# APPROXIMATIO ALGORITHMS

## MIN VERTEX COVER

- MINIMO INSIEME DI NODI CHE TOCCA TUTTI GLI ARCHI

APPROX-COVER ALGO {

$M = \{\emptyset\}$ //ARCHI
$U = \{\emptyset\}$ //NODI
WHILE $E \neq \{\emptyset\}$
  SELEZIONA $(u,v) \in E$
  $U = U \cup \{u, v\}$
  $E = E \setminus \{e \in E$ INCIDENTE SU $U\}$
  $M = M \cup \{(u,v)\}$
RET $U$

}

LEMMA
- M FORMA UN MATCHING

Proof BANALE, OGNI ARCO AGGIUNTO RIMUOVE OGNI ARCO ADIACENTE □

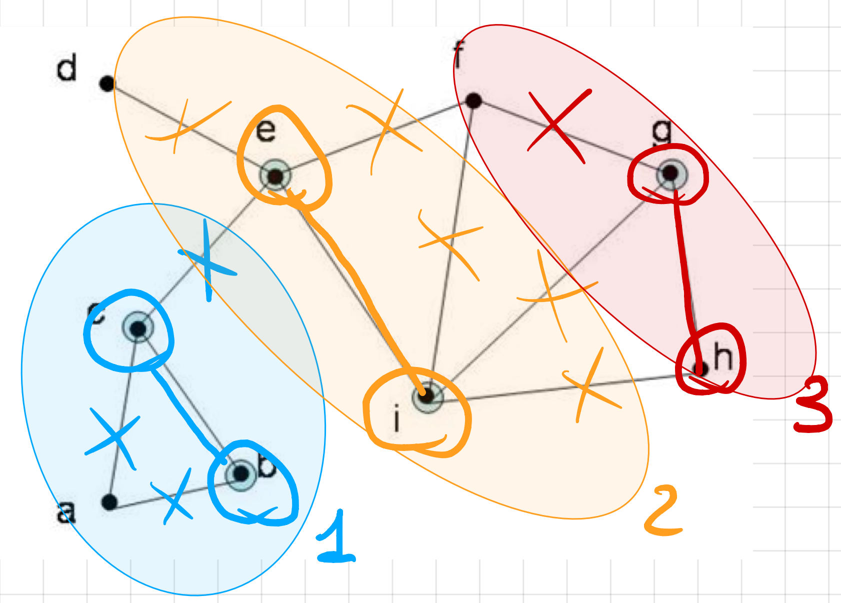
TH
- 2-APPROSSIMATIVO

Proof $m = |U| = 2 \cdot |M|$ → DUE NODI PER OGNI ARCO DI M

$|U^*| \geq |M|$ → SE $U^*$ OTTIMO, OGNI ARCO DI M DEVE AVERE ENDPOINT IN ESSO

$\Rightarrow m^* = |U^*| \geq |M| \Rightarrow m / m^* \leq 2 \cancel{|M|} / \cancel{|M|} = 2$ □

esempio




$\Rightarrow U = \{b, c, e, i, g, h\}$

# GREEDY

## MAX 0-1 KNAPSACK

IN

| | |
|---|---|
| $O$ | SET DI OGGETTI |
| $p_i$ | PROFITTO |
| $a_i$ | VOLUME |
| $b > 0$ | INTERO |

SOL $Q \subseteq O$ t.c. $\sum_{o_i \in Q} a_i \leq b$

MEASURE PROFITTO TOT. $\sum_{o_i \in Q} p_i$

## GREEDY KNAPSACK ALGO {

$Q = \{\emptyset\}$ // SOTTOINSIEME SCELTO

$V = \emptyset$ // VOLUME TOT. DI $Q$

ORDINA GLI $o_i$ IN ORDINE DECRESCENTE DI $p_i/a_i$ RINOMINANDOLI

FOR $i = 1, \dots, n$

  IF $V_i + a_i \le b$

    $Q = Q \cup \{o_i\}$

    $V_i = V_i + a_i$

RET $Q$

}

**TH** • PER OGNI $r < 1 \Rightarrow$ L'ALGO. NON $r$-APPROSSIMATIVO

Proof

• ASSUNTO $r < 1$

• SIA $k = \lceil 1/r \rceil$, $\forall n \ge 2$ CONSIDERA L'ISTANZA:

$b = kn$

$n-1$ OGGETTI CON $p_i = 1$, $a_i = 1$

$1$ OGGETTO CON $p_i = b-1$, $a_i = b$

• SOL: PRIMI $m = n-1$ OGGETTI

• SOL*: SOLO $o_n$, $m^* = b-1 = kn-1$

$$\Rightarrow \boxed{\frac{m}{m^*}} = \frac{n-1}{k \cdot n - 1} \le \frac{n-1}{\frac{n}{r} - 1} < \frac{n-1}{\frac{n}{r} - \frac{1}{r}} = \frac{n-1}{\frac{1}{r}(n-1)} = \boxed{r}. \quad \bot$$

$\frac{m}{m^*} = r$

$1/r > 1$

□

## MODIFIED GREEDY KNAPSACK - ALGO {

COMPUTA $Q_{GR}$ CON $m_{GR}$ USANDO GREEDY KNAPSACK

IF $Q_{GR} \ge p_{max}$

  RET $Q_{GR}$

ELSE

  RET $Q_{GR} = \{o_{max}\}$

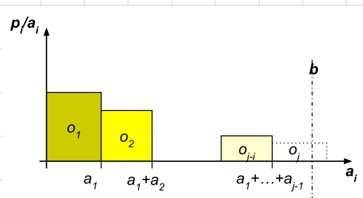
}

**LEMMA 1** (UPPER BOUND) • SIA $o_j$ IL PRIMO OGGETTO A ESSERE LASCIATO FUORI E

SIA $m = \sum_{i=1}^{j-1} p_i$

$\Rightarrow m^* \le m + p_j$

Proof




□

LEMMA 2 • $m^* \le m_{GR} + P_{max}$

Proof $m^* \le \underbrace{m_J}_{\le m_{GR}} + \underbrace{P_J}_{\le P_{max}} \le m_{GR} + P_{max}$

LEMMA 1

□

TH • 1/2 - APPROSSIMATIVO

Proof $\boxed{m_{Mo}} \ge \max(m_{Gr}, p_{max}) \ge (m_{Gr} + p_{max})/2 \boxed{\ge m^*/2}$

SOL MODIFIED GREEDY

□

---

MIN MULTIPROCESSOR SCHEDULING

IN

$m$ | JOB $P = \{P_1, \_, P_m\}$

$h$ | PROCESSORI

$t_J$ | RUNNING TIME DI $P_J$

SOL SCHEDULE $f: P \to \{1, \_, h\}$

MEASURE

MAKESPAN $\max_{i \in 1 \_ h} \sum_{P_J \in P \text{ t.c. } f(P_J) = i} t_J$

GREEDY-GRAHAM ALGO. {

FOR $J = 1, \_, m$

ASSEGNA $P_J$ AL PROCESSORE $i$ CON MINIMO $T_i(J-1)$

RET $f$

}

TH • $(2 - 1/h)$ - APPROSSIMATIVO

Proof • FACT: DATI $a_1 + \_ + a_m = S \Rightarrow \exists J, J'$ t.c. $a_J \ge S/m$, $a_{J'} \le S/m$

• OPT* $T_1^*, \_, T_h^*$ i COMPLETITION TIME

$\Rightarrow T_1^* + \_ + T_h^* = T \Rightarrow \exists\, T_J^* \ge T/h$ (FACT) $\Rightarrow \boxed{m^*} \ge T_j^* \boxed{\ge T/h.}$

• SOL

• SIA $T_k(m)$ IL MASSIMO COMP. TIME

• SIA $P_l$ LAST JOB AGGIUNTO AL PROCESSORE $k$

$\Rightarrow$ $T_k(l-1) \le (\Sigma_{j<l}\, t_j) / h \le (T - t_l) / h,$

• $\boxed{m} = \boxed{T_k(n)} = T_k(l-1) + t_l \le \frac{T - t_l}{h} + t_l = \underbrace{\frac{T}{h}}_{\le m^*} + \left(\frac{h-1}{h}\right) \cdot \underbrace{t_l}_{\le m^*} \le m^* + \frac{h-1}{h} \cdot m^* = \boxed{\left(2 - \frac{1}{h}\right) \cdot m^*}$

$$\Rightarrow \boxed{\frac{m}{m^*} \le 2 - \frac{1}{h}}$$

□

---

ORDERED GREEDY ALGO {

$P_1, \_, P_m$ IN ORDINE DECRESCENTE DI $t_i$

FOR $J = 1, \_, m$

ASSEGNA $P_J$ AL PROC. CON MINORE $T_i(J-1)$

RET $f$

}

**LEMMA** (LOW BOUND)

• SE $m > h$ ALLORA $t_{h+1} \le m^*/2$

Proof • I PRIMI $h$ JOB HANNO $t_i \ge t_{h+1}$

• IL JOB $h+1$ VA ASSEGNATO A CHI GIÀ HA UN JOB

$$\Rightarrow m^* = \underbrace{t_i}_{\ge t_{h+1}} + t_{h+1} \ge 2t_{h+1}$$

□

**TH** • $\left(\frac{3}{2} - \frac{1}{2h}\right)$-APPROSSIMATIVO

Proof • SIA $k$ IL PROC. PIÙ CARICO

• SE $k$ HA UN SOLO JOB $\Rightarrow$ SOL = OPT

• SENNO:

SIA $p_q$ L'ULTIMO JOB ASSEGNATO A $k$

ALLORA $q \ge h+1$ (NON ESSENDO IL PRIMO ASSEGNATOGLI)

$\Rightarrow t_q \le t_{h+1} \le m^*/2$

$$\Rightarrow \boxed{m} \le \frac{T}{h} + \left(\frac{h-1}{h}\right)\cdot t_l \le m^* + \left(\frac{h-1}{h}\right)\cdot \frac{m^*}{2} = \boxed{\left(\frac{3}{2} - \frac{1}{2h}\right)m^*}$$

□

## MAX CUT

**IN** $G = (V, E)$

**SOL** PARTIZIONE DI $V$ IN $V_1, V_2$

**MEASURE** $|\{(u,v)$ t.c. $u \in V_1, v \in V_2\}|$

**GREEDY MAX CUT ALGO** {

$V_1 = \{\emptyset\}$

$V_2 = \{\emptyset\}$

FOR $i = 1, \dots, m$

$\Delta_i = \{(i,j)$ t.c. $j < i\}$ // ARCHI TRA $i$ E I NODI $j$ GIÀ VISTI

$U_i = \{j$ t.c. $(i,j) \in \Delta\}$ // NODI $j$ GIÀ VISTI E ADIACENTI A $i$

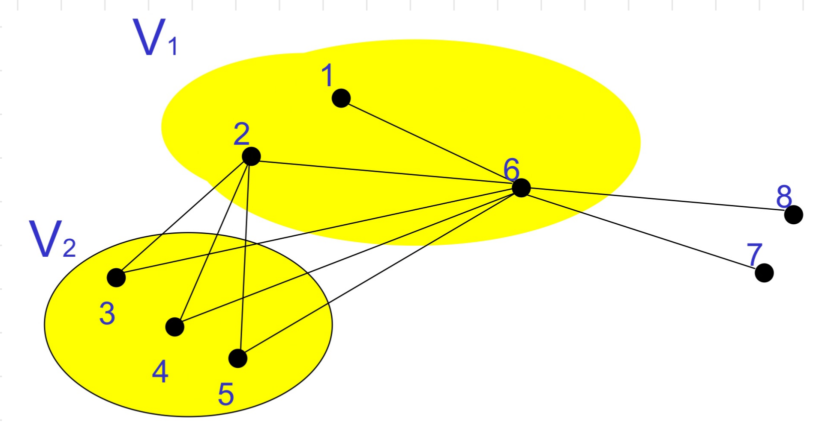
$\delta_i = |\Delta_i| = |U_i|$

$\delta_{1i} = |V_1 \cap U|$

$\delta_{2i} = |V_2 \cap U|$

IF $\delta_{1i} > \delta_{2i}$

$V_2 = V_2 \cup \{i\}$




ELSE

$V_1 = V_1 \cup \{i\}$

RET $V_1, V_2$

}

**TH** • $1/2$-APPROSSIMATIVO

Proof • $m^* \le |E|$ OBV.

• $\sum_{j=1}^{m} \delta_j = \sum_{i=1}^{m} |\Delta_i| = |E|$ (RICORDA CHE $\Delta_i$ HA SOLO $(i,j)$ CON $i < j$)

• #ARCHI AGGIUNTI A UNO STEP $i$ È:

$\max(\delta_{1i},\delta_{2i}) \geq (\delta_{1i}+\delta_{2i})/2=\delta_i/2,$

$\Rightarrow \boxed{m} = \sum_{i=1}^{n} \max(\delta_{1i},\delta_{2i}) \geq \sum_{i=1}^{n} \frac{\delta_i}{2} = \boxed{\frac{|E|}{2}}$

$\Rightarrow \frac{m}{m^*} \geq \frac{\frac{|E|}{2}}{|E|} = \boxed{\frac{1}{2}}$ $\square$

# LOCAL SEARCH

## MAX CUT

LOCAL SEARCH MAX CUT ALGO. {

• vedi sopra

INITIAL SOL.: $V_1 = V,\ V_2 = \{\emptyset\}$

NEIGHBOURS SOLS: $N((V_1,V_2))$ SONO LE COPPIE $(V_{1i},V_{2i})$ OTTENIBILI SPOSTANDO UN NODO DA UNO ALL'ALTRO.

}

COMPLEXITY

| | |
|---|---|
| • INIT. SOL. | $O(n)$ |
| • TROVARE $n$ VICINATI, CONTROLLO SE È MIGLIORE | $O(n)O(n^2) = O(n^3)$ |
| • #ITERAZIONI (OGNUNA ADD. ALMENO UN ARCO A SOL) | $\lvert E \rvert = O(n^2)$ |

$\Rightarrow O((n+n^3)\,n^2) = O(n^5)$

TH

• 1/2-APPROSSIMATIVO

Proof • FACT DATO $G=(V,E) \Rightarrow \sum_{s=1}^{n} \delta_s = 2|E|$

• DATO UN OTTIMO LOCALE CON $h$ ARCHI INTERNI

↳ $m + h = |E|$ OBV.

• DATO CHE $\underbrace{m((V_{1i},V_{2i}))}_{\text{QUALSIASI VICINO DEL LOCAL OPT}} \leq \underbrace{m((V_1,V_2))}_{\text{LOCAL OPT}}$

↳ $m - \delta_i^{ext} + \delta_i^{int} \leq m$ and thus $\delta_i^{int} - \delta_i^{ext} \leq 0$

• PER TUTTI I NODI

↳ $\underbrace{\sum_{v_i \in V} \delta_i^{int}}_{} - \underbrace{\sum_{v_i \in V} \delta_i^{est}}_{} \leq 0$

$\sum_{v_i \in V} \delta_i^{int} = 2h \qquad \sum_{v_i \in V} \delta_i^{est} = 2m$

$\Rightarrow 0 \geq 2h - 2m \Rightarrow 2m \geq 2h \Rightarrow m \geq h$

$\Rightarrow \frac{m+m}{2} \geq \frac{h+m}{2} \Rightarrow \boxed{m} \geq (m+h)/2 = \boxed{|E|/2}.$

$\Rightarrow \boxed{\frac{m}{m^*}} \geq \frac{\frac{|E|}{2}}{|E|} = \boxed{\frac{1}{2}}$ $\square$

$m^* \leq |E|$

# LINEAR PROGRAMMING

## MIN WEIGHTED VERTEX COVER

**IN**
$G=(V,E)$
$c_v$ COSTO INTERO

**SOL**
$U \subseteq V$ t.c. $v_i \in U$ o $v_j \in U$ $\forall (i,j) \in E$

**MEASURE**
$$COSTO(U) = \sum_{v_j \in U} c_j$$

| ILP | LP |
|---|---|
| $\min \sum_{j=1}^{m} c_j \cdot x_j$ | $\min \sum_{j=1}^{m} c_j \cdot x_j$ |
| $x_j + x_k \geq 1 \quad \forall (j,k) \in E$ | $x_j + x_k \geq 1 \quad \forall (j,k) \in E$ |
| $x_j \in \{0,1\} \quad \forall v_{j=1,\ldots,m} \in V$ | ~~$x_j \leq 1 \quad \forall v_{j=1,\ldots,m} \in V$~~ LINEAR RELAXATION |
| | $x_j \geq 0 \quad \forall v_{j=1,\ldots,m} \in V$ |

## ROUND VERTEX COVER ALGO {

DETERMINA ILP E RISOLVI RILASSAMENTO LP

SIA $\langle x_1^*, \ldots, x_m^* \rangle$ LP SOL

$$\forall v_j \begin{cases} x_j = 0, & x_j^* \leq 1/2 \\ x_j = 1, & x_j^* \geq 1/2 \end{cases}$$

RET U ASSOCIATO A $\langle x_1, \ldots, x_m \rangle$

}

**TH**

- 2-APPROSSIMATIVO

Proof
- FACT 1: $\langle x_1, \ldots, x_m \rangle$ FASIBLE PER ILP $(\Rightarrow$ U COVER$)$

  Proof: $\langle x_1^*, \ldots, x_m^* \rangle$ È FEASIBLE PER LP QUINDI $x_j^* \geq 1/2$ O $x_k^* \geq 1/2$ $\forall (j,k) \in E$

  $\Rightarrow$ O $x_j = 1$ O $x_k = 1$ $\Rightarrow$ $\langle x_1, \ldots, x_m \rangle$ FEASIBLE PER ILP

- FACT 2: $\dfrac{m}{m^*_{PL}} \leq 2$

  Proof: DAL ROUNDING $x_j \leq 2 \cdot x_j^*$

  $$\boxed{m} = \sum_{j=1}^{n} c_j \cdot x_j \leq \sum_{j=1}^{n} c_j \cdot 2 \cdot x_j^* = 2 \cdot \sum_{j=1}^{n} c_j \cdot x_j^* = 2 \cdot m^*_{LP}$$

$$\Rightarrow \boxed{\frac{m}{m^*}} \leq \frac{m}{m^*_{PL}} \underset{\text{FACT 2}}{\leq} \boxed{2}$$

$\square$

## MIN WEIGHTED SET COVER

**IN**
$U = \{o_1, \ldots, o_n\}$ UNIVERSO
$\hat{S} = \{S_1, \ldots, S_h$ t.c. $S_i \subseteq U\}$ FAMIGLIA
$c_j$ ASSOCIATO A $S_j$

**SOL**
$\hat{C} \subseteq \hat{S}$ t.c. $\sum_{S_j \in \hat{C}} S_j = U$

**MEASURE**
$$COSTO(U) = \sum_{S \in \hat{C}} c_j$$

| ILP | LP |
|---|---|
| $\min \sum_{j=1}^{h} c_j \cdot x_j$ | $\min \sum_{j=1}^{h} c_j \cdot x_j$ |
| $\sum_{j \mid o_i \in S_j} x_j \geq 1 \quad \forall o_i \in U$ | $\sum_{j \mid o_i \in S_j} x_j \geq 1$ |
| $x_j \in \{0,1\} \quad \forall S_j \in S$ | $\cancel{x_j \leq 1 \quad \forall S_j \in \hat{S}}$ LINEAR RELAXATION |
| | $x_j \geq 0 \quad \forall S_j \in \hat{S}$ |

ROUND SET COVER ALGO {

DETERMINA ILP E RISOLVI RILASSAMENTO LP

SIA $\langle x_1^* \_ x_m^* \rangle$ LP SOL

$$\forall S_j \begin{cases} x_j = 0, & x_j^* \leq 1/f \\ x_j = 1, & x_j^* \geq 1/f \end{cases}$$

RET $\hat{C}$ ASSOCIATO A $\langle x_1, \_, x_m \rangle$

}

QUANTE VOLTE APPARE L'OGGETTO IN $\hat{C}$?

TH • $f$-APPROSSIMATIVO

Proof • FACT 1: $\langle x_1, \_\_, x_m \rangle$ FASIBLE PER ILP ($\Rightarrow \hat{C}$ COVER)

Proof: $\langle x_1^*, \_, x_m^* \rangle$ È FEASIBLE PER LP $\Rightarrow \sum_{S_j \mid o_i \in S_j} x_j^* \geq 1$ E DATO CHE LA SOMMA HA AL PIÚ $f$ TERMINI C'È ALMENO UN $S_j$ T/C $x_j^* \geq 1/f \Rightarrow$ $\Rightarrow x_j = 1 \Rightarrow \sum_{S_j \mid o_i \in S_j} x_j \geq 1$

CONTENENTE $o_i$

L'OGG. $O_j$ APPARE ALMENO UNA VOLTA, VA PRESO

• FACT 2: $\frac{m}{m^*_{LP}} \leq f$

Proof: $$\boxed{m} = \sum_{j=1}^{h} c_j \cdot x_j \leq \sum_{j=1}^{h} c_j \cdot f \cdot x_j^* = f \cdot \sum_{j=1}^{f} c_j \cdot x_j^* = \boxed{f \cdot m^*_{LP}}$$

DAL ROUNDING $x_j \leq 2 \cdot x_j^*$

$$\Rightarrow \boxed{\frac{m}{m^*}} \leq \frac{m}{m^*_{LP}} \leq \boxed{f} \quad \square$$

# DYNAMIC PROGRAMMING

## MAX 0-1 KNAPSACK

PROG. DYN. KNAPSACK ALGO {

↳ NOTA: RITORNA LA MISURA CHE PUÒ RAGGIUNGERE LO ZAINO.

• Vedi sopra

FOR $w = 0, \_, b$
  $M[0, w] = 0$
FOR $i = 1, \_, m$
  FOR $w = 0, \_, b$
    IF $(w_i > w)$
      $M[i, w] = M[i-1, w]$
    ELSE
      $M[i, w] = \max\{M[i-1, w], M[i-1, w-w_i] + P_i\}$

}

esempio

| $i \backslash w$ | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 3 | 3 | 3 | 3 |
| 2 | 0 | 0 | 3 | 4 | 4 | 7 |
| 3 | 0 | 0 | 3 | 4 | 5 | 7 |
| 4 | 0 | 0 | 3 | 4 | 5 | 7 |

| Ogg | $w_i$ | $P_i$ |
|---|---|---|
| 1 | 2 | 3 |
| 2 | 3 | 4 |
| 3 | 4 | 5 |
| 4 | 5 | 6 |

MAX PROFITTO INSERIBILE IN KNAPSACK

FIND_OBJ_FROM_MATRIX {

i = n
k = W
WHILE (n>0 AND k>0)
  IF ( M[i,k] ≠ M[i-1,k] )
    MARCA i COME DENTRO_ZAINO
    i = i-1
    k = k - $w_i$
  ELSE
    i = i-1

}

| i\w | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 3 | 3 | 3 | 3 |
| 2 | 0 | 0 | 3 | 4 | 4 | 7 |
| 3 | 0 | 0 | 3 | 4 | 5 | 7 |
| 4 | 0 | 0 | 3 | 4 | 5 | 7 |

| Ogg | $w_i$ | $P_i$ |
|---|---|---|
| 1 | 2 | 3 |
| 2 | 3 | 4 |
| 3 | 4 | 5 |
| 4 | 5 | 6 |

i = ~~4~~ ~~3~~ ~~2~~ ~~1~~ 0   KNAPSACK = {2, 1}

k = ~~5~~ = 2 = 0   ($-w_2$, $-w_1$)

COMPLEXITY

TH · $O(n \cdot b)$

PROOF · $O(1)$ ∀ ENTRY ⇒ $O(nb)$
· $O(1)$ PER RISALIRE AGLI OGGETTI □

NOTA PSEUDO POLYNOMIAL
↳ es: $b = 2^n$

Def · $OPT \subseteq O$ t/c $P_i$ ALMENO $P \geq n \cdot P_{max} \geq n^*$ $\forall o_i \in OPT$

PROG. DYN. KNAPSACK DUAL {

FOR p = 1 _ P
  V[0,p] = ∞
FOR i = 1 _ n
  FOR p = 1 _ P
    IF ($P_i \geq p$)
      V[i,p] = min { V[i-1,p], $w_i$ }
    ELSE
      V[i,p] = min { V[i-1,p], V[i-1, p-$P_i$] + $w_i$ }

}

COMPLEXITY

TH · $O(n^2 \cdot P_{max})$

PROOF · $O(1)$ ∀ ENTRY ⇒ $O(n \cdot P) = O(n^2 P_{max})$
· $O(1)$ PER RISALIRE AGLI OGGETTI □

NOTA PSEUDO POLYNOMIAL
↳ es: $P_{max} = 2^n$

# PTAS - POLYNOMIAL TIME APPROXIMATION SCHEMES

**Def** · UN ALGO. PER $x \in NPO$ È PTAS SE DATO $x \in I_x$ e $\varepsilon > 0$, RITORNA $(1 \mp \varepsilon)$-APPROX SOL (PER max/min) IN POLY.T

↳ LA COMPLESSITÀ PUÒ ESSERE ESPONENZIALE A $1/\varepsilon$.

**MIN MULTIPROCESSOR SCHEDULING** · Vedi sopra

**RIC** · GRAHAM ALGORITHM:

$$\boxed{m} = \boxed{T_k(n)} = T_k(l-1) + t_l \le \frac{T - t_l}{h} + t_l = \underbrace{\frac{T}{h}}_{\le m^*} + \left(\frac{h-1}{h}\right) \cdot \underbrace{t_l}_{\le m^*} \le m^* + \frac{h-1}{h} \cdot m^* = \boxed{\left(2 - \frac{1}{h}\right) \cdot m^*}$$

↳ RISCRIVIAMO L'ULTIMO PASSAGGIO NON TOCCANDO $t_l$

$$m \le \underbrace{\frac{T}{h}}_{\le m^*} + \left(\frac{h-1}{h}\right) \cdot t_l \le m^* + \cancel{\left(\frac{h-1}{h}\right)} \cdot t_l \le m^* + t_l$$

↳ **IDEA** ↳ SE $t_l$ ABBASTANZA PICCOLO L'ERRORE DECRESCE

$\Rightarrow$ PER $t_l = \underbrace{\varepsilon \cdot m^*}$ $\Rightarrow m < m^* + \varepsilon m^*$ PTAS

**LEMMA** · SE $t_1, \dots, t_m$ IN ORDINE DECRESCENTE $\Rightarrow t_i \le T/i \quad \forall i$

Proof · ASSUMIAMO $t_i > T/i$

$$\Rightarrow \quad t_1 + t_2 + \ldots + t_i \boxed{\ge} i \cdot t_i > i \cdot \frac{T}{i} = T \quad \bot$$

□

**PTAS SCHEDULING ALGO** {

ORDINAMENTO DECRESCENTE DI $t_i$, RINOMINA JOB
COMPUTA OPT SCHEDULE $\delta$ PER I PRIMI $q = \lceil h/3 \rceil$ } OTTIMIZZAZIONE

FOR $J = q, \dots, m$
ASSEGNA $P_J$ AL PROC CON MINIMO $T_i(J-1)$ } GREEDY

RET $\delta$

}

**TH** · $(1+\varepsilon)$-APPROSSIMATIVO (PTAS QUINDI)

Proof · SIA $t \le m^*$ COMP. TIME DI $\delta^*(P_1, \dots, P_{h/3})$

· SE $m \le t$

$\delta$ OTTIMO (LA FASE GREEDY NON HA CAMBIATO NULLA

· SE $m > t$

SIA $K$ IL PROC. PIÙ CARICO

SIA $P_l$ L'ULTIMO ASSEGNATO A $K$

$$\Rightarrow \quad \boxed{m} = T_k(n) = T_k(l-1) + t_l \le \frac{T - t_l}{h} + t_l = \frac{T}{h} + \left(\frac{h-1}{h}\right) \cdot t_l < \underbrace{\frac{T}{h} + t_l \le m^* + \varepsilon \cdot m^*} = \boxed{(1+\varepsilon) \cdot m^*}$$

$$\Rightarrow \quad \frac{m}{m^*} \le \boxed{1+\varepsilon}$$

□

## COMPLEXITY

**NOTA** SE $h$ NON FISSATA MA DIPENDENTE DALL'INPUT

↳ CAZZI ESPONENZIALI!

Per $h=2$ diventa

**MIN PARTITION PROBLEM**

IN: $X$ set di oggetti, $a_i$ intero

SOL: partizione di $X$ in $X_1, X_2$

MEASURE: $\max\{\sum_{a_i \in X_1} a_i, \sum_{a_i \in X_2} a_i\}$

- ORDINAMENTO — $O(m \log m)$
- COMPUTAZIONE DI OPT — $O(h^{h/\varepsilon})$
- ITERAZIONE • # ITERAZIONI — $O(h) \cdot O(m)$

$\Rightarrow O(m \log m + h^{h/\varepsilon} + m \cdot h)$

**NOTA** • AL POSTO DI $\delta^*(P_1, \ldots, P_{h/3})$ TROVIAMONE UNA APPROSSIMATA

**LEMMA** • $\exists$ DYN. DUAL-APPROX ALGO. CHE IN POLY TIME TROVA UNO SCHEDULING PER I PRIMI $q = \lceil h/3 \rceil$ JOBS CON $t \leq (1+\varepsilon) m^*$

**TM** • $\exists$ PTAS PER IL PROBLEMA

## FPTAS - Fully Polynomial time approx. schemes

**Def** • UN ALGO. PER $\pi \in$ NPO È FPTAS SE DATO $x \in I_x$ E $\varepsilon > 0$, RITORNA $(1 \mp \varepsilon)$-APPROX SOL (PER $\max/\min$) IN POLY.T RISPETTO $x$ E $1/\varepsilon$

↳ LA COMPLESSITÀ RESTA BUONA ANCHE SE $\varepsilon$ CRESCE

| | |
|---|---|
| $\cancel{O\left(n \cdot \log n + 2^{\frac{1}{\varepsilon}}\right)}$ | $O\left(\frac{n \cdot \log n}{\varepsilon^2}\right)$ |
| NO FPTAS | FPTAS |

**MAX 0-1 KNAPSACK**

• Vedi sopra

**RIC** DYN-DUAL KNAPSACK PSEUDO POLY: $O(m^2 \cdot P_{max}) \Rightarrow$ L'IDEA È DI ABBASSARE $P_{max}$

**↳ IDEA** • APPROSSIMA A MULTIPLI DI $K > 0$ OGNI $P_i$ E POI DIVIDIAMOLO PER $K$

$\Rightarrow$ PER $K$ ABBASTANZA PICCOLO $P_i'$ APPROSSIMA $P_i$ E SOL OTTIMA DELLA NUOVA ISTANZA APPROSSIMA QUELLA DELLA VECCHIA

$$P_i' = \frac{\lfloor P_i / K \rfloor \cdot \cancel{K}}{\cancel{K}} = \left\lfloor P_i / K \right\rfloor$$

**COMPLESSITÀ** $O(m^2 \cdot P'_{max}) = O(m^2 \cdot P_{max} / K)$

**APPROSSIMAZIONE** • AL PIÙ UN ERRORE $K$ SU OGNI OGGETTO PRESO $\Rightarrow m \geq m^* - nk.$

LO DIMOSTRA NEL CORSO DOPO

$$\Rightarrow k = \left\lfloor \frac{\varepsilon \cdot p_{max}}{n} \right\rfloor$$

FPTAS KNAPSACK ALGO {

$k = \left\lfloor \frac{\varepsilon \cdot p_{max}}{n} \right\rfloor$

TROVA SOL PER L'ISTANZA CON $p_i = p_i/k$ TRAMITE DYN-DUAL KNAPSACK

RET $S$

}

LEMMA

- $m \geq m^* - nk$

Proof
- $S = \{o_{j1}, \dots, o_{jh}\} \Rightarrow m = p_{j1} + \dots + p_{jh}$
- $S^* = \{o_{i1}, \dots, o_{il}\} \Rightarrow m^* = p_{i1} + \dots + p_{il}$
- ASSUMIAM $m < m^* - nk$

$$\Rightarrow \underbrace{\left\lfloor \frac{p_{i_1}}{k} \right\rfloor + \dots + \left\lfloor \frac{p_{i_l}}{k} \right\rfloor}_{S} \geq \left(\frac{p_{i_1}}{k} - 1\right) + \dots + \left(\frac{p_{i_l}}{k} - 1\right) =$$

$$= \frac{p_{i_1} + \dots + p_{i_l}}{k} - l \geq \frac{p_{i_1} + \dots + p_{i_l}}{k} - n = \frac{m^*}{k} - n >$$

By hypothesis $m < m^* - n \cdot k$

$$> \frac{m + n \cdot k}{k} - n = \frac{m}{k} = \frac{p_{j_1} + \dots + p_{j_h}}{k} \geq \underbrace{\left\lfloor \frac{p_{j_1}}{k} \right\rfloor + \dots + \left\lfloor \frac{p_{j_h}}{k} \right\rfloor}_{S^*}$$

$\Rightarrow S \geq S^*$ $\perp$ $\square$

TH

- L'ALGO È FPTAS PER IL PROBLEMA

Proof

COMPLESSITÀ

- $O(n^2 \cdot p'_{max}) = O\left(n^2 \cdot \frac{p_{max}}{k}\right) = O\left(\frac{n^2 \cdot p_{max}}{\frac{\varepsilon \cdot p_{max}}{n}}\right) = \boxed{O\left(\frac{n^3}{\varepsilon}\right)}$

APPROSSIMAZIONE

K È SCELTO PER OTTENERE L'APPROSSIMAZIONE $(1-\varepsilon)$

- $\boxed{\frac{m}{m^*}} \geq \frac{m^* - n \cdot k}{m^*} = 1 - \frac{n \cdot k}{m^*} \geq 1 - \frac{n \cdot k}{p_{max}} \geq 1 - \frac{\frac{n \cdot \varepsilon \cdot p_{max}}{n}}{p_{max}} = \boxed{1 - \varepsilon}$ $\square$

$m^* \geq p_{max}$

---

RIDUCIAMO COMPLESSITÀ

- IN GENERALE È $O(n \cdot p_{max})$
- APPROSSIMANDO I PROFITTI È $O\left(n \cdot \frac{n \cdot p_{max}}{k}\right)$
- E LA MISURA $p_{max} \leq m^* \leq n \cdot p_{max}$

IDEA

- MIGLIORIAMO I BOUND
  - 1/2-APPROX-MODIFIED GREEDY $m_{GR} \leq m^*$
- LO USIAMO AL POSTO DEL DYN-DUAL KNAPSACK

NEW FPTAS KNAPSACK ALGO {

COMPUTA $m_{GR}$

$K = \left\lfloor \frac{\varepsilon \, m_{GR}}{n} \right\rfloor$

TROVA OPT

TROVA OPT PER L'ISTANZA CON $p_i = p_i/k$ TRAMITE MODIFIED-DYN-DUAL KNAPSACK

RET $S$

}

**COMPLEXITY**

- $p'_{max} = \left\lceil \frac{\varepsilon m_{ar}}{k} \right\rceil \Rightarrow O\left(\frac{n \cdot m_{mg}}{k}\right) = O\left(\frac{n \cdot m_{mg}}{\frac{\varepsilon \cdot m_{mg}}{n}}\right) = O\left(\frac{n^2}{\varepsilon}\right)$

**APPROSSIMAZIONE**

- $$\frac{m}{m^*} \geq 1 - \frac{n \cdot k}{m^*} \geq 1 - \frac{n \cdot k}{m_{mg}} \geq 1 - \frac{n \cdot \frac{\varepsilon \cdot m_{mg}}{n}}{m_{mg}} = 1 - \varepsilon.$$

# ALTERNATIVE APPROACHES

**RESTRIZIONE SET DI INPUT INSTANCES**

- PERFORMANCE GARANTITE SU UN SOTTOINSIEME DI ISTANZE

**AVERAGE OR PROBABILISTIC ANALYSIS**

- ASSUMENDO UNA DISTRIBUZIONE PROBABILISTICA DELLE ISTANZE VALUTA LE PERFORMANCE ATTESE

  ↳ PRO
  - SCOPRE BUONI COMPORTAMENTI DI UN ALGO
  - ANALISI BASATA SU PROVE MATEMATICHE

  ↳ CONTRO
  - ANALISI DIFFICILE
  - NON GARANTISCE PERFORMANCE (AL PIÙ ALTA PROBABILITÀ)
  - SPESSO LA DISTRIBUZIONE DELLE ISTANZE È SCONOSCIUTA

**HEURISTICS**

- ALGORITMI A VOLTE CON BUON COMPORTAMENTO PRATICO, MA LE PERFORMANCE NON SONO DIMOSTRABILI

**RANDOMIZED ALGORITHMS**

**Def**

- FANNO SCELTE RANDOM DURANTE L'ESECUZIONE (+ OUTPUT SU STESSO INPUT)

  ↳ PRO
  - SEMPLICI E VELOCI

  ↳ CONTRO
  - RISULTATO INCERTO
  - IMPOSSIBILITÀ DI RANDOM-REALE

- $m$ RANDOM VARIABLE
- $E(m)$ VALORE ATTESO DIPENDENTE DALLE RANDOM-CHOICES

**Def**

- UN RANDOMIZED ALGO. È $r$-APPROSSIMATIVO SE:

$$\frac{E(m)}{m^*} \leq r \qquad \text{(MIN)} \qquad\qquad \frac{E(m)}{m^*} \geq r \qquad \text{(MAX)}$$

**MAX WEIGHTED CUT**

| IN | SOL | MEAS |
|---|---|---|
| $G=(V,E)$<br>$w_{ij}\ \forall (i,j)\in E$ | PARTIZIONE $(V_1,V_2)$<br>T/C $V_1 \cup V_2 = V$,<br>$V_1 \cap V_2 = \{\emptyset\}$ | $\sum\limits_{(v_i,v_j)\in E \text{ T/c } v_i\in V_1, v_j \in V_2} w_{ij}$ |

**RANDOM CUT** {

$V_1 = \{\emptyset\}$

$V_2 = \{\emptyset\}$

FOR $i = 1, \ldots, m$

  PUT $V_i$ IN $V_1$ CON $1/2$ PROBABILITÀ

RET $V_1, V \setminus V_1$

}

**TH** — **1/2 - APPROSSIMATIVO**

*Proof* • SIA $X_{ij}$ UNA VAR. RANDOM "L'ARCO $(i,j)$ È NEL TAGLIO"

• $$m = \sum_{\{v_i,v_j\}\in E} \omega_{ij} \cdot x_{ij}$$

$$\Rightarrow E(m) = E\left(\sum_{\{v_i,v_j\}\in E} \omega_{ij} \cdot x_{ij}\right) = \sum_{\{v_i,v_j\}\in E} \omega_{ij} \cdot E(x_{ij}) =$$

$\sum_{\{v_i,v_j\}\in E} \omega_{ij} \cdot P(x_{ij}=1) = \sum_{\{v_i,v_j\}\in E} \omega_{ij} \cdot P((v_i \in V_1 \wedge v_j \in V_2) \vee (v_i \in V_2 \wedge v_j \in V_1)) =$

$$= \sum_{\{v_i,v_j\}\in E} \omega_{ij} \cdot P(x_{ij}=1) = \sum_{\{v_i,v_j\}\in E} \omega_{ij} \cdot P((v_i \in V_1 \wedge v_j \in V_2) \vee (v_i \in V_2 \wedge v_j \in V_1)) =$$

$$= \sum_{\{v_i,v_j\}\in E} \omega_{ij} \cdot \left(\frac{1}{2}\cdot\frac{1}{2} + \frac{1}{2}\cdot\frac{1}{2}\right) = \frac{1}{2} \cdot \sum_{\{v_i,v_j\}\in E} \omega_{ij} \geq \frac{m^*}{2}$$

$$\Rightarrow \frac{E(m)}{m^*} \geq \frac{1}{2}$$

$\square$

---

**MIN WEIGHTED SET COVER**

• Vedi sopra

**GREEDY - NOTA**

• NON POSSIAMO CONSIDERARE SOLO I COSTI → POCHI ELEMENTI COPERTI
• NON POSSIAMO CONSIDERARE SOLO IL NUMERO DI OGG. COPERTI → COSTO ECCESSIVO

⇒ PRENDIAMO IL SOTTOINSIEME CON MINIMO COSTO PER NUOVO OGG. COPERTO

↳ SIANO $S_1, \_\_, S_{j-1}$ GLI INSIEMI COPERTI ALLO STEP $i$:

COSTO DI $S_k$

$$eff(S_k) = \frac{c_k}{\left|S_k \cap \overline{C_{j-1}}\right|}$$

} OGG IN $S_k$ CHE NON HO PRESO

$C_{j-1} = (S_1 \cup \ldots \cup S_{j-1})$
$\overline{C_{j-1}} = U \setminus C_{j-1}$

↳ GREEDY CHOICE: $eff(S_j) = \min \{ eff(S_k) \mid S_k$ has not be chosen yet $\}$

**GREEDY ALGO** {

```
C = {∅}          // OGGETTI COPERTI
Ĉ = {∅}          // SOTTOINSIEMI PRESI
j = 1
WHILE C ≠ U
    SIA S_j IL SUBSET CON MINIMA effectiveness
    Ĉ = Ĉ ∪ S_j
    FOR o_i ∈ S_j ∩ C̄
        C = C ∪ S_j
RET C
```

}

**LEMMA** • $m = \sum_{S_j \in \mathcal{C}} c_j = \sum_{i=1}^{n} price(o_i)$ t.c. $price(o_i) = eff(S_j)$

**Proof** • BANALE: IL COSTO DEI $price(o_i)$ COPERTI ALLO STEP $i$ È SEMPLICEMENTE $c_j$ ASSEGNATO A $S_j$.

• QUINDI IL COSTO TOTALE È DIVISO UGUALMENTE TRA GLI OGGETTI COPERTI. □

**LEMMA**

• For every $j$, given any choice of subsets $S_j',\dots,S_t'$ that form a cover with the subsets $S_1,\dots,S_{j-1}$ chosen by the greedy algorithm at the beginning of step $j$, for every object $o_i$ not yet covered at the beginning of step $j$ $price'(o_i) \geq eff(S_j)$, where $S_j$ is the subset chosen by the greedy algorithm at step $j$, $eff(S_j)$ its effectiveness, $price'(o_i)$ is the effectiveness of the subset $S_l'$ covering $o_i$ assuming that, starting from step j, the greedy choice is done among subsets $S_j',\dots,S_t'$ only.

**Proof** It is sufficient to observe that $eff(S_j)$ is the minimum covering price per object at step $j$ and that the effectiveness of an unchosen subset can only increase during the steps, as its cost is fixed while some further objects in it can be covered during the steps due to the choice of other subsets.

Thus the price of $o_i$, that is the effectiveness of the subset $S_l'$ with $l \geq j$ among $S_j',\dots,S_t'$ that covers it, is at least equal to $eff(S_j)$. □

**LEMMA**

• Let $o_1,\dots,o_n$ be the objects listed in the covering order of the greedy algorithm, that is such that the objects covered during step $j$ are listed after the ones covered in the previous steps and before the ones covered in the successive steps,
Then, $\forall i$ such that $1 \leq i \leq n$, $price(o_i) \leq \frac{m^*}{n-i+1}$

**Proof** •Consider any object $o_i$ and let $j$ the step in which it is covered.

- At the beginning of step $j$, since the not yet chosen sets of an optimal solution can cover all the uncovered objects with overall cost at most $m^*$, there must exist a subset $S_k$ of effectiveness at most $m^*/|C_{j-1}|$, where $C_{j-1}$ is the subsets of objects not yet covered at the beginning of step $j$.

- In fact, if this is not the case, since the price of $o_i$ is equal to the effectiveness $eff(S_j)$ of the subset $S_j$ chosen by the greedy algorithm, by the previous lemma, for every possible choice of remaining subsets to complete the cover, that is for any possible pricing of the remaining objects

$$\Rightarrow \sum_{o_i \in C_{j-1}} price'(o_i) \geq \sum_{o_i \in C_{j-1}} eff(S_j) > \sum_{o_i \in C_{j-1}} \frac{m^*}{|C_{j-1}|} = |C_{j-1}| \cdot \frac{m^*}{|C_{j-1}|} = m^*$$

- A CONTRADICTION to the hypothesis that there exists a choice of subsets that covers the remaining objects with cost at most $m^*$.

↳ But $|C_{j-1}| \geq n-i+1$ as $o_i,\dots,o_n \in C_{j-1}$ and as a consequence

$$price(o_i) \leq \frac{m^*}{|C_{j-1}|} \leq \frac{m^*}{n-i+1}$$

□

**TM** • $H_n$ - APPROSSIMATIVO (CON $H_n = 1 + 1/2 + \dots + 1/n$)

Proof $m = \sum_{i=1}^{n} price(o_i) \le \sum_{i=1}^{n} \frac{m^*}{n-i+1} = m^* \cdot \left(\frac{1}{n} + \frac{1}{n-1} + \frac{1}{n-2} + \ldots + 1\right) = m^* \cdot H_n$

$$\Rightarrow \frac{m}{m^*} \le H_n$$

# SOCIAL NETWORK e LINK POPULARITY

## CENTRALITY

**Defs**

- DISTANZA: • SHORTEST PATH TRA u e v
- RAGGIO: • $r(u) = \max_v d(u,v)$ È LA DISTANZA DAL NODO PIÙ LONTANO
- CENTER OF GRAPH: • NODO CON MINIMO RAGGIO

DEGREE

- $c[v] = deg(v)$ (degree of $v$)

CLOSENESS

- $c[v] = \Sigma_{t \in V,\, t \neq v} 1/d(v,t)$ : INDICA QUANTO UN NODO È VICINO A TUTTI GLI ALTRI

BEETWEENES

- $c[v] = \Sigma_{s,t \in V,\, s \neq t \neq v} \sigma_{st}(v)/\sigma_{st}$
  - ↳ $\sigma_{st}(v)$ = number shortest paths from $s$ to $t$ containing $v$
    $\sigma_{st}$ = total number shortest paths from $s$ to $t$

## CO-CITATION

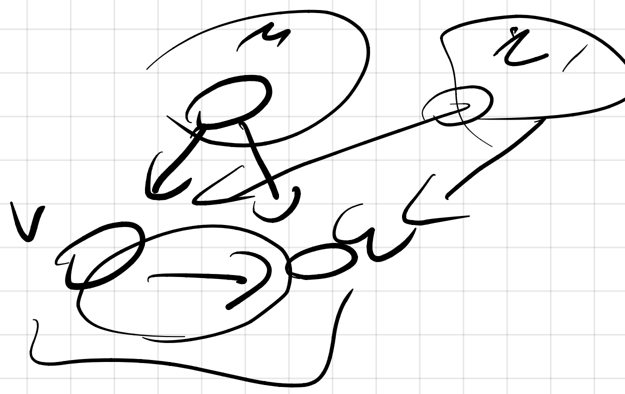
**Def**

- se u CITA v e w ⇒ u,v SONO CO-CITATI DA u
  ↳ ci fa intuire che c'è una correlazione tra i due

E

- SIA E ADJACENCY MATRIX DEL CITATION GRAPH
  ↳ $E[u,v] = 1$ se u cita v

$EE^T$




- $$(E^T E)[v,w] =$$
  $$= \sum_u E^T[v,u] E[u,w] =$$
  $$= \sum_u E[u,v] E[u,w] =$$
  $$= \left|\{u : (u,v) \in E, (u,w) \in E\}\right|$$

↳ UN ENTRY DI $EE^T$ È DETTA CO-CITATION INDEX, INDICA LA CORRELAZIONE TRA u,v.

↳ USATA PER CREARE CO-CITATION CLUSTERS.

## PRESTIGE

**Def**

- BANALMENTE L'IN-DEGREE
- DANDO A UN NODO v LA SOMMA DEI PRESTIGI DEGLI u T/C $E[u,v]=1$

$$p' = E^T p$$

POWER ITERATION {

INIT

$p = (1, \_, 1)^T$

ITERAZIONE

$p = E^T p$

NORMALIZZA $p = p / \sum_u p[u]$ (AVOID OVERFLOW)

}

NOTA

AUTOVETT. DOMIN. = QUELLO CON MINUO ABS. VAL.

- SE C'È UN AUTOVETTORE DOMINANTE, CONVERGE A QUELLO
  - ↳ DI SOLITO NEI WEB-GRAPH NON C'È

## RANKING ALGOS E POPULARITY

- GLI ALGORITMI DI RANKING VALUTANO LA POPOLARITÀ DELLE PAGINE

The basic steps are the following:

- Start from a collection of web pages
- Extract from them the graph of the hyperlinks
- Run the ranking algorithm on the graph
- Output : a weight of ***popularity*** for each node

- ESISTONO DUE TIPI:

Query independent: they make the ranking of the entire Web

Query dependent: they make the ranking of a relatively small subset of pages related to a specific query

NOTA

- IL PR SI TROVA CALCOLANDO IL PRINCIPALE AUTOVETTORE ASSOCIATO A UNA MATRICE DERIVATA DA E

We assume that ***E*** (and thus ***L***) are

- ***Irreducible***: there is a direct path from each node to every other node
- ***Aperiodic***: for each ***(u, v)*** there are paths that connect them with each possible number of links, every possible length, with the exception of a finite set of lengths which may be missing

## PAGE RANK

- $P_0[u]$ - PROBABILITÀ CHE IL RANDOM SURFER STARTI DA $u$
- $P_i[v]$ - PROBABILITÀ CHE DOPO $i$ CLICK IL SURFER SIA IN $v$
- $E$ - MATRICE DI ADIACENZA
  - ↳ $E[u,v]=1$ - C'È UN LINK DA $u$ A $v$
- $N_u = \sum_v E[u,v]$ - OUT-DEGREE DI $u$
  - ↳ È DATO DALLA SOMMA DEI VALORI IN $E$ ALLA RIGA DI $u$

⇒ DATO CHE NON CI SONO DUE ARCHI TRA $u$ E $v$ E CHE $P_1[v]$ DIPENDE DA $P_0[u]$:

$$p_1[v] = \sum_{u|(u,v)\in E} \frac{p_0[u]}{N_u}$$

- $L$ MATRICE CHE NORMALIZZA TUTTI I VALORI SU UNA RIGA DI $E$

$$L[u,v] = \frac{E[u,v]}{\sum_w E[u,w]} = \frac{E[u,v]}{N_u}$$

↳ $p_1[v] = \sum_u L[u,v] p_0[u]$ ⇒ $p_1 = L^T p_0$

↳ IN GENERALE:

$$p_{i+1} = L^T p_i$$

⇒ SE $E$ (e $L$) IRRIDUCIBILE e APERIODICA LA SEQUENZA $P_0, P_1, \ldots$ CONVERGE AL PRINCIPALE AUTOVETTORE $L^T$, RISOLVENTE $P = L^T P$

VETTORE

- IL PRESTIGIO $P[u]$ È DETTO PAGE RANK

RIC:

**Power iteration:**
A method for finding dominant eigenvector (the vector corresponding to the largest eigenvalue)

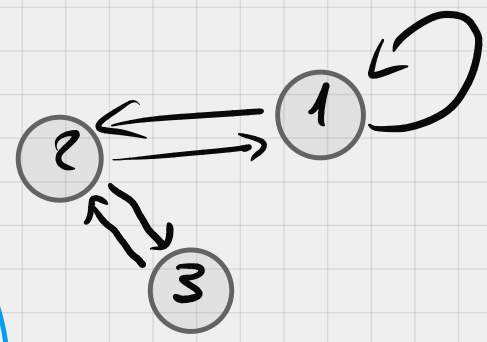
- $p_1 = L^T \cdot p_o$
- $p_2 = L^T \cdot p_1 = L^T(L^T p_1) = L^T{}_2 \cdot p_o$
- $p_3 = L^T \cdot p_2 = L^T(L^T{}_2 p_2) = L^T{}_3 \cdot p_o$

• esempio

$L^T =$

| | 1 | 2 | 3 |
|---|---|---|---|
| 1 | ½ | ½ | 0 |
| 2 | ½ | 0 | 1 |
| 3 | 0 | ½ | 0 |

ARCHI DI 1



0) $P = \begin{bmatrix} P_1 \\ P_2 \\ P_3 \end{bmatrix} = 1/3$

1)
$P_1 = 1/2\, P_1 + 1/2\, P_2 = 2/6 = 1/3$
$P_2 = 1/2\, P_1 + P_3 = 3/6$
$P_3 = 1/2\, P_2 = 1/3$

2)
$P_1 = 1/2\, P_1 + 1/2\, P_2 = 1/6 + 1/2 \cdot 3/6 = 5/12$
$P_2 = 1/2\, P_1 + P_3 = 1/6 + 1/3 = \frac{3}{6} = 1/3$
$P_3 = 1/2\, P_2 = 3/6 \cdot 1/2 = 3/12 = 3/12$

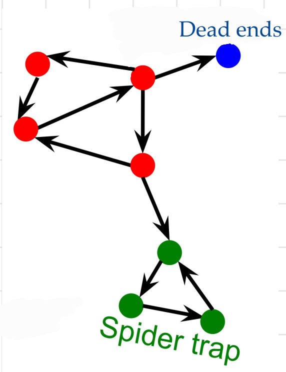
(3)
$P_1 = 1/2\, P_1 + 1/2\, P_2 = 5/24 + 1/6 = 9/24$
$P_2 = 1/2\, P_1 + P_3 = 5/24 + 3/12 = 11/24$
$P_3 = 1/2\, P_2 = 1/3 \cdot 1/2 = 1/6$

n)
$P_1 = 6/5$
$P_2 = 6/5$
$P_3 = 6/5$

COSA RAPPRESENTA P ALLA FINE DELLE ITERAZIONI?

• P È LA FREQUENZA CON CUI UN SURFER TOCCA OGNI PAGINA

## PROBLEMI
(DEAD ENDS e SPIDER TRAP)




• DEAD ENDS $A \longrightarrow B$ $\begin{cases} P_A = 1, 0, 0, 0 \dots \\ P_B = 0, 1, 0, 0 \dots \end{cases}$

• SPIDER TRAP $A \rightleftarrows B$ $\begin{cases} P_A = 1, 0, 1, 0 \dots \\ P_B = 0, 1, 0, 1 \dots \end{cases}$

⇒ PER RISOLVERLI INTRODUCIAMO TELETRASPORTI RANDOM CON PROBABILITÀ d, COSÌ $P_{i+1} = L^T P_i$ DIVENTA

$$\mathbf{p}_{i+1} = (1-d)L^T\mathbf{p}_i + d\begin{pmatrix} 1/N & \cdots & 1/N \\ \vdots & \ddots & \vdots \\ 1/N & \cdots & 1/N \end{pmatrix}\mathbf{p}_i = \left((1-d)L^T + \frac{d}{N}\boxed{1_N}\right)\mathbf{p}_i$$

PR CON d

• CIOÈ IL PR È IL PRINCIPALE AUTOVETTORE DELLA MATRICE

$$M=(1-d)L^T+(d/N)\, 1_N$$

$$p_{i+1} = (1-d)L^T p_i + \frac{d}{N}(1,\dots,1)^T$$

$$\Rightarrow p[v] = \frac{d}{N} + (1-d)\sum_{(u,v)} \frac{p[u]}{N_u}$$

# TOPIC SPECIFIC PR

**Def**
- MISURIAMO LA POPOLARITÀ SU DIVERSI TOPIC
- IL RANDOM SURFER SELEZIONA UNA CATEGORIA IN BASE ALLA QUERY E VI SI TELETRASPORTA CON IL 10% DI PROBABILITÀ

1) OFFLINE
- QUERY INDIPENDENT, OGNI PAGINA HA UN PR MULTIPLO (UNO A CATEGORIA) COMPUTATO STATICAMENTE, IL GRAFO È ^ ATO

2) ONLINE
- QUERY DEPENDENT, ESTRAPOLA PESI PER LE CATEGORIE E IL PR DI UNA PAGINA VIENE CALCOLATO COME WEIGHTED SUM PER OGNI CATEGORIA

$$PR(W,v) = \sum_j [w_j \cdot \underbrace{PR(W, v_j)}_{\text{PR NORMALE}}] = PR(W, \sum_j [w_j \cdot v_j])$$

CATEGORIA

## HITS (HYPERLINK INDUCED SPECIFIC TOPIC SEARCH)

ROOT SET
- LA QUERY È MANDATA A UN TEXT-BASED INFOR. RETRIEVAL SYS. PER GENERARE UN ROOT SET R

EXPANDED SET
- I VICINI $N(R)$ T/C $u \in N(R) \Leftrightarrow (u,v) \in E \vee (v,u) \in E$

QUERY GRAPH
- SONO RIMOSSI I NODI CHE PUNTANO A NODI DELLO STESSO SITO. OTTENIAMO QUINDI $G_q = (V_q, E_q)$ IN CUI OGNI NODO HA DUE INDICI (HUB e AUTHORITY SCORES)

AUTHORITIES e HUB
- UNA CONTIENE INFORMAZIONI SPECIFICHE SU UN ARGOMENTO, L'ALTRA NE PARLA TRA LE ALTRE COSE

In summary, the main steps of the HITS algorithm are submitting queries to an IR system based on texts and get the whole root expand along the root of a unit radius for obtaining the expanded graph run the power iterations simultaneously on the hub scores and authority return hubs and authority with the highest scores

- DEFINENDO $h$ e $a$ COME VETTORI DEGLI SCORES:
  - ↳ $a[u]$ DIPENDE DALLO SCORE $h[v]$ DI OGNI HUB $u$ T/C $\exists (v,u) \in E$
  - ↳ $h[u]$ DIPENDE DALLO SCORE $a[v]$ DI OGNI AUTH $u$ T/C $\exists (v,u) \in E$

$$a[u] = \sum_{v:(v,u)\in E} h[v] \qquad h[u] = \sum_{v:(u,v)\in E} a[v]$$

In matrix terms: $a = E^T h$ e $h = E a$

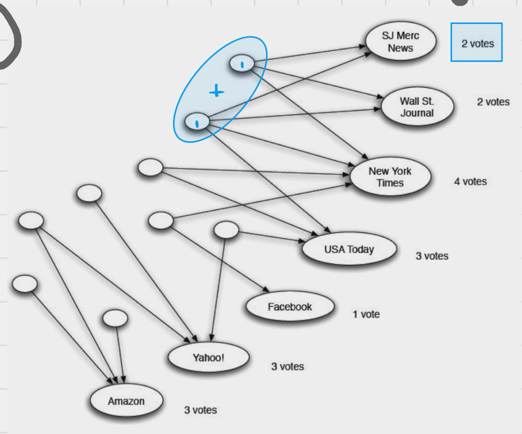
⇒ POSSIAMO RISOLVERLO CON POWER ITERATION
- INIZIALIZZIAMO $h[u] = a[u] = 1$
- A OGNI ITERAZIONE NORMALIZZIAMO $a$/$h$ CON LA NORMA $L_1$
- $a$, $h$ CONVERGONO AI PRINCIPALI AUTOVETTORI DI $E^T E$, $E E^T$

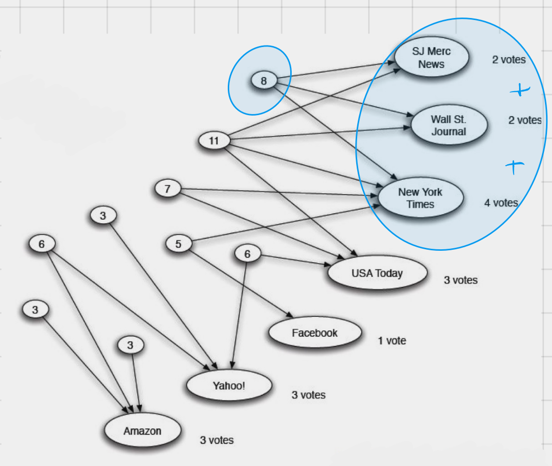
↳ TIPICAMENTE 20/30 ITERAZIONI

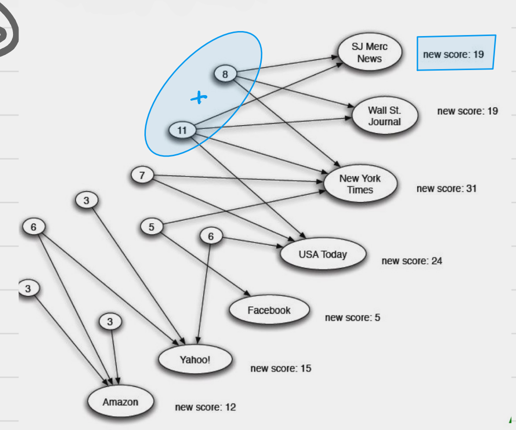
COME SCEGLIAMO HUB E AUTHORITIES?
↳ IDEA

BUONE AUTH. LINKATE DA BUONI HUB
BUONI HUB PUNTANO BUONE AUTH
SI INIZIA CON SCORE A 1
SI CAMBIANO ITERATIVAMENTE I PESI

- esempio: semplifichiamo il problema dando ad alcuni solo hub score e ad altri solo authority score

1



2



3


# SPAMMING

**GOOGLE SOL.**

- CREDI A CIÒ CHE DICONO DI TE PIUTTOSTO DI CIÒ CHE TI DICI DA SOLO

PROBLEMI: SPAM FARM

- LINK SPAM
  ↳ TRAMITE OWNED E ACCESSIBLE PAGES

## TRUST RANK

- TOPIC-SPECIFIC CON CATEGORIA TRUSTED
  - ↳ BASE PRINCIPLE: RARO CHE PAGINE BUONE LINKINO CATTIVE
  - ↳ SEED PAGES: POSSIAMO SCEGLIERE TRA QUELLE CON PIÙ PAGE RANK, O DOMINI FIDATI
  - ↳ ORACOLO UMANO (ISPEZIONA I SEED)

TRUST PROPAGATION

- LA TRUST DI UNA PAGINA È LA SOMMA DELLE PAGINE TRUSTY CHE LA LINKANO
- UN SEED HA TRUST=1
- INCREMENTO DI UNO LA TRUST DELLE PAGINE A CUI PUNTA

TRUST ATTENUATION

- PIÙ TI ALLONTANI DA UNA TRUSTED PAGE MENO BOOST ALLA TRUST PRENDI

TRUST SPLITTING

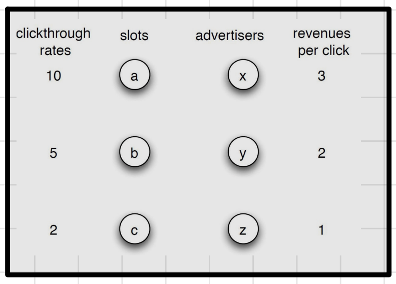
- LA TRUST CHE DAI È DIVISA PER IL NUMERO DEI TUOI LINK (+ PAGINE LINK − SEI AFFIDABILE)

# ADVERTISING AND MATCHING MARKET

## MATCHING MARKET

Matching market:

- Set of buyers (advertisers) and set of slots (sellers)
- Each buyer $j$ has a valuation $v_{i,j}$ for the item offered by seller $i$
- Goal: properly match up buyers and sellers




**DEFS**

- GRAFO BIPARTITO $G=(V_1+V_2,E)$ t.c. $\forall (u,v)\in E$ $u\in V_1 \wedge v\in V_2$ O VICEVERSA
- PERFECT MATCHING: MATCHING CHE NON LASCIA NODI SCOPERTI
- CONSTRICTED SET: INSIEME DI NODI DA UN LATO I CUI ARCHI IMPEDISCONO LA FORMAZIONE DI UN PERFECT MATCHING

HALL'S MATCHING TH.

- UN GRAFO BIPARTITO (IN CUI $|V_1|=|V_2|$) HA UN MATCHING PERFETTO ⟺ NON HA UN CONSTRICTED SET

Proof

1. SE C'È IL CONSTRICTED SET ⇒ NON C'È UN MATCHING PERFETTO
   ⇒ OVVIO!
2. SE NON C'È UN MAT. PERF. ⇒ C'È UN CONS. SET
   - TROVARE IL CONS. SET SAPENDO CHE G BIPARTITO NON HA UN MATCHING PERFETTO
     - DATO UN MATCHING M (NON PERFETTO), SE C'È UN AUGMENTING PATH, POSSIAMO ESPANDERE M

**Def**

AUGMENTING PATH: PATH ALTERNATO CON ENDPOINT UNMATCHED

DA G BIPARTITO CON $|V_1|=|V_2|$ COME COMPUTIAMO UN MATCHING PERFETTO?

1. Start with an empty matching
2. Look for an unmatched node *W* on the right
3. Use alternating BFS to search for an augmenting path beginning at *W*
4. If found, use this path to enlarge the matching and iterate, else indicate the constricted set

So, either we end up with a perfect matching, or we can provide a constricted set

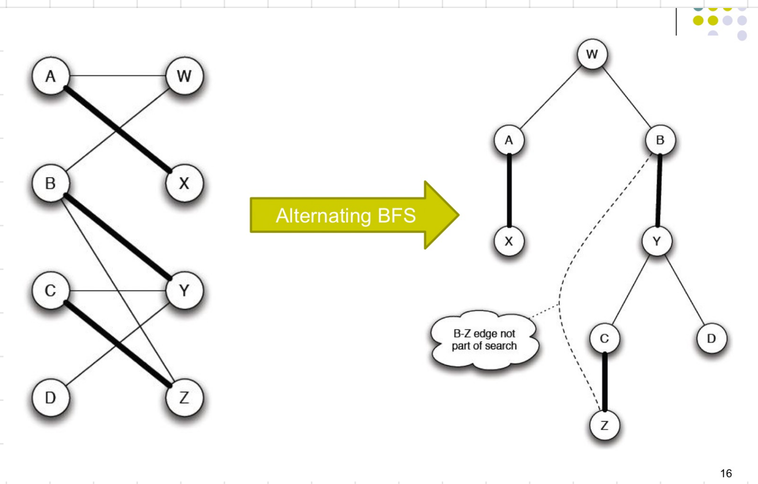
23

↳ SE ∀ NODO UNMATCHED SULLA DESTRA NON C'È L'AUGMENTING PATH, ALLORA IL MATCHING È MASSIMO

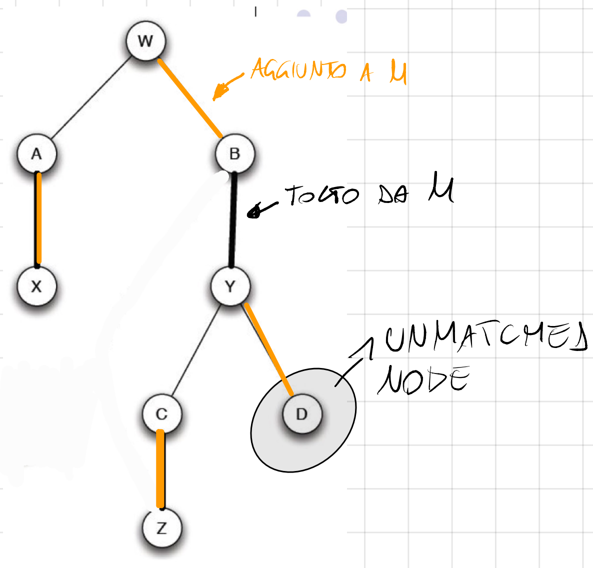
NOTA: IN QUESTO MODO POSSIAMO RISOLVERE IL MAT. MARK. SE C'È UNA CENTRAL AUTH. CHE COMPUTA IL MATCHING.

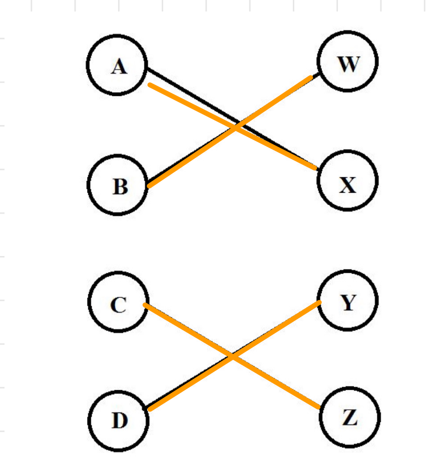
↳ PER DISTRIBUITO?

- ALTERNATING BFS
  (1) SCEGLI NODO UNMATCHED A DESTRA
  (2) COSTRUISCI ALBERO
    - AGLI STEP PARI ESPLORA MATCHING EDGES
    - AGLI STEP DISPARI GLI UNMATCHING EDGES



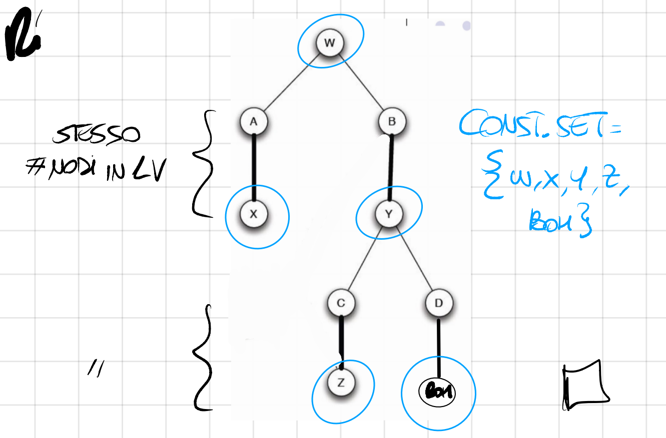

- PROPRIETÀ DEL BFS TREE
  - LIVELLI PARI: NODI $V_1$
  - LIVELLI DISPARI: NODI $V_2$
  - SE C'È UNMATCHED NODE A UN LV. DISPARI
    ⇒ ALLORA C'È UN AUGMENTING PATH (DA W A D) DALLA ROOT A TALE NODO



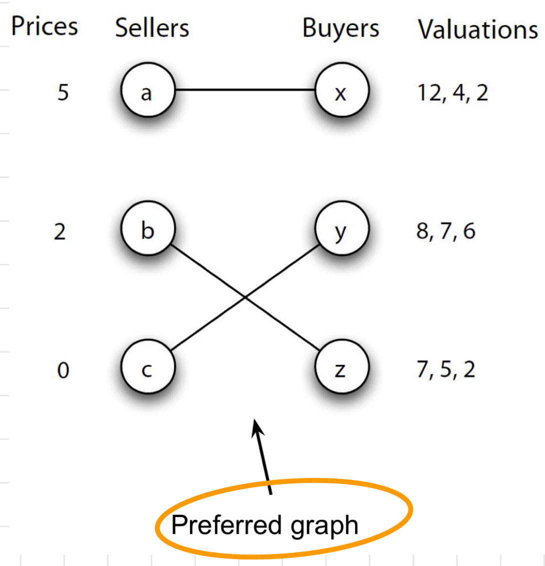

- SE NON C'È UNMATCHED
  - I NODI A LV $i$ DISPARI HANNO STESSO NUMERO DI NODI DEL LV $i+1$
  - OGNI NODO A LV PARI HA I VICINI (DI G ORIGINALE):
    - SOPRA SE SONO MATCHED
    - SOTTO ALTRIMENTI

- CONSTRICTED SET = W e TUTTI I NODI AI LIVELLI DISPARI




□

# MARKET CLEARING




Payoffs of each buyer for each house:

| | a | b | c |
|---|---|---|---|
| X | 7 | 2 | 2 |
| Y | 3 | 5 | 6 |
| Z | 2 | 3 | 2 |

5

- Se il preferred graph ha un perfect matching, il pricing è detto market clearing (tutti prendono il meglio)

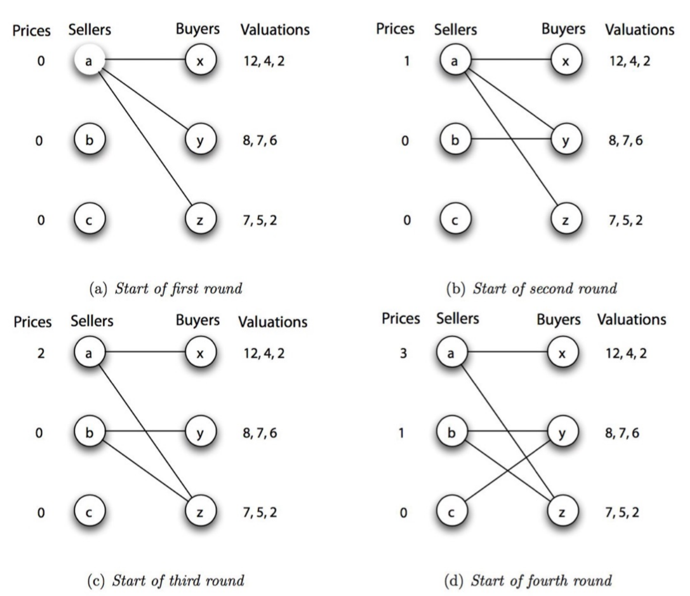
TM
- ∃ market-clearing pricing ∀ set di 'valutations'

Proof
- se P non è market clearing => ∃ const. set C
- |N(C)|>|C| ha alta domanda, alziamo il prezzo di 1 per N(C) finché un buyer non ha un'altra opzione
- prima di iniziare un round abbassiamo tutti i prezzi di $\delta$ in modo da averne uno con $p'_i = p_i - \delta = 0$ (così ogni buyer ha utilità $\geq 0$ per quel seller)
  => ogni buyer ha almeno un preferred seller.

## Procedure (for market clearing prices)

1. Set $p_i = 0 \ \forall i$
2. Construct the preferred-seller graph $G$
3. If perfect matching in $G$ STOP: current prices are market-clearing
4. If not find a constricted set of buyers $C$ and their neighbors $N(C)$
5. Raise the price of each seller in $N(C)$ by $1$
6. Reduce prices decreasing all of them by the same amount $\delta$=1 so that the lowest price becomes $0$
7. Repeat from step 2.

NOTA
market-clearing prices are not unique




(a) Start of first round
(b) Start of second round
(c) Start of third round
(d) Start of fourth round

• L'ALGORITMO TERMINA?

• MOSTRIAMO CHE LA PROCEDURA TERMINA

Potential pricing $P$ = sum potential of buyers and sellers

$$\Phi(P) = (\sum_{j \in B} v_{i(j),j} - p_{i(j)}) + (\sum_{i \in S} p_i)$$

- $\Phi(P_0) \geq 0$ (the initial potential function is at least 0)
- $\Phi(P_k) \geq 0 \;\forall\; k$ (it never goes below 0)
- $\Phi(P_{k+1}) < \Phi(P_k)$ (it strictly decreases at each round)

→ PRIMA O POI TERMINA

□

---

come valuto un assegnamento di prezzi?

- Social Welfare: global happiness of all the participants
  - In other words, the sum of all the sellers and buyers payoffs

$$SW(M) = (\sum_{j \in B} v_{i(j),j} - p_{i(j)}) + (\sum_{i \in S} p_i) = \sum_{j \in B} v_{i(j),j}$$

$$= \text{Total valuation of } M \text{ !!!}$$

- Revenue: total revenue of sellers
  - That is, $REV(M) = \sum_{i \in S} p_i$
  - Awful, as the procedure might return all prices equal $0$

    Example: Case in which all the buyers have the same valuation for all the sellers

Theorem. There exists an $\Omega(1/\log n)$-approximation

---

# AUCTIONS

PROBLEMA • NON POTENDO CONOSCERE LE VALUTAZIONI DEI BUYER, POSSIAMO CHIEDERE, MA QUESTI POSSONO MENTIRE.

SOL • ASTE

- Ascending-bid (or English) auctions:
  - Carried interactively in real time
  - Auctioneer gradually raises the price
  - Bidders drop out, until finally only one remain
  - Such a bidder wins the item at this final price
- Descending-bid (or Dutch) auctions:
  - Carried interactively in real time
  - Auctioneer gradually lowers the price, starting from a high price
  - When a bidders accepts he wins at the current price
  - Used to sell flowers in the Netherlands (Dutch)

- First-price sealed-bid auctions:
  - Bidders simultaneously submit “sealed bids” to the auctioneer
  - Once implemented writing down bids and providing them in sealed envelops
  - The highest bidder wins
  - She pays her bid

- Second-price sealed-bid (Vickrey) auctions:
  - Bidders simultaneously submit “sealed bids” to the auctioneer
  - The highest bidder wins
  - She pays the second highest bid
  - Vickrey wrote the first game-theoretical analysis of these auctions, getting the Nobel prize