u,v)\in E} c_{uv} \boxed{d_{uv}} \\
\text{subject to} \quad & d_{uv} - \boxed{z_u} + z_v \geq 0 \quad \text{for each } (u,v) \in E, u \neq s, v \neq t \\
& d_{su} + z_v \geq 1 \quad \text{for each } (s,u) \in E \\
& d_{ut} - z_u \geq 0 \quad \text{for each } (u,t) \in E \\
& d_{uv} \geq 0, \quad \text{for each } (u,v) \in E \\
& z_u \geq 0 \quad \text{for each } u \in V - \{t,s\} \\
& d_{uv} \in \{0,1\}, \quad \text{for each } (u,v) \in E \\
& z_u \in \{0,1\}, \quad \text{for each } u \in V - \{s,t\}
\end{aligned}
$$

If u is in S and v is
in T, then d_{uv} must be *1*.

If v is in T then d_{sv} must be *1*.

Constructing the dual

This is *1* if u is in S and v is in T and *0* otherwise.

This is *1* if u is in S and *0* otherwise.

$$
\begin{array}{lll}
\text{minimise} & \displaystyle\sum_{(u,v)\in E} c_{uv} \boxed{d_{uv}} & \\
\text{subject to} & d_{uv} - \boxed{z_u} + z_v \geq 0 & \text{for each } (u,v) \in E, u \neq s, v \neq t \\
& d_{su} + z_v \geq 1 & \text{for each } (s,u) \in E \\
& d_{ut} - z_u \geq 0 & \text{for each } (u,t) \in E \\
& d_{uv} \geq 0, & \text{for each } (u,v) \in E \\
& z_u \geq 0 & \text{for each } u \in V - \{t,s\} \\
& d_{uv} \in \{0,1\}, & \text{for each } (u,v) \in E \\
& z_u \in \{0,1\}, & \text{for each } u \in V - \{s,t\}
\end{array}
$$

If u is in S and v is in T, then d_{uv} must be *1*.

If v is in T then d_{sv} must be *1*.

If u is in S then d_{ut} must be *1*.

Minimum Cut as an ILP

$$
\begin{array}{llll}
\text{minimise} & \displaystyle\sum_{(u,v)\in E} c_{uv} d_{uv} & & \\
\text{subject to} & d_{uv} - z_u + z_v \geq 0 & \text{for each } (u,v) \in E, u \neq s, v \neq t & \\
& d_{su} + z_v \geq 1 & \text{for each} & (s,u) \in E \\
& d_{ut} - z_u \geq 0 & \text{for each} & (u,t) \in E \\
& d_{uv} \geq 0, & \text{for each} & (u,v) \in E \\
& z_u \geq 0 & \text{for each } u \in V - \{t,s\} & \\
& d_{uv} \in \{0,1\}, & \text{for each } (u,v) \in E & \\
& z_u \in \{0,1\}, & \text{for each } u \in V - \{s,t\} &
\end{array}
$$

LP-relaxation

An *LP-relaxation* of an Integer Linear Program is a linear program which is identical to the ILP, except all the integrality constraints have been removed ("*relaxed*"), or replaced with non-integral constraints.

Minimum Cut as an ILP

$$
\begin{array}{lrl}
\text{minimise} & \displaystyle\sum_{(u,v)\in E} c_{uv} d_{uv} & \\
\text{subject to} & d_{uv} - z_u + z_v \geq 0 & \text{for each } (u,v) \in E, u \neq s, v \neq t \\
& d_{su} + z_v \geq 1 & \text{for each } \quad (s,u) \in E \\
& d_{ut} - z_u \geq 0 & \text{for each } \quad (u,t) \in E \\
& d_{uv} \geq 0, & \text{for each } \quad (u,v) \in E \\
& z_u \geq 0 & \text{for each } u \in V - \{t,s\} \\
& d_{uv} \in \{0,1\}, & \text{for each } (u,v) \in E \\
& z_u \in \{0,1\}, & \text{for each } u \in V - \{s,t\}
\end{array}
$$

Minimum Cut as an ILP

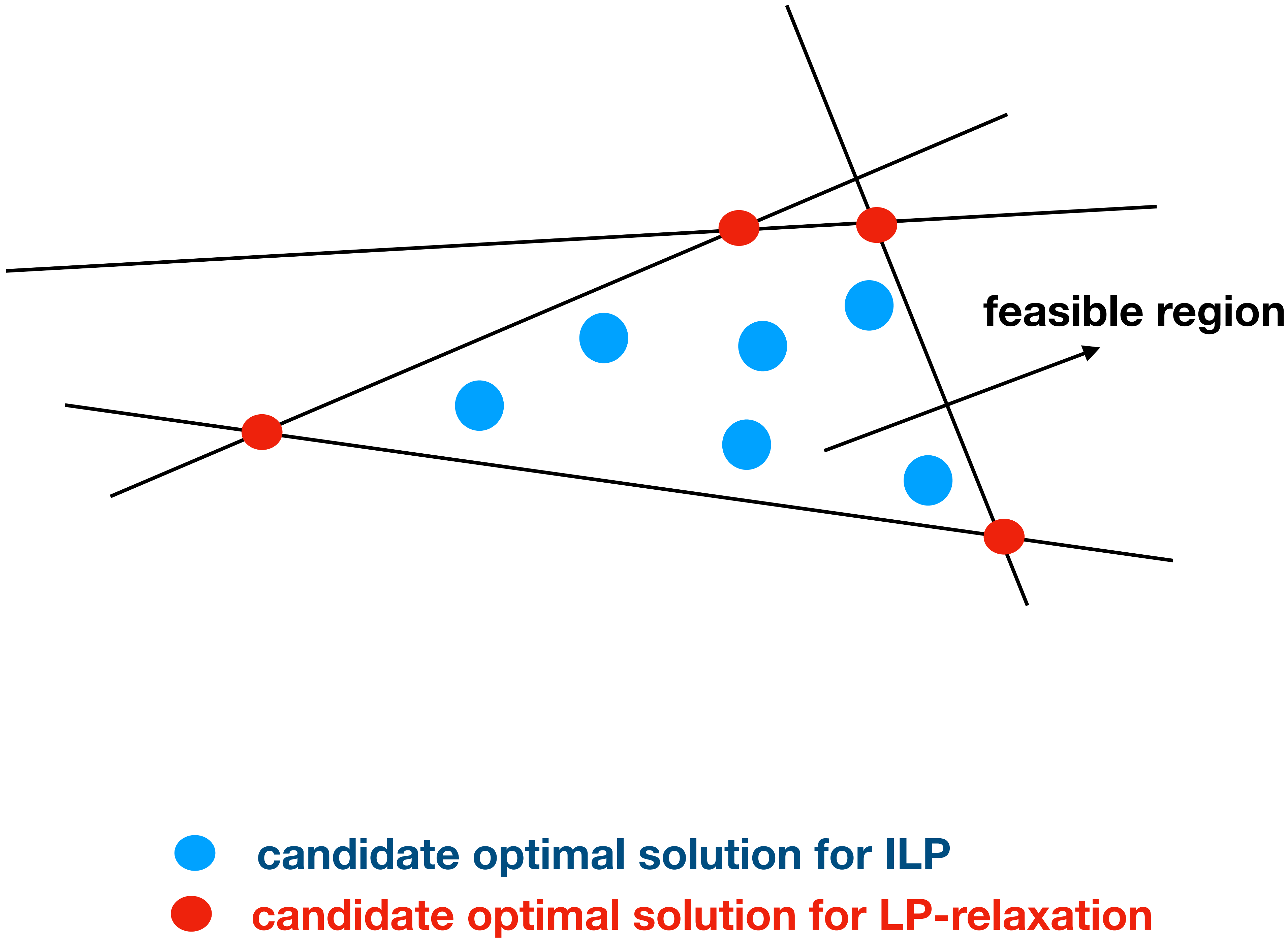
LP-relaxation

$$
\begin{array}{lrll}
\text{minimise} & \displaystyle\sum_{(u,v)\in E} c_{uv} d_{uv} & & \\
\text{subject to} & d_{uv} - z_u + z_v \geq 0 & \text{for each } (u,v) \in E, u \neq s, v \neq t & \\
& d_{su} + z_v \geq 1 & \text{for each} & (s,u) \in E \\
& d_{ut} - z_u \geq 0 & \text{for each} & (u,t) \in E \\
& d_{uv} \geq 0, & \text{for each} & (u,v) \in E \\
& z_u \geq 0 & \text{for each } u \in V - \{t,s\} &
\end{array}
$$

$$d_{uv} \in \{0,1\}, \quad \text{for each } (u,v) \in E$$

$$z_u \in \{0,1\}, \quad \text{for each } u \in V - \{s,t\}$$

ILP vs LP-relaxation



ILP vs LP-relaxation

ILP vs LP-relaxation

For a maximisation problem:

ILP vs LP-relaxation

For a maximisation problem:

The optimal value of the ILP is not larger than the optimal value of the LP-relaxation.

ILP vs LP-relaxation

For a maximisation problem:

The optimal value of the ILP is not larger than the optimal value of the LP-relaxation.

The ratio

max_value(LP-relaxation) / max_value(LP)

is called the integrality gap of the LP-formulation.

Let’s put some facts together.

Let’s put some facts together.

The Max-Flow LP and the Min-Cut LP-relaxation are duals of each other.

Let’s put some facts together.

The Max-Flow LP and the Min-Cut LP-relaxation are duals of each other.

By *strong duality*, it holds that the optimal value of the Max-Flow LP is equal to the optimal value of the Min-Cut LP-relaxation.

Let’s put some facts together.

The Max-Flow LP and the Min-Cut LP-relaxation are duals of each other.

By *strong duality*, it holds that the optimal value of the Max-Flow LP is equal to the optimal value of the Min-Cut LP-relaxation.

By the *Max-Flow-Min-Cut Theorem,* the value of the maximum flow is *equal* to the capacity of the minimum cut.

Let's put some facts together.

The Max-Flow LP and the Min-Cut LP-relaxation are duals of each other.

By *strong duality*, it holds that the optimal value of the Max-Flow LP is equal to the optimal value of the Min-Cut LP-relaxation.

By the *Max-Flow-Min-Cut Theorem,* the value of the maximum flow is *equal* to the capacity of the minimum cut.

That can only mean one thing:

Let’s put some facts together.

The Max-Flow LP and the Min-Cut LP-relaxation are duals of each other.

By *strong duality*, it holds that the optimal value of the Max-Flow LP is equal to the optimal value of the Min-Cut LP-relaxation.

By the *Max-Flow-Min-Cut Theorem,* the value of the maximum flow is *equal* to the capacity of the minimum cut.

That can only mean one thing:

The value of the Min-Cut LP-relaxation is equal to the value of the Min-Cut LP.

Let's put some facts together.

The Max-Flow LP and the Min-Cut LP-relaxation are duals of each other.

By *strong duality*, it holds that the optimal value of the Max-Flow LP is equal to the optimal value of the Min-Cut LP-relaxation.

By the *Max-Flow-Min-Cut Theorem,* the value of the maximum flow is *equal* to the capacity of the minimum cut.

That can only mean one thing:

The value of the Min-Cut LP-relaxation is equal to the value of the Min-Cut LP.

In other words, the Min-Cut LP-formulation has integrality gap *1*.

Let's put some facts together.

The Max-Flow LP and the Min-Cut LP-relaxation are duals of each other.

By *strong duality*, it holds that the optimal value of the Max-Flow LP is equal to the optimal value of the Min-Cut LP-relaxation.

By the *Max-Flow-Min-Cut Theorem,* the value of the maximum flow is *equal* to the capacity of the minimum cut.

That can only mean one thing:

The value of the Min-Cut LP-relaxation is equal to the value of the Min-Cut LP.

In other words, the Min-Cut LP-formulation has integrality gap *1*.

In other words, the Min-Cut LP has *an integer optimal solution*.

Back to Maximum Flow

What if we wanted an integer flow instead of any flow?

$$
\begin{array}{llll}
\text{maximise} & \displaystyle\sum_{v \in V} f_{sv} - \sum_{v \in V} f_{vs} & & \\
\text{subject to} & f_{uv} \le c_{uv}, & \text{for each} & u, v \in V \\
& \displaystyle\sum_{v \in V} f_{vu} = \sum_{v \in V} f_{uv}, & \text{for each} & u \in V - \{s, t\} \\
& f_{uv} \ge 0, & \text{for each} & u, v \in V
\end{array}
$$

Back to Maximum Flow

What if we wanted an integer flow instead of any flow?

$$
\begin{aligned}
\text{maximise} \quad & \sum_{v \in V} f_{sv} - \sum_{v \in V} f_{vs} \\
\text{subject to} \quad & f_{uv} \le c_{uv}, \quad \text{for each } u, v \in V \\
& \sum_{v \in V} f_{vu} = \sum_{v \in V} f_{uv}, \quad \text{for each } u \in V - \{s, t\} \\
& f_{uv} \ge 0, \quad \text{for each } u, v \in V \\
& f_{uv} \in \mathbb{R}, \quad \text{for each } u, v \in V
\end{aligned}
$$

Back to Maximum Flow

Does the LP-relaxation of this ILP always have an integer solution?

$$
\begin{array}{llll}
\text{maximise} & \displaystyle\sum_{v \in V} f_{sv} - \sum_{v \in V} f_{vs} & & \\
\text{subject to} & f_{uv} \le c_{uv}, & \text{for each} & u, v \in V \\
& \displaystyle\sum_{v \in V} f_{vu} = \sum_{v \in V} f_{uv}, & \text{for each } u \in V - \{s, t\} & \\
& f_{uv} \ge 0, & \text{for each} & u, v \in V \\
& f_{uv} \in \mathbb{R}, & \text{for each } u, v \in V &
\end{array}
$$

LPs for Max-Flow and Min-Cut

LPs for Max-Flow and Min-Cut

The Max-Flow problem for integer flows can be written as an ILP.

LPs for Max-Flow and Min-Cut

The Max-Flow problem for integer flows can be written as an ILP.

The Min-Cut problem can be written as an ILP (cuts are always integers).

LPs for Max-Flow and Min-Cut

The Max-Flow problem for integer flows can be written as an ILP.

The Min-Cut problem can be written as an ILP (cuts are always integers).

We can write the LP-relaxations of those two ILPs.

LPs for Max-Flow and Min-Cut

The Max-Flow problem for integer flows can be written as an ILP.

The Min-Cut problem can be written as an ILP (cuts are always integers).

We can write the LP-relaxations of those two ILPs.

For Max-Flow, it finds the maximum fractional flow.

LPs for Max-Flow and Min-Cut

The Max-Flow problem for integer flows can be written as an ILP.

The Min-Cut problem can be written as an ILP (cuts are always integers).

We can write the LP-relaxations of those two ILPs.

For Max-Flow, it finds the maximum fractional flow.

For Min-Cut, it finds the minimum “fractional” cut.

LPs for Max-Flow and Min-Cut

The Max-Flow problem for integer flows can be written as an ILP.

The Min-Cut problem can be written as an ILP (cuts are always integers).

We can write the LP-relaxations of those two ILPs.

For Max-Flow, it finds the maximum fractional flow.

For Min-Cut, it finds the minimum “fractional” cut.

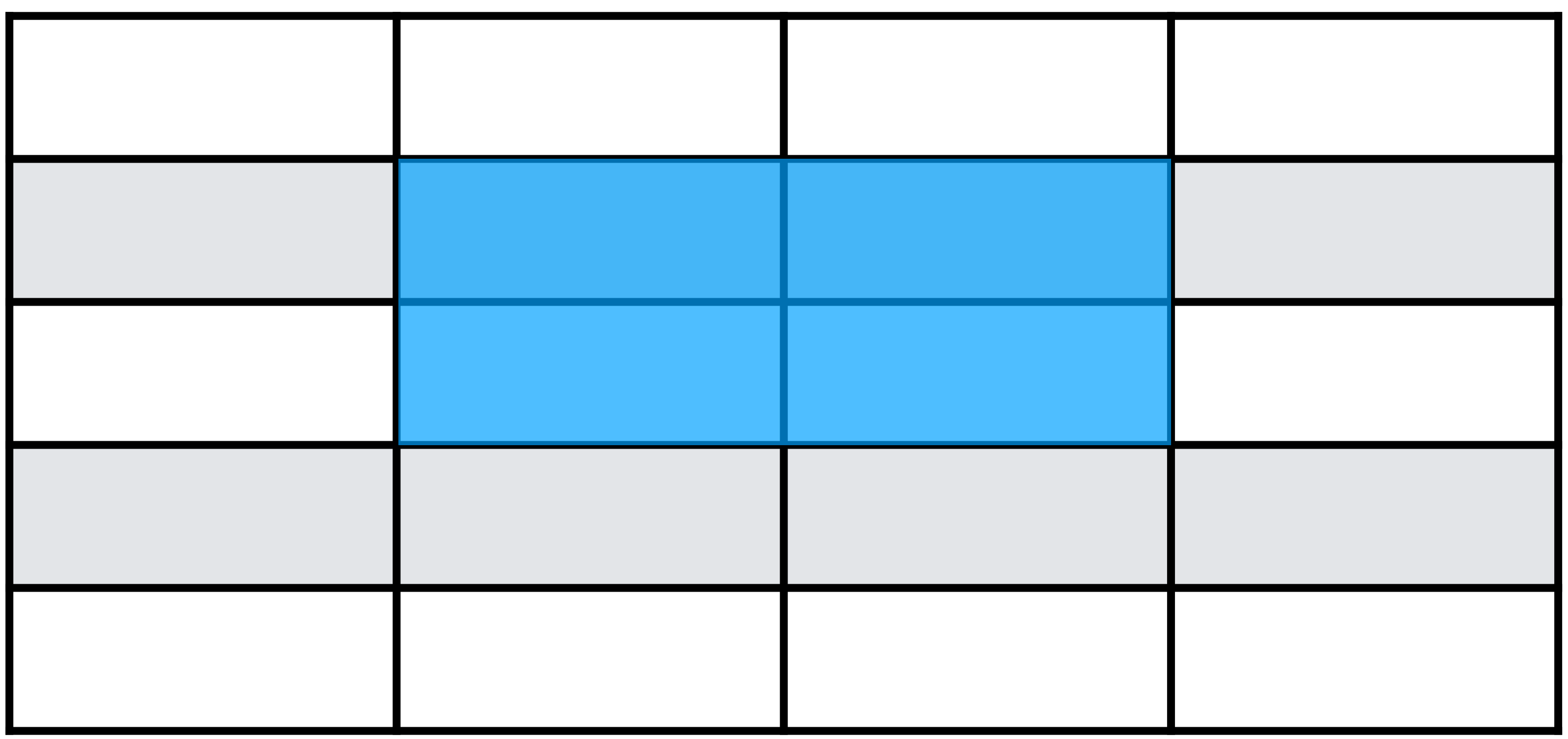
If we solve those LP-relaxations, we will get *integer solutions*.

Totally Unimodular Matrices

Let A be a real *m* x *n* matrix. Suppose that every square submatrix of A has determinant in {*0*, *+1*, *-1*}. Then A is *totally unimodular*.

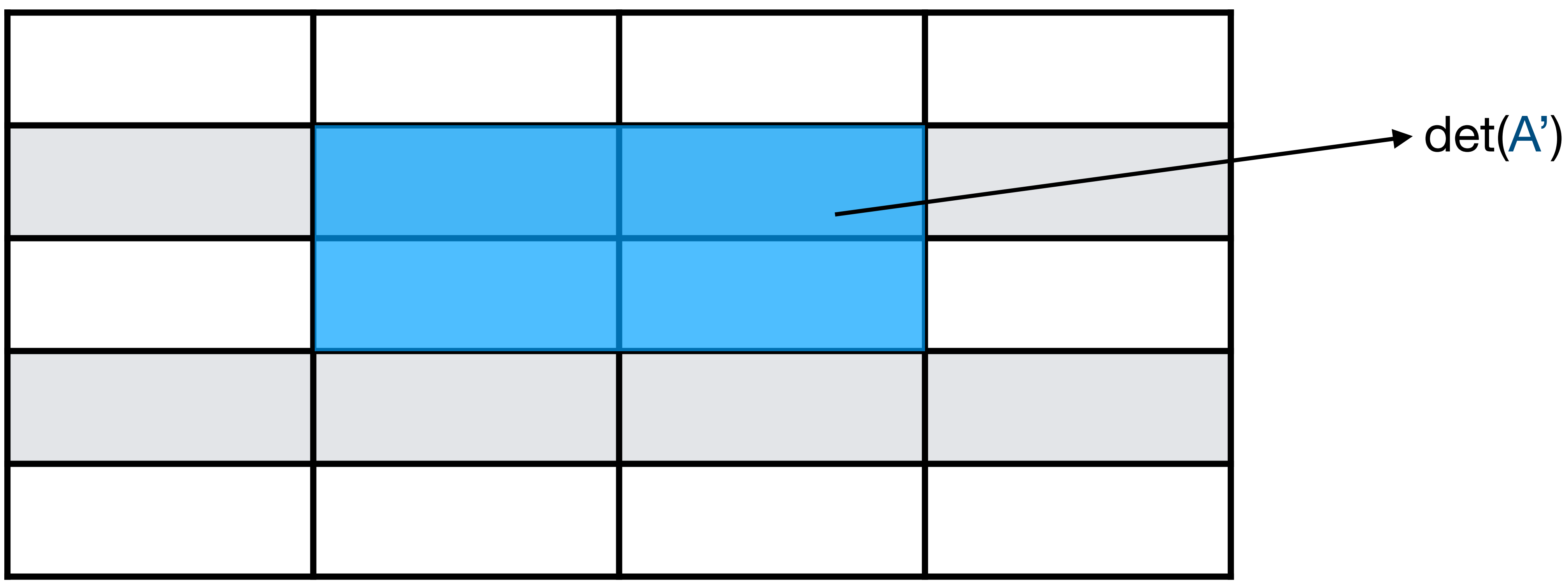
Totally Unimodular Matrices

Let A be a real *m* x *n* matrix. Suppose that every square submatrix of A has determinant in {*0*, *+1*, *-1*}. Then A is *totally unimodular*.



Totally Unimodular Matrices

Let A be a real *m* x *n* matrix. Suppose that every square submatrix of A has determinant in {*0*, *+1*, *-1*}. Then A is *totally unimodular*.



Total Unimodularity

If the constraint matrix A is totally unimodular and b is an *integer vector*, then the LP has an *integer solution*.

The Max-Flow and Min-Cut LP-relaxations admit integer solutions because their constraint matrices are totally unimodular.

$$\begin{aligned} \text{maximise} \quad & c^{\mathrm{T}} x \\ \text{subject to} \quad & Ax \le b, \\ & x \ge 0 \end{aligned}$$

To be more precise

To be more precise

Lemma: Suppose A is a totally unimodular matrix and b is an integer vector. Then every extreme point of

$P = \{x: Ax < b\}$ is integral.

To be more precise

Lemma: Suppose A is a totally unimodular matrix and b is an integer vector. Then every extreme point of

$P = \{x: Ax < b\}$ is integral.

Claim: Suppose A is totally unimodular. Then the matrix $A' = (A\ {-A}\ I\ {-I})^T$ is also totally unimodular.

To be more precise

Lemma: Suppose A is a totally unimodular matrix and b is an integer vector. Then every extreme point of

$P = \{x: Ax < b\}$ is integral.

Claim: Suppose A is totally unimodular. Then the matrix $A' = (A\ {-A}\ I\ {-I})^T$ is also totally unimodular.

Corollary: Suppose A is a totally unimodular matrix and b is an integer vector. Then every extreme point of

$P = \{x: Ax = b, 0 \leq x \leq c\}$ is integral.

Back to maximum flow

$$
\begin{array}{lrll}
\text{maximise} & \displaystyle\sum_{v\in V} f_{sv} - \sum_{v\in V} f_{vs} & & \\
\text{subject to} & f_{uv} \le c_{uv}, & \text{for each} & u, v \in V \\
& \displaystyle\sum_{v\in V} f_{vu} = \sum_{v\in V} f_{uv}, & \text{for each} & u \in V - \{s, t\} \\
& f_{uv} \ge 0, & \text{for each} & u, v \in V
\end{array}
$$

Back to maximum flow

$$
\begin{aligned}
\text{maximise} \quad & \sum_{v \in V} f_{sv} - \sum_{v \in V} f_{vs} \\
\text{subject to} \quad & f_{uv} \le c_{uv}, \quad \text{for each } u, v \in V \\
& \sum_{v \in V} f_{vu} = \sum_{v \in V} f_{uv}, \quad \text{for each } u \in V - \{s, t\} \\
& \boxed{f_{uv} \ge 0, \quad \text{for each } u, v \in V}
\end{aligned}
$$

Back to maximum flow

$$
\begin{aligned}
\text{maximise} \quad & \sum_{v \in V} f_{sv} - \sum_{v \in V} f_{vs} \\
\text{subject to} \quad & \boxed{f_{uv} \leq c_{uv}, \quad \text{for each} \quad u, v \in V} \\
& \sum_{v \in V} f_{vu} = \sum_{v \in V} f_{uv}, \quad \text{for each } u \in V - \{s, t\} \\
& \boxed{f_{uv} \geq 0, \quad \text{for each} \quad u, v \in V}
\end{aligned}
$$

Back to maximum flow

$$
\begin{array}{lll}
\text{maximise} & \displaystyle\sum_{v \in V} f_{sv} - \sum_{v \in V} f_{vs} & \\
\text{subject to} & \boxed{f_{uv} \le c_{uv}, \quad \text{for each} \quad u, v \in V} & \\
& \displaystyle\sum_{v \in V} f_{vu} = \sum_{v \in V} f_{uv}, \quad \text{for each } u \in V - \{s, t\} & \\
& \boxed{f_{uv} \ge 0, \quad \text{for each} \quad u, v \in V} &
\end{array}
$$

0 ≤ x ≤ c

Back to maximum flow

$$
\begin{aligned}
\text{maximise} \quad & \sum_{v \in V} f_{sv} - \sum_{v \in V} f_{vs} \\
\text{subject to} \quad & \boxed{f_{uv} \le c_{uv}, \quad \text{for each} \quad u, v \in V} \\
& \boxed{\sum_{v \in V} f_{vu} = \sum_{v \in V} f_{uv}, \quad \text{for each } u \in V - \{s, t\}} \\
& \boxed{f_{uv} \ge 0, \quad \text{for each} \quad u, v \in V}
\end{aligned}
$$

0 ≤ x ≤ c

Back to maximum flow

$$\text{maximise} \quad \sum_{v \in V} f_{sv} - \sum_{v \in V} f_{vs}$$

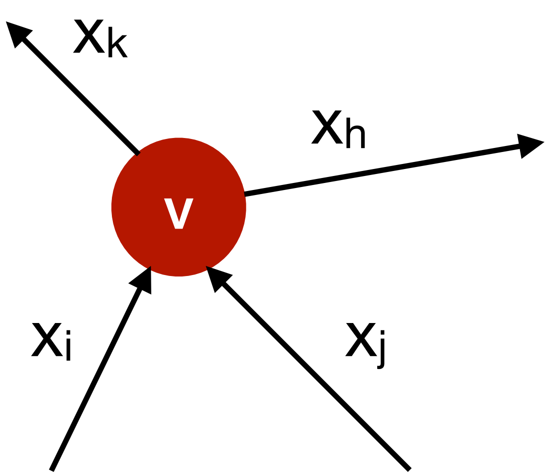
$$\begin{aligned}
\text{subject to} \quad & \boxed{f_{uv} \le c_{uv}, \quad \text{for each} \quad u, v \in V} \\
& \boxed{\sum_{v \in V} f_{vu} = \sum_{v \in V} f_{uv}, \quad \text{for each } u \in V - \{s, t\}} \\
& \boxed{f_{uv} \ge 0, \quad \text{for each} \quad u, v \in V}
\end{aligned}$$

$0 \le x \le c$

$Ax = 0$

Back to maximum flow

$$
\begin{aligned}
\text{maximise} \quad & \sum_{v \in V} f_{sv} - \sum_{v \in V} f_{vs} \\
\text{subject to} \quad & f_{uv} \le c_{uv}, \quad \text{for each} \quad u, v \in V \\
& \sum_{v \in V} f_{vu} = \sum_{v \in V} f_{uv}, \quad \text{for each } u \in V - \{s, t\} \\
& f_{uv} \ge 0, \quad \text{for each} \quad u, v \in V
\end{aligned}
$$

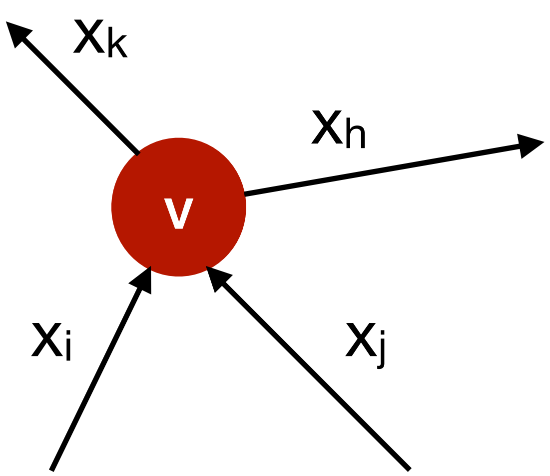


$0 \le x \le c$

$Ax = 0$

Back to maximum flow

$$
\begin{aligned}
\text{maximise} \quad & \sum_{v \in V} f_{sv} - \sum_{v \in V} f_{vs} \\
\text{subject to} \quad & f_{uv} \le c_{uv}, \quad \text{for each} \quad u, v \in V \\
& \sum_{v \in V} f_{vu} = \sum_{v \in V} f_{uv}, \quad \text{for each } u \in V - \{s, t\} \\
& f_{uv} \ge 0, \quad \text{for each} \quad u, v \in V
\end{aligned}
$$



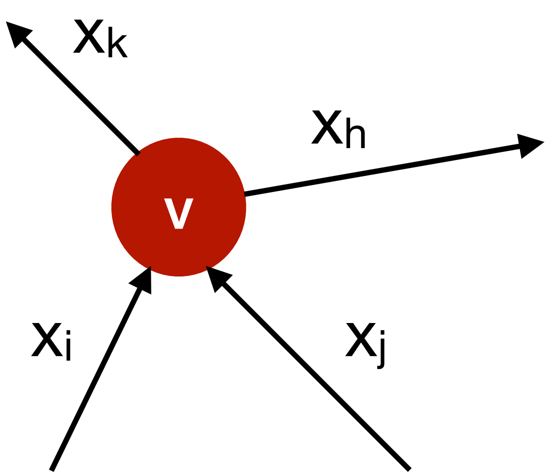
$0 \le x \le c$

$Ax = 0$

$x_k + x_h - x_i - x_j = 0$

Back to maximum flow

$$
\begin{array}{lll}
\text{maximise} & \displaystyle\sum_{v \in V} f_{sv} - \sum_{v \in V} f_{vs} & \\
\text{subject to} & f_{uv} \le c_{uv}, & \text{for each } u, v \in V \\
& \displaystyle\sum_{v \in V} f_{vu} = \sum_{v \in V} f_{uv}, & \text{for each } u \in V - \{s, t\} \\
& f_{uv} \ge 0, & \text{for each } u, v \in V
\end{array}
$$



$0 \le x \le c$

$Ax = 0$

$x_k + x_h - x_i - x_j = 0$

$A_{vi} = A_{vj} = -1$

$A_{vk} = A_{vh} = 1$

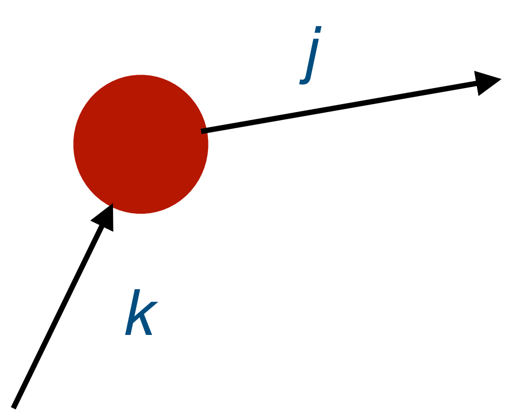
Back to maximum flow

- Consider the *incidence matrix* of the flow network (without s and t):
  - $A_{ij} = 1$ if edge j **starts** at node i in G_f.
  - $A_{ij} = -1$ if edge j **ends** at node i in G_f.
  - $A_{ij} = 0$ otherwise.

Nodes/Edges		*j*	*k*
i		1	-1

Back to maximum flow

- Consider the *incidence matrix* of the flow network (without s and t):
  - $A_{ij} = 1$ if edge j **starts** at node i in G_f.
  - $A_{ij} = -1$ if edge j **ends** at node i in G_f.
  - $A_{ij} = 0$ otherwise.



Nodes/Edges		*j*	*k*
i		1	-1

Back to maximum flow

Consider the *incidence matrix* of the flow network (without s and t):

Nodes/Edges		*j*	*k*
i		1	-1

Back to maximum flow

Consider the *incidence matrix* of the flow network (without s and t):

This is precisely the matrix A of the max flow LP.

Nodes/Edges		*j*	*k*
i		1	-1

Back to maximum flow

Consider the *incidence matrix* of the flow network (without s and t):

This is precisely the matrix A of the max flow LP.

It suffices to prove that A is totally unimodular, by Corollary.

Nodes/Edges		*j*	*k*
i		1	-1

Back to maximum flow

Consider the *incidence matrix* of the flow network (without s and t):

This is precisely the matrix A of the max flow LP.

It suffices to prove that A is totally unimodular, by Corollary.

Lemma: The incidence matrix of any directed graph is totally unimodular.

Nodes/Edges		*j*	*k*
i		1	-1