Programación lineal

Simona Bernardi

Universidad de Zaragoza

Grado en Ingeniería Informática

Resumen

- Introducción
- Método general
  - Formulación del problema general
  - Interpretación geométrica
  - Algoritmo símplex
  - Dualidad
  - Reducciones
- Aplicación
  - El problema de la fábrica de cerveza
  - El problema de flujo máximo/mínimo corte
  - El problema de emparejamiento bipartido

Aplicaciones

- Agricultura (problema de la dieta)
- Energía (problema de máximo flujo de la red de distribución)
- Finanzas (optimización de portafolios de inversión)
- Logística (gestión de cadenas de suministro)
- Marketing (asignación territorios de ventas)
- Producción industrial (problema cutting-stock)
- Telecomunicaciones (diseño de redes) ... y muchas más

El problema de la fábrica de cerveza

- Dos nuevos tipos diferentes de cerveza: Ale y Lambic
- Hay una cantidad limitada de materia prima
  - Trigo: 384Kg
  - Lúpulo: 4Kg
  - Cebada: 476Kg
- Cada tipo requiere diferentes proporciones de materia prima
- Hay un beneficio diferente en la venta de cada tipo
- ¿Cuántos barriles hay que producir para maximizar el beneficio?

1 barril (50 litros)	trigo (Kg.)	lúpulo (Kg.)	cebada (Kg.)
Ale	4	0.1	14
Lambic	12	0.1	8

	1 barril
Ale	12€
Lambic	20€

Formalización en un problema de programación lineal

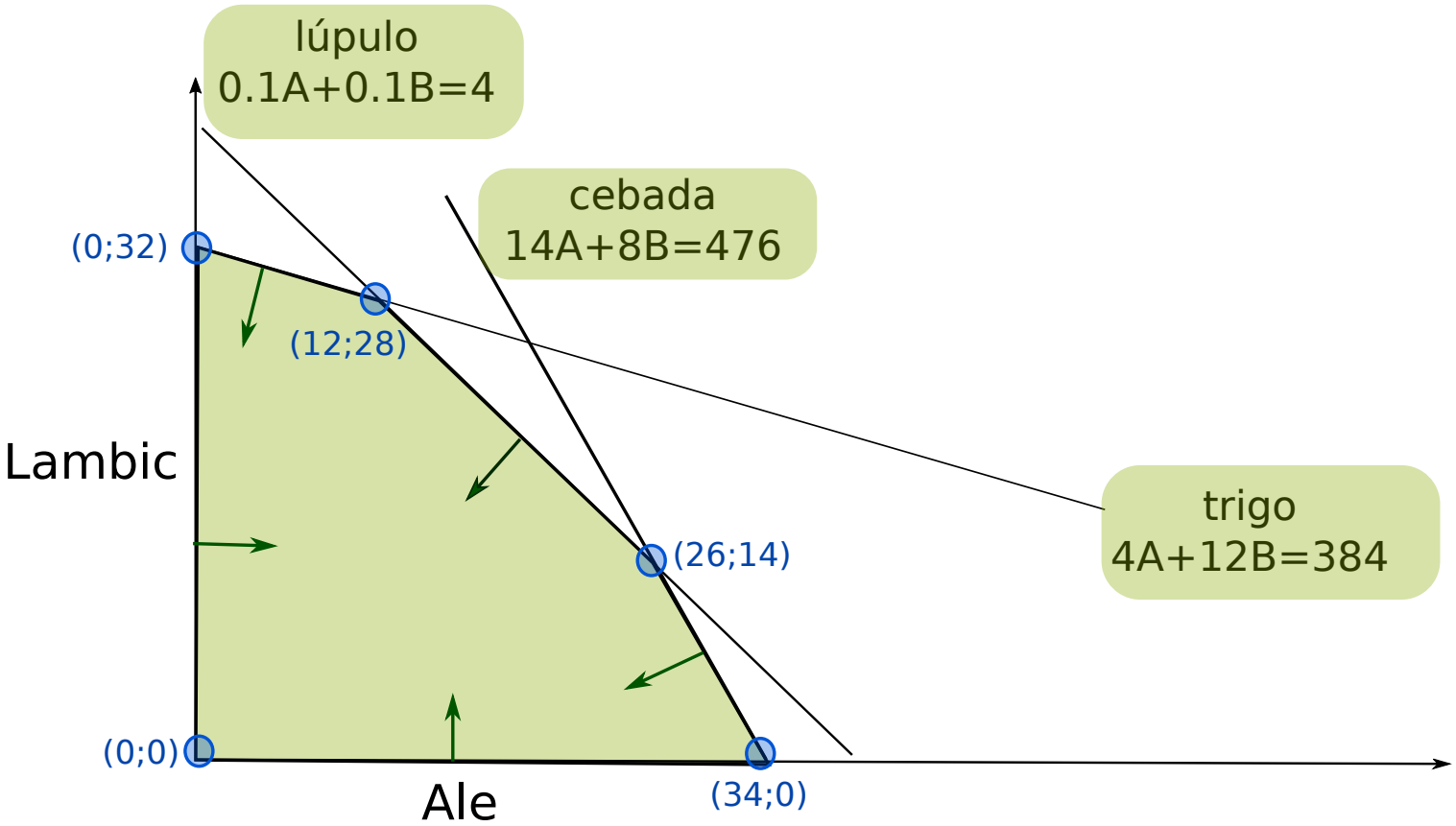
- Sean
  - A número de barriles del tipo Ale
  - B número de barriles del tipo Lambic

$$
\begin{array}{lll}
max & 12A + 20B & (beneficio) \\
t.q. & 4A + 12B \leq 384 & (trigo) \\
 & 0{,}1A + 0{,}1B \leq 4 & (lúpulo) \\
 & 14A + 8B \leq 476 & (cebada) \\
 & A, B \geq 0 &
\end{array}
$$

Resolución en el plano cartesiano

Región factible

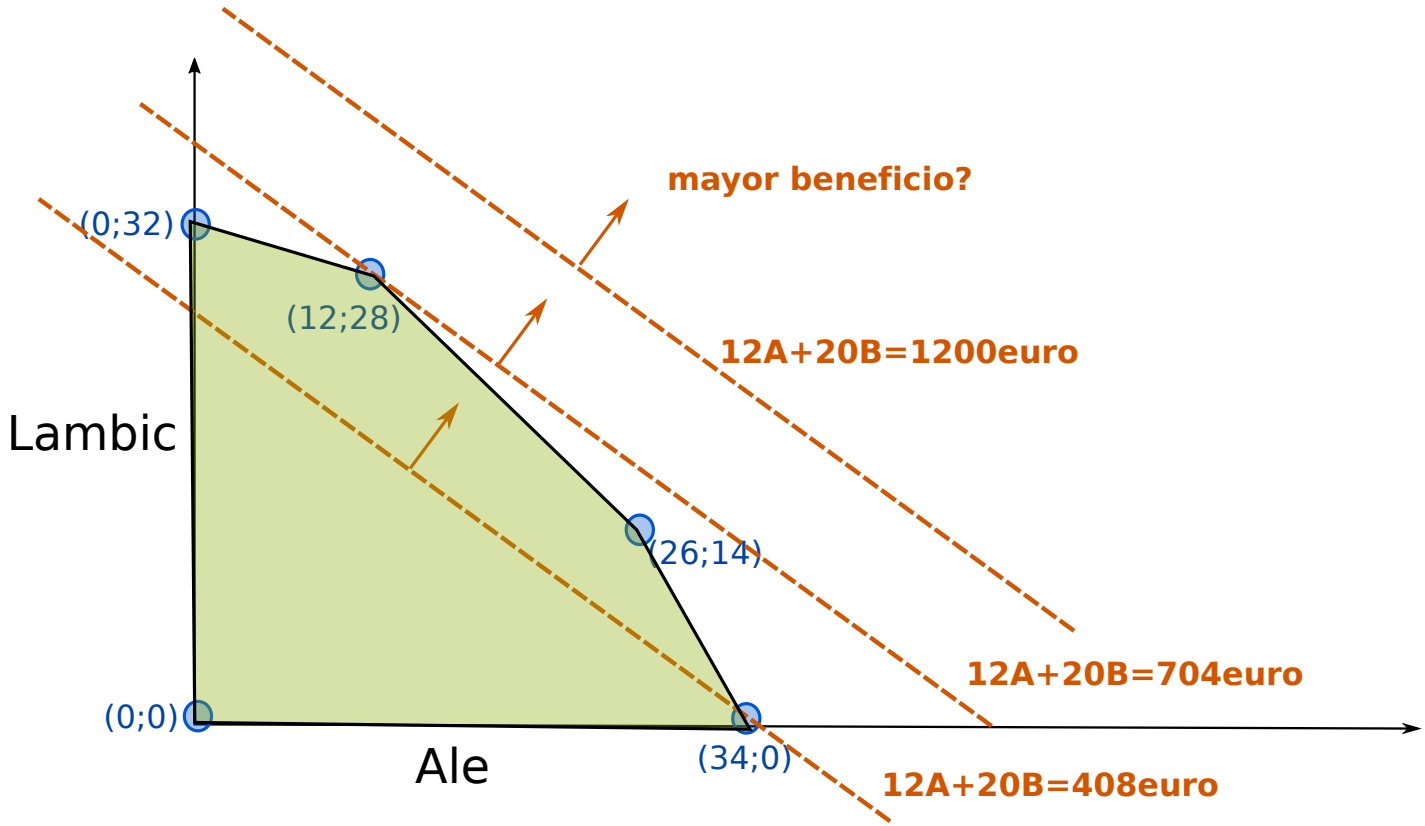
- Las desigualdades definen semiplanos
- La región factible es un polígono convexo



Resolución en el plano cartesiano

Función objetivo

- Gradiente en el origen de la función $\nabla = (12, 20)$
- La solución óptima se encuentra en un vértice
  - Intersección de dos rectas en $\mathbb{R}^2$



Formulación del problema LP

Objetivo

Maximizar (o minimizar) una función lineal de n variables, sujeta a m desigualdades o igualdades $(\leq, \geq, =)$ lineales.

$$
\begin{aligned}
max(min) \quad & c_1x_1 + c_2x_2 + \cdots + c_nx_n \\
t.q. \quad & a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \lessgtr b_1 \\
& a_{12}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \lessgtr b_2 \\
& \vdots \qquad\qquad \vdots \\
& a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \lessgtr b_m \\
& x_1, x_2, \ldots, x_n \in \mathbb{R}
\end{aligned}
$$

- Datos en entrada:
  - $\mathbf{A} = (a_{ij}) \in \mathbb{R}^{m \times n}$
  - $\mathbf{b} = (b_j) \in \mathbb{R}^m$
  - $\mathbf{c} = (c_i) \in \mathbb{R}^n$
- Variables:
  - $\mathbf{x} = (x_i) \in \mathbb{R}^n$

$$max(min)\, \mathbf{c^T x} \quad s.t. \quad \mathbf{Ax} \lessgtr \mathbf{b}$$

Formas estándar

- Se añaden las restricciones de signo
- Algunos autores llaman estándar a la canónica otros a la *slack*

Forma canónica

$$
\begin{aligned}
max\ & c_1x_1 + c_2x_2 + \cdots + c_nx_n \\
t.q.\ & a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \leq b_1 \\
& a_{12}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \leq b_2 \\
& \vdots \qquad\qquad\qquad\qquad \vdots \\
& a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \leq b_m \\
& x_1, x_2, \ldots, x_n \geq 0
\end{aligned}
$$

$$max\ \mathbf{c^T x}\quad t.q.\quad \mathbf{Ax} \leq \mathbf{b}, \mathbf{x} \geq \mathbf{0}$$

Forma(s) *slack*

$$
\begin{aligned}
max\ & c_1x_1 + c_2x_2 + \cdots + c_nx_n \\
t.q.\ & a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\
& a_{12}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\
& \vdots \qquad\qquad\qquad\qquad \vdots \\
& a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n = b_m \\
& x_1, x_2, \ldots, x_n \geq 0
\end{aligned}
$$

$$max\ \mathbf{c^T x}\quad t.q.\quad \mathbf{Ax} = \mathbf{b}, \mathbf{x} \geq \mathbf{0}$$

Transformación del problema en forma *slack*

Problema original

$$
\begin{aligned}
max\ & 12A + 20B \\
t.q.\ & 4A + 12B \leq 384 \\
& 0{,}1A + 0{,}1B \leq 4 \\
& 14A + 8B \leq 476 \\
& A, B \geq 0
\end{aligned}
$$

Forma *slack*

$$
\begin{array}{lrrrrrrr}
max & Z & & & & & & \\
 & 12A & +20B & & & & -Z = & 0 \\
\hline
t.q. & 4A & +12B & +S_c & & & = & 384 \\
 & 0{,}1A & +0{,}1B & & +S_d & & = & 4 \\
 & 14A & +8B & & & +S_e & = & 476 \\
 & A, & B, & S_c, & S_d, & S_e, & Z \geq & 0
\end{array}
$$

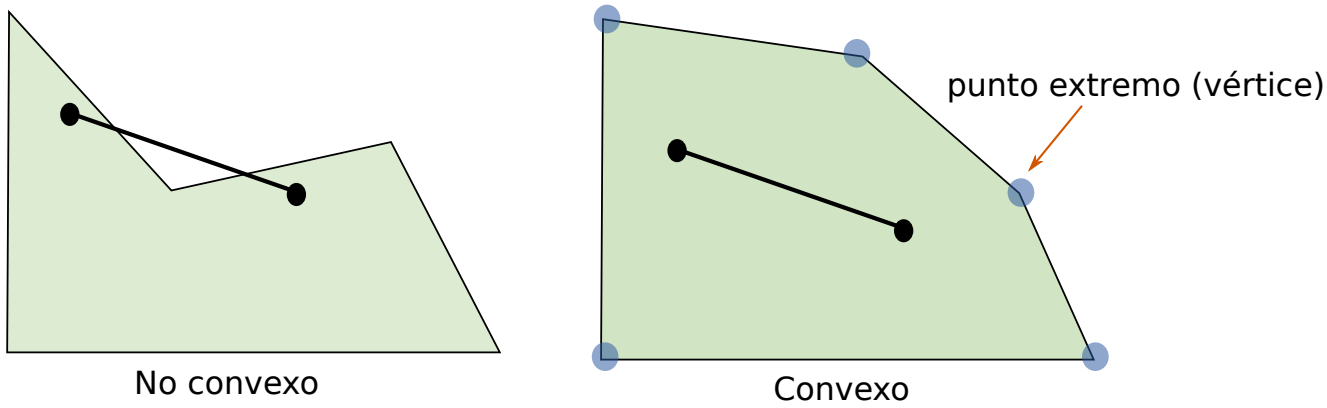
- Se añade
  - Una variable Z y la ecuación correspondiente a la función objetivo
  - Una variable *slack* para transformar una desigualdad en igualdad
  - Restricciones de signo
- Problema en $\mathbb{R}^6$

Resolución en el espacio $\mathbb{R}^n$

Dado un problema LP:

- Un punto $x \in \mathbb{R}^n$ es factible si satisface a todas restricciones lineales
- El conjunto de los puntos factibles se llama **región factible**

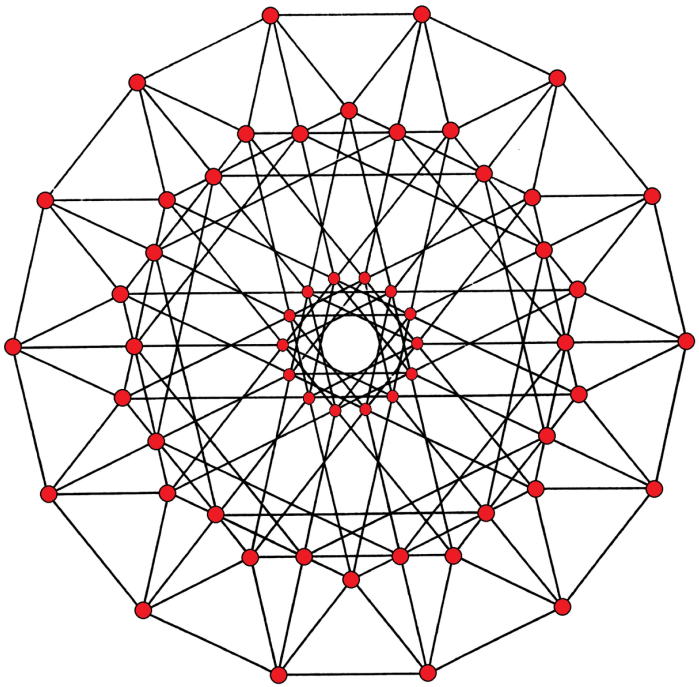
- Las restricciones de igualdad definen **hiperplanos** en $\mathbb{R}^n$
- Las restricciones de desigualdad definen **semiespacios** $\mathbb{R}^n$
- La región factible es la intersección de los hiperplanos y semiespacios $\Rightarrow$ **poliedro convexo**



Ejemplo en $\mathbb{R}^2$

Propiedad de los vértices

- Si existe una solución óptima al problema LP, entonces existe una **solución óptima que es un vértice**
  - El número de vértices a considerar es **finito**
  - El número de vértices puede ser **exponencial**



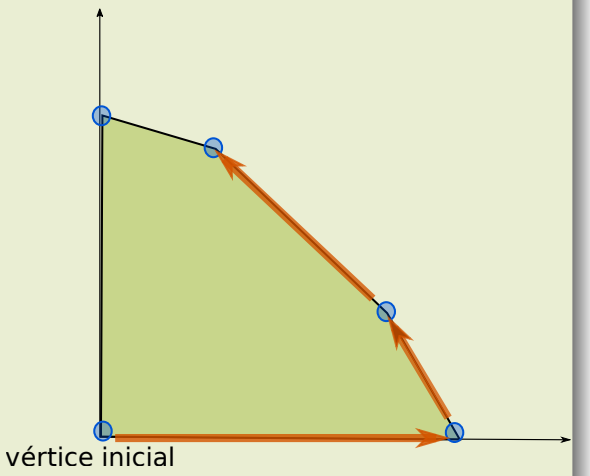
- Los óptimos locales son óptimos globales porque la función objetivo es lineal y la región factible es convexa
- Propiedad **voraz**: un vértice es óptimo si y sólo si no hay mejor vértice adyacente

Algoritmo símplex [George Dantzig, 1947]

- Desarrollado después de la II guerra Mundial para resolver problemas de lógistica
- Clasificado como uno de los 10 mejores algoritmos del siglo XX

Idea básica

1. Empieza en un vértice de la región factible
2. Se mueve a un vértice adyacente que genera un mejor valor para la función objetivo
3. Repite el paso (2) hasta que no exista mejor vértice
4. El último vértice visitado es un óptimo global

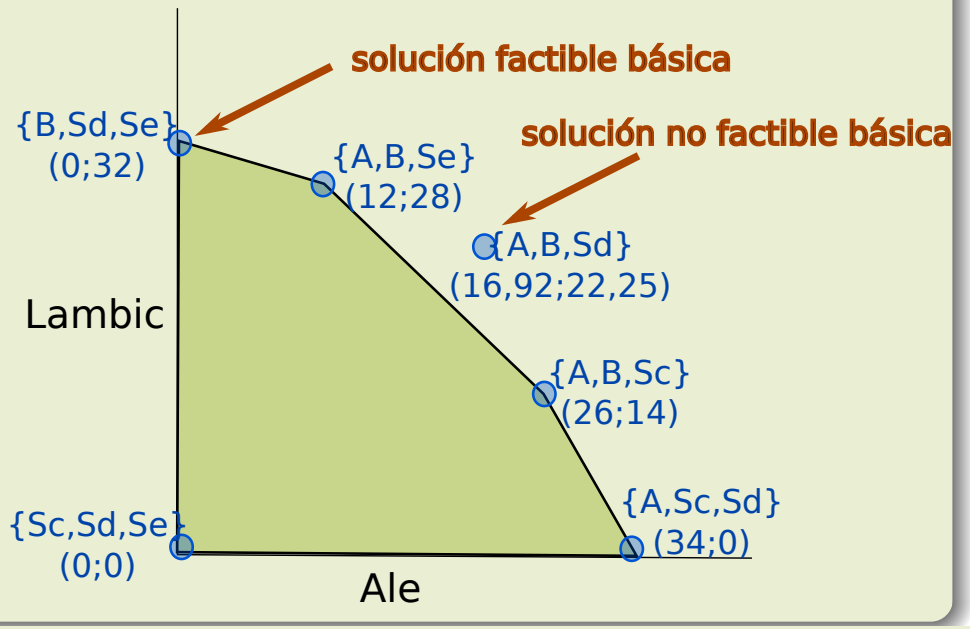


Ejemplo en $\mathbb{R}^2$

Solución factible básica

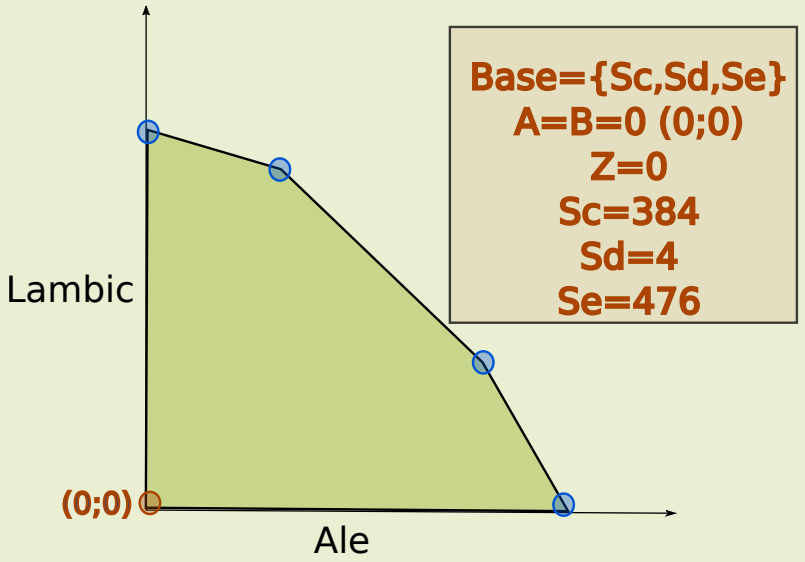
- Una **base** es un subconjunto de $p < n$ variables
- **Solución factible básica (SFB)**
  - Inicializa a cero las $n - p$ variables que no están en la base
  - Resolve p ecuaciones en p variables
  - Si existe una solución única y factible, entonces es una SFB
  - SFB $\Leftrightarrow$ vértice

$$
\begin{array}{lrrrrrrr}
max & Z & & & & & & \\
 & 12A & +20B & & & -Z & = & 0 \\
\hline
t.q. & 4A & +12B & +S_c & & & = & 384 \\
 & 0{,}1A & +0{,}1B & & +S_d & & = & 4 \\
 & 14A & +8B & & & +S_e & = & 476 \\
 & A, & & B, & S_c, & S_d, & S_e, Z \geq & 0
\end{array}
$$



Inicialización

$$
\begin{array}{lrcl}
max & Z & & \\
 & 12A + 20B & -Z = & 0 \\
\hline
t.q. & 4A + 12B + S_c & = & 384 \\
 & 0{,}1A + 0{,}1B + S_d & = & 4 \\
 & 14A + 8B + S_e & = & 476 \\
 & A, \quad B, \; S_c, \; S_d, \; S_e, \; Z & \geq & 0
\end{array}
$$



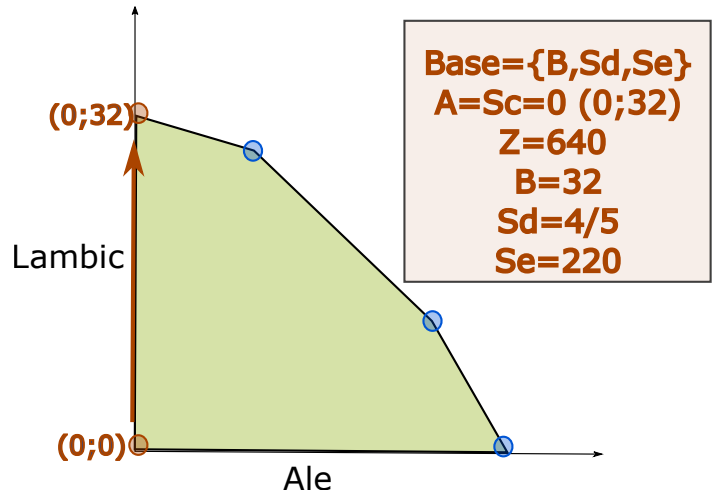
- Solución factible inicial:
  - Variables *slack* en la base $\{S_C, S_D, S_E\}$
  - Variables A, B inicializadas a cero
  - Solución 4 ecuaciones, 4 variables

Pivote – primera iteración

$$
\begin{array}{lrrrrrrr}
max & Z & & & & & & \\
 & 12A & +20B & & & -Z= & 0 & \\
\hline
t.q. & 4A & +12B & +S_c & & = & 384 & \\
 & \frac{1}{10}A & +\frac{1}{10}B & & +S_d & = & 4 & \\
 & 14A & +8B & & +S_e & = & 476 & \\
 & A, & & B, & S_c, S_d, S_e, Z \geq & & 0 &
\end{array}
$$

$$
\begin{array}{lrrrrrrr}
max & Z & & & & & & \\
 & \frac{16}{3}A & & -\frac{5}{3}S_c & & -Z= & -640 & (1) \\
\hline
t.q. & \frac{1}{3}A & +B & +\frac{1}{12}S_c & & = & 32 & (2) \\
 & \frac{1}{15}A & & -\frac{1}{120}S_c & +S_d & = & \frac{4}{5} & (3) \\
 & \frac{34}{3}A & & -\frac{2}{3}S_c & +S_e & = & 220 & (4) \\
 & A, & B, & & S_c, S_d, S_e, Z \geq & & 0 &
\end{array}
$$

- Sustituir $B = \frac{1}{12}(384 - 4A - S_c)$
- Añadir B a la base
- Volver a escribir la ecuación (2)
- Eliminar B en las ecuaciones (1,3,4)



Selección del pivote

$$
\begin{array}{lrcl}
max & Z \downarrow & & \\
 & 12A + 20B - Z = & 0 & (1) \\
\hline
t.q. \rightarrow & 4A + 12B + S_c = & 384 & (2) \\
 & \frac{1}{10}A + \frac{1}{10}B + S_d = & 4 & (3) \\
 & 14A + 8B + S_e = & 476 & (4) \\
 & A, B, S_c, S_d, S_e, Z \geq & 0 &
\end{array}
$$

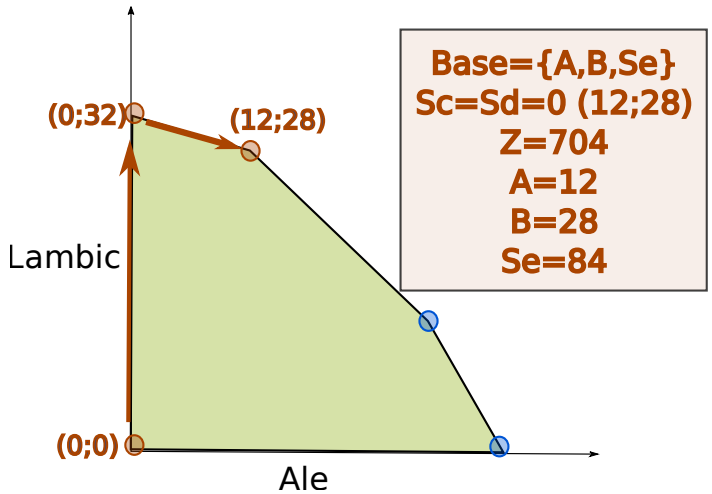
- ¿Porqué elegimos el pivote en la columna de la variable B?
  - Tiene un coeficiente positivo en la función objetivo (cada incremento de una unidad en B incrementa la función objetivo de 20€)
  - La columna de la variable A también vale
- ¿Porqué elegimos el pivote en la fila (2)?
  - Preserva la factibilidad asegurando que $\mathbf{b} \geq \mathbf{0}$
  - Regla mínimo ratio: $min\{\frac{384}{12}, 40, \frac{476}{8}\}$

Pivote – segunda iteración

$$
\begin{array}{lrcl}
max & Z & & \\
 & \frac{16}{3}A \quad -\frac{5}{3}S_c \quad -Z = -640 & & (1) \\
\hline
t.q. & \frac{1}{3}A + B + \frac{1}{12}S_c = 32 & & (2) \\
 & \frac{1}{15}A - \frac{1}{120}S_c + S_d = \frac{4}{5} & & (3) \\
 & \frac{34}{3}A - \frac{2}{3}S_c + S_e = 220 & & (4) \\
 & A,\ B,\ S_c,\ S_d,\ S_e,\ Z \geq 0 & &
\end{array}
$$

$$
\begin{array}{lrcl}
max & Z & & \\
 & -S_c \ -80S_d \quad -Z = -704 & & \\
\hline
t.q. & +B + \frac{1}{8}S_c \ -5S_d = 28 & & \\
 & A \ -\frac{1}{8}S_c \ +15S_d = 12 & & \\
 & +\frac{3}{4}S_c - 170S_d + S_e = 84 & & \\
 & A,\ B,\ S_c,\ S_d,\ S_e,\ Z \geq 0 & &
\end{array}
$$

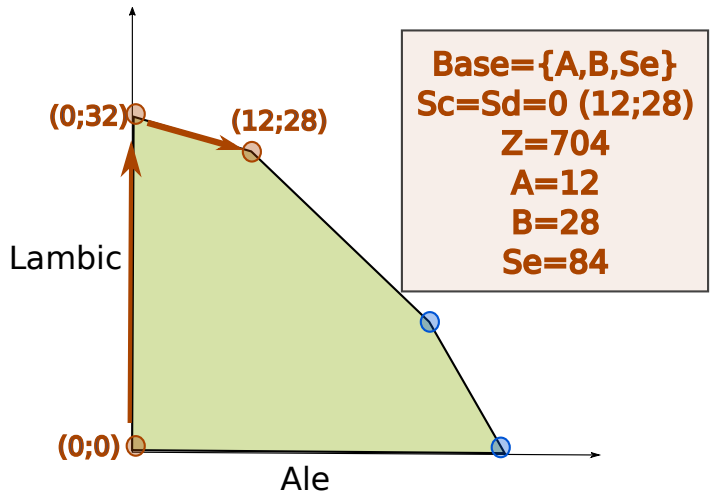
- Sustituir $A = 15(\frac{4}{5} + \frac{1}{120}S_c - S_d)$
- Añadir A a la base
- Volver a escribir la ecuación (3)
- Eliminar A en las ecuaciones (1,2,4)



Optimalidad

$max\ Z$

$$
\begin{array}{llllll}
 & & -S_c & -80S_d & -Z = -704 & (1) \\
\hline
t.q. & +B + \frac{1}{8}S_c & -5S_d & = & 28 & (2) \\
A & -\frac{1}{8}S_c & +15S_d & = & 12 & (3) \\
 & +\frac{1}{3}S_c & -170S_d + S_e & = & 84 & (4) \\
A,\ B, & S_c, & S_d, & S_e, & Z \geq 0 &
\end{array}
$$

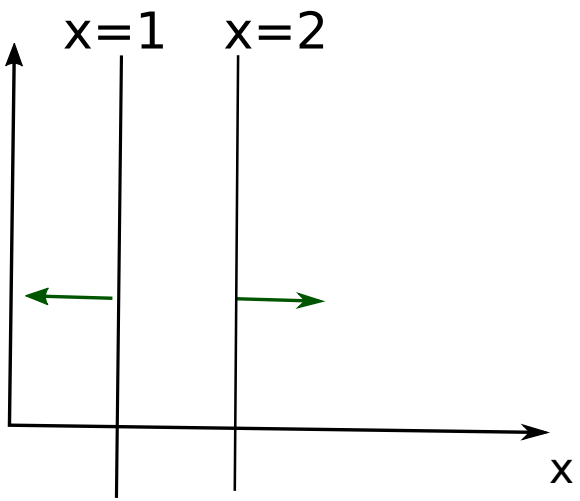


- ¿Cuándo se termina?
  - Cuando no hay coeficientes positivos en la función objetivo
- ¿Porqué la última SFB es óptima?
  - Cualquier SFB satisface el sistema de ecuaciones en curso
  - En particular la (1): $Z = 704 - S_c - 80S_d$
  - El valor óptimo $z^* \leq 704$ porque $S_c, S_d \geq 0$
  - La SFB en curso tiene valor 704 $\Rightarrow$ es óptima

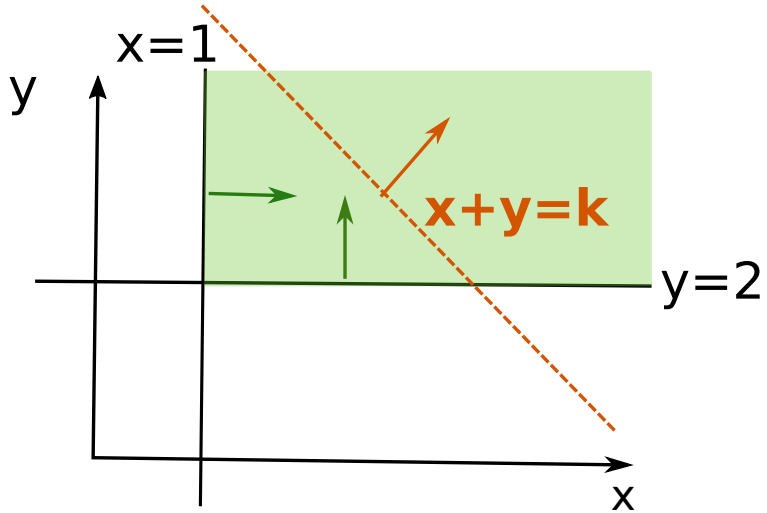
Excepciones

- El problema LP no es factible
  - No es posible satisfacer todas las restricciones (región factible $= \emptyset$)
- El problema LP no es acotado

max x
t.q. x <=1;
x >=2



max x+y
t.q. x >=1;
y >=2



Ejemplos en $\mathbb{R}^2$

Símplex: construcción del cuadro inicial

Problema inicial (forma canónica ⇒ forma *slack*)

$$max \sum_{i=1}^{n} c_i x_i$$

$$t.q. \sum_{j=1}^{n} a_{ij} x_j \leq b_i, i = 1, \ldots, m$$

$$x_i \geq 0 \qquad i = 1, \ldots, n$$

$$max\ z$$

$$t.q. \sum_{j \in N} a_{ij} x_j + x_i = b_i, i \in B$$

$$\sum_{j \in N} c_j x_j - z = -v$$

$$x_i, z \geq 0, v = 0 \qquad i \in N \cup B$$

- $N = \{i \mid x_i \text{ fuera de la base}\}$ $(|N| = n)$
- $B = \{i \mid x_i \text{ en base}\}$ $(|B| = m)$
- v término constante en la función objetivo

m	A	I	b
1	c	0	-v
	n	m	1

Símplex: esquema algorítmico


```
1  algoritmo símplex(entsal A: matriz, entsal b,c: vector)
2  variables ... {no se especifican por razones de espacio}
3  principio
4   factible := inicializaciónSímplex(A,b,c,N,B,v);
5    si not factible entonces escribir("Problema no factible")
6    sino
7     óptimo := falso; noAcotado := falso;
8     mientrasQue not óptimo and not noAcotado hacer
9      si not hayPositivosFO(c,N) entonces óptimo := verdadero
10     sino
11      colP := eligePositivo(c,N) {variable que entra en la base}
12      si not hayPositivos(A,B,colP) entonces noAcotado := verdadero
13      sino {b[filaP]/a[filaP][colP]=min(b[i]/a[i][colP]: i\in B,a[i][colP]>0)}
14       filaP := filaPivote(A,b,B,colP); {variable que sale de la base}
15       pivote(N,B,A,b,c,v,colP,filaP); {cambio de base}
16      fsi fsi
17    fmientrasQue
18    si óptimo entonces escribirSolución(b,B,v)
19    sino escribir("Problema no acotado")
20 fin
```

Cuestiones

1. ¿Cómo determinar si un problema es factible?
2. ¿Qué hacemos si un problema es factible, pero la solución básica inicial no lo es?

   ⇒ `inicializaciónSímplex`

3. ¿Cómo sabemos si un problema no está acotado?

   ⇒ `hayPositivos`

4. ¿Cómo elegimos las variables que entran en la base y las que salen?

   ⇒ `eligePositivo` y `filaPivote`

Factibilidad del problema y solución factible básica

Cuestiones 1 & 2

- La función `inicializaciónSimplex` determina si el problema es factible o no. Si lo es, devuelve la forma *slack* con la solución factible básica.

Enfoque `inicializaciónSimplex` (Fase 1 del símplex)

- Si $\mathbf{b} \geq \mathbf{0}$
  - La solución que corresponde al vertex origen es factible
  - Devuelve la forma *slack* del problema P (cuadro inicial)
- Sino crea un problema auxiliar P'
- Soluciona P'
- Dependiendo del valor de la solución de P' decide si el problema inicial P es factible ...

Factibilidad del problema y solución factible básica

Problema inicial P y problema auxiliar P'

$$P: \ max \sum_{i=1}^{n} c_i x_i$$

$$t.q. \sum_{j=1}^{n} a_{ij} x_j \leq b_i, i = 1, \ldots, m$$

$$x_i \geq 0 \qquad i = 1, \ldots, n$$

$$P': \ max \ -x_0$$

$$t.q. \sum_{j=1}^{n} a_{ij} x_j - x_0 \leq b_i, i = 1, \ldots, m$$

$$x_i \geq 0 \qquad i = 0, \ldots, n$$

Lemma

P es factible si y solo si el valor óptimo de P' es 0 (cero).

Factibilidad del problema y solución factible básica

Cuestiones 1 & 2

Enfoque `inicializaciónSimplex`

- Si $\mathbf{b} \geq \mathbf{0}$
  - La solución que corresponde al vertex origen es factible
  - Devuelve la forma *slack* del problema P (cuadro inicial)
- Sino crea un problema auxiliar P'
- Soluciona P'
- Dependiendo del valor de la solución de P' decide si el problema inicial P es factible ...
  - Si $x_0 = 0$ entonces P es factible y devuelve la forma *slack* final de P' borrando x_0 y recuperando la función objetivo original de P
  - Sino el problema P no es factible

Caso no acotado

Cuestión 3

- La función `hayPositivos(A,B,colP)` determina si hay elementos $A[i][colP], i \in B$ positivos: si no hay entonces el problema no está acotado
- Ninguna restricción del problema limita la cantidad a incrementar para la variable que entra en la base ($colP \in N$)

Ejemplo

$$P' : max\ x_1 + x_2$$
$$t.q.\ -x_1 \leq 1$$
$$-x_2 \leq 2$$
$$x_1, x_2 \geq 0$$

	x_1	x_2	x_3	x_4	b
c1:	-1	0	1	0	1
c2:	0	-1	0	1	2
z:	1	1	0	0	0

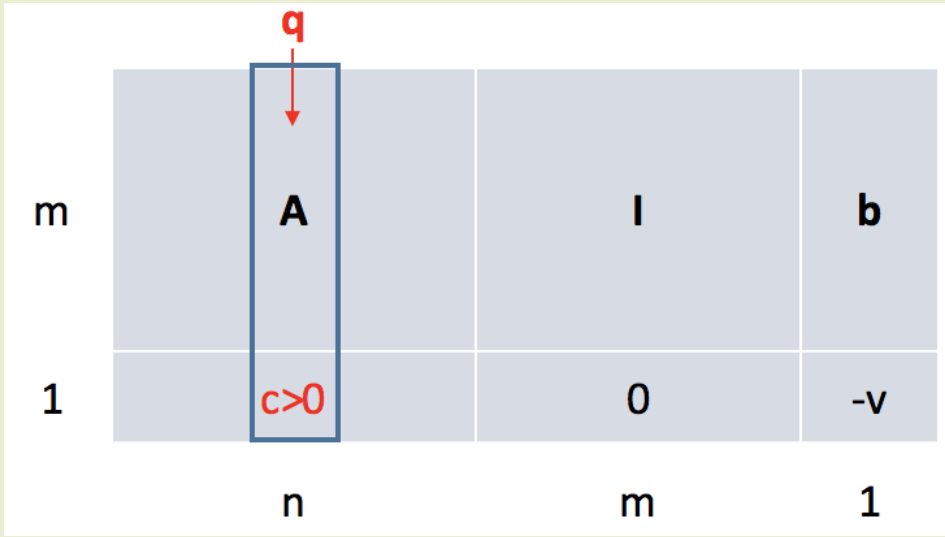
$B = \{3, 4\}$, $N = \{1, 2\}$

Elección del pivote

Cuestión 4: variable entrante

Función `eligePositivo(c,N)`

- La función elige la variable que **entra** en la base B
- ¿Cuál?
  - La variable con índice $q \in N$ que tiene un coeficiente positivo en la función objetivo



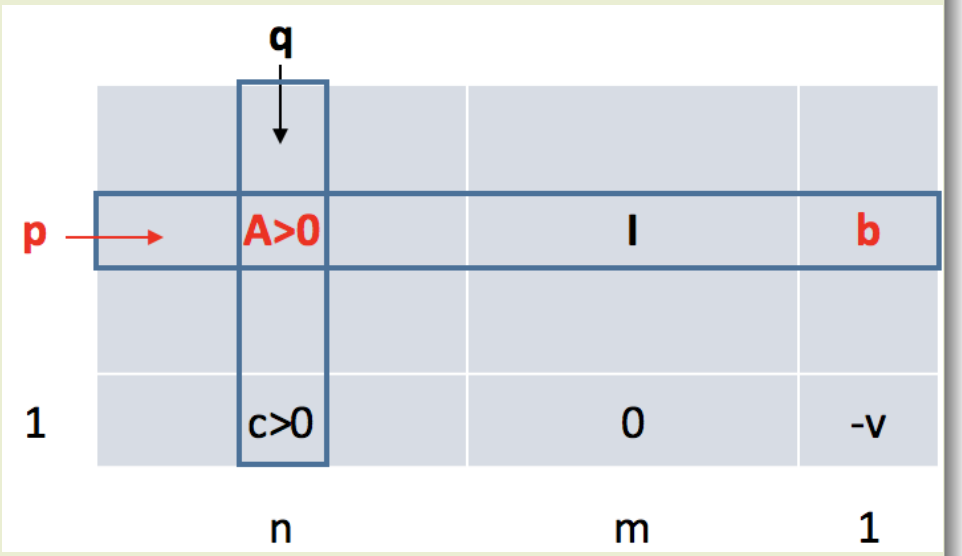
- ¿Si hay varias posibilidades?. . . hay diferentes criterios
- **Regla de Bland**: la variable con el índice menor

Elección del pivote

Cuestión 4: variable saliente

Función `filaPivote(A,b,B,q)`

- La función elige la variable que **sale** de la base B
- ¿Cuál?
- **Regla mínimo ratio**: la variable con índice $p \in N$ tal que:

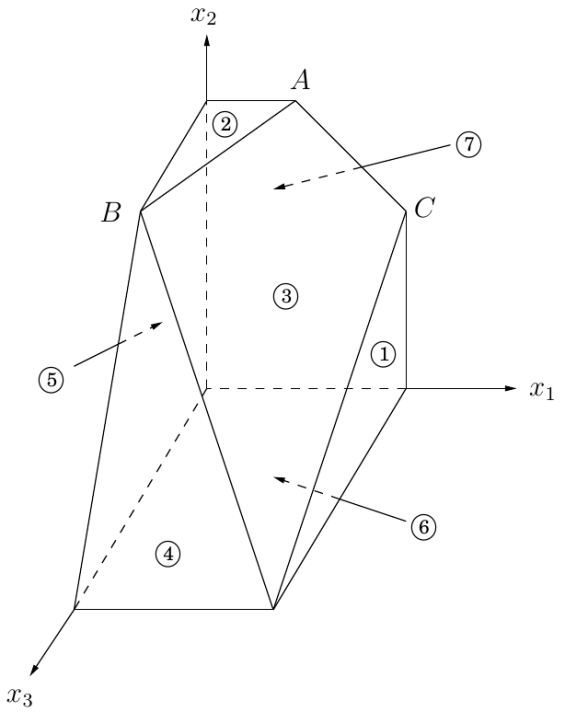


$$\frac{b[p]}{A[p][q]} = min\{\frac{b[i]}{A[i][q]} : i \in B, A[i][q] > 0\}$$

- ¿Si hay varias posibilidades?. . . hay diferentes criterios
- **Regla de Bland**: la variable con el índice menor

Caso de degeneración

- El vértice B es degenerado
  - Es la intersección de más de $n = 3$ caras del poliedro: ② ③ ④ ⑤
- Si se elige uno de los siguientes conjuntos de desigualdades
  - ② ③ ④ – ② ③ ⑤ – ② ④ ⑤ – ③ ④ ⑤
- y se soluciona el sistema de 3 ecuaciones en 3 variables de obtiene la misma solución en todo los casos



Fuente: S. Dasgupta et al. *Algorithms*

- **Problema!** El símplex puede pasar en ciclo por la diferentes bases que corresponden al mismo vértex
- Es importante el criterio de elección del pivote
  - P.ej. la regla de Bland asegura la selección de un número finito de pivotes

Tiempo de ejecución del símplex

Problema general

$$max\ \mathbf{c^T x}$$
$$t.q.\ \mathbf{Ax} \leq \mathbf{b}$$
$$\mathbf{x} \geq \mathbf{0}$$

n variables, m desigualdades

- Un vértice v es la intersección de n hiperplanos
- Cada vértice vecino u comparte con v $n-1$ hiperplanos
- v tiene como máximo $n \cdot m$ vecinos (elegir un hiperplano a quitar y otro a añadir)

- Tiempo de ejecución en cada iteración: $O(nm)$
- ¿Cuántas iteraciones puede haber?
  - Cota superior del número de vértices: $\binom{n+m}{n}$
  - **Exponencial en n!**
- Sin embargo **en la mayoría de las aplicaciones prácticas es polinomial**

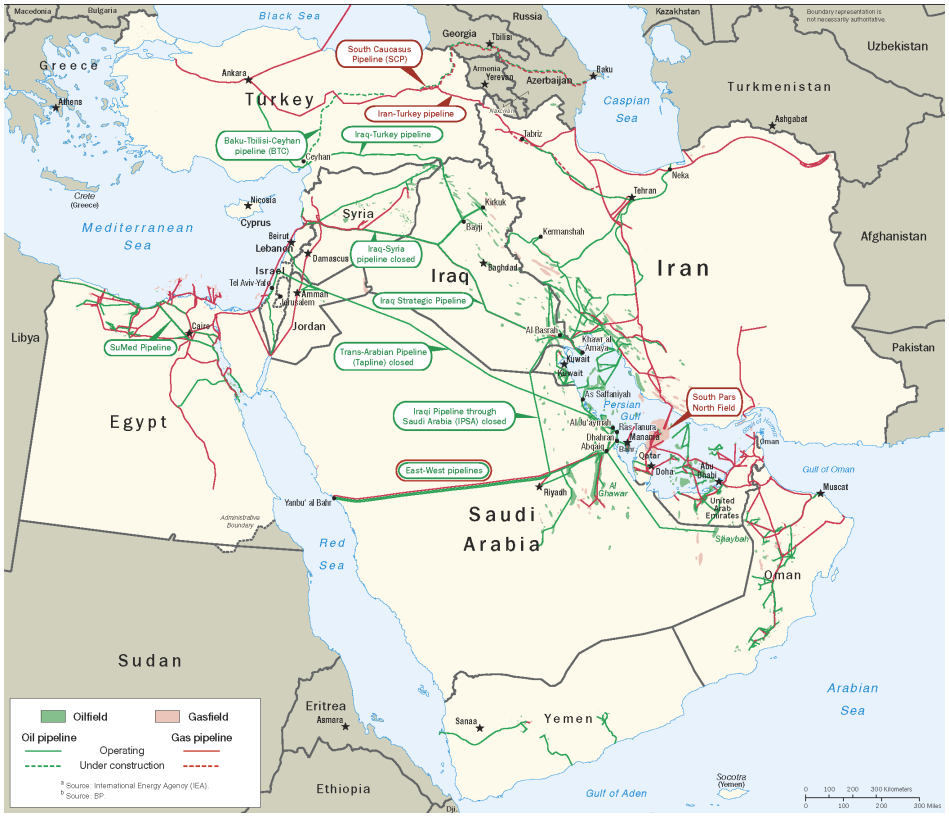
Observaciones

Problemas de implementación

- No factibilidad ⇒ fase 1 del símplex
- Caso no acotados ⇒ regla bien definida
- Mantener la dispersión ⇒ estructuras de datos ad hoc
- Caso de degeneración
- Estabilidad numérica

- Buena práctica: **no** implementarlo!
- Implementaciones disponibles en muchos entornos de programación
- Hay muchas herramientas disponibles que pueden resolver problemas con **millones** de variables
- Hay lenguajes de modelado para la especificación de problemas LP

La red de oleoductos de Arabia Saudita



- **Problema1** ¿Cuál es la máxima velocidad a la que se puede enviar el crudo desde los campos petroleros a las terminales de los puertos marítimos?

  $\Rightarrow$ Máximo flujo

- **Problema2** ¿Cuál es la forma más barata de sabotear la red?

  $\Rightarrow$ Mínimo corte

Fuente: `https://www.eia.gov/beta/international/analysis.php?iso=SAU`

Definición de flujo

- La red es un grafo dirigido $G = (V, E)$ con dos nodos especiales $s, t \in V$ (origen y destino)
- Un flujo-(s, t) es una función $f : E \to \mathbb{R}$ que satisface a la siguiente **restricción de conservación** en cada nodo $v \in V$ (excepto posiblemente s y t):

$$\sum_u f(u \to v) = \sum_w f(v \to w)$$

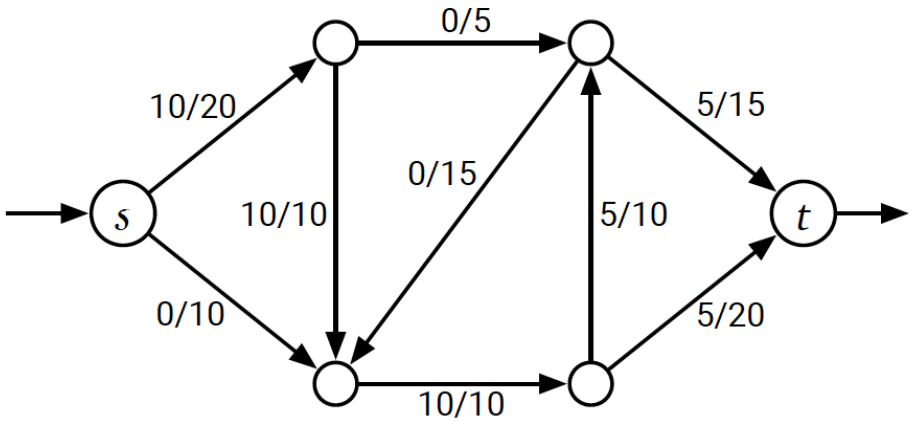
- El **valor** del flujo f es el flujo total de la red que sale de la origen $s \in V$:

$$|f| = \sum_w f(s \to w) - \sum_u f(u \to s)$$

- Se puede demostrar que $|f| = \sum_u f(u \to t) - \sum_w f(t \to w)$

Problema de flujo máximo

- Sea $c : E \to \mathbb{R}_{\geq 0}$ una función que asigna **capacidades** no negativas a cada arco $e \in E$
- Un flujo f es **factible** si $0 \leq f(e) \leq c(e)$ para cada arco $e \in E$
  - f **satura** $e \Leftrightarrow f(e) = c(e)$
  - f **evita** $e \Leftrightarrow f(e) = 0$



Flujo-(s, t) factible con valor $|f| = 10$

Fuente: J. Erickson, *Algorithms*, cap. 10.

Problema LP

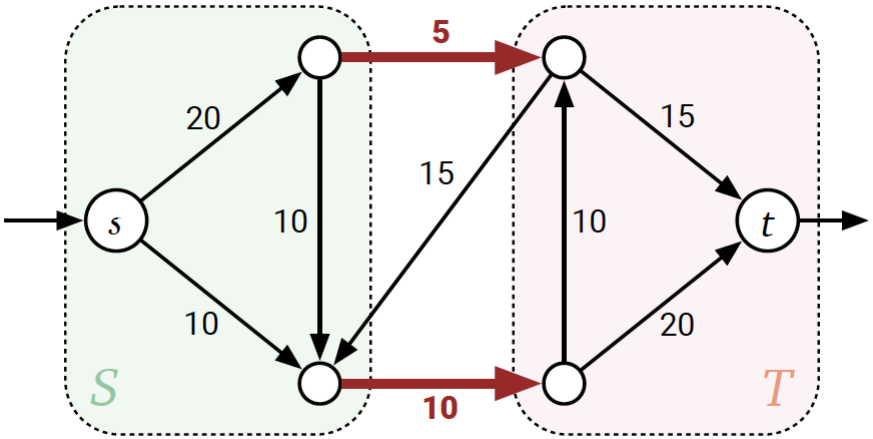
$$
\begin{aligned}
& max \quad |f| \\
& t.q. \sum_u f(u \to v) = \sum_w f(v \to w), \quad \forall v \in V \setminus \{s, t\} \\
& \qquad 0 \leq f(e) \leq c(e), \qquad\qquad \forall e \in E
\end{aligned}
$$

Definición de corte

- Un corte-(s, t) es una partición de los nodos del grafo G en dos subconjuntos S, T ($S \cup T = V, S \cap T = \emptyset$) tales que $s \in S$ y $t \in T$
- Sea c la función de capacidad de los arcos, la **capacidad** del corte, se define como:

$$||S, T|| = \sum_{v \in S} \sum_{w \in T} c(v \to w)$$

- Intuitivamente, el corte mínimo es la forma más barata de interrumpir todo el flujo de s a t



Corte-(s, t) con capacidad $||S, T|| = 15$

Fuente: J. Erickson, *Algorithms*, cap. 10.

Relación entre flujos y cortes

Lemma

Sea f un flujo-(s,t) factible y (S,T) un corte-(s,t), entonces el valor de f es **menor o igual** a la capacidad de (S,T). Además $|f| = ||S,T||$ si y solo si f **satura** cada arco de S a T y **evita** cada arco de T a S.

Teorema máximo flujo-mínimo corte [Ford-Fulkerson 1954]

En cada red de flujo $G = (V,E)$ con origen s y destino t, el valor del máximo flujo-(s,t) es igual a la capacidad del mínimo corte-(s,t).

- El algoritmo de Ford-Fulkerson se basa en:
  1. Considerar el flujo-(s,t) factible $f \equiv 0$
  2. Elegir cualquier camino *apropiado* de s a t e incrementar el flujo en los arcos del camino *cuanto más posible*
  3. Repetir el paso (2) hasta que no hay caminos apropiados

Capacidad residual

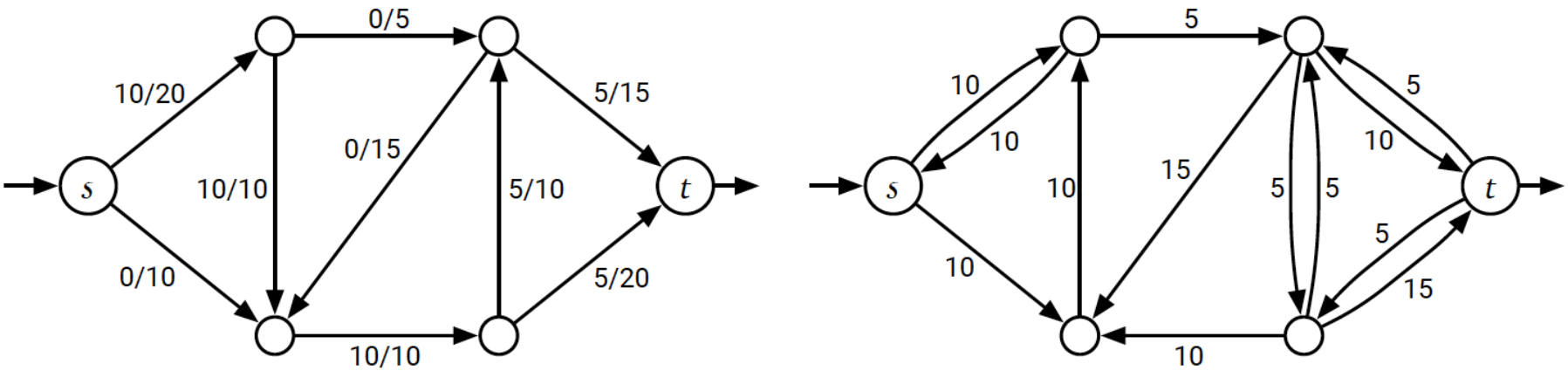
- Dada una red de flujo $G = (V, E)$ con capacidad c y un flujo-(s, t) factible, la **capacidad residual** es una función $c_f : E \to \mathbb{R}_{\geq 0}$ así definida:

$$c_f(u \to v) = \begin{cases} c(u \to v) - f(u \to v) & \text{si } u \to v \in E \\ f(v \to u) & \text{si } v \to u \in E \\ 0 & \text{de otra manera} \end{cases}$$

- La capacidad residual de un arco indica cuanto más flujo se puede introducir en el arco
- Es posible tener una capacidad residual $c_f(u \to v) > 0$ aunque $u \to v \notin E$

Grafo residual

- El **grafo residual** $G_f = (V, E_f)$ es el grafo asociado a la red de flujo $G = (V, E)$ con capacidad c y flujo-(s, t) factible, donde E_f es el conjunto de arcos cuya capacidad residual c_f es positiva.



Flujo f en la red de flujo G (izda.) y su grafo residual G_f (dcha.)

Fuente: J. Erickson, *Algorithms*, cap. 10.

Incremento del flujo en un camino de s a t

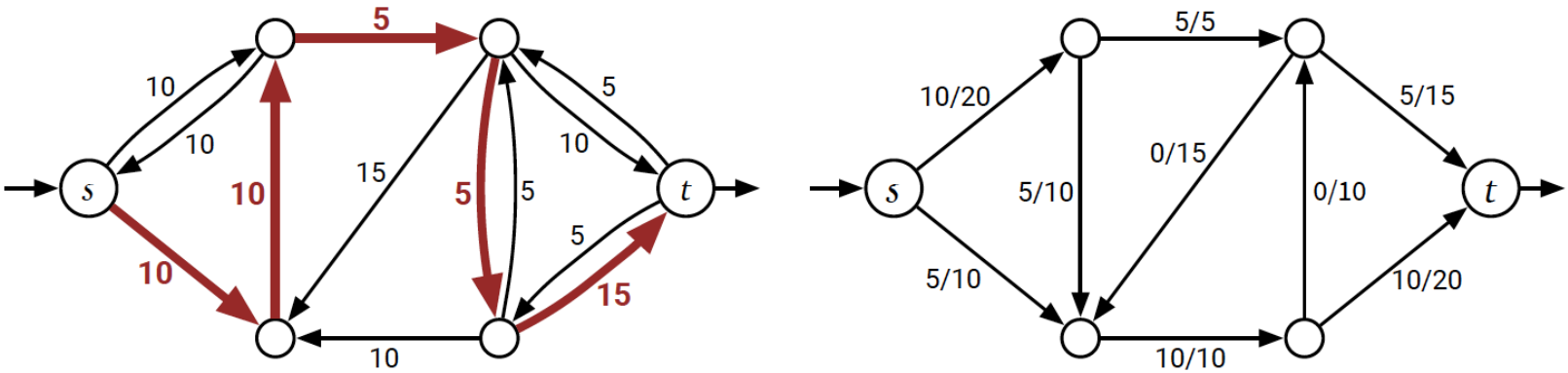
- Sea P un camino de s a t en el **grafo residual** G_f, la máxima cantidad de flujo que se puede introducir en P es:

$$F = min_{u \to v \in P}\, c_f(u \to v)$$

- Se define un nuevo flujo $f' : E \to \mathbb{R}$ en el **grafo original** G:

$$f'(u \to v) = \begin{cases} f(u \to v) + F & \text{si } u \to v \in P \\ f(u \to v) - F & \text{si } v \to u \in P \\ 0 & \text{de otra manera} \end{cases}$$

Incremento del flujo en un camino de s a t



Camino P con $F = 5$ en G_f (izda.) y el nuevo flujo f' en G (dcha.)

Fuente: J. Erickson, *Algorithms*, cap. 10.

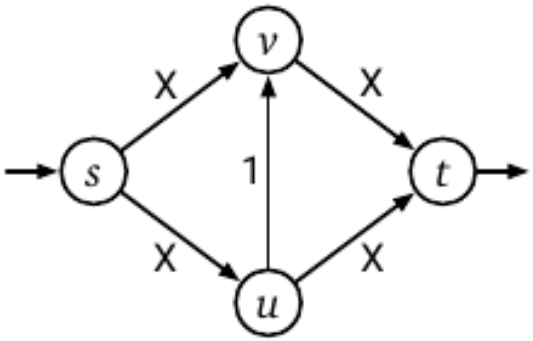
- El nuevo flujo f' es **factible** con respecto a las capacidades originales c de G
- $|f'| = |f| + F \geq |f| \Rightarrow f$ no es flujo máximo

Terminación del algoritmo

- ¿Qué pasa si el grafo residual G_f no incluye un camino P de s a t?
- Sea S el conjunto de nodos de G_f que se pueden alcanzar de s y $T = V \setminus S$
  - Se puede demostrar que el flujo f satura cada arco de S a T y evita cada arco de T a S
- Por el lemma (transpa. 27): $|f| = ||S, T||$

Eficiencia

- En cada iteración se puede utilizar una búsqueda en profundidad o amplitud para encontrar un camino de s a t
  - Complejidad en tiempo: $O(|E|)$
- ¿Cuántas iteraciones hay?
  - Asumimos todos los arcos con capacidad entera $\mathbb{N}$
  - El algoritmo de Ford-Fulkerson termina después como mucho $|f^*|$ iteraciones, donde $|f^*|$ es el flujo máximo
  - Complejidad en tiempo (caso peor): $O(|E||f^*|)$

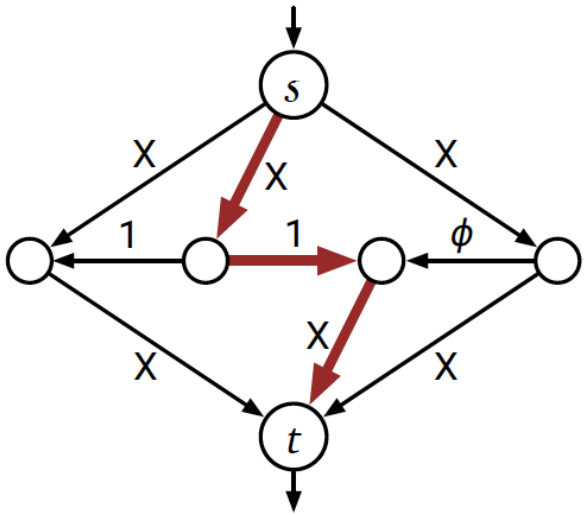


Fuente: J. Erickson, *Algorithms*, cap. 10.

- Exponencial en el tamaño del input (número de bits necesario para describir las capacidades en entrada)
- ¿Qué compleijdad tiene el algoritmo en el caso de capacidades racionales $\mathbb{Q}$?
  - Igual que en el caso entero

Sobre la terminación y convergencia al valor max

- ¿Qué pasa en el caso de capacidades irracionales $\mathbb{R} \setminus \mathbb{Q}$?
- ¡No está garantizada ni la terminación ni la convergencia al flujo máximo!

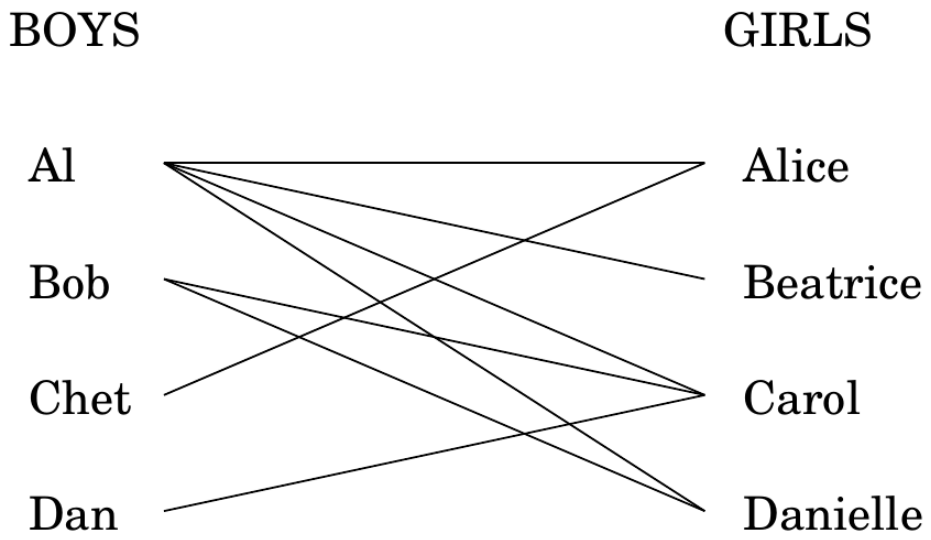


Ejemplo de Uri Zwick

Fuente: J. Erickson, *Algorithms*, cap. 10.

- ¿Es un problema únicamente teórico?
- ¡No! **En práctica**: una mala implementación del algoritmo puede provocar los mismos problemas con capacidades no enteras debido a errores de redondeo

El problema de emparejamiento bipartido



Fuente: S. Dasgupta, et al. *Algorithms*

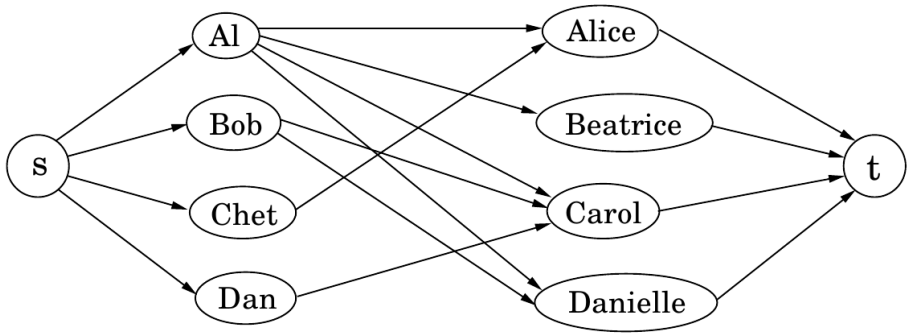
Objetivo del emparejamiento perfecto

¿Es posible elegir las parejas para que todos estén contentos?

- El problema se puede reducir en un problema de máximo flujo . . .

Reducción en un problema de máximo flujo

1. Crear un nuevo nodo origen s y conectarlo a todos los chicos
2. Crear un nuevo nodo destino t y conectar todas las chicas a t
3. Asociar capacidad 1 a cada arco



Fuente: S. Dasgupta, et al. *Algorithms*

Hay un emparejamiento perfecto $\Leftrightarrow$ la red de flujo tiene un flujo con valor igual al número de parejas

Dualidad

Problema de la fábrica de cerveza

$$
\begin{aligned}
max\ & 12A + 20B \\
t.q.\ & 4A + 12B \leq 384 \\
& 0{,}1A + 0{,}1B \leq 4 \\
& 14A + 8B \leq 476 \\
& A, B \geq 0
\end{aligned}
$$

- Solución óptima: $(A = 12, B = 28)$
- Valor max: 704

- Multiplicamos la primera, segunda y tercera restricción por $1, 80$ y 0 respectivamente ...
- Obtenemos: $12A + 20B \leq 704$ 

Dualidad

- ¿De dónde vienen los multiplicadores $(1, 80, 0)$?

Mult.	*Desigualdades*
y_1	$4A + 12B \leq 384$
y_2	$0{,}1A + 0{,}1B \leq 4$
y_3	$14A + 8B \leq 476$

- Sean $y_i \geq 0$ (preservamos las desigualdades $\leq$)
- Después haber multiplicado y sumado ...

$$(4y_1 + 0{,}1y_2 + 14y_3)A + (12y_1 + 0{,}1y2 + 8y_3)B \leq 384y_1 + 4y_2 + 476y_3$$

- Queremos que la parte izquierda de la desigualdad sea parecida a la función objetivo del problema inicial ... $12A + 20B$
- Es decir: $(4y_1 + 0{,}1y_2 + 14y_3) \geq 12$ y $(12y_1 + 0{,}1y_2 + 8y_3) \geq 20$

Dualidad

- Cota superior de la función objetivo:

$$12A + 20B \leq 384y_1 + 4y_2 + 476y_3$$

- Si:

$$y_1, y_2, y_3 \geq 0$$
$$4y_1 + 0{,}1y_2 + 14y_3 \geq 12$$
$$12y_1 + 0{,}1y_2 + 8y_3 \geq 20$$

- Para obtener una buena cota, tenemos que minimizarla
- Se obtiene un nuevo problema!

Dualidad

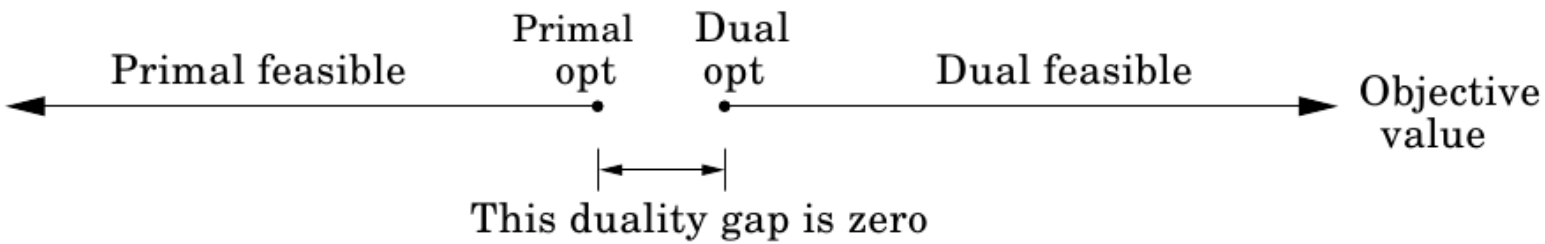
Relación entre soluciones primal-dual

Problema dual

$$
\begin{aligned}
min\ & 384y_1 + 4y_2 + 476y_3 \\
t.q.\ & 4y_1 + 0{,}1y_2 + 14y_3 \geq 12 \\
& 12y_1 + 0{,}1y_2 + 8y_3 \geq 20 \\
& y_1, y_2, y_3 \geq 0
\end{aligned}
$$

- Cualquier solución factible del dual es una cota superior del problema primal
- Las soluciones factibles
  - Primal: $(A, B) = (12, 28)$, sol.: 704
  - Dual: $(y_1, y_2, y_3) = (1, 80, 0)$, sol.: 704

son soluciones óptimas.



Fuente: S. Dasgupta, et al. *Algorithms*

Dualidad

Primal en forma canónica

- Para cada problema de programación lineal, existe un problema **dual** que se obtiene aplicando un conjunto de reglas.
- Si el primal se encuentra en forma canónica, la transformación es más simple . . .

Problema *primal*

$$\begin{aligned} max\ & \mathbf{c^T x} \\ t.q.\ & \mathbf{Ax} \leq \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

$\Longleftrightarrow$

Problema *dual*

$$\begin{aligned} min\ & \mathbf{y^T b} \\ t.q.\ & \mathbf{A^T y} \geq \mathbf{c} \\ & \mathbf{y} \geq \mathbf{0} \end{aligned}$$

Dualidad

Primal y dual en forma canónica

- Podemos escribir el problema dual también en forma canónica como el primal
- La dualidad es una involución: el dual del dual es el problema primal!

Problema *primal*

$$\begin{aligned} max\ & \mathbf{c}^\mathbf{T}\mathbf{x} \\ t.q.\ & \mathbf{A}\mathbf{x} \leq \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

$\Longleftrightarrow$

Problema *dual*

$$\begin{aligned} max\ & -\mathbf{b}^\mathbf{T}\mathbf{y} \\ t.q.\ & -\mathbf{A}^\mathbf{T}\mathbf{y} \leq -\mathbf{c} \\ & \mathbf{y} \geq \mathbf{0} \end{aligned}$$

Dualidad

Teorema de dualidad

Problema *primal*

$$\begin{aligned} max\ & \mathbf{c^T x} \\ t.q.\ & \mathbf{Ax} \leq \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

$\Longleftrightarrow$

Problema *dual*

$$\begin{aligned} min\ & \mathbf{y^T b} \\ t.q.\ & \mathbf{A^T y} \geq \mathbf{c} \\ & \mathbf{y} \geq \mathbf{0} \end{aligned}$$

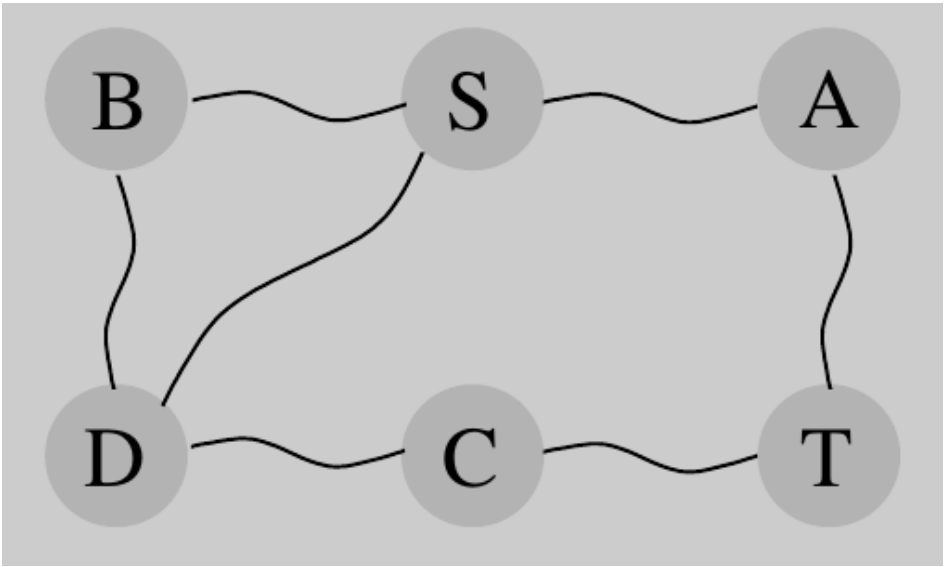
El problema primal tiene una solución óptima x^* si y solo si el problema dual tiene una solución óptima y^* tal que:

$$c \cdot x^* = {y^*}^T A x^* = {y^*}^T \cdot b$$

Dualidad

Visualización del dual

- Problema primal: problema del camino mínimo entre dos nodos S y T
- Problema dual: es un problema de máximo
- ¿Cómo se interpreta?

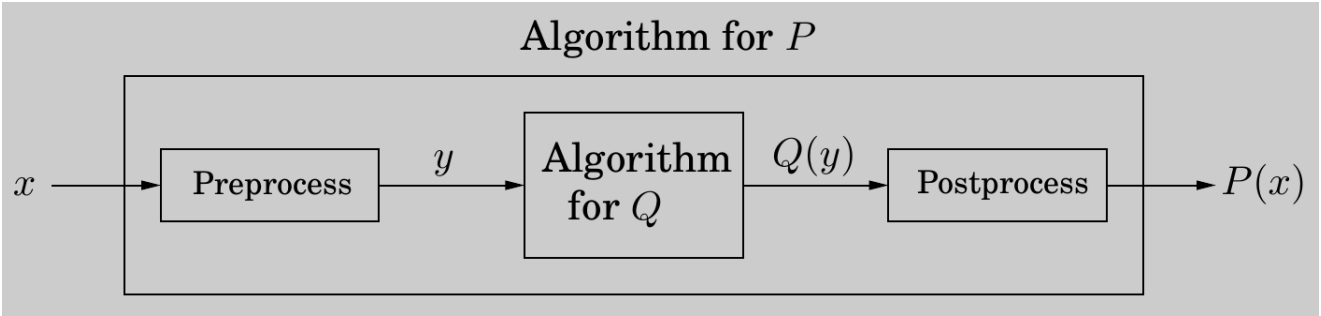


- Las cuerdas que unen parejas de nodos tiene una longitud dada
- Problema dual: *estirar* los nodos S y T cuanto más posible, sin romper las cuerdas que unen las parejas de nodos

Fuente: S. Dasgupta, et al. *Algorithms*

Transformación de un problema en otro

- P es **reducible** a Q si se puede utilizar un algoritmo eficiente para resolver Q como sub-rutina de un algoritmo eficiente para resolver P



Fuente: S. Dasgupta et al. *Algorithms*

```
1  {P: encontrar el camino más largo en un DAG;
2   Q: encontrar el camino más corto en el DAG}
3  funcion caminoMasLargo(G: grafo) devuelve camino
4  principio
5    {Pre-procesado: negar todos los pesos de los arcos en G}
6    devuelve caminoMasCorto(G)
7  fin
```

Reducciones en problemas de optimización

- Las reducciones se aplican a menudo a problemas de optimización
- Un problema LP general tiene muchos grados de libertad:
  - Puede ser un problema de *max* o de *min*
  - Puede tener restricciones de igualdad o de desigualdad
  - Las variables a menudo tienen que ser no negativas

Todas las variantes pueden reducirse entre sí

Variantes de problemas LP

- Reducción de un problema *max* a un problema *min* (o viceversa)
  - Multiplicar la función objetivo por "-1"

Ejemplo: problema de la mochila 0-1

$$max \quad \sum_{i=1}^{n} b_i x_i \qquad min \quad -\sum_{i=1}^{n} b_i x_i$$

$$s.t. \quad \sum_{i=1}^{n} p_i x_i \leq C \qquad s.t. \quad \sum_{i=1}^{n} p_i x_i \leq C$$

$$x_i \in \{0,1\}, b_i, p_i > 0, 1 \leq i \leq n \qquad x_i \in \{0,1\}, b_i, p_i > 0, 1 \leq i \leq n$$

- Aplicación de la técnica de ramificación y poda al problema *min*

Variantes de problemas LP

- Reducción de una restricción de desigualdad a una de igualdad
  - Añadir una nueva variable *slack* s

$$\sum_{i=1}^{n} a_i x_i \leq b_i \quad \Rightarrow \quad \sum_{i=1}^{n} a_i x_i + s = b_i$$
$$s \geq 0$$

- Reducción de una restricción de igualdad a restricciones de desigualdad

$$\sum_{i=1}^{n} a_i x_i = b_i \quad \Rightarrow \quad \sum_{i=1}^{n} a_i x_i \leq b_i$$
$$\sum_{i=1}^{n} a_i x_i \geq b_i$$

Variantes de problemas LP

- Reducción de una variable x sin restricciones de signo a dos variables x^+, x^- no negativas
  - Añadir las dos variables $x_1, x_2 \geq 0$
  - Sustituir x por $x_1 - x_2$ (en las restricciones y función objetivo)
- Reducción de un problema primal a un problema dual

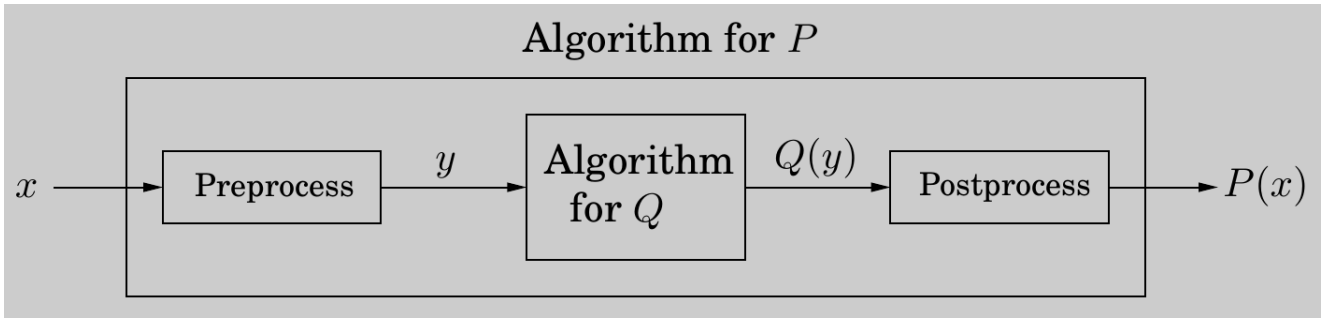
Primal	Dual	Primal	Dual
$\max \mathbf{c^T x}$	$\min \mathbf{y^T b}$		
$\sum_j a_{ij} x_j \leq b_i$	$y_i \geq 0$	$x_j \geq 0$	$\sum_i a_{ij} y_i \geq c_j$
$\sum_j a_{ij} x_j \geq b_i$	$y_i \leq 0$	$x_j \leq 0$	$\sum_i a_{ij} y_i \leq c_j$
$\sum_j a_{ij} x_j = b_i$	$y_i \gtreqless 0$	$x_j \gtreqless 0$	$\sum_i a_{ij} y_i = c_j$

Ejemplo

Problema de máximo flujo (primal) / problema de mínimo corte (dual)

Reducciones en problemas de optimización

- P: problema de programación lineal entera (ILP)
- Q: problema de programación lineal *relajado*



Fuente: S. Dasgupta et al. *Algorithms*

- Algoritmo para P: técnica de ramificación y poda
- Algoritmo para Q: algoritmo símplex

Problemas ILP

Ejemplo

Máquina	Espacio	Precio	Beneficio (por unidad y día)
prensa	$15m^2$	8.000€	100€
torno	$30m^2$	4.000€	150€

Espacio disponible: $200m^2$, Presupuesto: 40.000€

$$
\begin{aligned}
P: \; & max \; 100x_1 + 150x_2 \\
t.q. \quad & 15x_1 + 30x_2 \leq 200 \\
& 8000x_1 + 4000x_2 \leq 40000 \\
& x_1, x_2 \geq 0 \quad \textbf{enteros}
\end{aligned}
$$

- x_1: número de prensas
- x_2: número de tornos

Reducción del problema a un problema LP

$$
\begin{aligned}
max\ & 100x_1 + 150x_2 \\
t.q.\ & 15x_1 + 30x_2 \leq 200 \\
& 8000x_1 + 4000x_2 \leq 40000 \\
& x_1, x_2 \geq 0 \quad \textbf{enteros}
\end{aligned}
$$

- x_1: número de prensas
- x_2: número de tornos

- Q: problema LP obtenido de P quitando las restricciones de integridad, (i.e., $x_1, x_2 \in \mathbb{R}_0^+$)

UB=1.055,56 {x1=2,22;x2=5,56}
LB = 950 {x1=2;x2=5}

- Cota superior: valor de la solución óptima del problema Q
- Cota inferior: valor de la solución óptima **redondeada** por defecto (es solución factible de P)

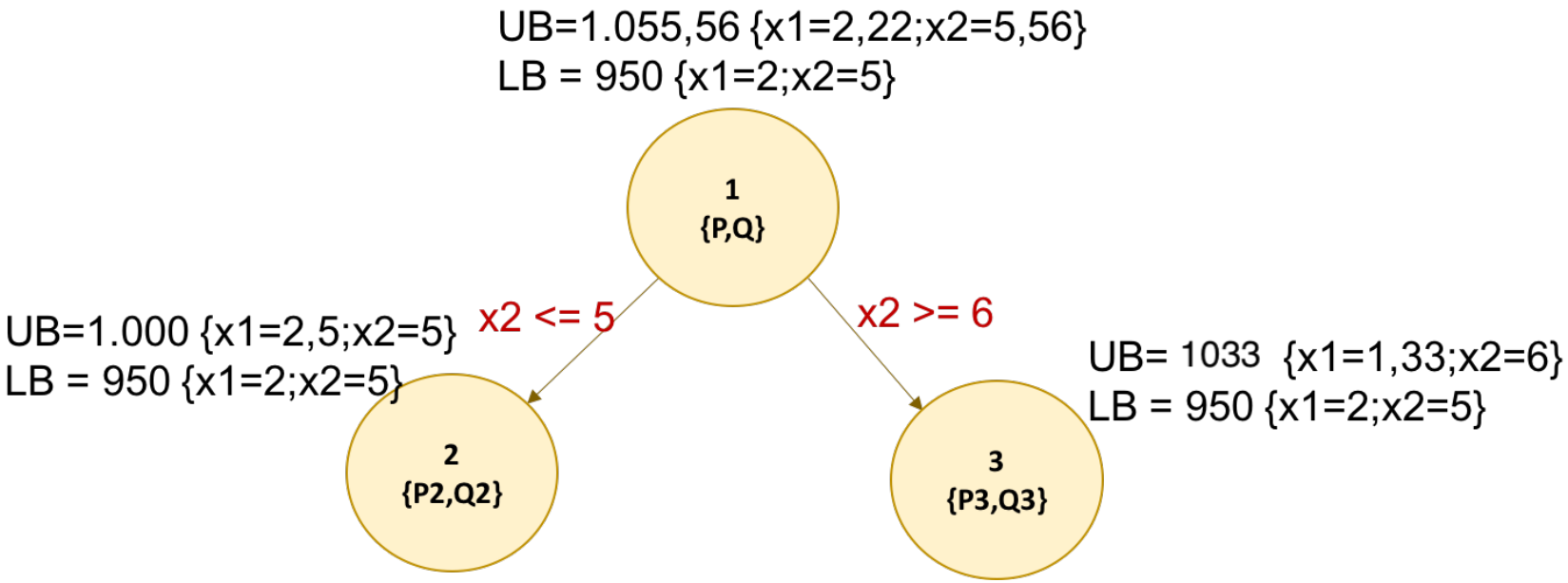
Técnica de ramificación y poda

Criterio de ramificación

- Dado el nodo X del árbol y la solución óptima asociada al problema Q_X, elegir la variable con la mayor parte decimal
  - $x_1 = 2,22$ y $x_2 = 5,56$
- Crear dos nuevas restricciones que definen una partición en el conjunto de soluciones de Q
  - Problema P_2 : $P+$ la restricción $x_2 \leq 5$ asociado al hijo izquierdo de X
  - Problema P_3 : $P+$ la restricción $x_2 \geq 6$ asociado al hijo derecho de X
- Resolver los problemas *relajados* correspondientes: Q_2, Q_3

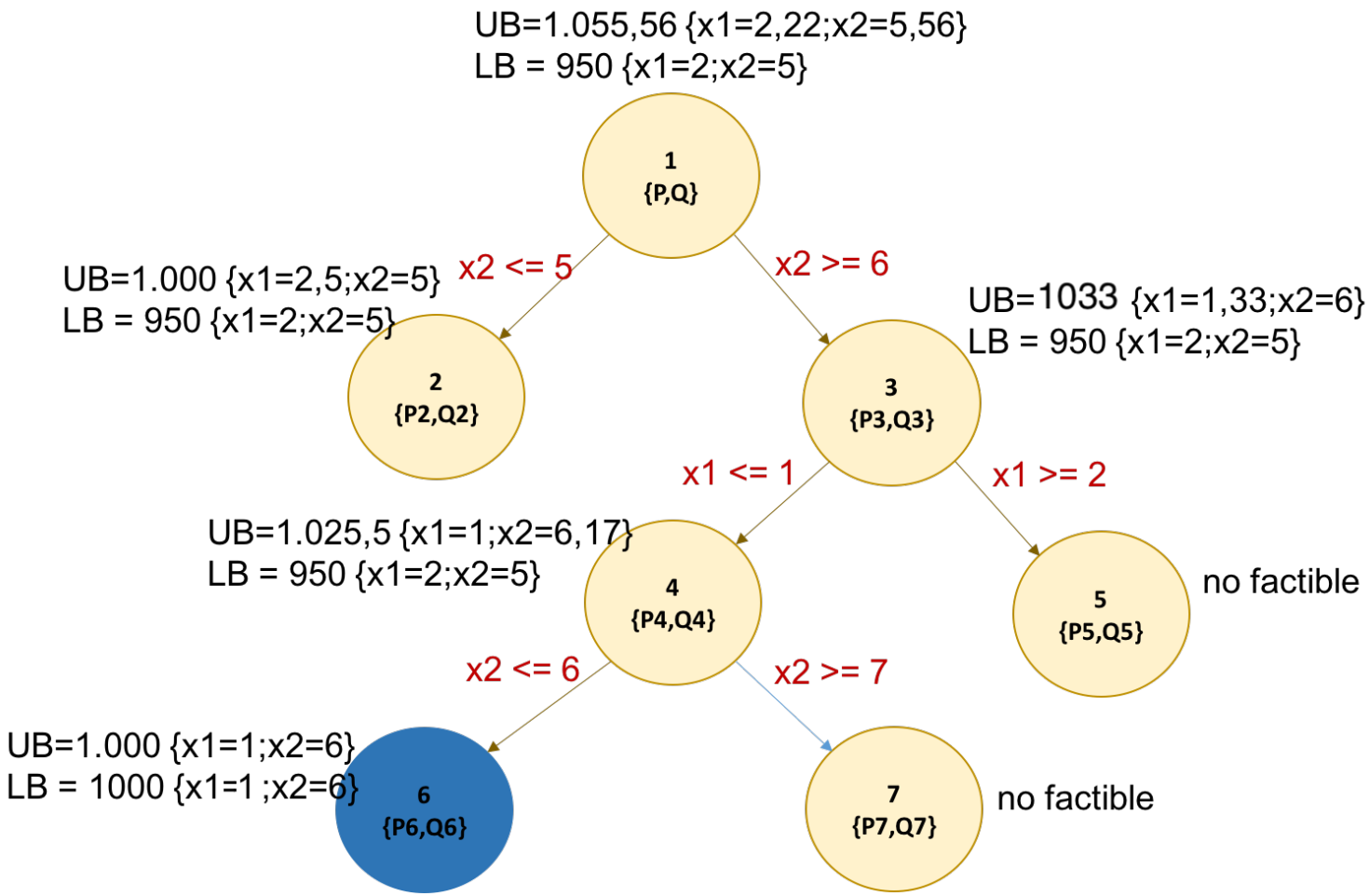
Técnica de ramificación y poda

- Los valores de las soluciones óptimas de Q_2 y Q_3 son las cotas superiores asociadas a los nodos (2) y (3)



- No hay soluciones enteras óptimas
- ¿Cuál es el siguiente nodo en expansión?
  - El nodo que tiene el valor más alto como cota superior

Técnica de ramificación y poda



Referencias

- T. H. Cormen et al. *Introduction to Algorithms* (3rd edition), the MIT Press, 2009 – Capítulo 29
- S. Dasgupta et al., *Algorithms*, McGraw-Hill, 2008 – Capítulo 7
- J. Erickson, *Algorithms*, libro en línea, 2019: http://jeffe.cs.illinois.edu/teaching/algorithms/ – Capítulos 10, H
- B. Taylor, *Introduction to Management Science*, mod. C: "Integer programming the branch&bound method"(en línea)
- Transparencias de R. Sedgewick, K. Wayne (Princeton University): https://algs4.cs.princeton.edu/lectures/keynote/99LinearProgramming-2x2.pdf