

ESAME: 3(4) sezioni; max 33: 31-33 vuol dire lode

- teoria: 10p; obbligatoria, almeno 6 punti per passare -> 3 parti (domande): definizione (qualche riga), dimostrazione (5/6 in tutto il corso, da una pagina), modello
  - applicare algoritmi (main: algoritmo del simplex) (da numeri forniti)
  - problema di modellazione
- Niente libro/appunti all'esame

Altra dispensa opzionale: Agnelis (uni Siena), suo sito ha dispense in italiano

SCOPO: cercare vie efficienti per migliorare operazioni (es. manifattura) -> applicare certi strumenti

Corso in modo specifico su mathematical programming (non coding, più simile a planning) -> termine migliore: optimization

Problema reale (descrivibile a parole) da risolvere con ottimizzazione -> si fa modello matematico, convertire carat. importanti di problema in equazioni -> problema di ottimizzazione su modello matematico

Esempio: turni in ospedale (p. 11) -> coprire tutti i turni

Tabella con giorni settimana, per ogni giorno: num. minimo di infermieri

Ogni infermiere lavora 5 giorni di fila, poi 2 di pausa

Si potrebbero provare tutte le combinazioni

Output: num. infermieri che inizia nei vari giorni -> DECISION VARIABLES (incognite)

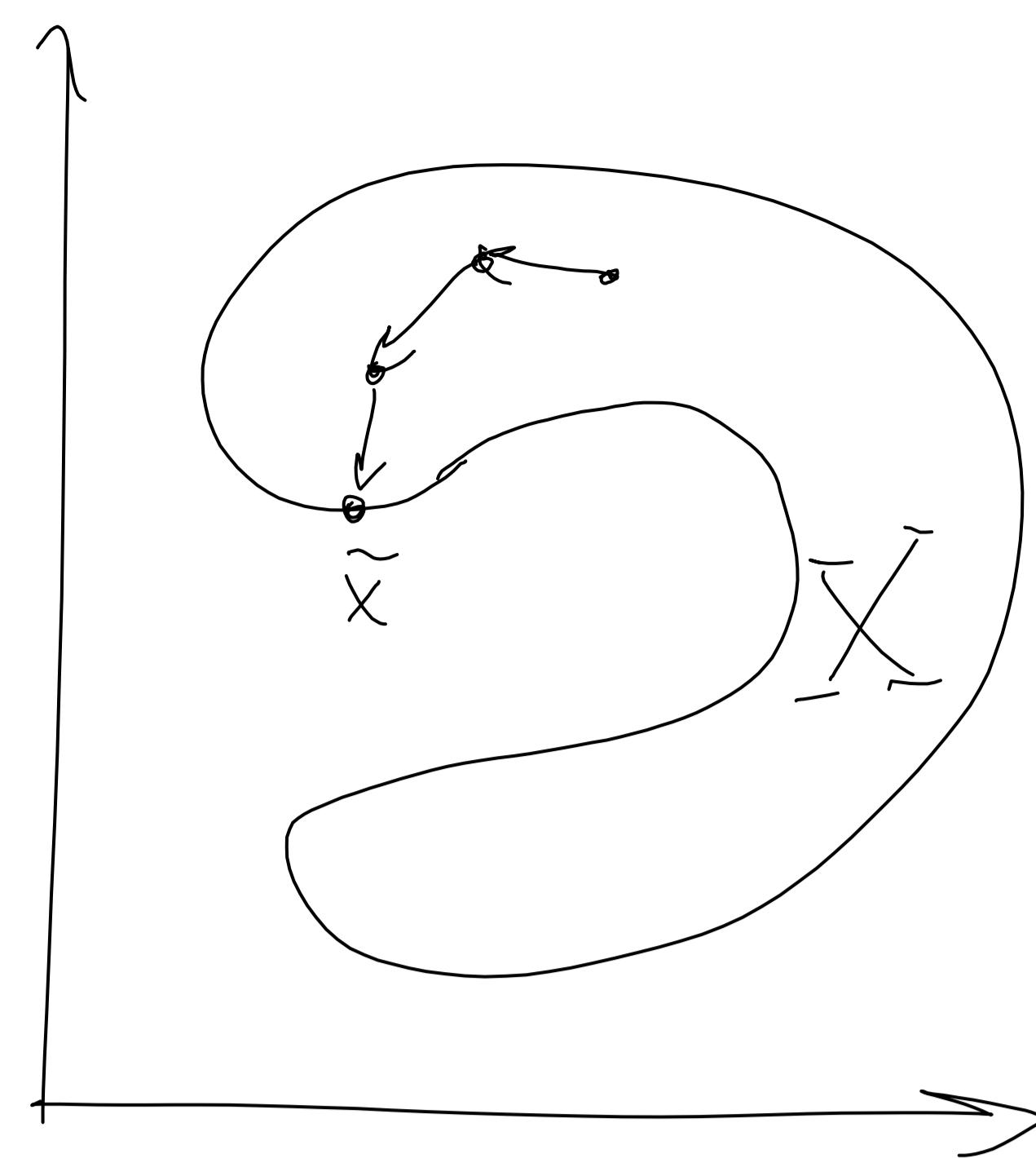
Scopo: minimizzare n. infermieri -> OBJECTIVE FUNCTION

Poi servono CONSTRAINTS (vincoli) -> es. # infermieri in MON:  $x_1+x_4+x_5+x_6+x_7 \geq 17$

Specificare che variabili sono naturali

Ora alcune combinazioni di valori variabili non sono accettabili -> rappresentabili in spazio -> si crea forma

Strumento per risolvere LP: IBM CPLEX



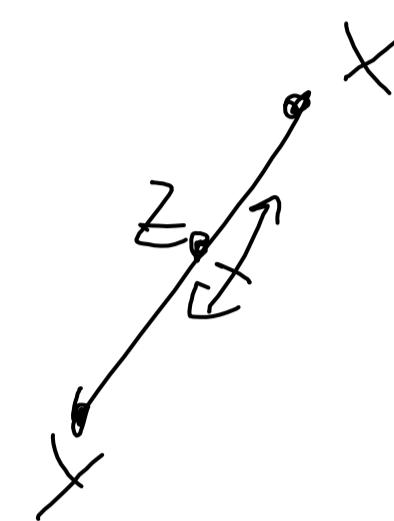
$\tilde{x}$  può essere locale, e si può convergere lì

regularità in  $X$ : CONVESSITÀ → allora locale è  
regularità in  $f(x)$ : CONVESSITÀ anche globale

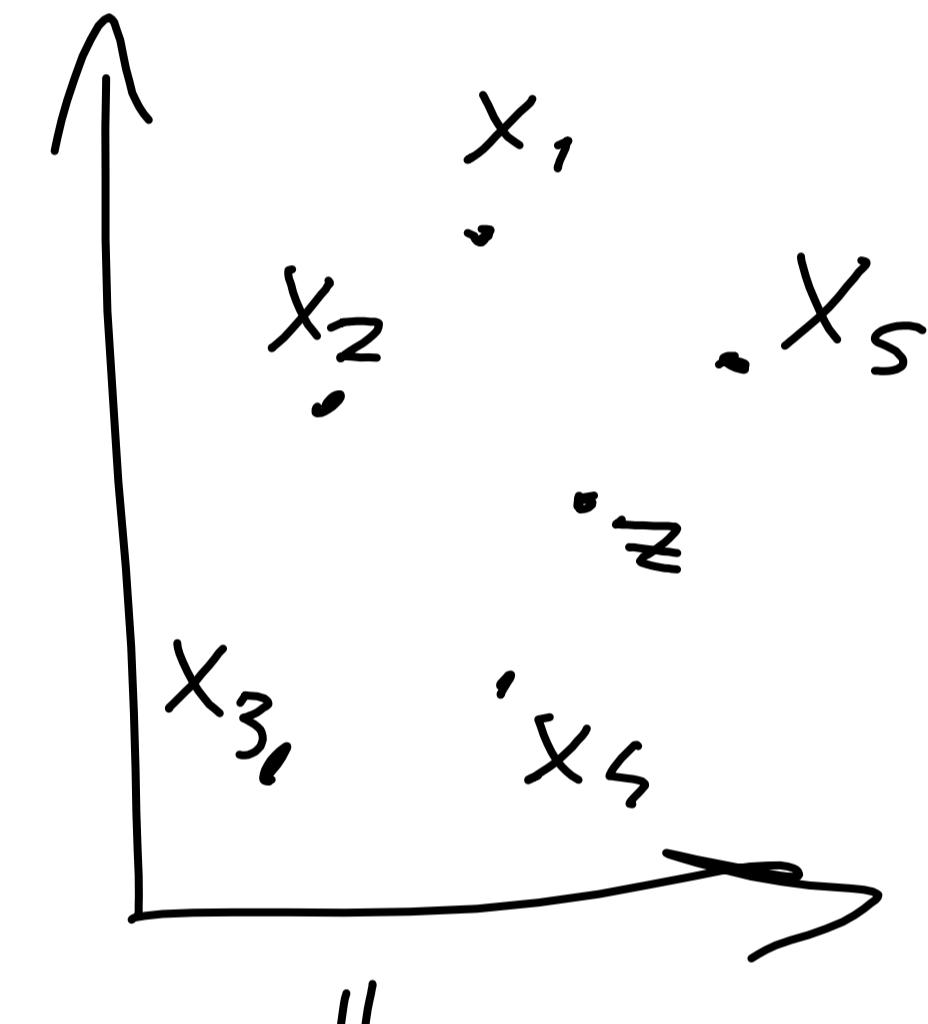
Def:  $x, y \in \mathbb{R}^n$   
 $z = \lambda x + (1-\lambda)y, \lambda \in [0, 1]$

CONVEX COMBINATION:

- STRICT se  $\lambda \neq 0, \lambda \neq 1$



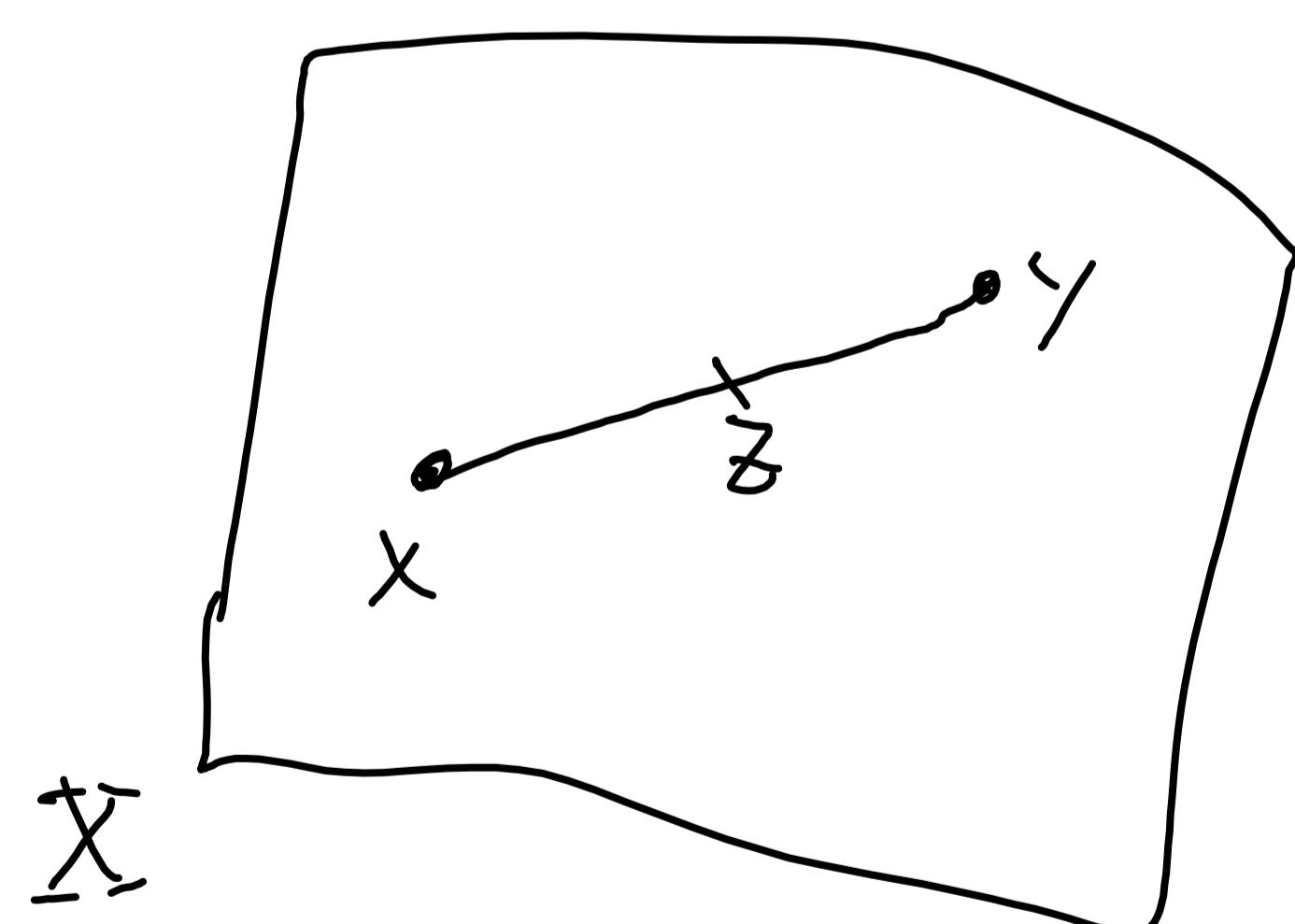
Def:  $x^1, x^2, \dots, x^k \in \mathbb{R}^n$   
 $z = \sum \lambda_i x^i, \lambda_1, \dots, \lambda_k \geq 0 | \sum \lambda_i = 1$



Le comb. panno solido: CONVEX HULL

CONVEX SET:

$X \subseteq \mathbb{R}^n$  è convesso



$z = \lambda x + (1-\lambda)y$  sempre  $\in X$   
 (non serve solo strict)

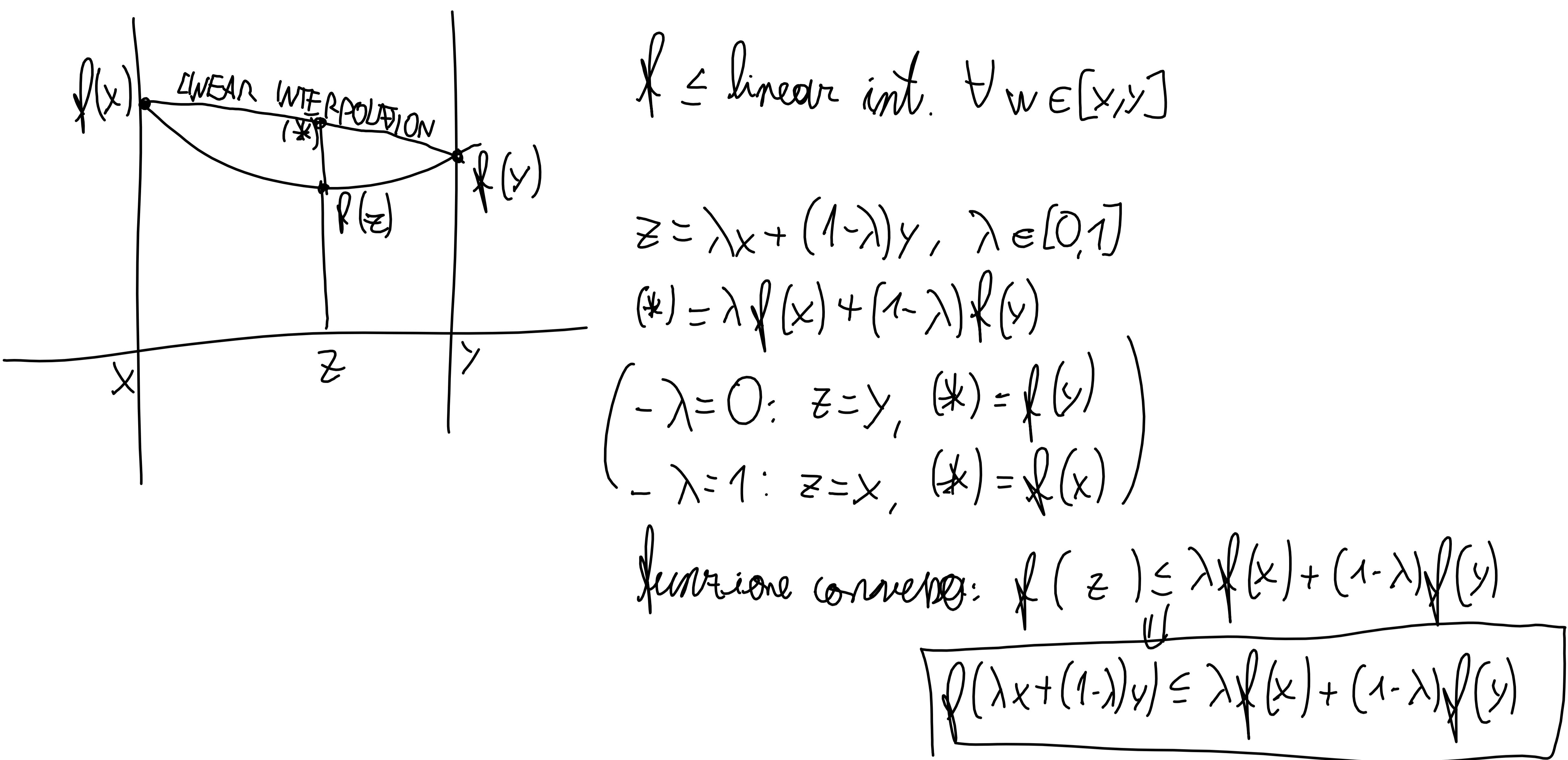
commentari per funzioni

$f: X \rightarrow \mathbb{R}, X$  convesso

$f$  è CONVEXA



F va giù andando avanti, quando inizia a salire non cambierà più



come dimostrare convessità insieme?

Serve  $\bar{X}$  definito come

$$\bar{X} = \{x \in \mathbb{R}^n \mid g_i(x) \leq 0 \forall i \in [1, m]\} \quad (\text{soddisfa insieme di diseguaglianze})$$

THEOREM 1.1.1.

se  $g_1, \dots, g_m$  convessa, allora  $\bar{X}$  convesso

$$\text{Dim: } \bar{X} = \bigcap_{i=1}^m \bar{X}_i, \bar{X}_i = \{x \in \mathbb{R}^n \mid g_i(x) \leq 0\}$$

serve dimostrare che ogni  $\bar{X}_i$  convesso

dati qualsiasi  $x, y \in \bar{X}_i \Rightarrow z = \lambda x + (1-\lambda)y, \lambda \in [0,1]$

$$g_i(z) = g_i(\lambda x + (1-\lambda)y) \leq \lambda g_i(x) + (1-\lambda)g_i(y)$$

$$\begin{aligned} \lambda &\geq 0, g_i(x) \leq 0 \quad (\text{perché } x \in \bar{X}_i) \\ (1-\lambda) &\geq 0, g_i(y) \leq 0 \end{aligned} \Rightarrow \text{somma negativa}$$

$$\begin{gathered} \downarrow \\ g_i(\lambda x + (1-\lambda)y) \leq 0 \\ \downarrow \\ z \in \bar{X}_i \end{gathered}$$

# CONVEX OPTIMIZATION PROBLEM

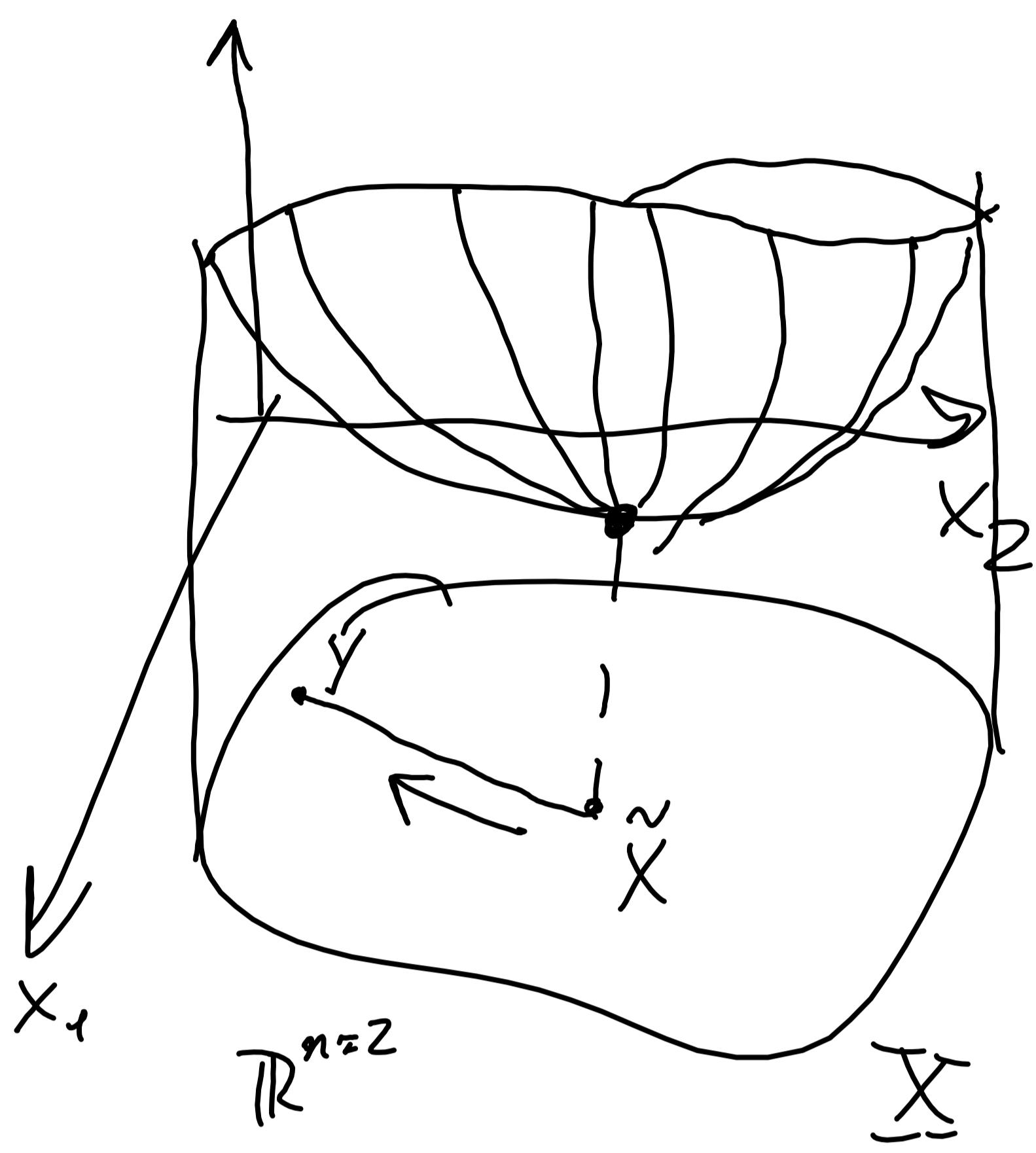
$$\min f(x)$$

$x \in X$

~~f, X~~ conversi

TEOREMA: ogni soluzione ottima locale è anche globale

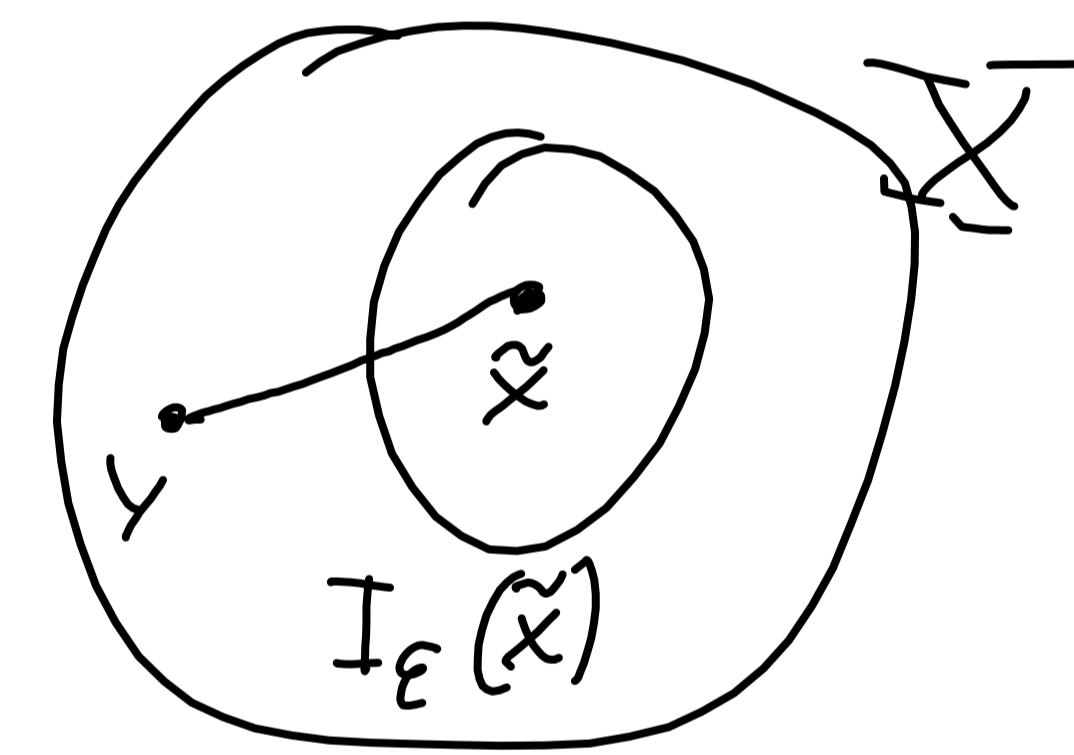
*Dim:*



(ver completa)

$\tilde{x}$  sia ottimo locale  $\Rightarrow \exists \varepsilon > 0 \mid f(\tilde{x}) \leq f(z) \quad \forall z \in X \mid \|z - \tilde{x}\| \leq \varepsilon$ ,

$I_E(x)$



que el vector  $y \in \underline{X} \Rightarrow z = \lambda \underline{x} + (1-\lambda) y$

per  $\lambda \lesssim 1 \Rightarrow z \in I_\varepsilon(\tilde{x})$

$$\Rightarrow (1-\lambda) f(\tilde{x}) \leq (1-\lambda) f(y) \Rightarrow f(\tilde{x}) \leq f(y) \quad \forall y \Rightarrow \tilde{x} \text{ global}$$

$\uparrow$   
 $\lambda < 1$

## LINEAR PROGRAMMING/OPT.

$$\begin{cases} \min f(x) \rightarrow f \text{ lineare: } f(x) = \sum_j c_j x_j \\ g_i(x) \leq 0, g_i \text{ convessa} \rightarrow b_i - \sum a_{ij} x_j \leq 0 \quad \forall i \end{cases}$$

## MATRIX NOTATION

$$\sum_i c_i x_i = \langle c, x \rangle = c^T x$$

$(c, x \in \mathbb{R}^n)$

$$x = \begin{bmatrix} x_1 \\ \vdots \\ x_m \end{bmatrix} \quad c = \begin{bmatrix} c_1 \\ \vdots \\ c_n \end{bmatrix}$$

vettore  
colonna

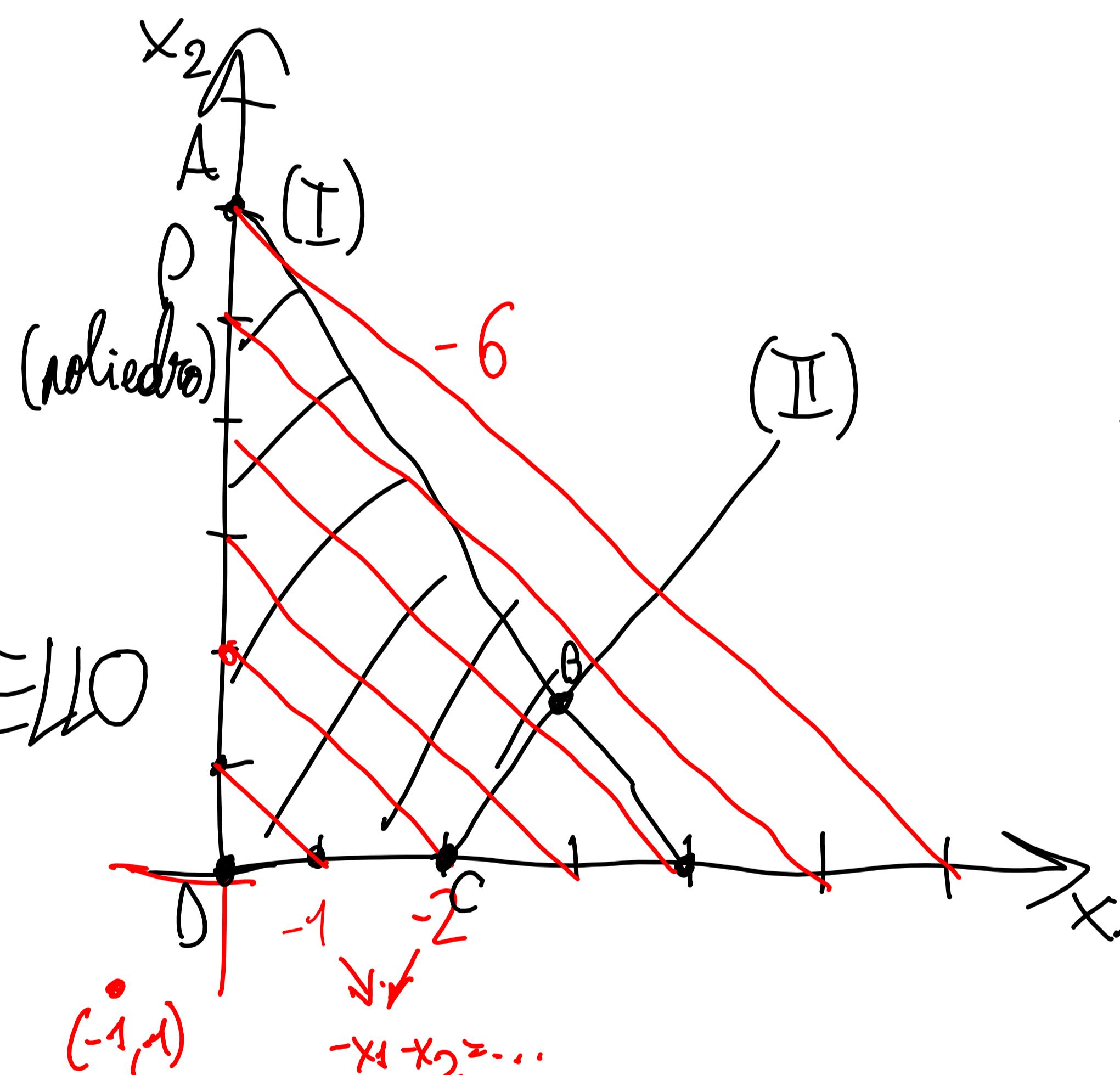
$$a_i^T x \geq b_i \quad i=1, \dots, m$$

$$Ax \geq B$$

Ese:

$$\begin{cases} \min -x_1 - x_2 \\ 6x_1 + 4x_2 \leq 24 \quad (\text{I}) \\ 3x_1 - 2x_2 \leq 6 \quad (\text{II}) \\ x_1, x_2 \geq 0 \end{cases}$$

per risolvere, LINEE DI LIVELLO  
meglio disegnare gradiente  
funzione  $\Rightarrow (-1, -1) \Rightarrow$   
 $\Rightarrow +$  lunghezza del grad.,  
valori migliori



$-x_1 - x_2 = \text{parametro} \Rightarrow$   
 $\Rightarrow$  cambia posz., linea si sposta

$-x_1 - x_2 = -6 \Rightarrow$  aumenta posz., si esce da P  $\Rightarrow$   
 $\Rightarrow$  abbiamo sol. ottima

stessa sol. A per molte linee, ma per alcune è B oppure C oppure D  $\Rightarrow$   
 $\Rightarrow$  ci si può ridurre a pochi punti: VERTICI di P  $\Rightarrow$  iniziazione: sol. ottima  
è sempre vertice (non necessariamente)

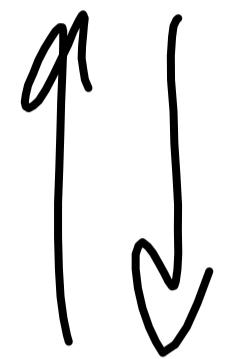
Possono esserci tanti vertici (es. in 3D)

Approccio migliore: dal punto possibile, si arriva al vertice, poi ad altri vertici, fino a raggiungere ottimo  $\Rightarrow$  possibili lunghi percorsi  $\Rightarrow$  SIMPLEX METHOD

Funziona in convex programming  $\Rightarrow$  ottimo locale = ottimo globale

$$\begin{cases} \min c^T x \\ Ax \geq b \\ x \geq 0 \end{cases}$$

CANONICAL FORM: solo diseguaglianze in senso



$$\begin{cases} \min c^T x \\ Ax = b \\ x \geq 0 \end{cases}$$

STANDARD FORM: solo uguaglianze  $\Rightarrow$  usata per simplex

Metodo per passare tra forme:

$$a_i^T x \geq b_i \rightsquigarrow \begin{cases} a_i^T x - s_i = b_i \\ s_i \geq 0 \end{cases} \quad \text{SURPLUS variable}$$

$$a_i^T x \leq b_i \rightsquigarrow \begin{cases} a_i^T x + s_i = b_i \\ s_i \geq 0 \end{cases} \quad \text{SLACK variable}$$

$$a_i^T x = b_i \rightsquigarrow \begin{cases} a_i^T x \leq b_i \\ a_i^T x \geq b_i \end{cases}$$

Se abbiamo  $x_i \geq 0$  (" $x_i$  free")  $\Rightarrow$

$$\begin{cases} x_i = x_i^+ - x_i^- \\ x_i^+ \geq 0 \\ x_i^- \leq 0 \end{cases}$$

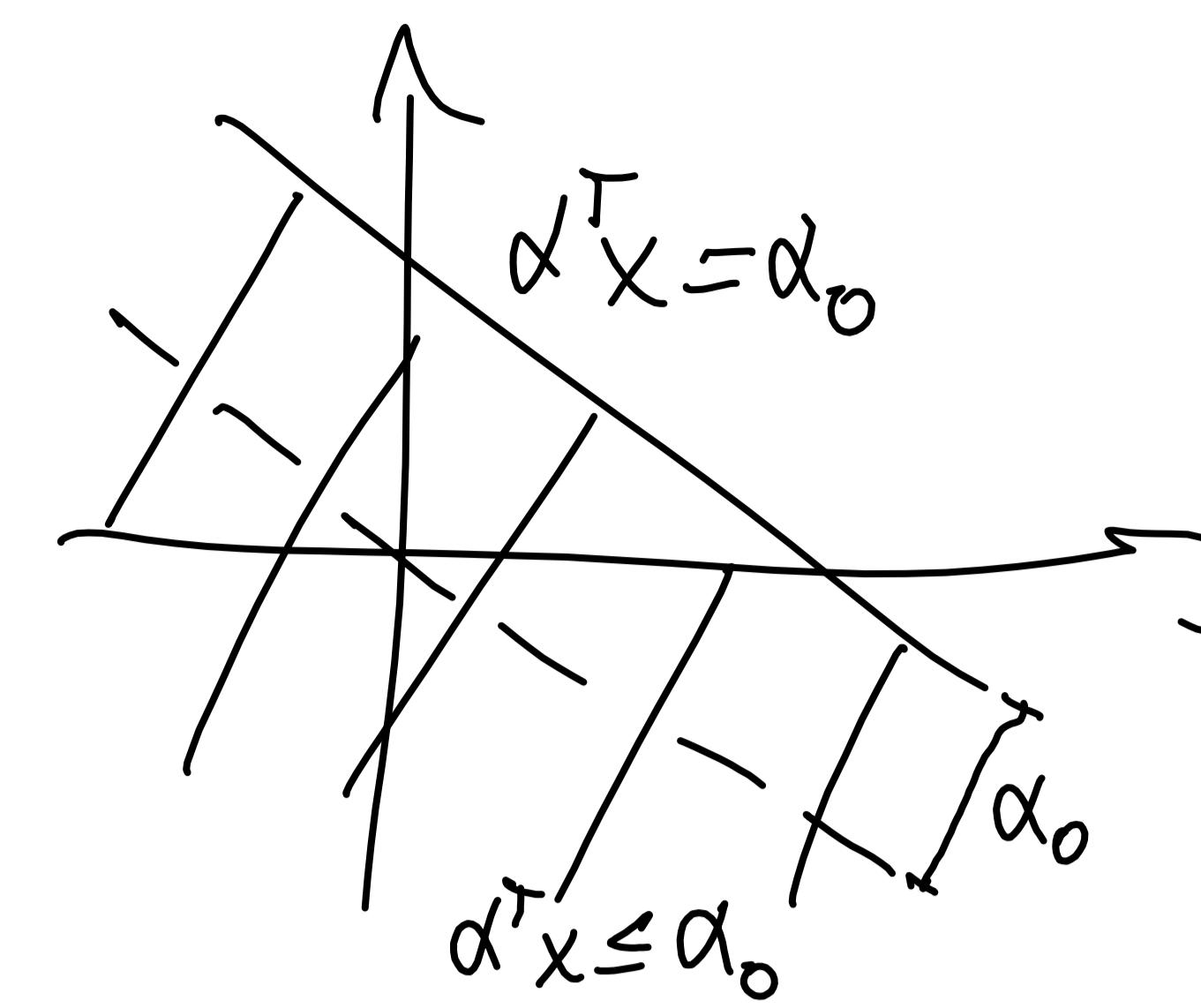
Vincolo  $a_i^T x \leq b_i \Rightarrow$  non ci è modo sicuro per convertire  $\Rightarrow$  si approssima:  $\leq 2.999..$

9/10

# SIMPLEX METHOD

Def:  $\{x \in \mathbb{R}^n \mid \underline{\alpha}^T x \leq \underline{\alpha}_0\}$ : AFFINE HALF-SPACE  
 ↘  
 aff.

$\Downarrow$   
 deve passare per  
 origine

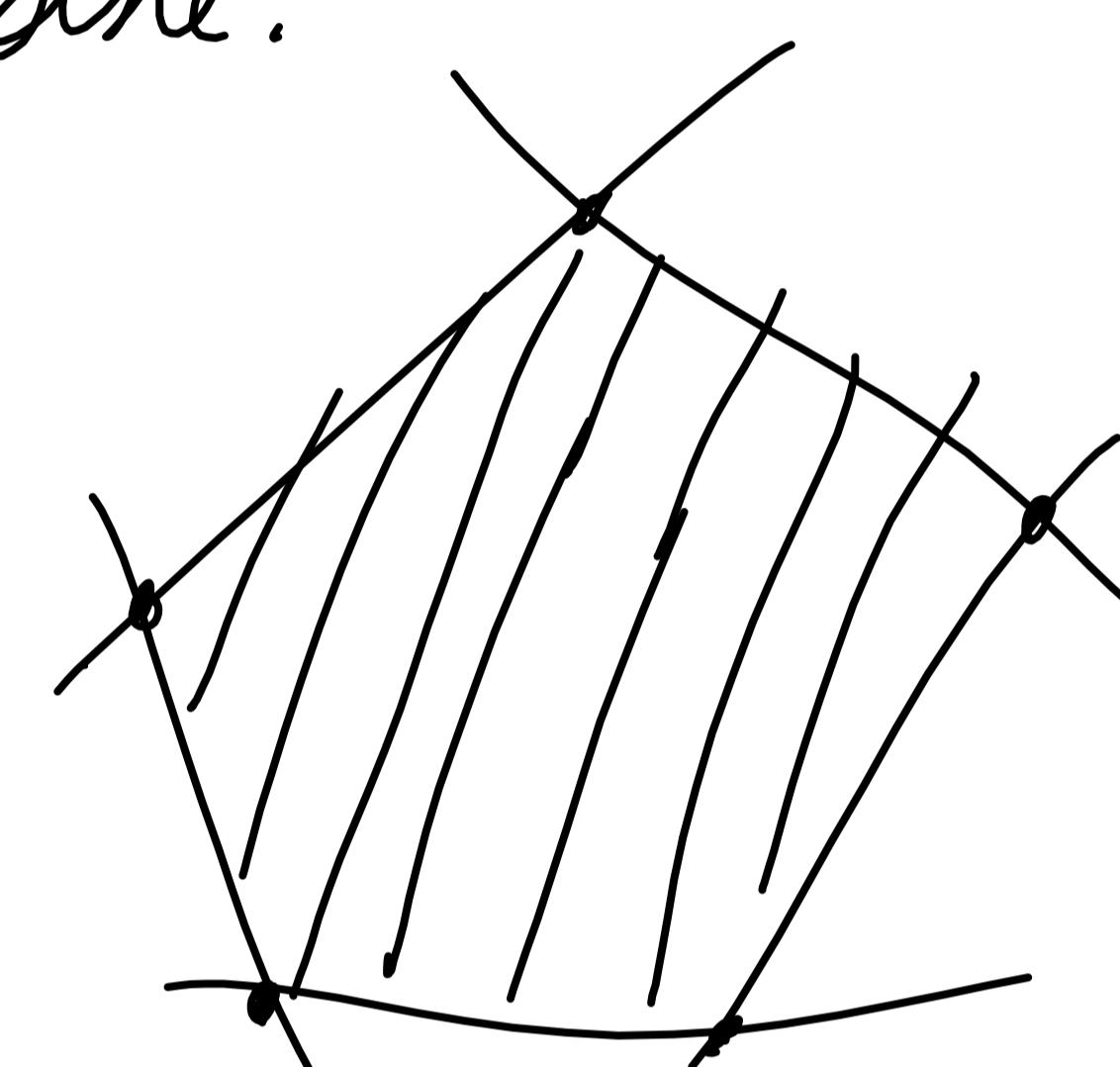


$\{x \in \mathbb{R}^n \mid \alpha^T x = \alpha_0\}$ : HYPERPLANE

In LP, si dà set di dis./eq.  $\Rightarrow$  set di iperpiani e semispazi

Def: intersezione di  $\#$  finito di semispazi affini e iperpiani:  
 CONVEX POLYHEDRON

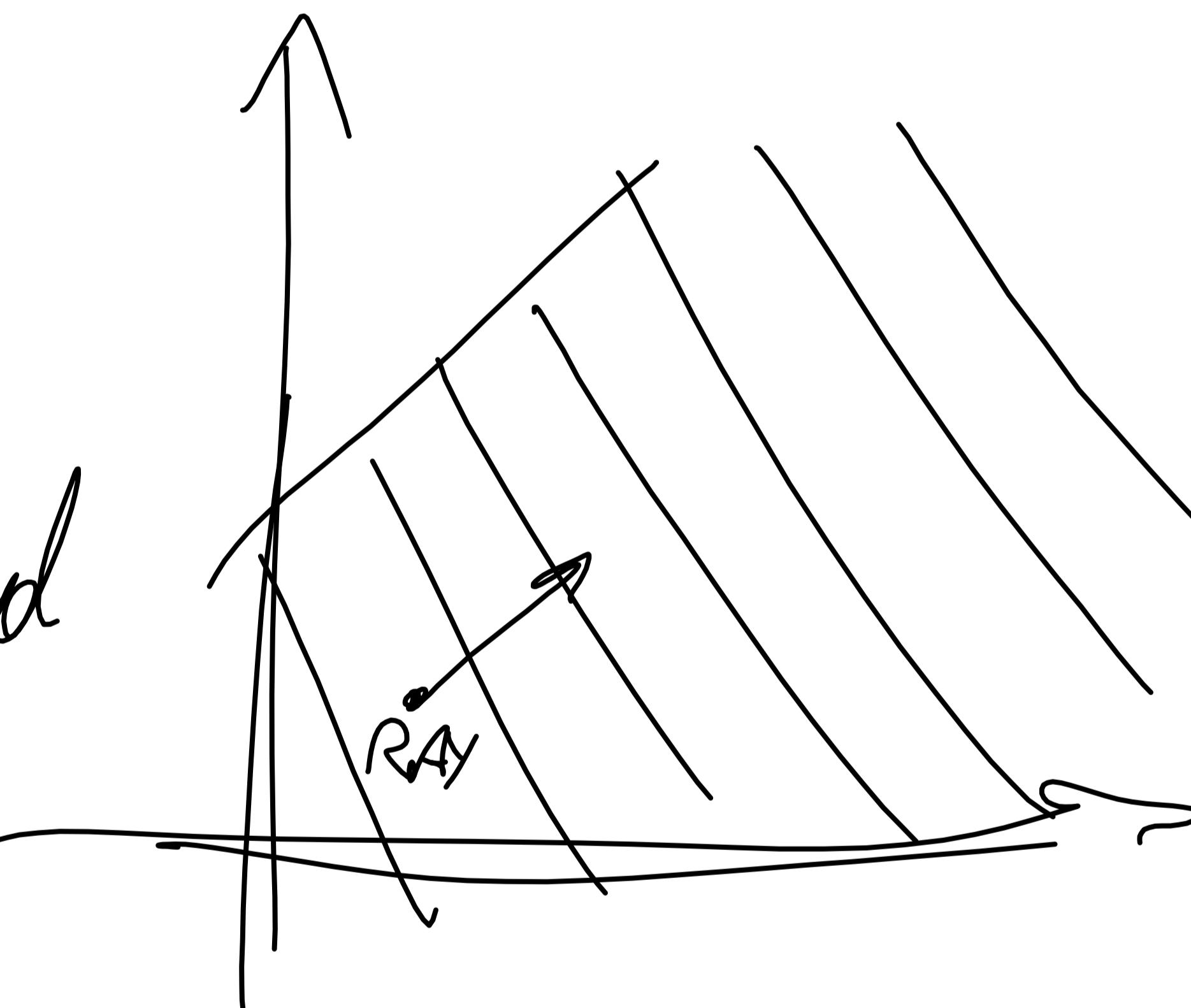
(tutti oggetti in inter. sono convessi)



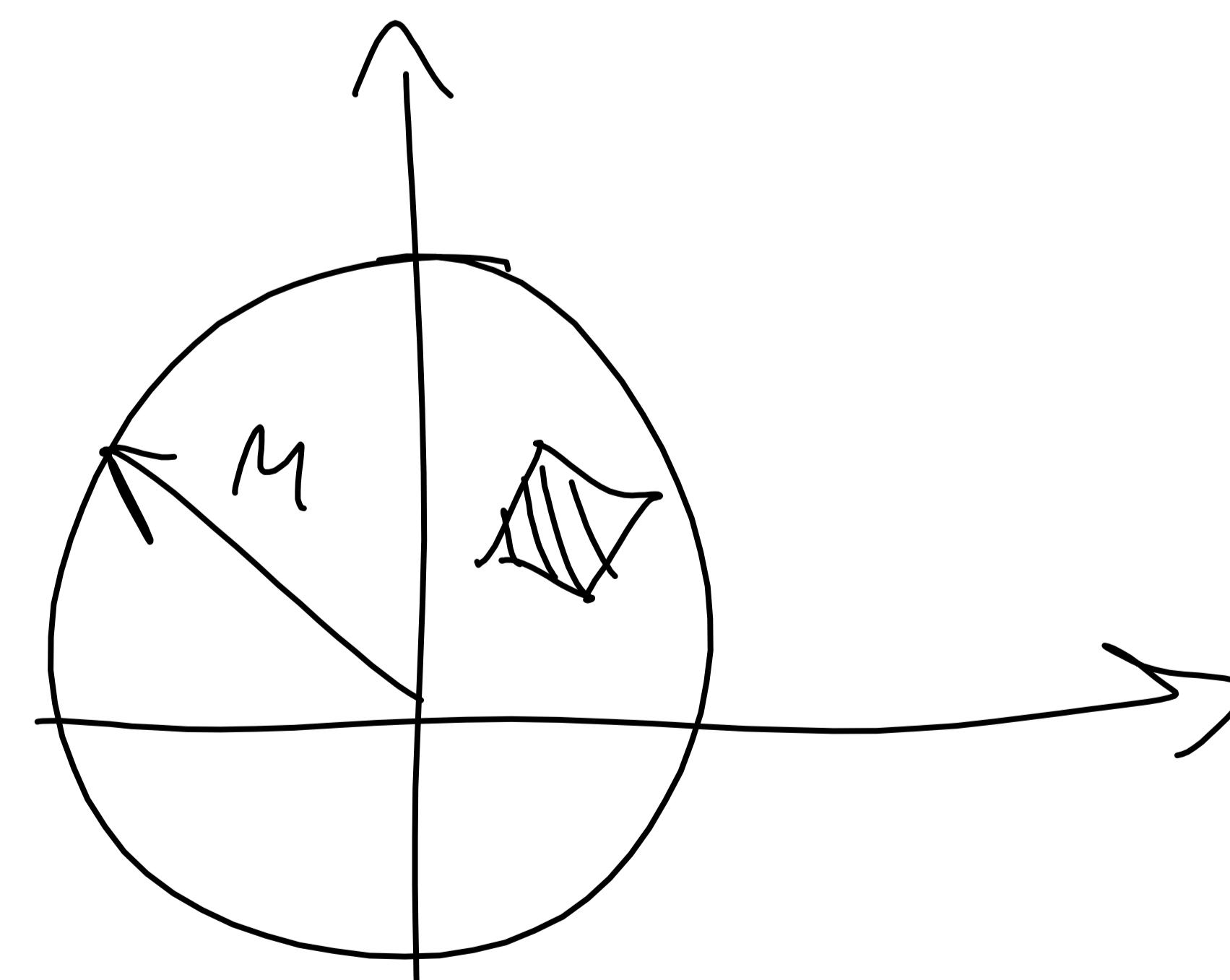
Se accettiamo  $\#$  infinito, ci avvicineremo a cerchio  
 non raggiamo fisi vertici

Spesso, poliedro non limitato  $\Rightarrow$

$\Rightarrow$  non siamo trovare direzione in cui si va  
 all'infinito  $\Rightarrow$  new more problema unbounded



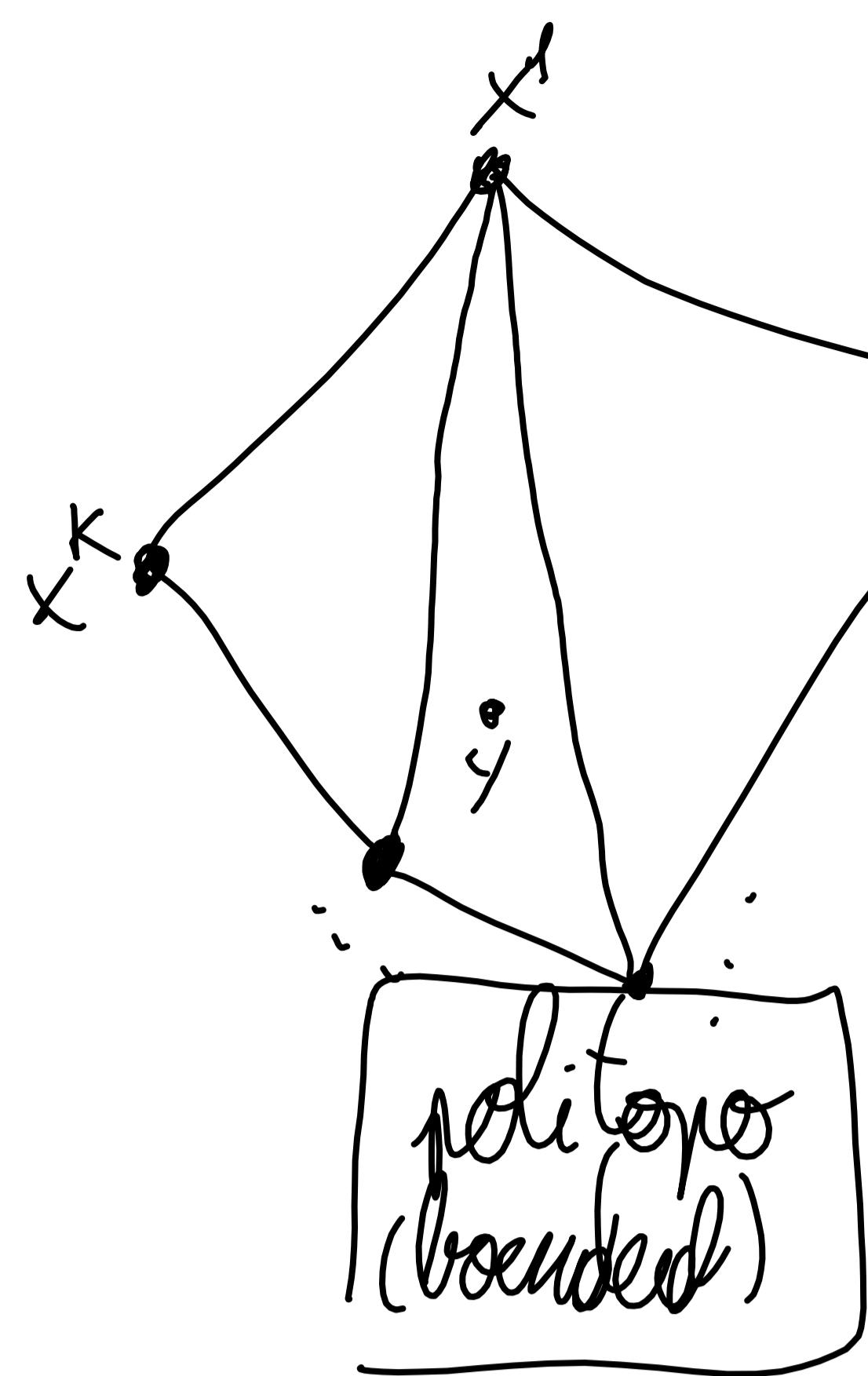
Spesso, poliedri contenuti in sfera di  
 raggio  $M \geq 0$ ,  $M \in \mathbb{R}$ ,  
 BOUNDED POLYHEDRON/POLYTOPE



Def: un punto  $x \in P$  è un VERTICE di  $P$  se non può essere espresso come  
 comb. convessa [stretta] di due punti  $y, z \in P$   $\square$

(punti ottenibili con comb. convessa non servono per ottimizzazione)

# TEOREMA DI MINKOWSKI-WEYL



vettori  $x^1, \dots, x^K \in P$

$$\forall y \in P \exists \lambda_1, \dots, \lambda_K \geq 0 \mid \sum_{i=1}^K \lambda_i = 1, \sum_{i=1}^K \lambda_i x^i = y$$

non vale per unbounded  $\Rightarrow$

$\Rightarrow$  si può estendere: serve coinvolgere raggi  $\Rightarrow$   
 $\Rightarrow$  allora si possono aggiungere limiti fissi

## THEOREM

$P$  bounded polyhedron (poliporto)  $\Rightarrow \min_{x \in P} c^T x$  ha soluzione ottima al vertice di  $P$  (arrivo in a vertex)

Dim:  $x^1, \dots, x^K$  vertici di  $P$

calcolare  $z^* = \min\{c^T x^i, i \in [1, K]\}$

consideriamo qualunque  $y \in P \Rightarrow$  M-W:  $\exists \lambda \geq 0 \mid y = \sum_{i=1}^K \lambda_i x^i, \sum_{i=1}^K \lambda_i = 1$

calcolare:  $c^T y = c^T \sum_{i=1}^K \lambda_i x^i = \sum_{i=1}^K \lambda_i c^T x^i \geq \sum_{i=1}^K \lambda_i z^* = z^* \Rightarrow c^T y \geq z^*$

Ora dobbiamo calcolare vertici

$$\text{ES: } \begin{cases} \min -x_1 - x_2 \\ 6x_1 + 4x_2 + x_3 = 24 \\ 3x_1 - 2x_2 + x_4 = 6 \\ x_1, x_2, x_3, x_4 \geq 0 \end{cases}$$

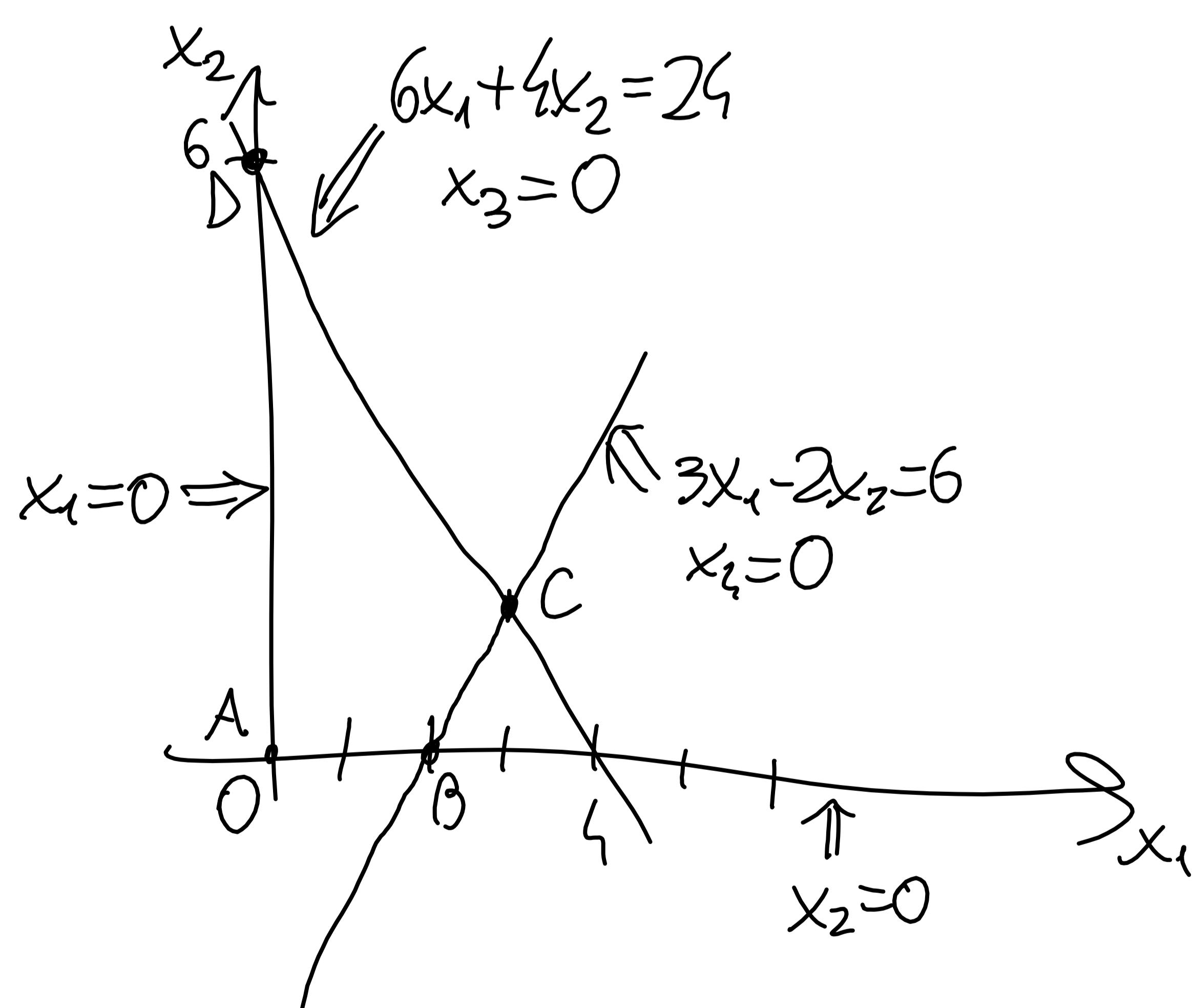
condizione usare forma standard

vertice A:  $x_1 = x_2 = 0$

vertice B:  $x_2 = x_4 = 0$

vertice C:  $x_3 = x_4 = 0$

vertice D:  $x_1 = x_3 = 0$



punto  $x_2 = x_3 = 0$ : no vertice  $\Rightarrow x_1 < 0$

linee parallele: no intersezione (e non coincidenti)

In general:

$$Ax = b \quad (x \geq 0) \quad (\text{non serve sapere var. slack}) \Rightarrow m \times n, m \leq n$$

supponiamo che tutte righe di  $A$  siano lin. ind.  $\Leftrightarrow \text{rank}(A) = m$

Def: BASIS di matrice  $A$ : insieme di  $m$  colonne lin. ind. di  $A$

$A = \left[ \begin{array}{c|cc|c} 0 & A_1 & \dots & A_m \end{array} \right] \Rightarrow$  prenderne alcune, in questo ordine:

$$B = \left[ A_{\beta(1)} \mid A_{\beta(2)} \mid \dots \mid A_{\beta(m)} \right] \Rightarrow m \times m \Rightarrow \det(B) \neq 0$$

*notazione per distinguere  
da  $A_1, \dots, A_n$*

$$Ax = b \Rightarrow A_1 x_1 + A_2 x_2 + \dots + A_n x_n$$

prendo qualsiasi basis non-singolare  $B$

$$x = \begin{bmatrix} x_B \\ x_F \end{bmatrix} \Rightarrow \text{alcune saranno in basis} \Rightarrow x = \begin{bmatrix} x_B \\ 0 \end{bmatrix}, \quad A = \left[ \begin{array}{c|c} B & F \end{array} \right]$$

*(sono, altrimenti blu)*

$$Ax = b \Rightarrow \left[ B \mid F \right] \begin{bmatrix} x_B \\ 0 \end{bmatrix} = Bx_B + Fx_F = b \quad \forall \text{ basis } B \Rightarrow$$

$$\Rightarrow Bx_B = b - Fx_F \Rightarrow B^{-1}Bx_B = B^{-1}[b - Fx_F] \Rightarrow \underbrace{x_B = B^{-1}b - B^{-1}F x_F}_{\text{CANONICAL FORM}}$$

$$\underbrace{x_F = 0}_{\text{ecco numero diverso}} \Rightarrow x_B = B^{-1}b \Rightarrow x = \begin{bmatrix} x_B = B^{-1}b \\ x_F = 0 \end{bmatrix}: \text{BASIC SOLUTION corresponding to } B$$

*feasible se  $B^{-1}b \geq 0$*

mentre a 0, come fatto prima  $\Rightarrow$  lin. ind: non possono essere linee parallele

10/10

$(B^{-1}b)_i = 0$ : possibile  $\Rightarrow$  DEGENERATE basic solution  $\forall i$

Basic solutions = vertici: TEOREMA

un punto  $x \in P$  è vertice di  $P := \{x \geq 0 | Ax = b\} \neq \emptyset \Leftrightarrow x$  è basic feasible solution associata al sistema  $Ax = b$

Dim: proviamo " $x$  è BFS  $\Rightarrow x$  è vertice"

$$x = \begin{bmatrix} x_1, \dots, x_k, 0, \dots, 0 \end{bmatrix}^T \text{ per } k \geq 0$$

$\Rightarrow$  basic variables  $\Rightarrow$  colonne  $A_1, \dots, A_k \in \text{basis} \Rightarrow$  lin. ind.

Proviamo per dimostrazione:  $x$  non vertice

$$\exists y \neq z \in P, \lambda \in ]0, 1[ \Rightarrow x = \lambda y + (1-\lambda)z$$

$$y = [y_1, \dots, y_k, 0, \dots, 0], z = [z_1, \dots, z_k, 0, \dots, 0]$$

$$y \in P \Rightarrow A_1 y_1 + \dots + A_k y_k = b; z \in P \Rightarrow A_1 z_1 + \dots + A_k z_k = b \Rightarrow$$

$$\Rightarrow A_1 (\underbrace{y_1 - z_1}_{d_1}) + \dots + A_k (\underbrace{y_k - z_k}_{d_k}) = 0$$

$d_i$  non sono tutti nulli  $\Rightarrow$  se  $y \neq z$

$\Rightarrow A_1, \dots, A_k$  lin. dip.  $\Rightarrow$  CONTR.: avevamo detto lin. ind.  $\Rightarrow x$  è vertice

Proviamo "x vertice  $\Rightarrow$  x BFS"

Feasibility point: vertice  $\in P$

$$x = [x_1, \dots, x_k, 0, \dots, 0]^T \text{ con } k \in [0, n] \Rightarrow x \geq 0, \text{ nemuna comp. nulla}$$

$$x \in P \Rightarrow A_1 x_1 + \dots + A_k x_k = b$$

Due casi possibili:

1)  $A_1, \dots, A_k$  lin. ind.

$$x = \underbrace{\begin{bmatrix} x_1, \dots, x_k, 0, \dots, 0 \end{bmatrix}}_{m \downarrow} \Rightarrow m < k \text{ per degeneracy} \Rightarrow$$

$$\Rightarrow A = [A_1, \dots, A_k, \underbrace{A_m, \dots}_m]$$

2)  $A_1, \dots, A_k$  lin. dip.  $\Rightarrow \exists d_1, \dots, d_k$ , non tutte = 0 |  $A_1d_1 + \dots + A_kd_k = 0$

prendo  $\epsilon \geq 0 \Rightarrow (A_1x_1 + \dots + A_kx_k = b) + \epsilon(A_1d_1 + \dots + A_kd_k = 0) \Rightarrow$   
 $\Rightarrow A_1(\underbrace{x_1 + \epsilon d_1}_{z_1}) + \dots + A_k(\underbrace{x_k + \epsilon d_k}_{z_k}) = b$

$$(A_1x_1 + \dots + A_kx_k = b) - \epsilon(A_1d_1 + \dots + A_kd_k = 0) \Rightarrow$$

$$\Rightarrow (\underbrace{x_1 - \epsilon d_1}_{y_1})A_1 + \dots + (\underbrace{x_k - \epsilon d_k}_{y_k})A_k = b$$

$$Y = [x_1, \dots, x_k, 0, \dots, 0], Z = [z_1, \dots, z_k, 0, \dots, 0] \Rightarrow Y, Z \geq 0$$

oltre  $Ax = b, Az = b \Rightarrow x, z \in P; x \neq z; x = \frac{1}{2}y + \frac{1}{2}z \Rightarrow x$  non vertice  $\Rightarrow$   
 CONTR.:  $A_1, \dots, A_k$  non sono esp. lin. dip.

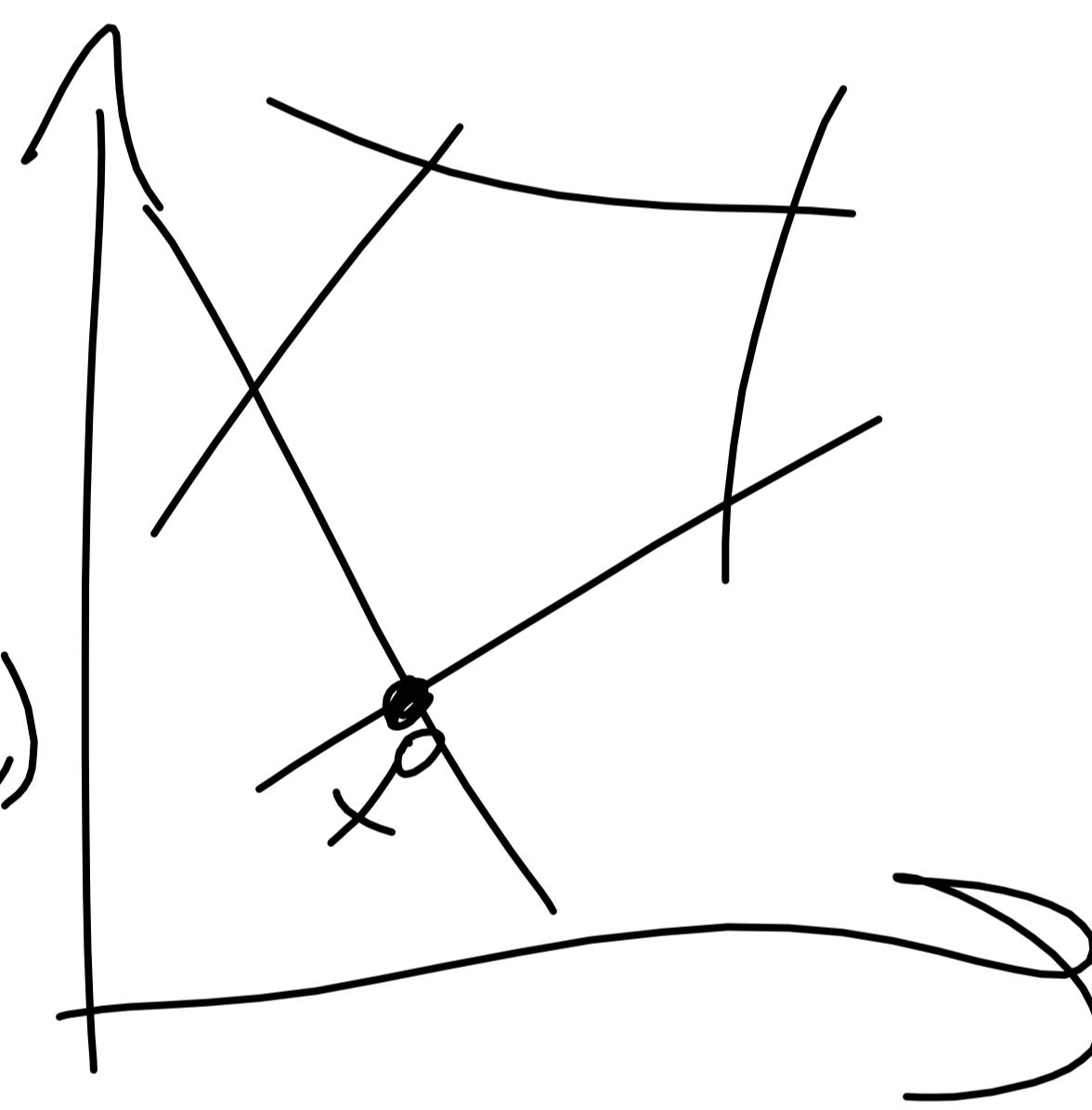
Problema: alto numero di BFS  $\Rightarrow$  metodo migliore, di George Dantzig

computare vertice per una BFS  $x^*$ , poi fare

OPTIMALITY TEST:  $x^*$  è ottimo locale?

Se non lo è, si cambia base (moveni in altro vertice)

finché raggiunge ottimo



OPTIMALITY TEST

$$Ax = b, x \geq 0; \text{ base attuale } B \Rightarrow x_B = B^{-1}b - B^{-1}F x_F \quad (\forall x \in P)$$

$$\text{recalcoliamo } C^T x = [C_B^T, C_F^T] \begin{bmatrix} x_B \\ x_F \end{bmatrix} = C_B^T x_B + C_F^T x_F =$$

$$= C_B^T (B^{-1}b - B^{-1}F x_F) + C_F^T x_F = \underbrace{C_B^T B^{-1}b}_{c_0} + \underbrace{0^T x_B}_{0} + \underbrace{(C_F^T - C_B^T B^{-1}F)}_{\bar{C}_F^T} x_F$$

$\bar{C}_F^T$ : REDUCED COST VECTOR

$$x_F = 0 \Rightarrow \text{siamo su BFS corrente} \Rightarrow C^T x = C_B^T B^{-1}b = c_0$$

costo di BFS/vertice corrente

Se  $\bar{C}_F^T \geq 0 \Rightarrow C^T x \geq c_0 \Rightarrow$  ogni altra soluzione avrà costo maggiore  $\Rightarrow$   
 termine calcolarlo  $\Rightarrow$  trovato ottimo

Optimality test: condizione sufficiente per fermarsi

In alternativa:  $\bar{c}^T = c^T - C^T B^{-1} A$  (costo totale, se tutte componenti)  
||

$$[\bar{c}_B^T, \bar{c}_F^T] = \left( \underbrace{c_B^T - C_B^T B^{-1} B}_{0}, \underbrace{c_F^T - C_B^T B^{-1} C}_{\bar{c}_F^T} \right)$$

11 / 10

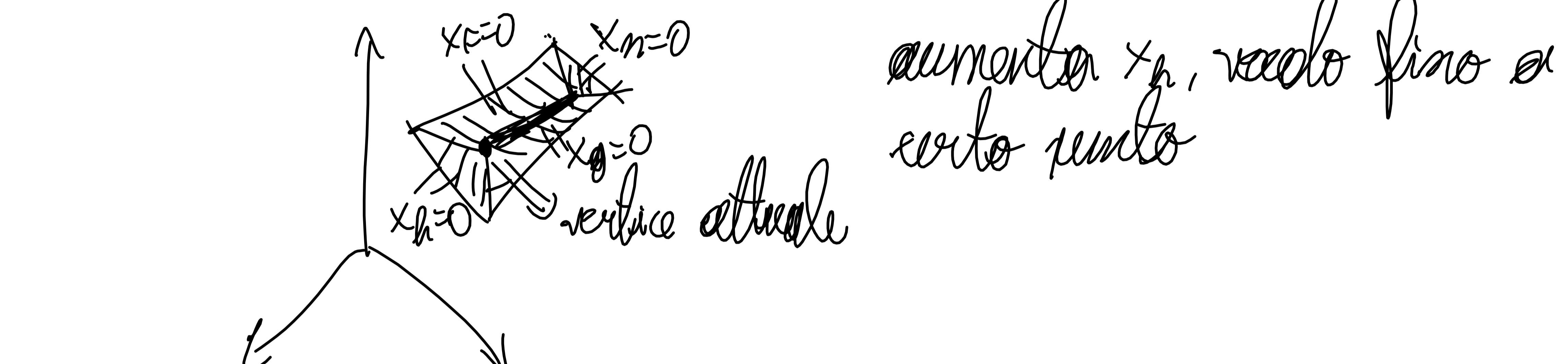
B base covariate  $\Rightarrow \bar{C}^T = C^T - U^T A, \quad U^T = C_B^T B^{-1}$

Supponiamo  $h \mid \bar{c}_h = c_h - u^T A_h < 0$ :

$$\dot{C}^T X = \underline{\underline{C}}^T B^{-1} b + \bar{C}_h^T X_h; \quad C^T X = \underline{\underline{C}}^T B^{-1} b + \bar{C}_F^T X_F$$

$\text{Co}$   $\text{Zn}$   $\text{O}$

$x_h = 0$  in sol. teoriche, voglio aumentarla  $\Rightarrow$   
 $\rightarrow$  altre componenti devono adattarsi  $\Rightarrow$  fissa a val. max possibile: 0  
(com. basiche)



$$(x_3 = \beta^{-1} b - \beta^{-1} f x_f)$$

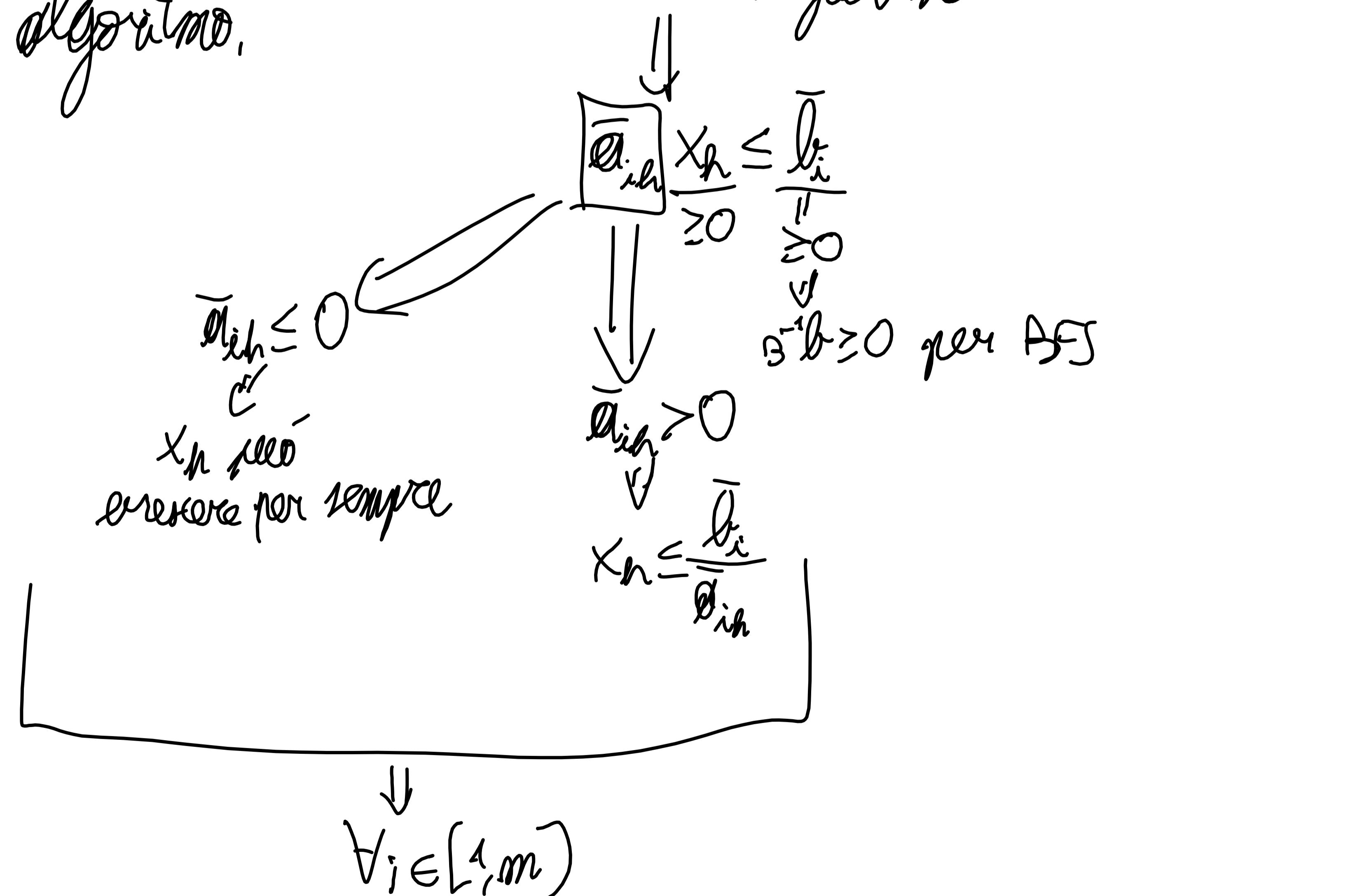
$$x_B = \underline{B^{-1}b} - \underline{\frac{B^{-1}A_h}{\tilde{A}_h} x_h}; \text{ anche alle vor. au-henche sind } \neq 0$$

$$x_B = \begin{bmatrix} x_{\beta[0]} \\ \vdots \\ x_{\beta[i]} \\ \vdots \\ x_{\beta[m]} \end{bmatrix} = \begin{bmatrix} b_1 \\ \bar{b}_2 \\ \vdots \\ b_n \\ \bar{b}_{m+1} \end{bmatrix} - \begin{bmatrix} \bar{\alpha}_{1h} \\ \bar{\alpha}_{2h} \\ \vdots \\ \bar{\alpha}_{ih} \\ \vdots \\ \bar{\alpha}_{mh} \end{bmatrix} x_h$$

$\Rightarrow x_{\beta(i)} = b_i - \bar{\alpha}_{ih} x_h \quad \forall i \in [1, m]$

$\downarrow$   
 $x_h^1, x_h^2, \dots, x_h^m$ :  $x_h$  von oben

base sommiserà durante algoritmo,  
quindi notazione  $\Theta(i)$



$x_h \Rightarrow$  diversi limiti:  $\frac{\bar{t}_i}{\bar{a}_{ih}}$  con  $\bar{a}_{ih} > 0 \Rightarrow$  si si forma a primo limite  $\Rightarrow \theta = \min \left\{ \frac{\bar{t}_i}{\bar{a}_{ih}} \mid \bar{a}_{ih} > 0 \right\}$

$x_n: \mathbb{O} \rightarrow \Theta$ : si scrive  $x_{\beta(t)}: \bar{\mathbb{O}}_t \rightarrow \mathbb{O}$   $\Rightarrow$  mi interessa anche  $t = \arg\min_i \left\{ \frac{\bar{v}_i}{\bar{a}_{ih}} \mid \bar{a}_{ih} > 0 \right\}$

$B = [A_{B(1)} \mid \dots \mid A_{B(t)} \mid \dots \mid A_{B(m)}]$   $\Rightarrow x_h$  deve entrare in base in prox. iterazione  
 $A_{B(t)}$  si può togliere da base

## SIMPLEX METHOD

1. inizializzazione

trovare base di partenza  $B = [A_{B(1)} \mid \dots \mid A_{B(m)}]$

2. optimality test

$$\bar{U}^T := C_B^T B^{-1}, \text{ se } \bar{C}^T := C^T - U^T A \geq 0, \text{ STOP} \Rightarrow x = \begin{bmatrix} B^{-1} b \\ 0 \end{bmatrix}$$

3. cambio di base

seguiamo  $\bar{c}_h = c_h - u^T A_h < 0 \Rightarrow x_h$  vuole entrare in base

$t := \arg\min_i \left\{ \frac{\bar{v}_i}{\bar{a}_{ih}} \mid \bar{a}_{ih} > 0 \right\}$ ,  $\bar{B}_t := B^{-1} B_t$ ,  $\bar{A}_h := B^{-1} A_h \Rightarrow x_{B(t)}$  deve uscire  $\Rightarrow$

$$\Rightarrow B(t) := h$$

4. ripetere

Procedimento finito, se non si parametrizza calcolo  $B^{-1} \Rightarrow$  aggiornare inverso

Se  $t = \arg\min \emptyset \Rightarrow$  non si può scindere  $\Rightarrow$  no limite ad aumento  $\Theta \rightarrow +\infty \Rightarrow$   
 $\Rightarrow C^T x: C_0 \rightarrow -\infty \Rightarrow$  problema unbounded

Ad ogni iterazione, tenere sistema in forma canonica rispetto a base attuale

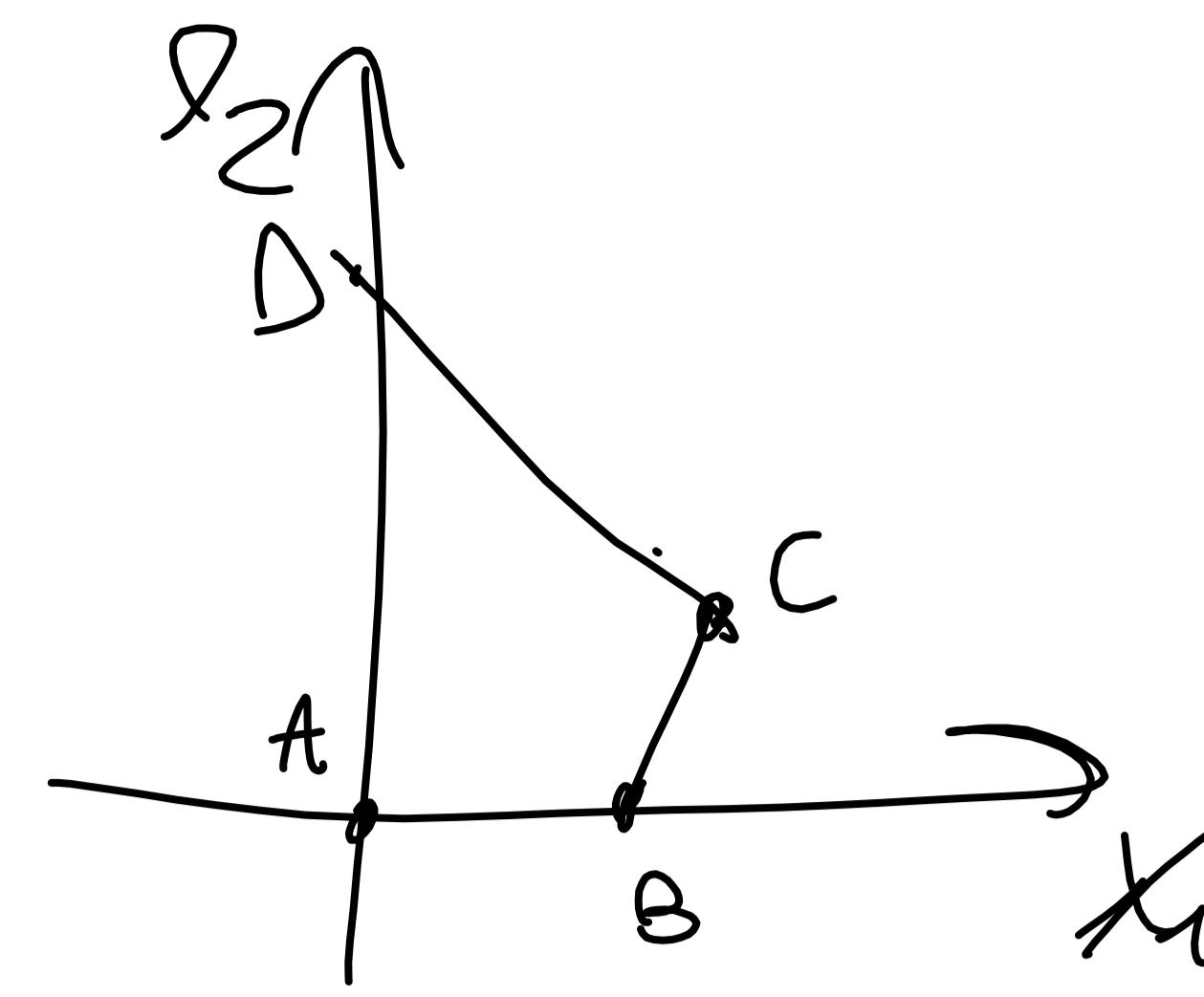
$$C^T x = z: nuova variabile \Rightarrow z = C_B^T B^{-1} b + (C_F^T - C_B^T B^{-1} F) x_F$$

$$x_B = B^{-1} b - B^{-1} F x_F \geq 0 \Rightarrow \text{così ricavo } B^{-1} b / B^{-1} F$$

solo a sinistra:  
calcolabili con funzione ex

per questo è forma canonica

$$\left\{ \begin{array}{l} \min z = -x_1 - x_2 \\ 6x_1 + 4x_2 + x_3 = 24 \\ 3x_1 - 2x_2 + x_4 = 6 \\ x_1, x_2, x_3, x_4 \geq 0 \end{array} \right.$$



punto da A, aumento  $x_1 \Rightarrow$

$\Rightarrow$  non vado oltre B, altrimenti  $x_1 < 0$

(vedere libro)

16/10

## TABLEAU form di simplex

$$\begin{cases} x_B = B^{-1}b - B^{-1}F x_F \\ z = C_B^T B^{-1}b + (C_F - C_B^T B^{-1}F) x_F \end{cases}$$

L'iterazione originale:  $\begin{cases} \min c^T x = z \\ Ax = b \\ x \geq 0 \end{cases}$

$$c^T x - z = 0$$

membru di  
destra

0	$x_1 x_2 \dots x_n$	$z$
0	$c_1 c_2 \dots c_n$	-1
b	A	0
		0
		0
		0
		m

riga 0  
obj. function  
1 riga  
:

↓  
tableau in forma canonica

$$\begin{cases} B^{-1}b = I x_B + B^{-1}F x_F \\ -C_B^T B^{-1}b = 0^T x_B + (C_F - C_B^T B^{-1}F) x_F - z \end{cases}$$



$-C_B^T B^{-1}b$

	$x_1 x_2 \dots x_n$	$z$
0 .. 0	$\bar{c}_m \dots \bar{c}_n$	-1
$\bar{b}$	I	0
		0
		0
		0

$\left(\bar{b} = B^{-1}b\right)$  base non base

$\leftarrow$  reduced cost

: CANONICAL TABLEAU:  
ottenuto da originale moltiplicato per  $B^{-1}$   
(tranne riga 0)

Dal ogni iterazione, verificare forma canonica  $\Rightarrow$  sanity check

	$0 \dots 0$	$\dots$	$\bar{c}_h < 0$	$\dots$
$\bar{b} \geq 0$	$\bar{b}_1$	I	$\bar{a}_{1h}$	$\dots$
	$\vdots$		$\vdots$	
	$\bar{b}_m$		$\bar{a}_{mh}$	

$\leftarrow$  riga t: la evidenza  $\Rightarrow$  contiene PIVOT ELEMENT

$\Rightarrow x_h$  vuole entrare in base

$\theta = \min \left\{ \frac{\bar{b}_i}{\bar{a}_{ih}} \mid \bar{a}_{ih} > 0 \right\}$

Come scegliere var. entrante  $x_h < 0$

Vogliamo arrivare a ottimo in 1 iterazione  $\Rightarrow$  facile se conosciamo poliedro intero  $\Rightarrow$   
 $\Rightarrow$  no teorema unico  $\Rightarrow$  solo regole empiriche

1 - più semplice: scegliere prima  $\bar{c}_h < 0$  in tableau

2 - scegliere con  $|\bar{c}_h|$  massimo

3 -  $\min |\Delta z| = |\bar{c}_h| \cdot \theta_h$ : valuto in base a cambiamento  $\varepsilon \Rightarrow$   
 $\Rightarrow$  scelgo tasso di miglioramento massimo  $\Rightarrow$   
 $\Rightarrow$  tuttavia,  $O(mn)$ : si spende "tanto" tempo

4 - random

Come fare initializzazione?

Base di partenza deve avere  $\bar{b} \geq 0 \Rightarrow$  random fino a trovare base valida  $\Rightarrow$   
 $\Rightarrow$  non va bene

Potrei anche avere infeasibile

17/10

Se  $w^* = 0 \Rightarrow y = 0 \Rightarrow$  sol. feasible per originale

Così posso usare simplex: ho T per tableau

26/10

## DUALITY

Una diseguaglianza  $a^T x \geq a_0$  valida per  $x \Leftrightarrow a^T x \geq a_0 \quad \forall x \in X$

EJ 1 (pag. 66)

Da es. problema, eq. con fattori  $\Rightarrow$  infiniti modi per i vari (rentabili) /  
daventare lato destro

$$\left. \begin{array}{l} 1 - \text{d'origine} \quad u^T A x = u^T b \\ 2 - // \quad u^T A x \geq u^T B \\ 3 - c^T x \geq c_0 \end{array} \right\} \begin{array}{l} \text{no validi per} \\ \text{ottenere sol.} \end{array}$$

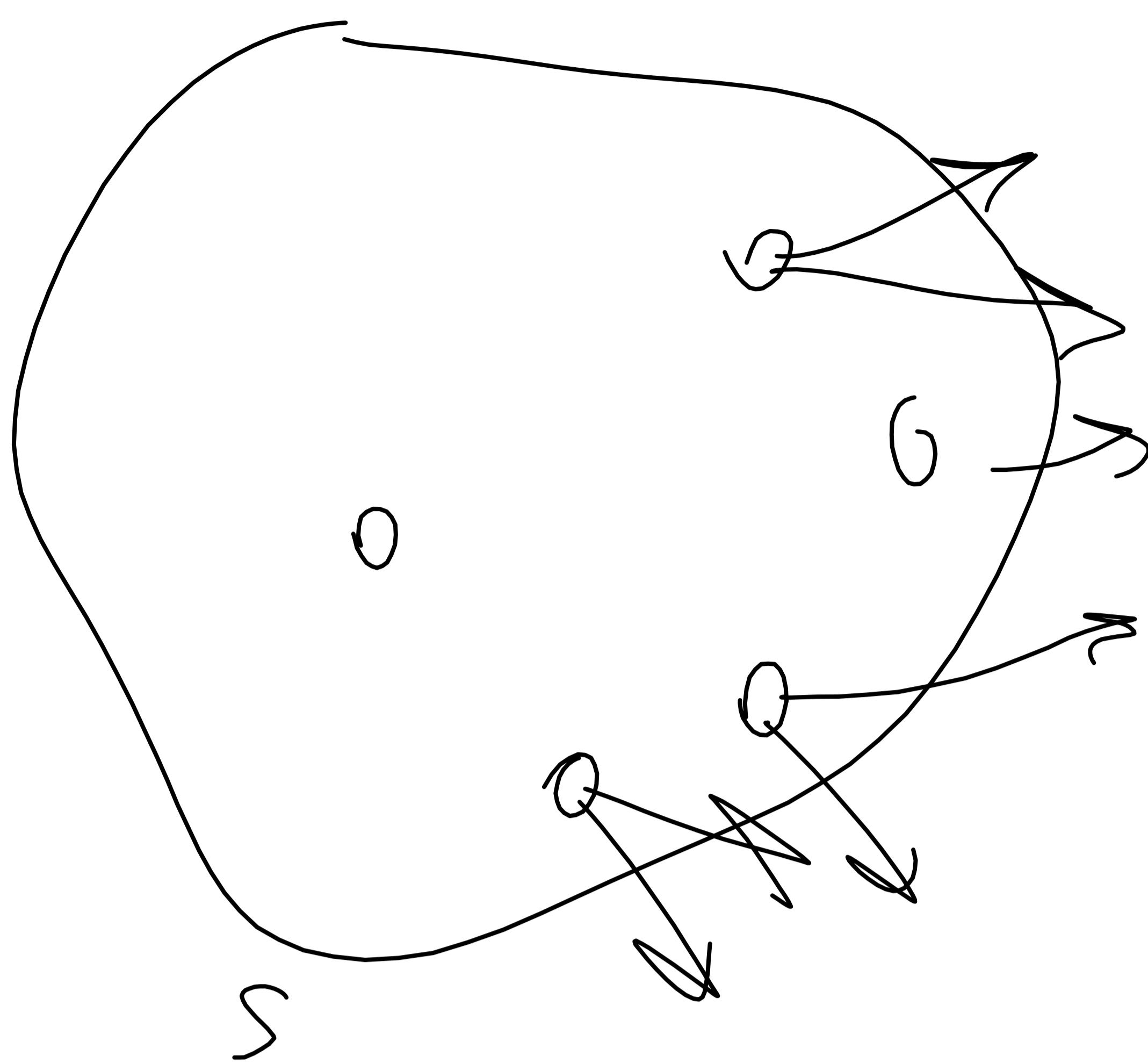
The: LEMMA DI FARKAS:

$c^T x \geq 0$  inequality  $\rho = \{x \geq 0 \mid Ax = b\} \neq \emptyset \iff \exists u \in \mathbb{R}^m \mid c^T \geq u^T A$ ,

$\vdash 0, 0]$

S12

SECTION/CUT:  $(S, V \setminus S)$  I set, tagli (c'è anche perfezione nodi)



Definiamo feasible flow  $x$ :

$$\varphi(S) := \sum_{(i,j) \in \delta^+(S)} x_{ij} - \sum_{(i,j) \in \delta^-(S)} x_{ij}$$

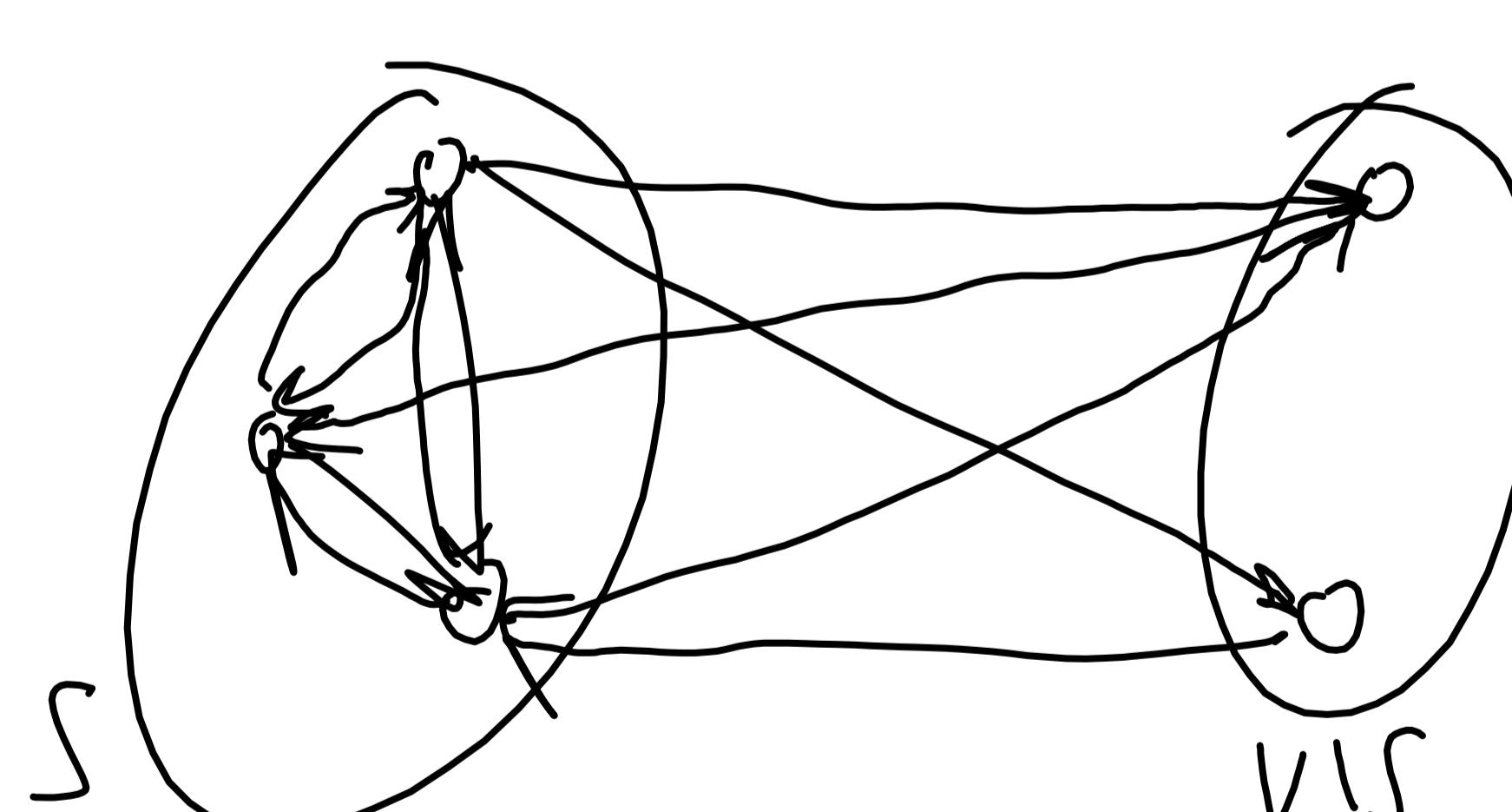
$$\frac{k(S)}{k} := \sum_{(i,j) \in \delta^+(S)} k_{ij}$$

*capacità  
di cut*

Teo1: se  $x$  è qualsiasi feasible flow;  $\forall$  sezione  $(S, V \setminus S)$ ,  $\varphi(S)$  è costante

Teo2:  $\varphi(S) \leq k(S)$

$$\begin{aligned} \text{Dim1: } \varphi_0 &= \sum_{(i,j) \in \delta^+(S)} x_{ij} - \sum_{(i,j) \in \delta^-(S)} x_{ij} = \\ &= \sum_{A \in S} \left( \sum_{(i,j) \in \delta^+(A)} x_{ij} - \sum_{(i,j) \in \delta^-(A)} x_{ij} \right) = \\ &\quad \downarrow \\ &= 0 \quad \forall h \in S \setminus \{s\} \\ &= \sum_{h \in S} \sum_{(i,j) \in \delta^+(A)} x_{ij} - \sum_{h \in S} \sum_{(i,j) \in \delta^-(A)} x_{ij} = \\ &\quad \downarrow \\ &= \left( \sum_{(i,j) \in A(S)} x_{ij} + \sum_{(i,j) \in \delta^+(S)} x_{ij} \right) - \left( \sum_{(i,j) \in A(S)} x_{ij} + \sum_{(i,j) \in \delta^-(S)} x_{ij} \right) =: \varphi(S) \end{aligned}$$



$$\text{Dim2: } \varphi(S) = \underbrace{\sum_{(i,j) \in \delta^+(S)} x_{ij}}_{\leq k_{ij}} - \underbrace{\sum_{(i,j) \in \delta^-(S)} x_{ij}}_{=0} \leq K(S)$$

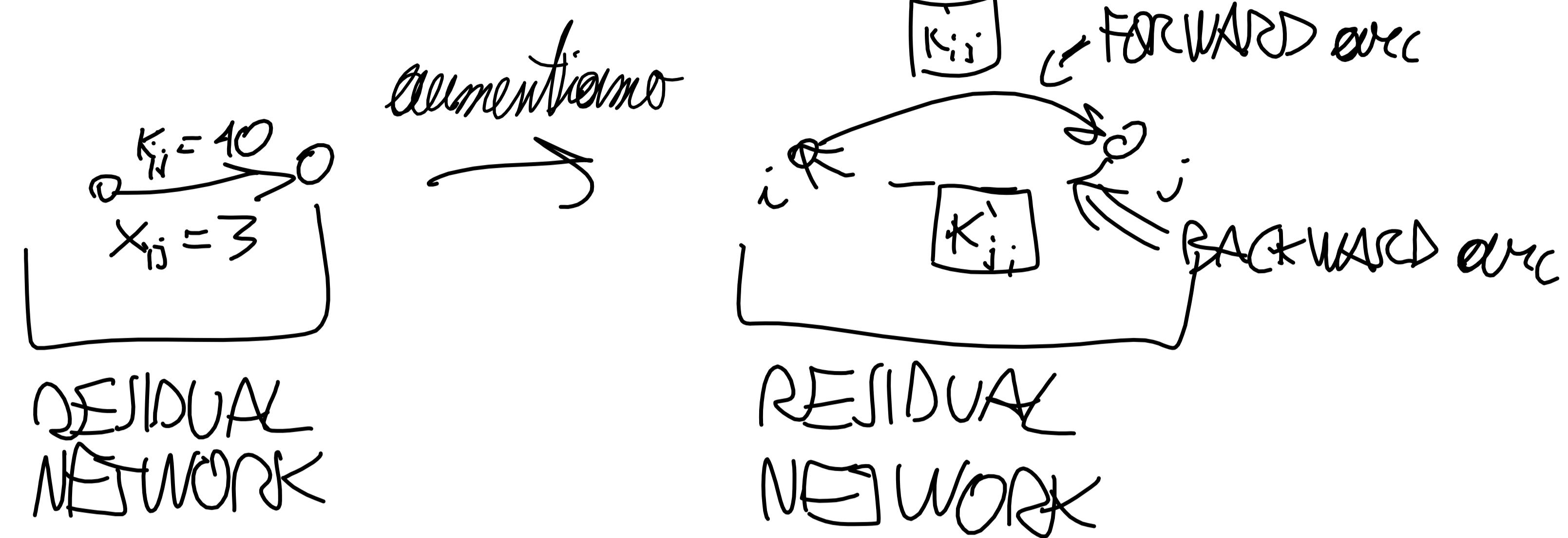
PRIMAL METHOD:

all'inizio,  $x=0$

flow attuale  $x$

(RESIDUAL) NETWORK

rispetto a  $x$

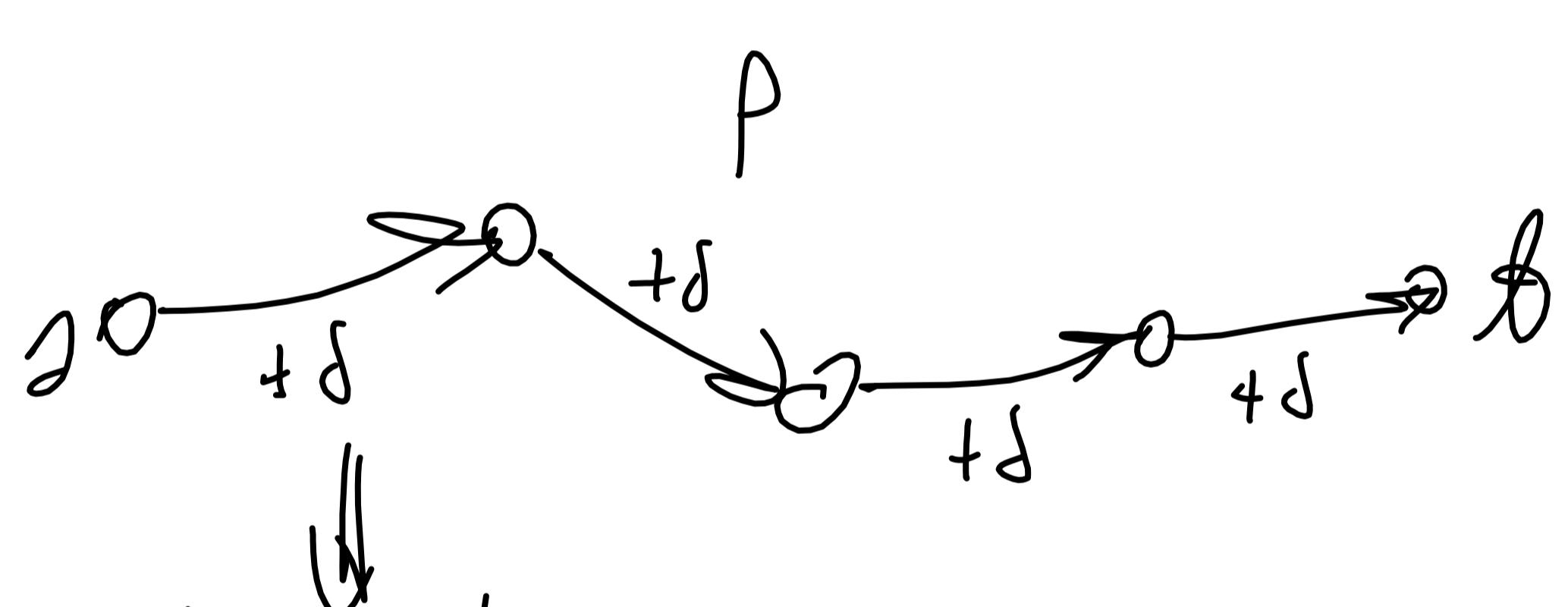


introduciamo RESERVE ARC:

$$k'_{ij} = x_{ij}$$

Se abbiamo SATURAZIONE ( $x_{ij} = k_{ij}$ ), avremo  $k'_{ij} = 0 \Rightarrow$   
 $\Rightarrow$  bisogna togliere archi residuali con  $k'_{ij} = 0$

Residual network



Dico percorso con labeling

$$\delta := \min \{k'_{ij} \mid (v_i, v_j) \in P\}$$

flow extra mandato

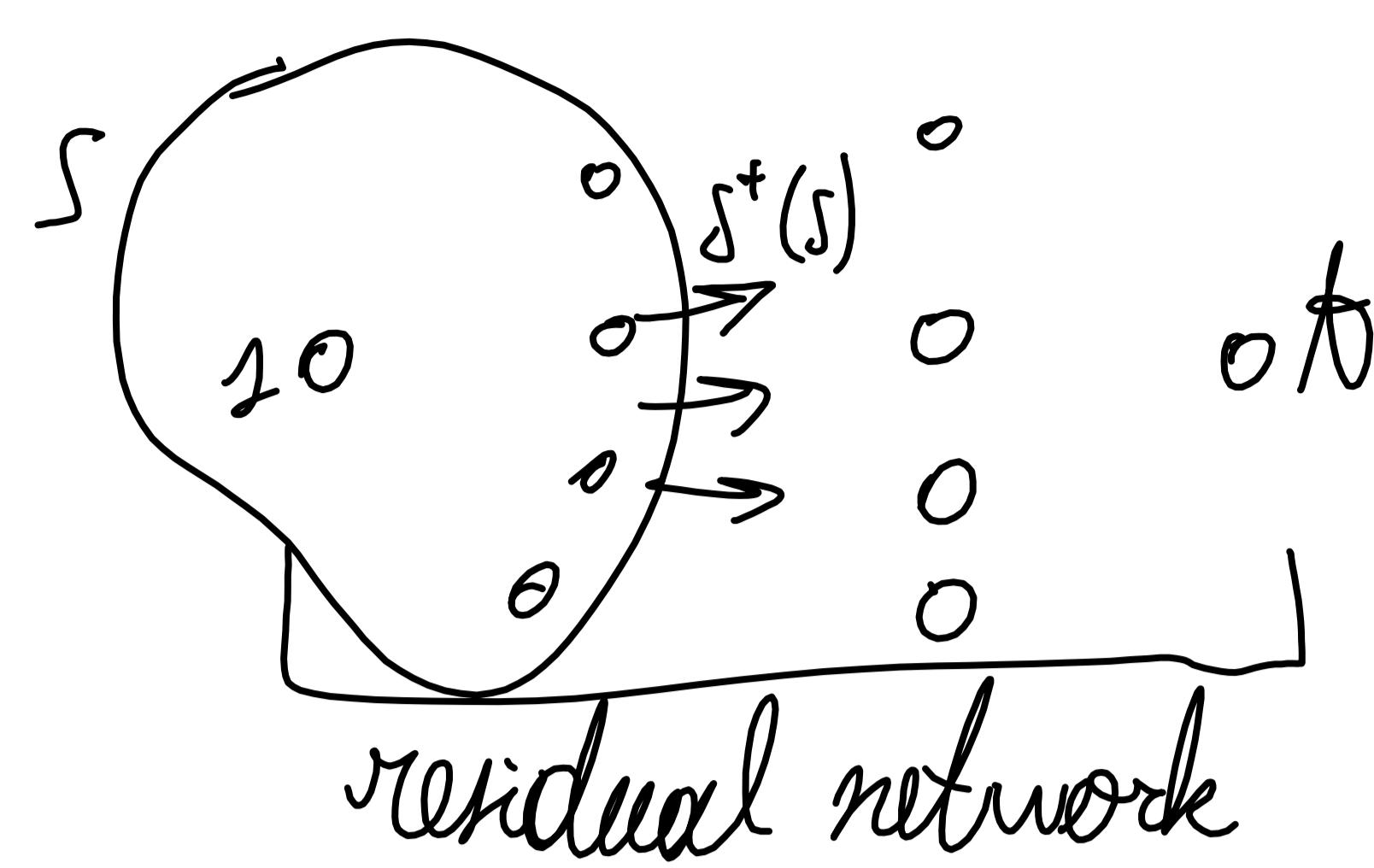
da s a t lungo P  $\Rightarrow$  faccio di più, se non esistono di archi

Se  $\delta > 0$ , ho aumentato flow

Poi cerco altro percorso e ripeto fino a quando non posso  
aumentare (t non raggiungibile in residual network)

Dimostrazione correttanza:

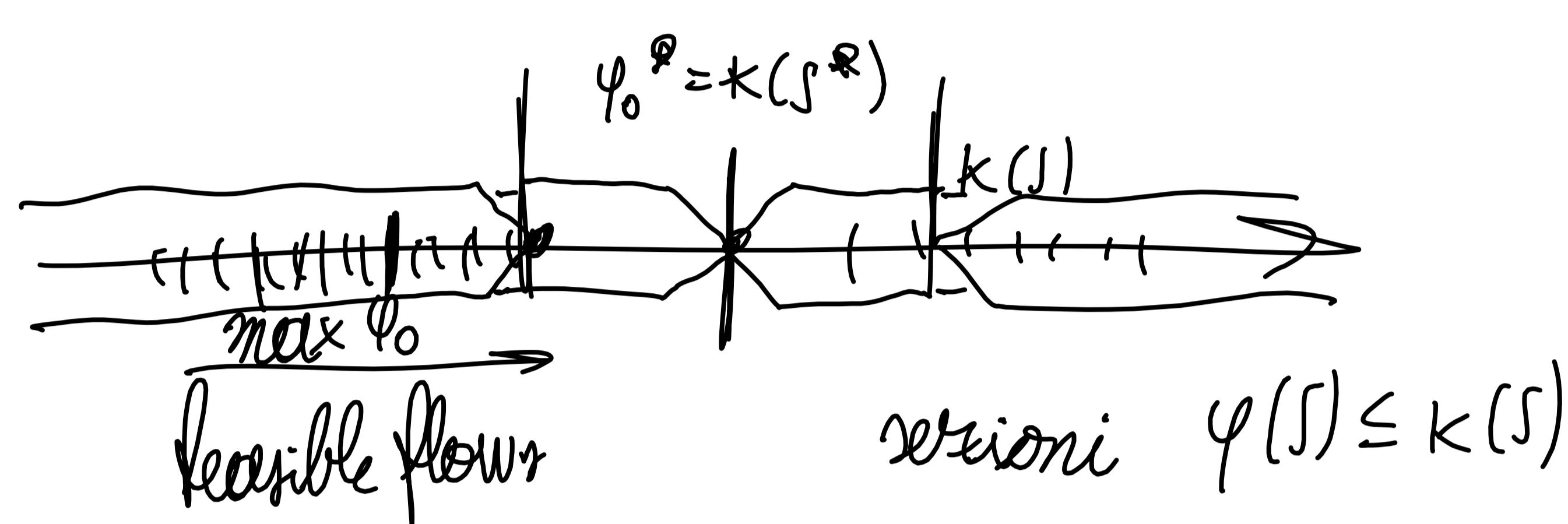
ultima it.



$\delta_{\text{def}}^+(s) = \emptyset \Rightarrow (i, j) \in \delta_{\text{def}}^+(s)$  tutti saturati

$(i, j) \in \delta_{\text{def}}^-(s)$  tutti vuoti

$$\varphi(s) = \underbrace{\sum_{(i,j) \in \delta^+(s)} x_{i,j}}_{= k_{i,j}} - \underbrace{\sum_{(i,j) \in \delta^-(s)} x_{i,j}}_{= 0} = \sum_{(i,j) \in \delta^*(s)} x_{i,j} = k(s) \quad \blacksquare$$



simile a dualità

$$\downarrow$$

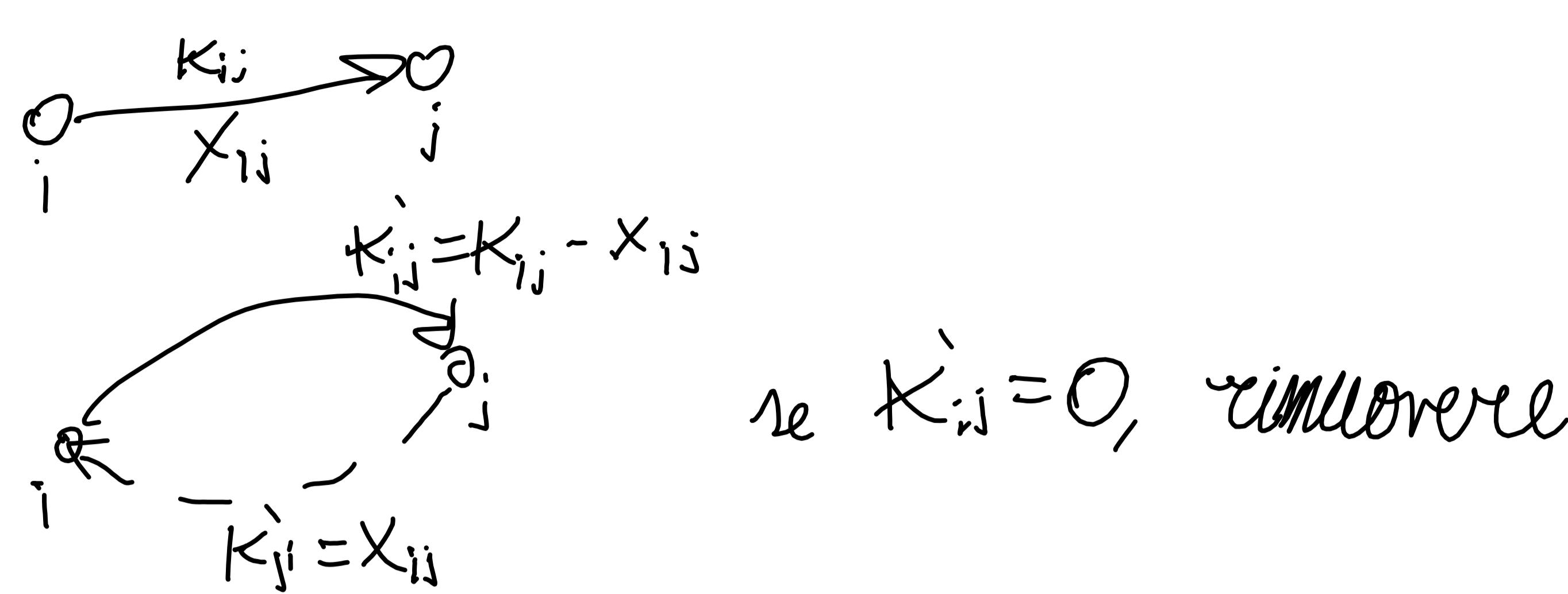
$$\max \varphi_0 = \min k(s)$$

6/12

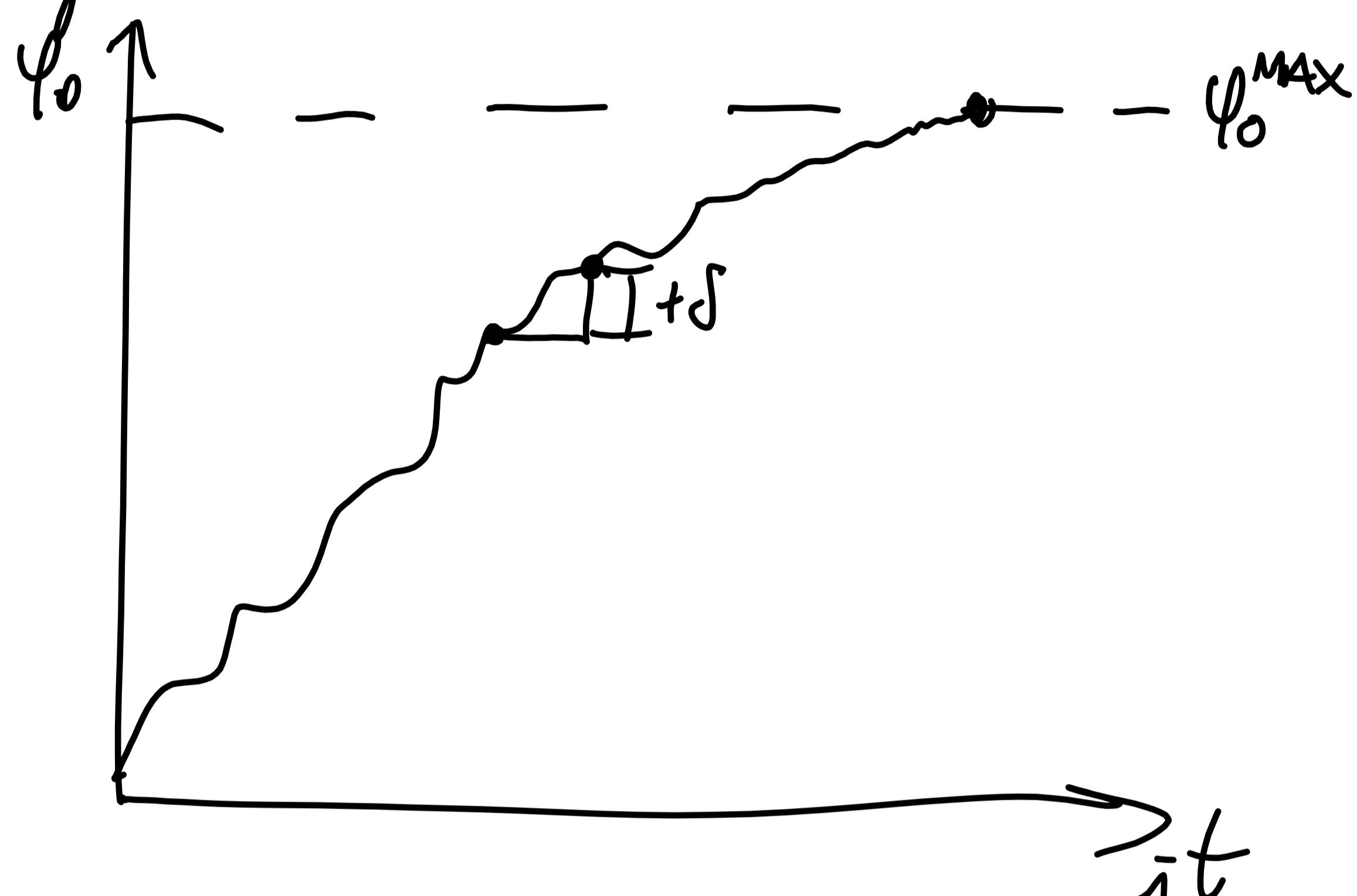
## FORD-FULKERSON

iniziiamo da

residual network



convergenza



se  $\delta \geq 1$   $\forall t$ . (dati interi),  
raggiungo max in al più  $\phi_0^{\max}$   $t$ .  
capacità razionale: può essere  
convergenza solo asintotica

Velocità

poniamo interza: capacità interi, < byte

alg. veloce con numeri piccoli

Dinic: variante di FF  $\Rightarrow$  usava politica FIFO per labeling

$\Downarrow$   
così, tempo esecuzione dipende solo  
da dim. grafo (non dim. numeri)

Serie di problemi NP-HARD: no soluzione veloce nota

KNAPSACK problem

oggetti  $1, \dots, n \Rightarrow$  peso  $w_i$ , profitto  $p_i$ ,  $\sum_{i=1}^n w_i \leq W$

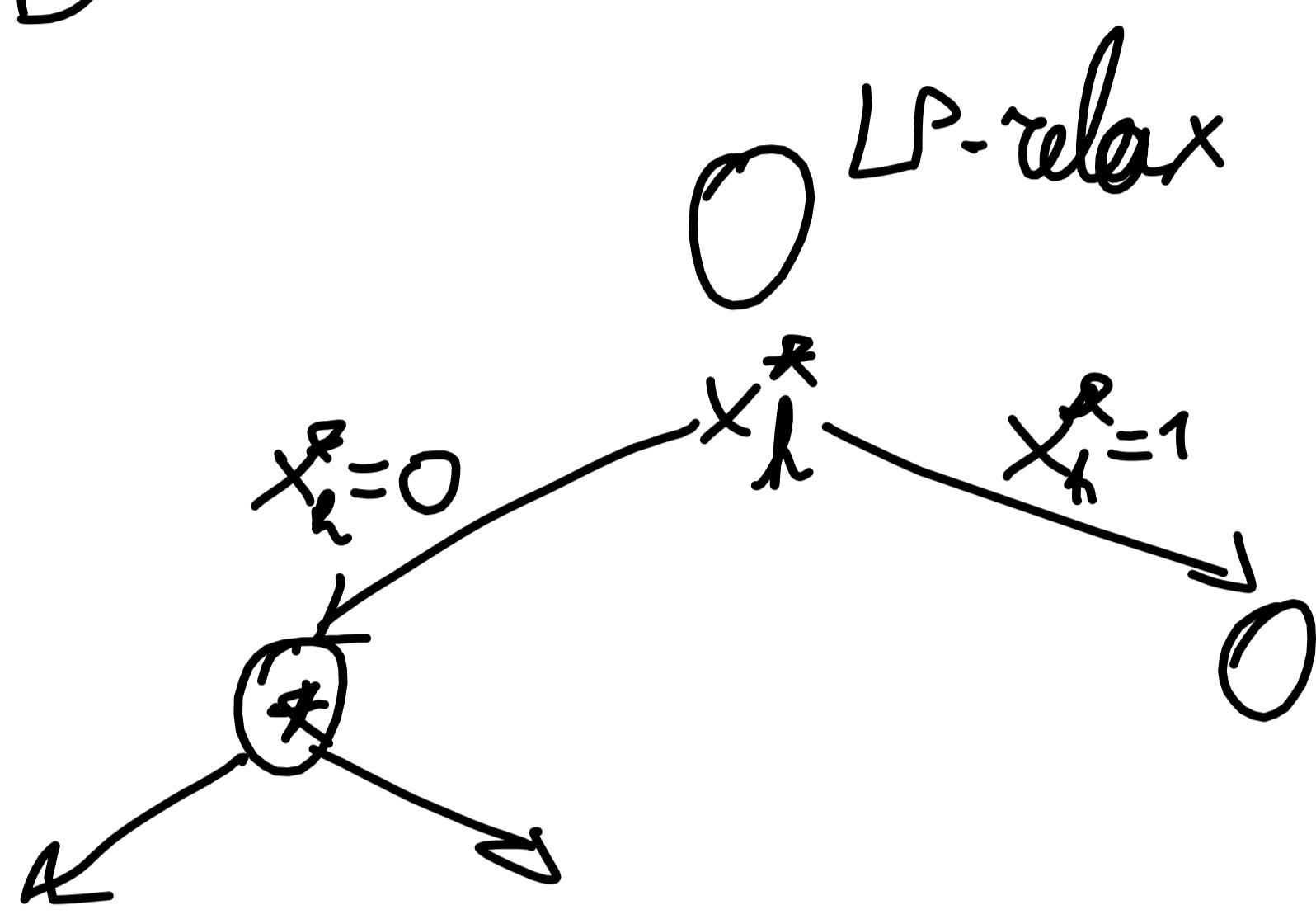
capacità contenitore  $W$

OBB:  $S \subseteq \{1, \dots, n\} \mid \sum_{i \in S} w_i \leq W$   
 $\sum_{i \in S} p_i$  massimo

Modello ILP:  $x_j = \begin{cases} 1 & \text{se oggetto } j \text{ scelto} \\ 0 & \text{else} \end{cases} \quad \forall j \in [1, n]$

$$\left\{ \begin{array}{l} \max \sum_{j=1}^n p_j x_j \\ \sum_{j=1}^n w_j x_j \leq W \\ 0 \leq x_j \leq 1 \text{ intero } \forall j \in [1, \dots, n] \end{array} \right.$$

Risolviamo con B&B



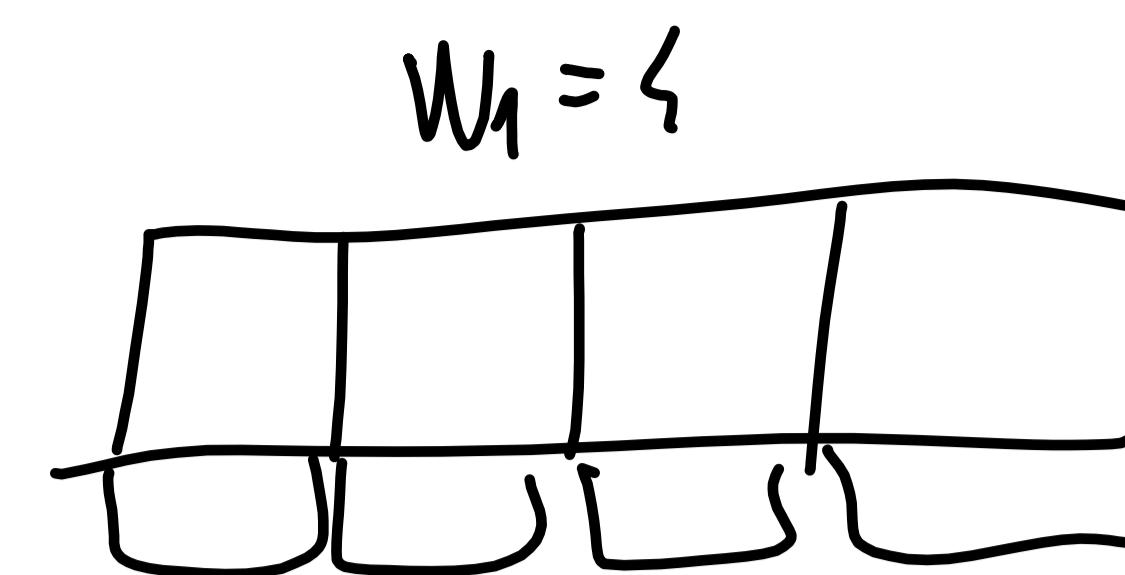
Come risolvere LP-relax?

- simplex con var. limitate
- DANTZIG algorithm  $\Rightarrow$  molto + veloce:
  - ordinare oggetti 1 volta (solo per radice)
  - $\Downarrow$  solo peggiore comunque esp.  $\Rightarrow$  rapido

11/12

DANTZIG alg.

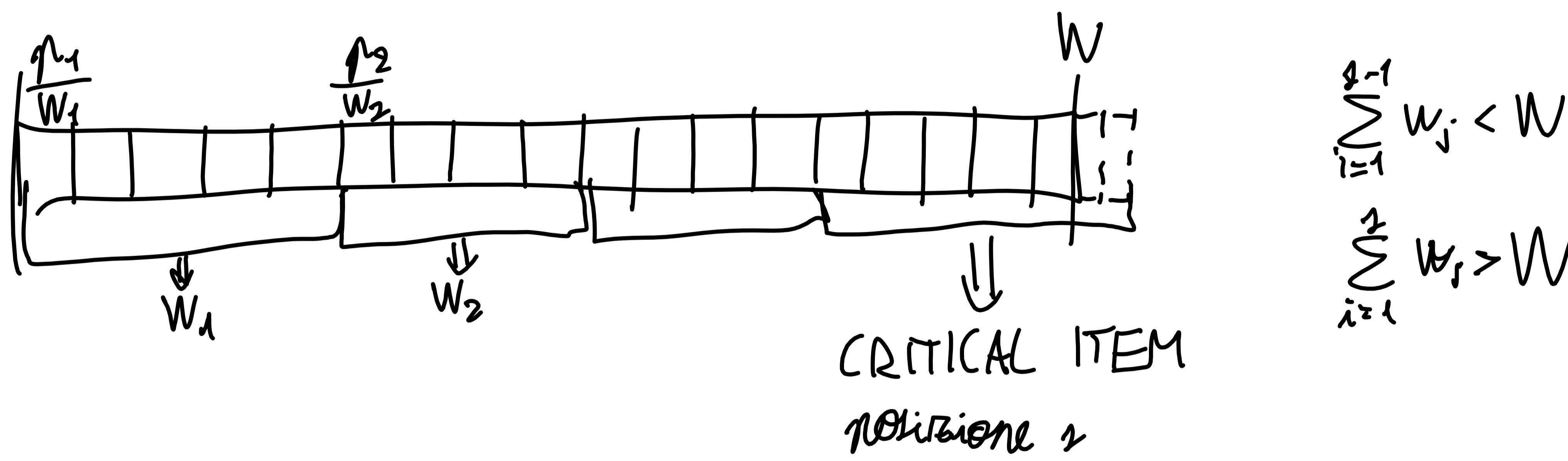
pero "compro" oggetti in preprocessing



$W_1$  MICROITEMS con peso 1  
profitter:  $n_j$  divisor in  $s$

caricherò  $N$  oggetti  $\Rightarrow$  devo scegliere migliori  $N$  oggetti  $\Rightarrow$

$\Rightarrow$  devo riordinare oggetti per sceglierli:  $\frac{v_1}{w_1} \geq \frac{v_2}{w_2} \geq \dots \geq \frac{v_n}{w_n}$



sorting:  $O(n \log n) \Rightarrow O(n)$  con PARTIAL SORTING algorithm

$x^*$  di LP-relax:  $x_1^* = \dots = x_{j-1}^* = 1, x_{j+1}^* = \dots = x_n^* = 0$ ,

$$x_j^* = \frac{W - \sum_{i=1}^{j-1} w_i}{w_j} \Rightarrow \text{solo 1 var. frazionaria}$$

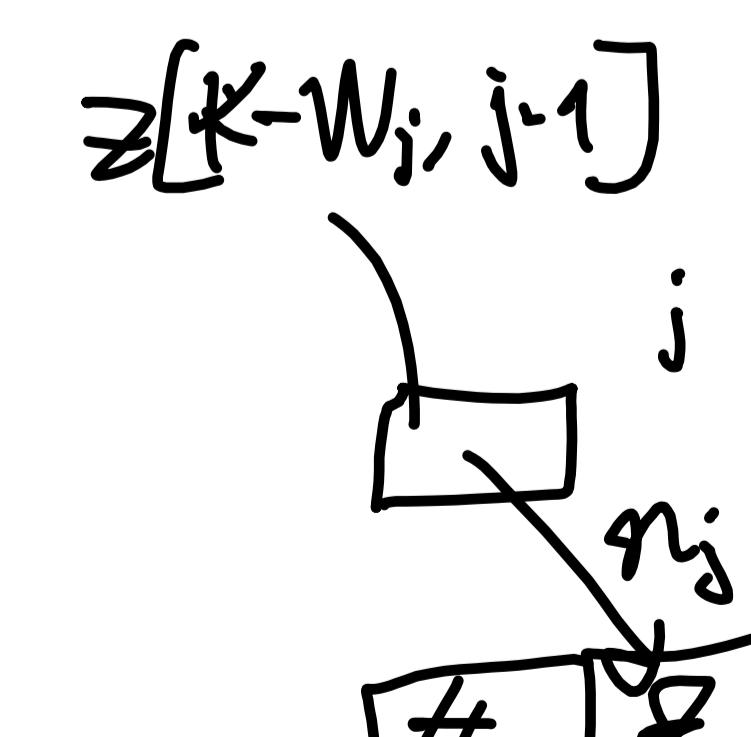
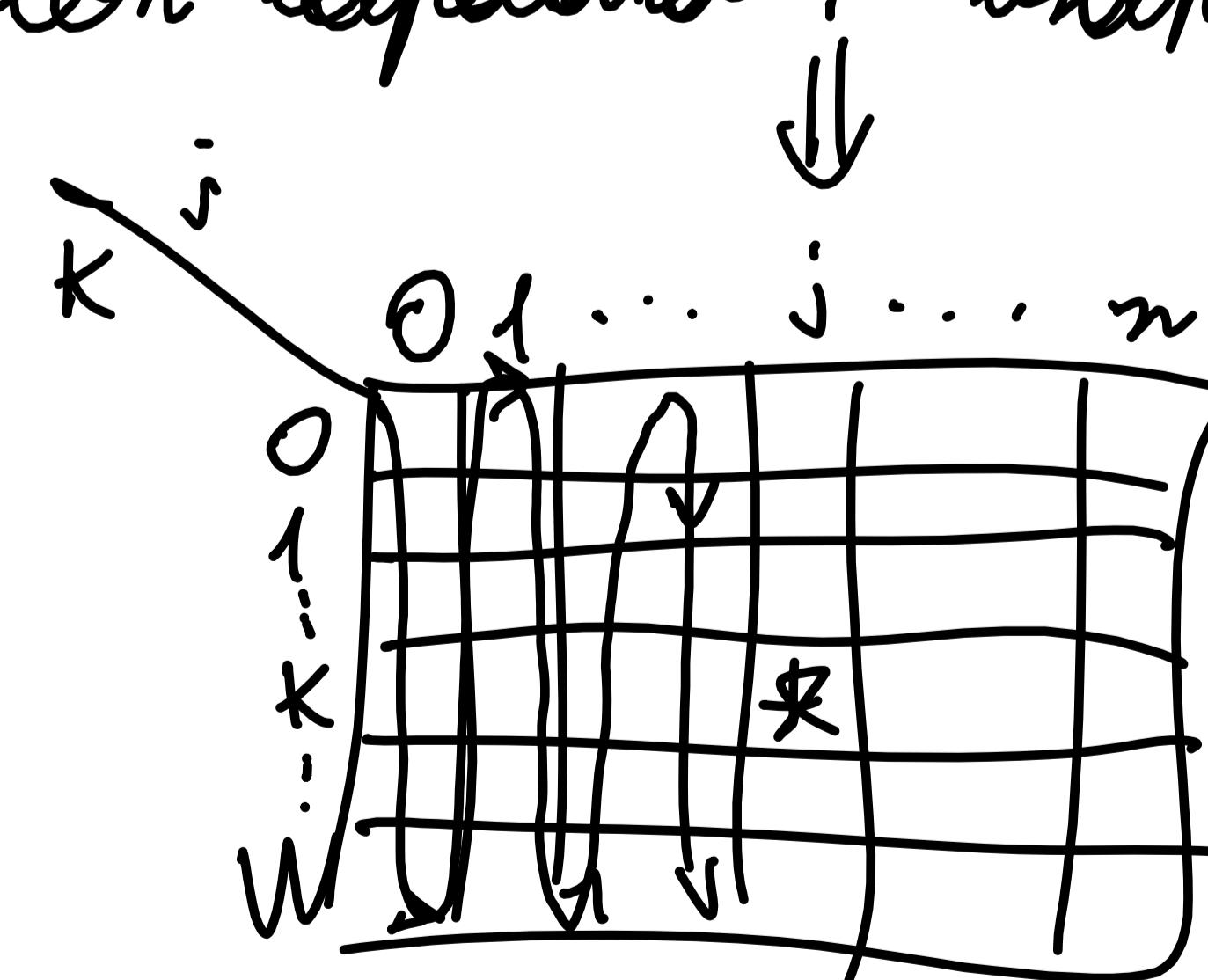
ha senso: tableau ha solo 1 riga,  
quindi solo 1 var. frazionaria

degliendo oggetto critico, notizie 2 si sposta a sx fino a  $w_{j-1}$

Per "piccolo"  $W$ , DYNAMIC PROGRAMMING

$Z[k, j] =$  valore ottimo di KP con capacità  $k$  usando solo oggetti  $[1, j]$

$$Z_{LP} = Z[W, n]$$



$$\#1 \quad \#2 \quad \#3$$

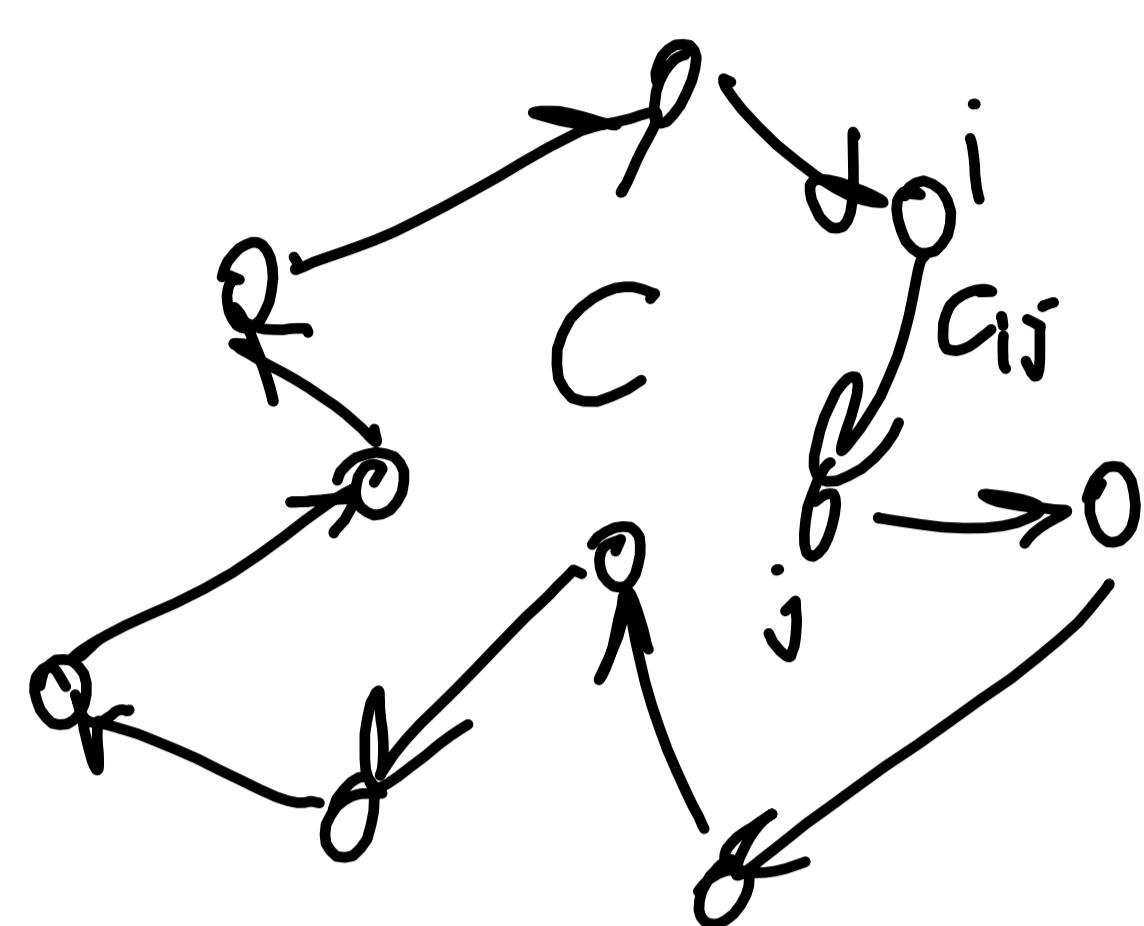
\*  $Z[k, j] = \max \left\{ \underbrace{Z[k, j-1]}_{x_j=0}, \underbrace{v_j + Z[k-w_j, j-1]}_{x_j=1} \right\} \Rightarrow$  ricorsione

tempo  $O(nW)$

indice  $K - w_j$  può essere  $< 0 \Rightarrow$  si salta valore

importante  $W$  intero  $\Rightarrow$  conviene se è piccolo

## TRAVELLING SALESMAN problem



$$\text{cost}(C) = \sum_{(i,j) \in C} c_{ij} \Rightarrow \min.$$

(circuit)  
Hamiltonian cycle (ogni nodo esattamente 1 volta)  
e non sovrapposti (visitare tutti i nodi)

branch & cut

modello

$$x_{ij} = \begin{cases} 1 & (i,j) \text{ preso} \\ 0 & \text{else} \end{cases}$$

$$G = (V, A)$$

$$\left\{ \begin{array}{l} \min \sum_{(i,j) \in A} c_{ij} x_{ij} \\ \sum_{(i,j) \in \delta^-(v)} x_{ij} = 1 \quad \forall v \in V \quad (\text{IN-DEGREE constraint}) \\ \sum_{(i,j) \in \delta^+(v)} x_{ij} = 1 \quad \forall v \in V \quad (\text{OUT-DEGREE constraint}) \\ SEC_1 \\ x_{ij} \in \{0, 1\} \quad \forall (i,j) \in A \end{array} \right.$$

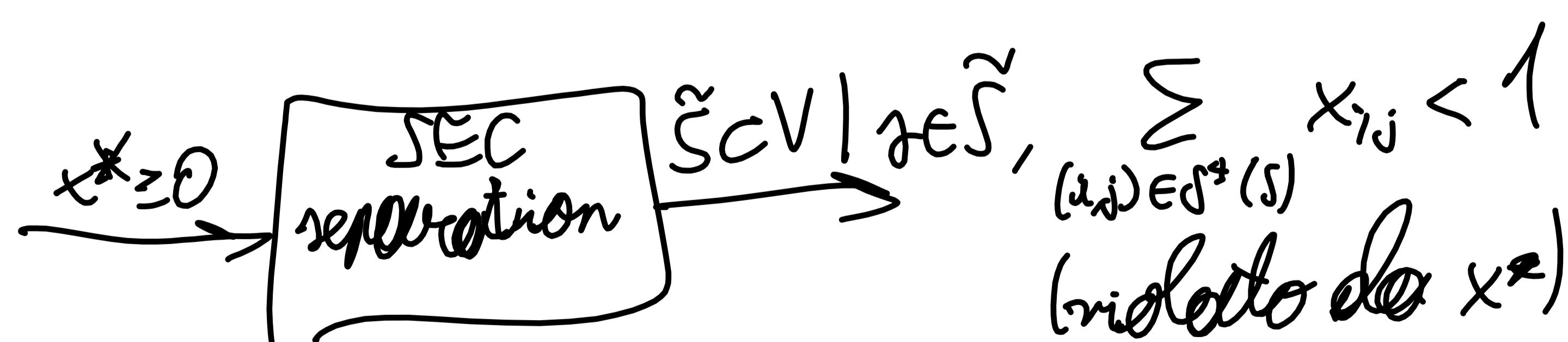
SEC: come in path problem oppure

$$\sum_{(i,j) \in \delta^-(S)} x_{ij} \geq 1 \quad \forall S \subset V \setminus \{s\} \quad (\text{CONNECTIVITY constraint})$$

(CUT-FORM SEC)

↓  
# esp. di vincoli

↓  
separazione di vincoli in B&C

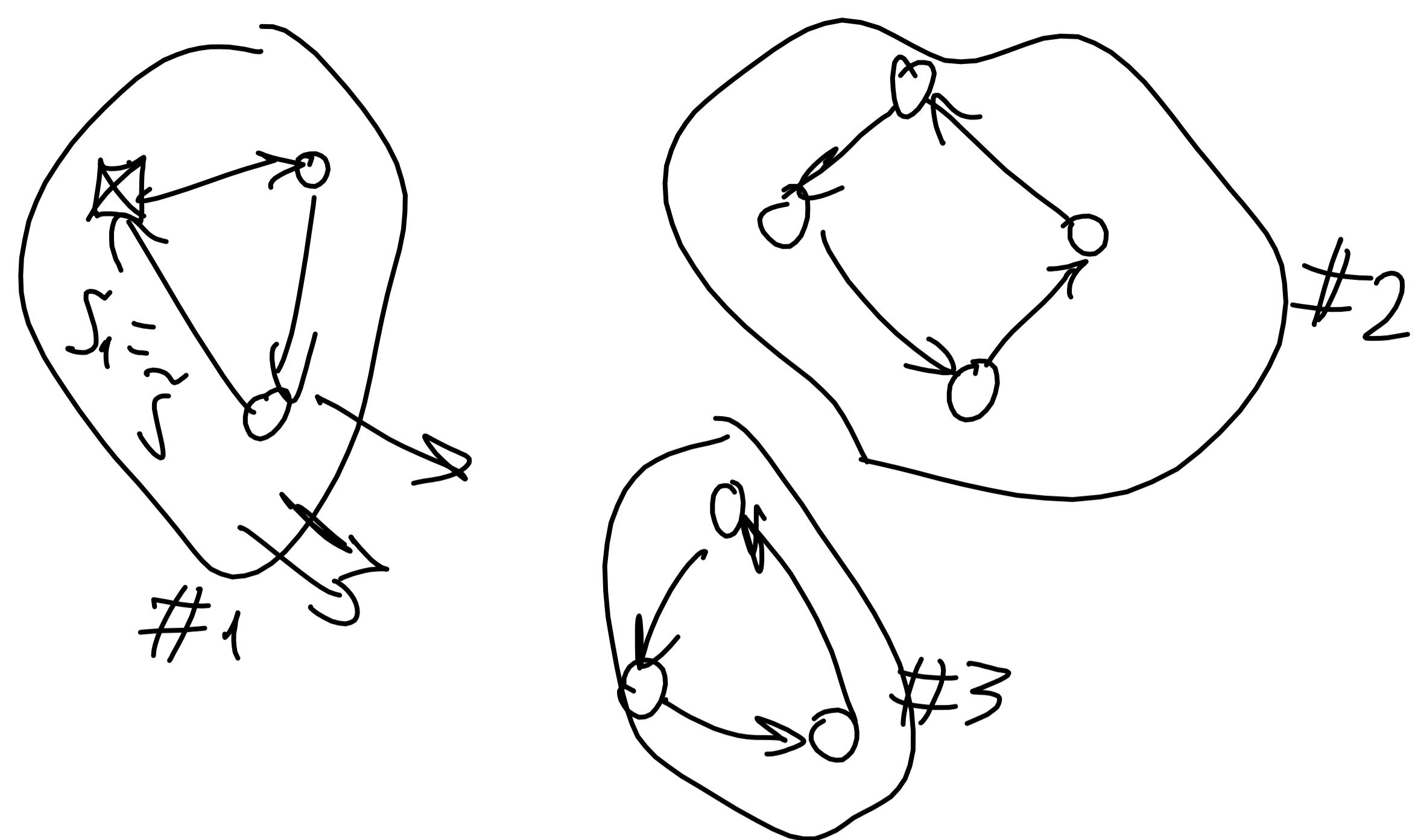


12/12

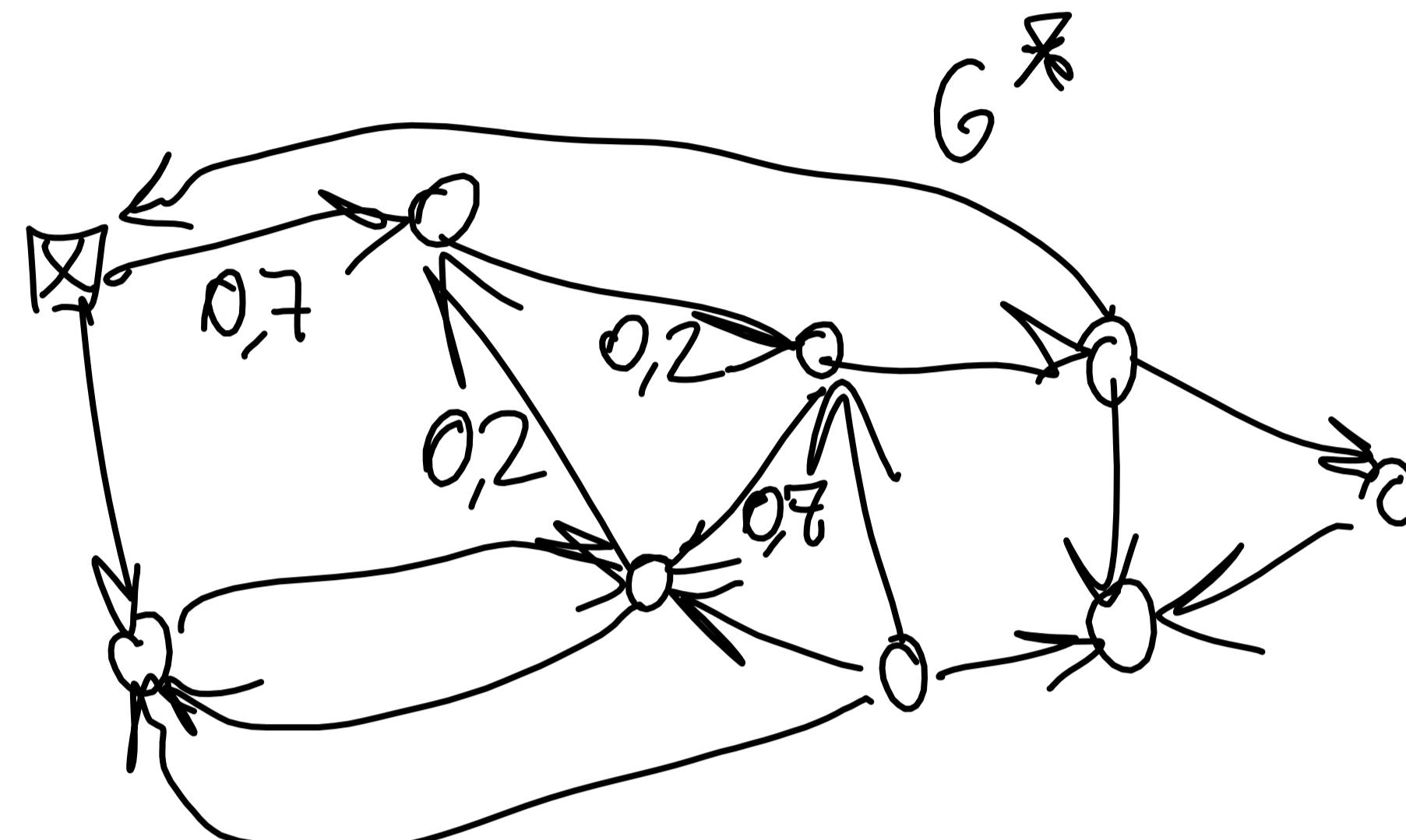
L'esi:

1 -  $x \in \text{intero} \cdot (x \in \{0,1\}^{|A|})$

me ne basta 1, ma più è meglio



2 -  $x \in \text{fraccionario}$



serve cert intelligent

sol: voglio min.  $\sum_{(i,j) \in \delta^+(j)} x_{ij} < 1 \Rightarrow \downarrow \text{lt.} + \text{violate è cert}$

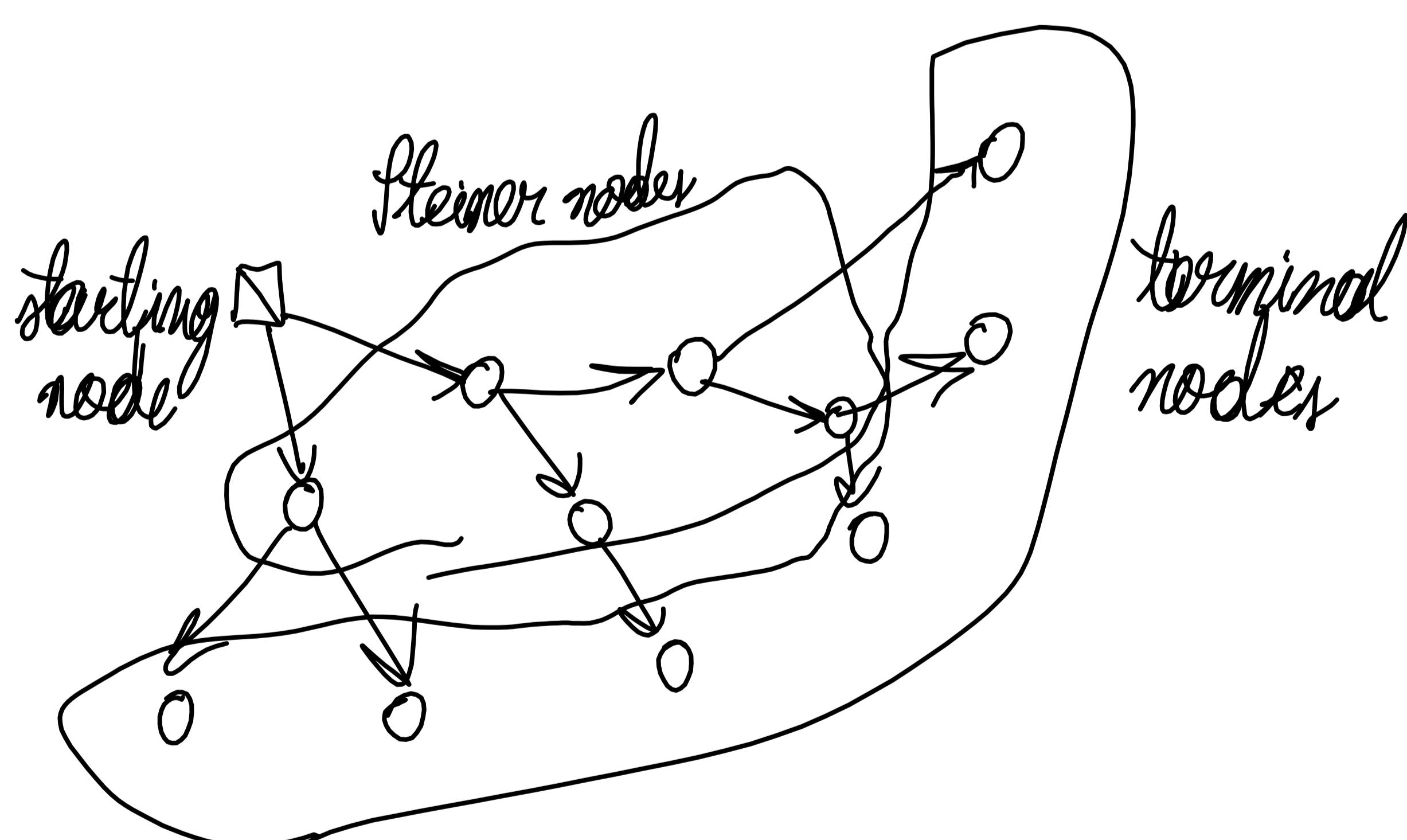
$K_{ij} = x_{ij}$  capacità su arco  $(i,j)$  (interpretazione)

otteniamo  $K(j) \Rightarrow$  problema di min-cost (max-flow)  $\Rightarrow$  Ford-Fulkerson

per FF, devo scegliere  $s, t \Rightarrow s = \text{node 1}, t = \text{ogni possibile} \Rightarrow$

$\Rightarrow$  risolvo FF  $\forall t$ , poi controllo se cost  $\equiv$  vincolo violato

## STEINER TREE PROBLEM



costi  $c_{ij} \Rightarrow$  vogliamo percorso min-cost  
non è spanning tree  $\Rightarrow$  non serve coprire  
ogni nodo  $\Rightarrow$  dobbiamo prendere ogni  
terminal node

$G = (V, A), c: A \rightarrow \mathbb{R}$

radice  $r \in V$ , terminal set  $T \subset V \setminus \{r\}$   
(di solito  $T \neq V \setminus \{r\}$ )

Modello ILP:

$$x_{ij} = \begin{cases} 1 & \text{se } (i,j) \text{ preso} \\ 0 & \text{else} \end{cases}$$

$$\left\{ \begin{array}{l} \min \sum_{(i,j) \in A} c_{ij} x_{ij} \\ \sum_{(i,j) \in \delta^-(h)} x_{ij} = \begin{cases} 0 & h = r \\ 1 & \forall h \in T \\ \leq 1 & \forall h \in V \setminus (T \cup \{r\}) \end{cases} \\ 0 \leq x_{ij} \leq 1 \quad \forall (i,j) \in A \\ \sum_{(i,j) \in \delta^+(S)} x_{ij} \geq \sum_{(i,j) \in \delta^-(t)} x_{ij} \quad \forall S \subseteq V \setminus \{r\}, \forall t \in V \setminus S \quad (\text{connectivity constraint}) \end{array} \right.$$

Funzione di separazione

$$x^* \geq 0 \rightarrow \boxed{\text{---}} \xrightarrow{\tilde{s}, f} \forall \tilde{s}, \forall \tilde{t} \in V \setminus \tilde{S}$$

$$\sum_{(i,j) \in \delta^+(\tilde{S})} x_{ij}^* \leq \sum_{(i,j) \in \delta^-(\tilde{t})} x_{ij}^*$$

Interpretazione reale

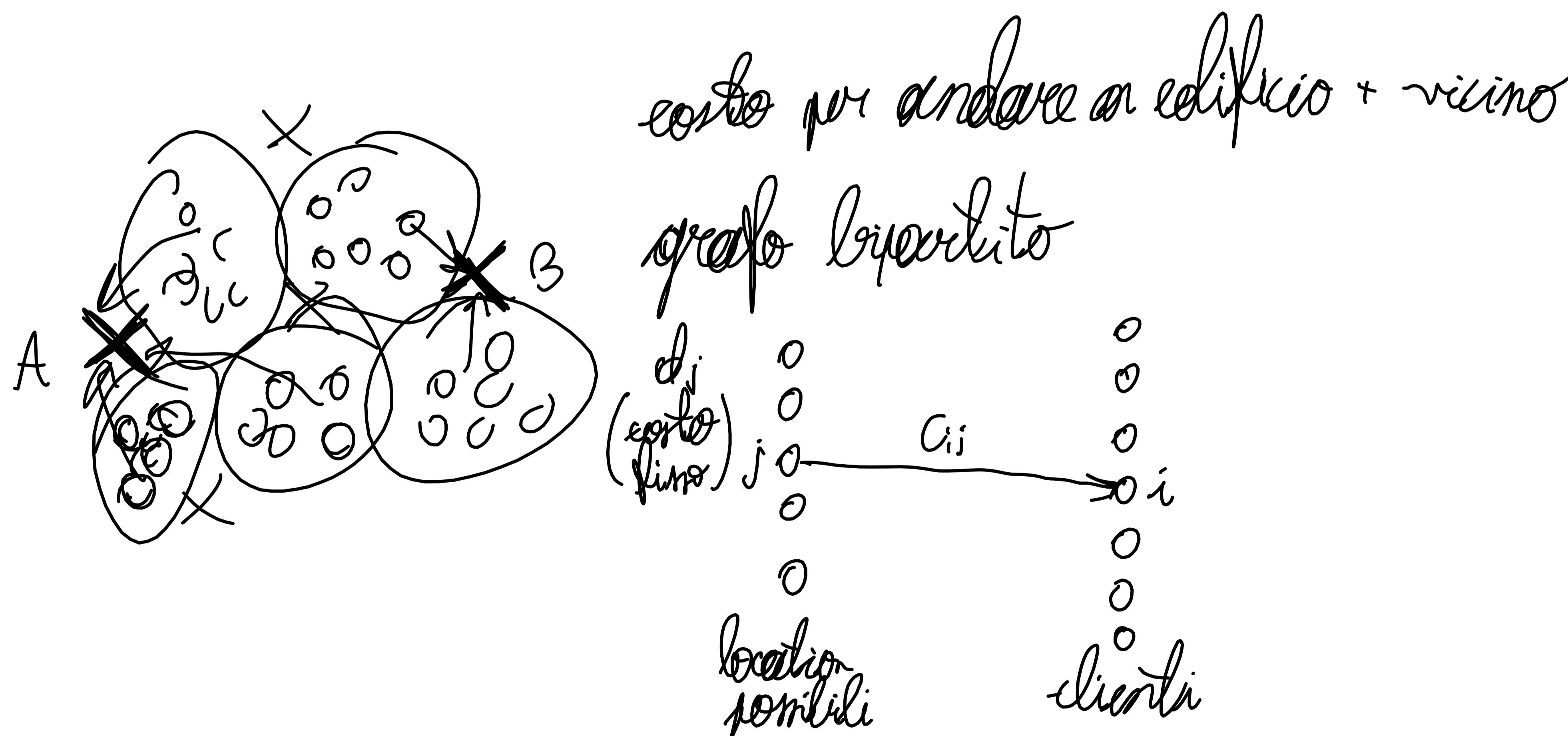
$$j = v$$

$$\emptyset = 1, \dots, \emptyset, \dots, n$$

$$z \cdot \begin{array}{c} \xrightarrow{\hspace{1cm}} \\ \max\text{-flow} \end{array} \cdot b$$

Danno max. capacity wif  
con  $k_{ij} = x_{ij}^*$

PLANT LOCATION PROBLEM



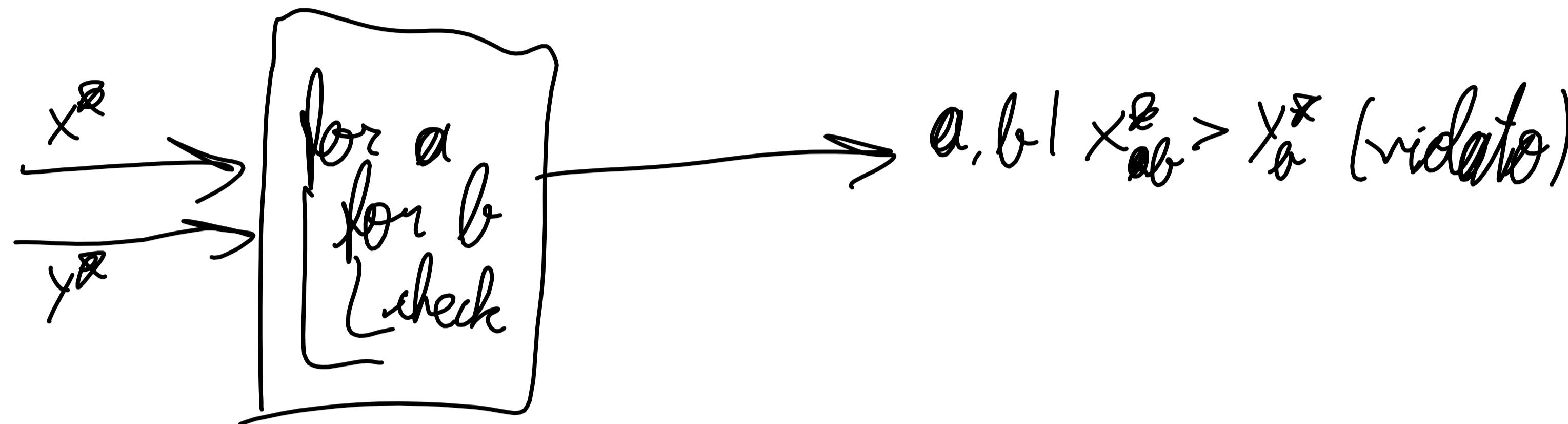
## ILP modello

$$Y_j = \begin{cases} 1 & \text{se porto } j \text{ velto} \\ 0 & \text{else} \end{cases} \quad \forall j \in [1, m]$$

$$X_{ij} = \begin{cases} 1 & \text{se utente } i \text{ connesso al porto } j \\ 0 & \text{else} \end{cases}$$

$$\forall i \in [1, n], \forall j \in [1, m]$$

$$\left\{ \begin{array}{l} \min \sum_{j=1}^m d_j Y_j + \sum_{i=1}^n \sum_{j=1}^m c_{ij} X_{ij} \\ \sum_{j=1}^m X_{ij} = 1 \quad \forall i \in [1, n] \\ 0 \leq X_{ij} \leq 1 \quad \text{int} \forall i, j \\ 0 \leq Y_j \leq 1 \quad \text{int} \forall j \\ X_{ij} \leq Y_j \quad \forall i, j \end{array} \right.$$



13/12

## SET COVERING

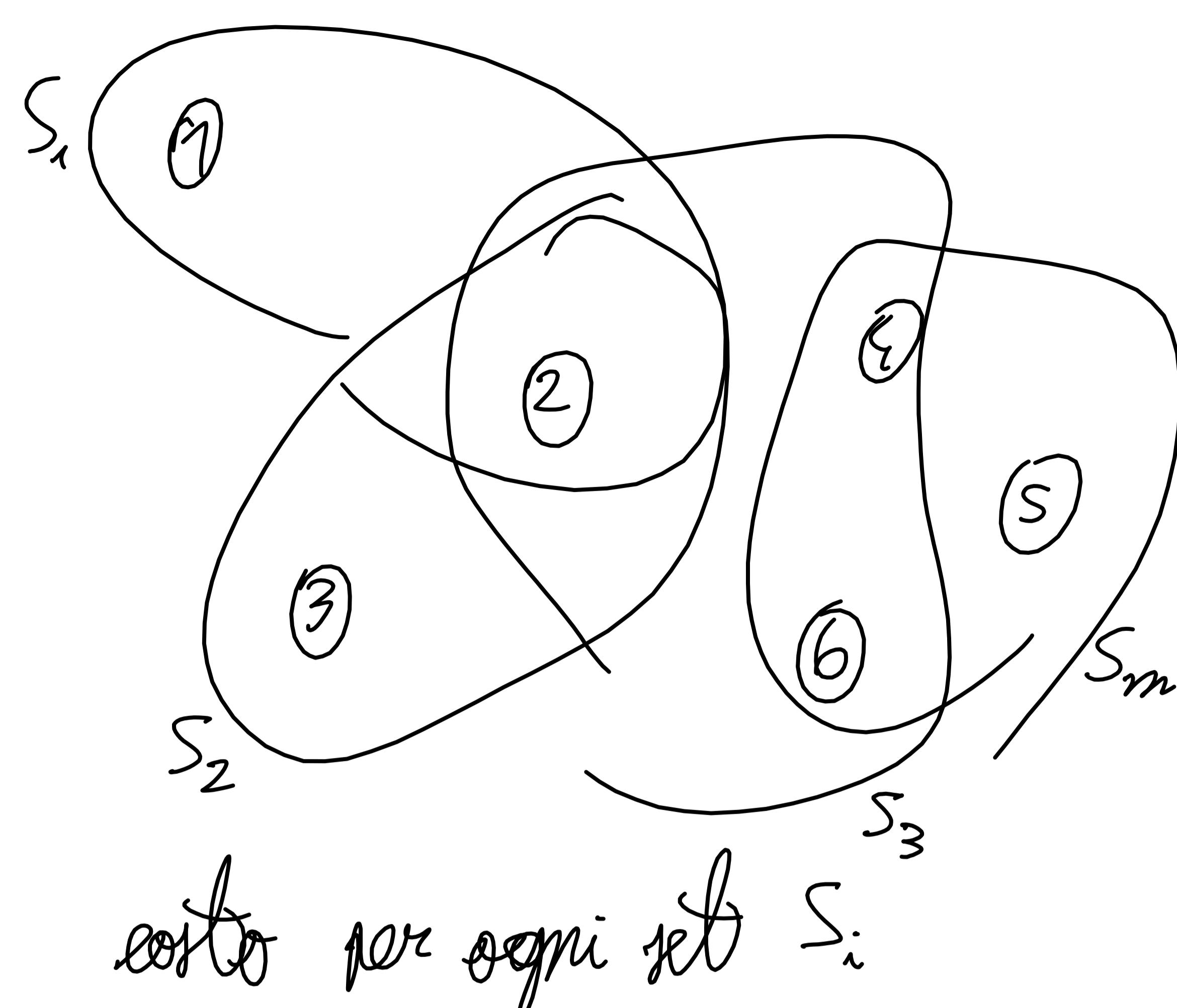
$$\begin{cases} \min c^T x \\ Ax \geq 1 \\ x \in \{0,1\}^n \\ \text{con } A \in \{0,1\}^{n \times m} \end{cases}$$

voglio coprire tutti gli elementi con sottoinsiemi disponibili (cachesi in matrice A)

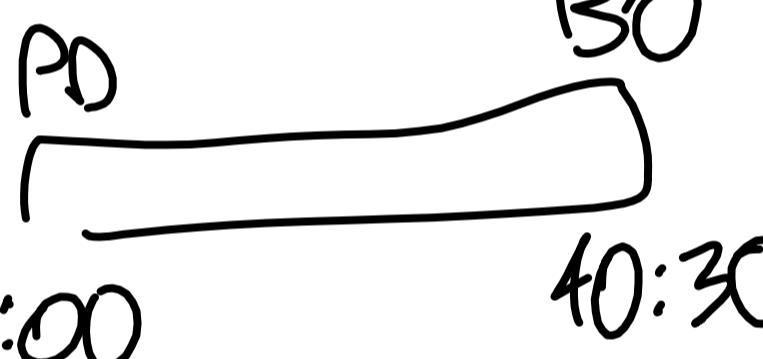
$$x_j = \begin{cases} 1 & \text{se } S_j \text{ scelto} \\ 0 & \text{else} \end{cases}$$

$$\begin{cases} \min \sum_{j=1}^n c_j x_j \\ \sum_{j=1}^n a_{ij} x_j \geq 1 \quad \forall i \in [1, n] \\ x_j \in \{0,1\} \quad \forall j \in [1, n] \end{cases}$$

(= voglio che 1 set per 1 el.: partizioni)



E. applicazione: crew scheduling (e.g. bus driver)

corso: vettore 

# corsi  $\approx 5000$

Averni: sequenza di corsi con certe condizioni  $\Rightarrow$  operai ha corso assegnare turni legali

difficile esprimere regole in modello  $\Rightarrow$  usiamo set covering, dove set sono turni  $\Rightarrow$   $\Rightarrow$  prima fase dove genero turni come percorsi in grafo, poi risolvo problema

## MODELING TRICKS

## BIG-M CONSTRAINTS

$a_i^T x_i \geq b_i \Rightarrow$  disug. valida solo in certi momenti

un metodo: ACTIVATION VARIABLE: " $x_i = 0 \Rightarrow a_i^T x_i \leq b_i$ "

$$a_i^T x_i \geq b_i - M y_i, \quad M \gg 0$$

$$- \text{ se } y_i = 0 \Rightarrow a_i^T x_i \geq b_i$$

$$- \text{ se } y_i = 1 \Rightarrow a_i^T x_i \geq b_i - M \approx -M \Rightarrow -M \text{ molto piccolo} \Rightarrow \text{vincolo disattivato}$$

Problemi:

- stabilità numerica
- sol. frazionario (B&B/C)





































