“Sapienza” Università di Roma - Dipartimento di Ingegneria Informatica, Automatica e Gestionale

Scheduling

Renato Bruni
bruni@dis.uniroma1.it

Il materiale presentato è derivato da quello dei proff. A. Sassano e C. Mannino
Alcune parti introduttive o esempi sono tratte dal materiale del prof. S. Smriglio (www.di.univaq.it/~oil/index_ita.htm) e del prof. A. Agnetis (www.dii.unisi.it/~agnetis/scheduling.pdf)

Problemi di scheduling

- I problemi di scheduling nascono nel contesto **dell'automazione della produzione**
- Generalmente si tratta di trovare un opportuno **sequenziamento di un insieme di attività** che impiegano risorse scarse, tipicamente macchinari dedicati a specifiche operazioni
- Più in generale, sono tutti i problemi decisionale in cui ha importanza il **tempo**, visto come risorsa scarsa da allocare in modo ottimo
- I modelli di scheduling trovano applicazione anche in molte altre aree, dalla logistica al project management

Esempio 1: industria meccanica

- In una industria meccanica, i centri di lavorazione devono effettuare delle lavorazioni (per es. taglio, fresatura, tornitura) per produrre vari pezzi che vengono poi assemblati insieme
- Ognuna di queste lavorazioni richiede un certo tempo e impegna un certo macchinario
- Spesso esistono tempi di riconfigurazione (*set-up*) delle macchine (per es. cambio della lama, della fresa, etc.), per cui un pezzo non può essere immediatamente lavorato quando la macchina si libera
- Il problema consiste, ad esempio, nel determinare l'ordinamento delle operazioni in modo da terminare tutte le lavorazioni il prima possibile

Esempio 2: elaborazione dati

- Ogni programma che gira su un computer vorrebbe usare la CPU, ma vogliamo poter far girare più programmi contemporaneamente su uno stesso computer
- Uno dei compiti di un sistema operativo è allora quello di disciplinare l'utilizzo della CPU da parte dei diversi programmi
- Ciascun programma ha una certa priorità
- L'obiettivo del sistema operativo è quello di gestire l'insieme dei programmi in modo tale da minimizzare il tempo complessivo di attesa dei programmi, tenendo conto della priorità
- Il sistema operativo può eventualmente interrompere (e poi magari riprendere) certi programmi prima del loro termine per consentire la prosecuzione anche degli altri. Questa modalità operativa prende il nome di *preemption*
- L'impossibilità di interrompere processi una volta avviati si chiama invece *no-preemption* (in alcuni problemi sarà così)

Esempio 3: officina meccanica

- In un'officina di carrozzeria, vi sono quattro stazioni, dedicate rispettivamente a messa in forma, ribattitura, verniciatura, essiccatura a forno
- In ciascuna stazione è attivo un operaio, che può lavorare su una sola autovettura alla volta
- Le autovetture sinistrate richiedono il servizio da parte di alcune stazioni, in un dato ordine (ad esempio non si può riverniciare la carrozzeria prima di avere aggiustato le parti danneggiate)
- Il problema consiste nel gestire le varie operazioni in modo da terminare tutte le lavorazioni nel minor tempo possibile, rispettando l’ordine di sequenziamento sulle macchine

Esempio 4: voli in atterraggio

- In un aeroporto abbiamo l'insieme dei velivoli prossimi all'atterraggio
- Il sistema di controllo dell'aeroporto permette di far atterrare al più un volo alla volta per ogni pista e la durata della manovra di atterraggio è nota per ogni velivolo
- Ad ogni velivolo è inoltre associato un tempo previsto di atterraggio
- Obiettivo del controllore può essere quello di decidere la sequenza in modo da minimizzare la somma (pesata) dei ritardi
- Vincoli aggiuntivi:
  - precedenze fra voli dovute alle coincidenze
  - ….

Come condurre una analisi: esempio fotocopie

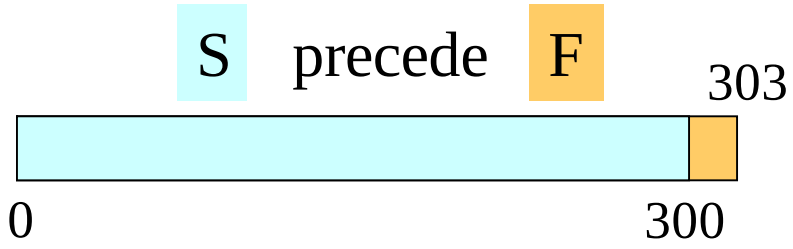
Silvia e Francesca giungono nello stesso momento ad una macchina fotocopiatrice. F deve fare 1 fotocopia, S ne deve fare 100

Il galateo imporrebbe a S di lasciar passare per prima F, che impegna la macchina per un tempo breve e poi la lascia libera

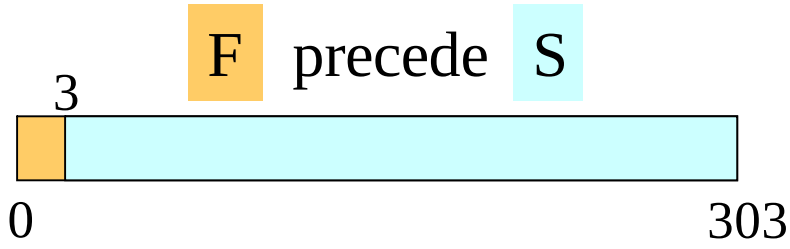
Come valutare se è una buona idea? Serve una **analisi quantitativa**

Supponiamo che l'esecuzione di una fotocopia richieda 3 secondi

Due casi:



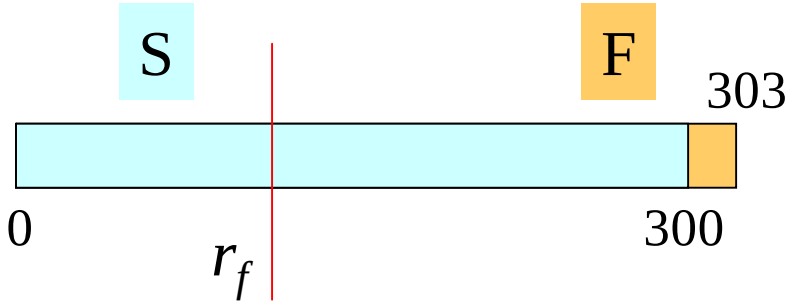
Attesa totale = 300 + 303 = 603



Attesa totale = 3 + 303 = 306

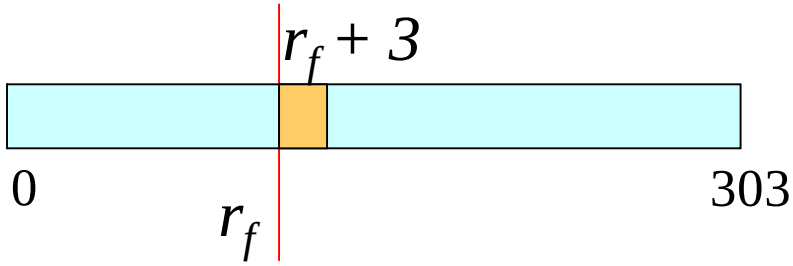
Come condurre una analisi: esempio fotocopie

E se F giungesse dopo che S ha già cominciato? (ad esempio all'istante r_f) Se F attende che S finisca, abbiamo:



Attesa totale = 300 + 303 – r_f = 603 – r_f

Se invece S interrompe le proprie copie, lascia F fare la sua copia e poi riprende:



Attesa totale = 3 + 303 = 306

Primo approccio alla risoluzione

- Quattro parti meccaniche (A), (B), (C), (D) devono essere lavorate e poi assemblate per realizzare un prodotto finale
- Ogni parte deve subire 4 processi su 4 macchinari diversi e condivisi. La sequenza di macchine e il tempo di processamento è fissato per ogni parte, così come l'istante di arrivo del pezzo in catena

Parte	Arrivo	Sequenza Lavoro + tempo in minunti			
A	8.30	M1(60)	M2(30)	M3(2)	M4(5)
B	8.45	M2(75)	M3(3)	M1(25)	M4(10)
C	8.45	M3(5)	M2(15)	M1(10)	M4(30)
D	9.30	M4(90)	M1(1)	M2(1)	M3(1)

Caratteristiche del problema

- Nessuna parte cede la macchina su cui è lavorata prima di aver completato la lavorazione (*no-preemption*)
- La sequenza di lavoro che serve per ottenere una parte **non può essere modificata**: se per fare B devo usare prima M2 e poi M3 (e poi altre) non posso usare prima M3 e poi M2 (e poi altre)
- Tutte le parti devono essere completate per poter cominciare l'assemblaggio

Problema:

A che ora (al minimo) è possibile cominciare ad assemblare le quattro parti?

Una Soluzione Ammissibile

- Il problema equivale a determinare in quale sequenza ciascuna macchina lavorerà le singole parti meccaniche, in modo da minimizzare il tempo di completamento totale

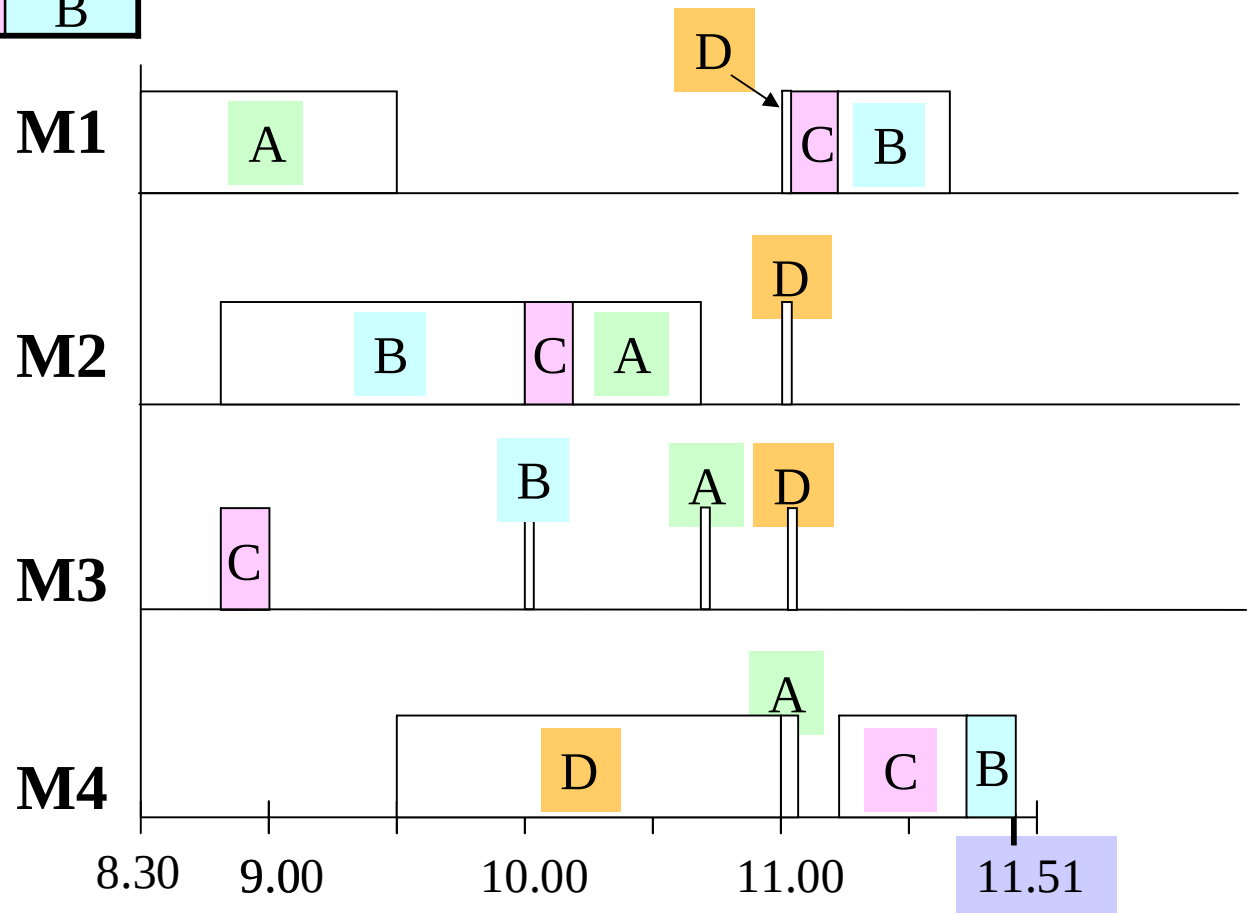
Una possibile soluzione è :

Macch.	P1	P2	P3	P4
M1	A	D	C	B
M2	B	C	A	D
M3	C	B	A	D
M4	D	A	C	B

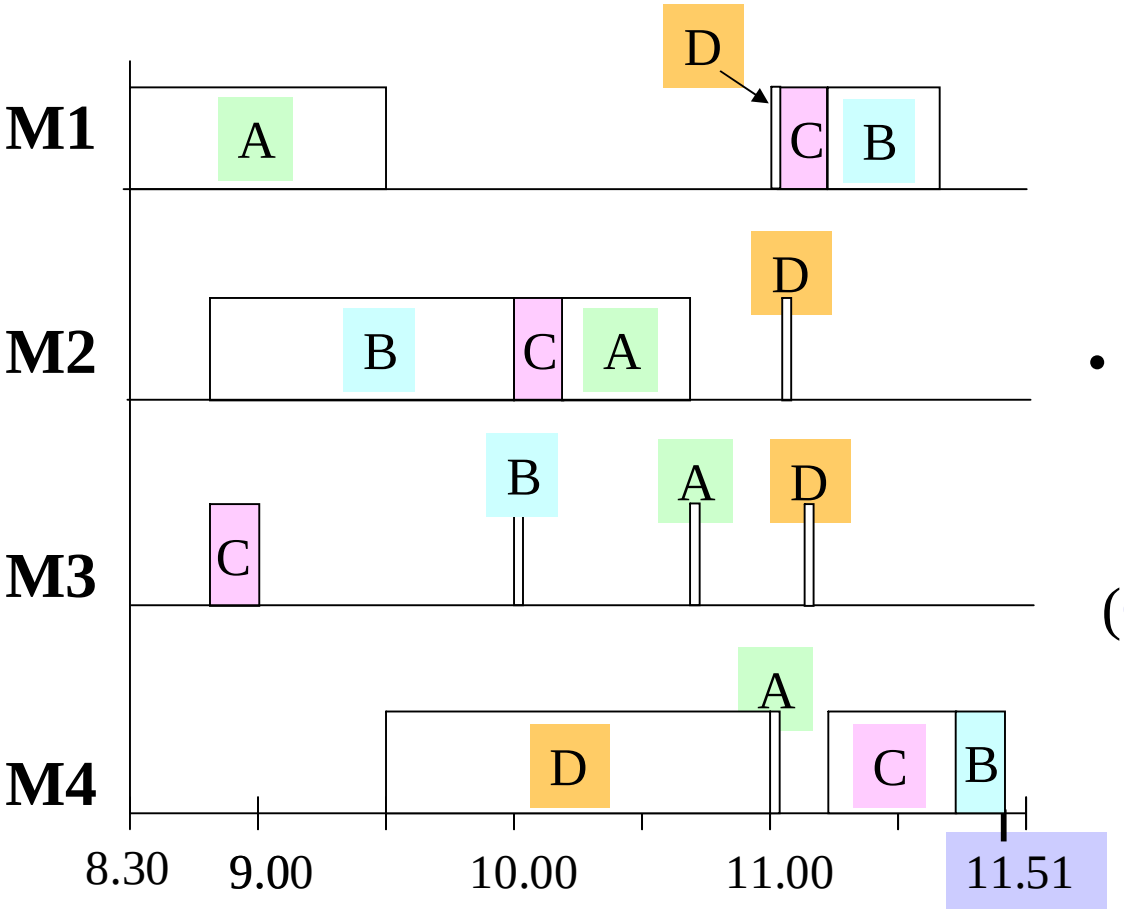
Diagramma di Gantt

Macc	P1	P2	P3	P4
M1	A	D	C	B
M2	B	C	A	D
M3	C	B	A	D
M4	D	A	C	B

Un modo di rappresentare la soluzione è il diagramma di Gantt:



Commenti

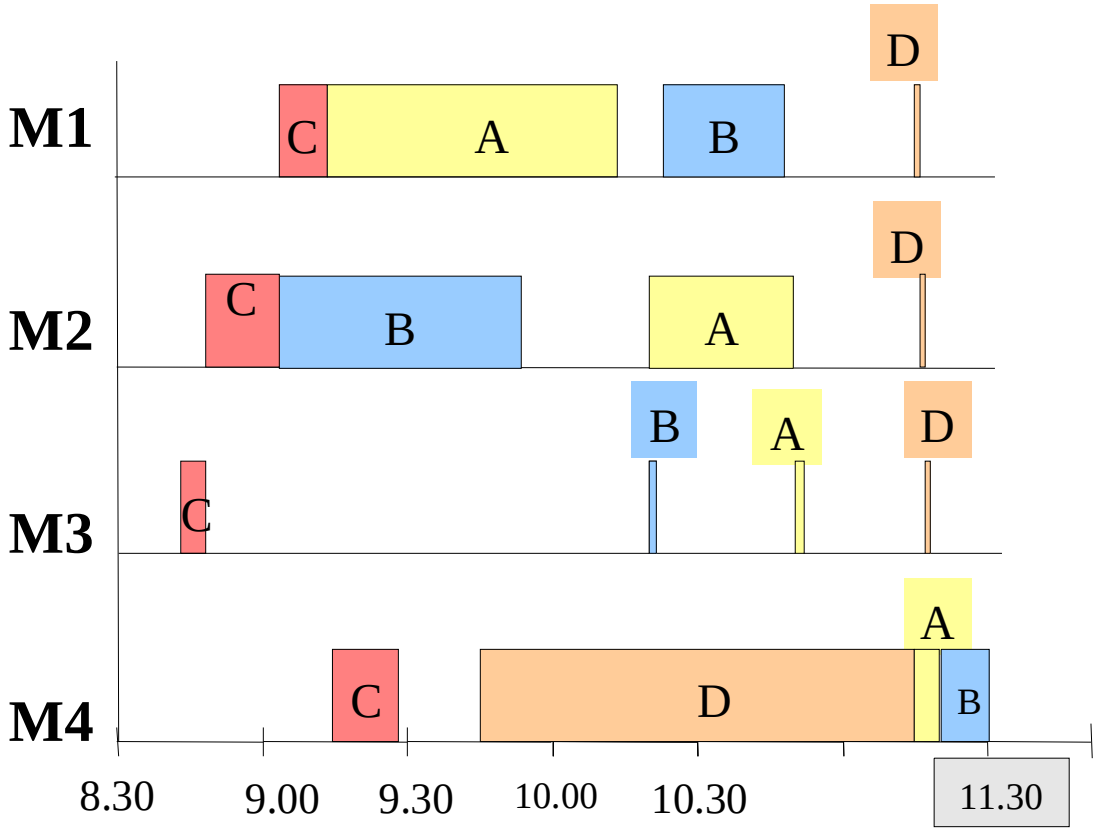


- Un pezzo può essere inattivo (in *idle*) in certi periodi

(C in [10.15, 11.01])

- Ci può essere una macchina che deve restare "*ferma*" anche se c'è una parte che deve ancora essere lavorata: **M4** potrebbe lavorare *B* alle 11.05, ma ciò violerebbe la sequenza di *B*; **M1** potrebbe lavorare *C* alle 9.30 ma ciò violerebbe la sequenza di *C*

Soluzione Ottima



- Soluzione ottima ottenuta tramite formulazione di PL01

Elementi del problema di scheduling

• ***job***: collezione di attività da svolgere. L'insieme dei *job* è spesso indicato con *J*

Documento da fotocopiare, esecuzione di un programma di calcolo, produzione di un autoveicolo (fatta da più attività)

Un job può essere costituito da una singola attività (*task*) o da un insieme di attività tecnologicamente o logicamente legate

• ***macchina***: risorsa necessaria all'esecuzione delle attività. L'insieme delle macchine disponibili è *M*

Fotocopiatrice, CPU, robot industriali

• ***operazione***: parte di un job eseguita su una macchina (un job può richiedere più macchine, e quindi una operazione su ognuna)

Documento 1 su fotocopiatrice xerox, veicolo 5 in verniciatura (è una delle varie operazioni del job produzione autoveicolo, quella fatta sulla macchina verniciatrice)

Definizione di Operazione

Def. Operazione: Coppia (j,m) con $j \in J$ job, e $m \in M$ macchina

- Le operazioni sono l'elemento centrale dei modelli di scheduling
- Le relazioni di precedenza, infatti, riguardano coppie di operazioni (dello stesso *job* o di *job* differenti)

  La verniciatura del veicolo 5 deve avvenire prima della *verniciatura del veicolo 6* e dopo *il montaggio del motore sul veicolo 5*

- Il concetto di job è comunque utile per classificare le operazioni
- Nei modelli che considereremo, una macchina può eseguire al più un'operazione alla volta
- L'insieme di tutte le operazioni di un problema è indicato con O

Soluzioni di un problema di scheduling

- Consideriamo solo il caso in cui le operazioni non possono essere interrotte (*no-preemption*)
- Una **soluzione** di un problema di scheduling è uno *schedule* (*piano*) che fornisce per ogni operazione $o \in O$ il suo istante iniziale s_o (o finale C_o)
- Una permutazione dei job che definisce l'ordine con cui i job sono processati su una certa macchina si dice *sequenza*
- Nei problemi con più macchine, in genere l'insieme di macchine utilizzate da un *job* e la loro sequenza è fissata in partenza. Altrimenti bisogna risolvere anche un problema di *routing*
- In genere, è possibile identificare per ogni job un'operazione *inizio job* e un'operazione *fine job*. Nel caso di job con una sola operazione, *inizio* e *fine* coincidono
- Nel caso di problema con più job, si considera anche un'operazione di inizio lavoro *l* (che precede tutti i job) e una di fine lavoro *t* (che li segue)

Attributi del job e delle operazioni

• ***tempo di processamento*** p_{ji} : durata del processamento del job j sulla macchina i (= durata dell'operazione (j,i))

• ***release date*** r_j : tempo in cui il job j arriva nel sistema, ovvero primo istante ammissibile per l'operazione *inizio job*

• ***due date*** d_j: tempo entro il quale si desidera che il job j sia completato (data di consegna), ovvero *deadline* sull'operazione *fine job*

• ***vincoli di precedenza***: vincoli fra operazioni dello stesso job o di job diversi

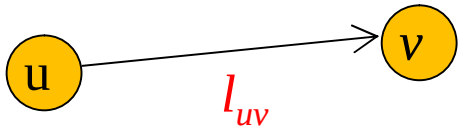
Problema di scheduling:

Trovare un piano s che minimizza una data funzione di costo $c(s)$, soddisfacendo tutti i vincoli di precedenza, di *release date* e *due date*, e tale che due operazioni su una stessa macchina non si sovrappongano temporalmente

Relazioni di precedenza fra operazioni

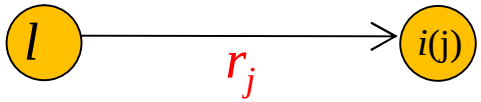
- Le relazioni di precedenza fra due operazioni u e v possono essere tutte poste nella forma del tipo $s_v - s_u \geq l_{uv}$

- Possiamo rappresentare associando un nodo a ogni operazione e un arco pesato a ogni relazione di precedenza



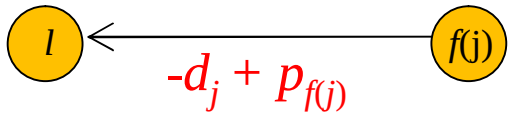
- *Release date* r_j e *due date* d_j di un job j sono vincoli di precedenza fra l'operazione inizio lavoro l e le operazioni di inizio $i(j)$ e fine job $f(j)$ (con tempo di processamento $p_{f(j)}$)

Release date $s_{i(j)} - s_l \geq r_j$ ⇒



Due date

$s_{f(j)} - s_l \leq d_j - p_{f(j)}$ ⇒ $s_l - s_{f(j)} \geq -d_j + p_{f(j)}$ ⇒



Relazioni di precedenza disgiuntive

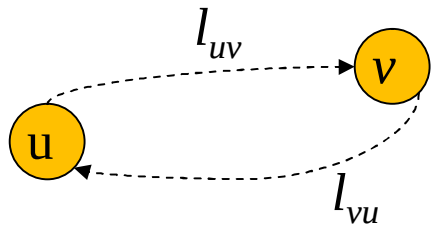
- Un ruolo particolare è svolto dalle relazioni di precedenza fra operazioni su una stessa macchina
- Infatti, l'ordine con cui due operazioni $u = (j,m)$, e $v = (k,m)$ si svolgono su m non è dato a priori, ed è anzi la principale decisione da prendere

> *l'operazione u viene effettuata prima dell'operazione v*
>
> **oppure**
>
> *l'operazione v viene effettuata prima dell'operazione u*

- Vincolo disgiuntivo: esprime matematicamente la disgiunzione

$$(s_v - s_u \geq l_{uv}=p_{(jm)}) \vee (s_u - s_v \geq l_{vu}=p_{(km)})$$

- Ogni soluzione ammissibile deve soddisfare almeno uno dei due vincoli (e prima di risolvere non sappiamo quale dei due!)
- Si rappresenta con un *arco disgiuntivo*, ovvero una *coppia di archi orientati antiparalleli*



Scheduling e programmazione disgiuntiva

- I problemi di scheduling vengono allora modellati con la programmazione disgiuntiva
- O insieme delle operazioni (*incluse l'operazione di inizio e fine lavoro*)
- A insieme dei vincoli di precedenza
- D insieme dei vincoli di precedenza disgiuntivi

Problema di Scheduling

$$
\begin{aligned}
&\min c(s) \\
&s.t. \\
&\quad s_v - s_u \geq l_{uv} && uv \in A \\
&\quad (s_v - s_u \geq l_{uv}) \ \vee \ (s_u - s_v \geq l_{vu}) && \{(uv),(vu)\} \in D \\
&\quad s_u \in R_+ && u \in O
\end{aligned}
$$

Programmazione Disgiuntiva: formulazione

- E' possibile rappresentare un problema di programmazione disgiuntiva come problema di programmazione intera mista
- Dato un vincolo disgiuntivo $d \in D$: $(s_v - s_u \geq l_{uv}) \vee (s_u - s_v \geq l_{vu})$ ogni soluzione ammissibile s deve soddisfare una delle due condizioni della disgiunzione, ma **non** tutte e due
- Se ad esempio s soddisfa la prima condizione, la seconda deve "*scomparire*" dal problema (non potrebbe essere soddisfatta!)
- A tal scopo si usa la cosiddetta "*big M*"
- M è una costante molto grande, scelta in modo da rendere banalmente soddisfatta una condizione del tipo visto tramite il seguente meccanismo (nella scelta del valore di M tenere sempre conto di problemi numerici!)

$$s_v^* - s_u^* \geq l_{uv} - M \qquad \text{o} \qquad s_u^* - s_v^* \geq l_{vu} - M$$

Programmazione Disgiuntiva: formulazione

- Per ogni $d \in D$: $(s_v - s_u \geq l_{uv}) \vee (s_u - s_v \geq l_{vu})$ introduciamo una variabile binaria x_d tale che

$$\begin{cases} x_d = 0 \Rightarrow & s_v - s_u \geq l_{uv} \\ x_d = 1 \Rightarrow & s_u - s_v \geq l_{vu} \end{cases}$$

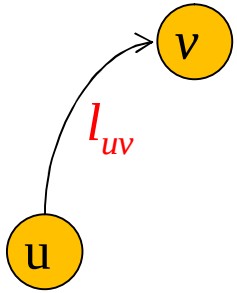
- In pratica, la variabile indicatrice x_d determina quale delle due condizioni del vincolo disgiuntivo è soddisfatta dalla soluzione (rendendo l'altra ridondante)

- Per ottenere una formulazione lineare, si sostituisce il vincolo disgiuntivo con la coppia di vincoli (congiuntivi):

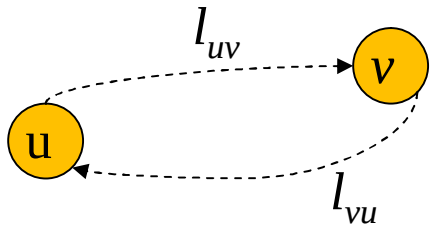
$$\begin{cases} s_v - s_u \geq l_{uv} - M\, x_d \\ s_u - s_v \geq l_{vu} - M\,(1 - x_d) \end{cases}$$

Il grafo disgiuntivo

- Grafo disgiuntivo $G(V, A, D)$: rappresentazione per problemi di scheduling. Contiene nodi V, archi orientati A e *archi disgiuntivi* D
- Insieme nodi $V = O$: coincide con le operazioni O del problema (inclusi eventualmente i nodi $\{l,t\}$ di inizio e fine lavoro)
- Insieme archi: A . Rappresentano precedenze stabilite fra operazioni (e.g. la sequenza di operazioni di un *job*).



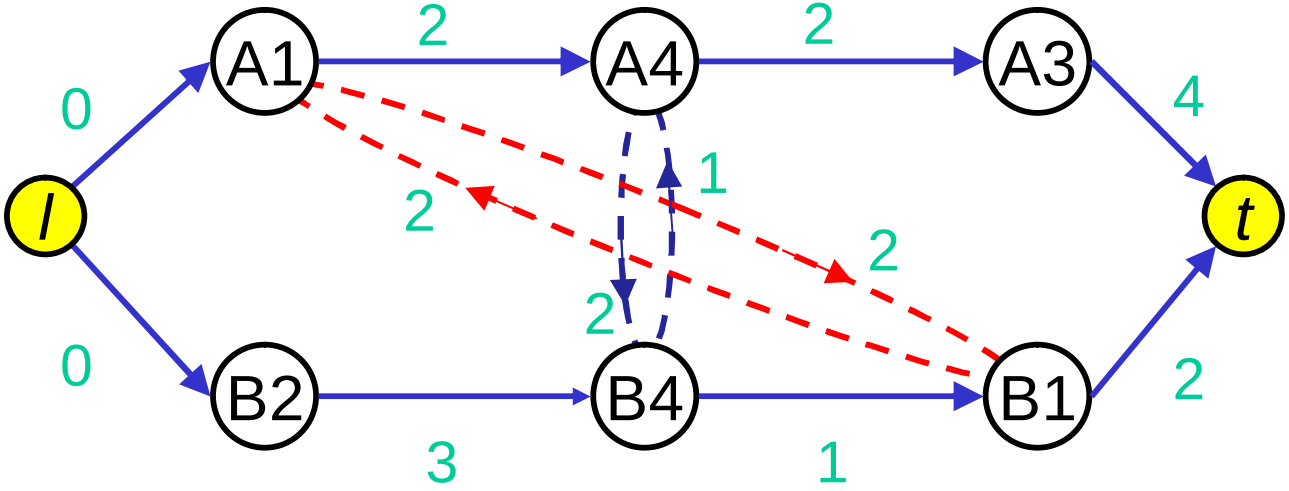
- Insieme archi disgiuntivi: D. Un arco disgiuntivo è una coppia di archi orientati $\{(u,v),(v,u)\}$ e rappresenta una precedenza *disgiuntiva*



Esempio di grafo disgiuntivo

- Esempio: 2 *job* (A, B), 4 *macchine*
- 3 operazioni per ogni job: A deve essere lavorato sulla macchina 1, poi sulla 4 e quindi sulla 3; B sulla 2, poi sulla 4 e quindi sulla 1
- Tempi di lavorazione delle operazioni sono i seguenti

job	$m_j(1)$ [$p_{j,\,m(1)}$]	$m_j(2)$ [$p_{j,m(2)}$]	$m_j(3)$ [$p_{j,m(3)}$]
A	1 [2]	4 [2]	3 [4]
B	2 [3]	4 [1]	1 [2]



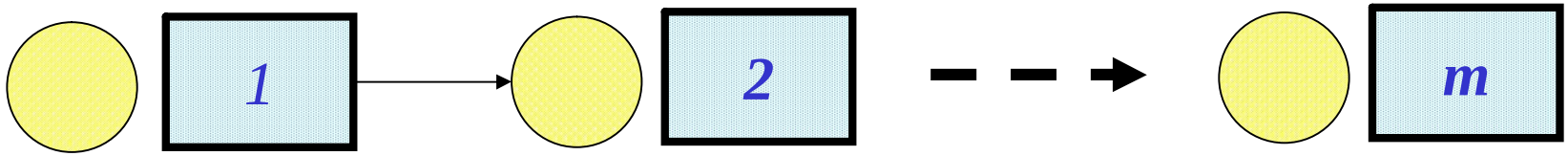
- OSS: *G* ha 2 archi disgiuntivi $D = \{\{(A1,B1),(B1,A1)\}, \{(A4,B4),(B4,A4)\}\}$

La classificazione a tre campi $\alpha/\beta/\gamma$

- Data che esistono molti tipi d problemi di scheduling, si usa una classificazione standard di questi problemi con notazione a tre campi $\alpha/\beta/\gamma$:
- α indica il sistema di macchine
- β rappresenta vincoli e modalità di processamento
- γ indica l'obiettivo
- Di seguito si descrivono brevemente questi campi

Campo α

- Macchina singola **(1)** : ciascun job richiede una singola operazione da eseguirsi sull'unica macchina disponibile
- Macchine identiche parallele **(P_m)** : ciascun job richiede una singola operazione da eseguirsi su una qualunque delle *m* macchine identiche.
- Flow shop **(F_m)** : *m* macchine in serie. Ciascun job deve essere processato su ciascuna di esse (*m* operazioni). Tutti i job hanno lo stesso *routing*. Una volta terminata un'operazione il job viene posto nel buffer della macchina successiva



- Job shop **(J_m)** : ciascun job deve essere processato da un sottoinsieme di un insieme di *m* macchine, secondo un proprio routing (non necessariamente prestabilito)

Campo β

- Release dates **(r_j)** : il job *j* non può iniziare il processamento prima dell'istante r_j
- Tempi di set-up **(s_{jk})** : tempo richiesto per il riattrezzaggio delle macchine fra i job *j* e *k*. Se dipende dalla macchina *i*, si esprime con s^i_{jk}
- Preemption ***(prmp)*** : è ammesso interrompere un'operazione su una macchina prima del suo completamento (e.g. per iniziare una nuova operazione). Quando l'operazione viene ripresa, essa richiede solo il tempo *rimanente* di processamento
- Vincoli di precedenza ***(prec)*** : un job può essere processato solo se tutti i job di un certo insieme sono stati completati. Se ciascun job ha al più un predecessore e un successore β = ***chain***
- Breakdown ***(brkdwn)*** : le macchine non sono sempre disponibili, ma hanno periodi fissati di interruzione del servizio
- …

Campo γ

C_j	tempo di completamento del job j
$L_j = C_j - d_j$	*lateness* del job j (rispetto alla *due date* d_j del job)
$T_j = max(L_j, 0)$	*tardiness* del job j (è la lateness se >0, 0 altrim.)
$U_j = 1$ se $C_j > d_j$ (*tardy* job) e 0 altrimenti	

- Oss: se $f(j)$ è l'ultima operazione del *job j*, e $p_{f(j)}$ la sua durata, allora in regine di *no-preemption* si ha $C_j = s_{f(j)} + p_{f(j)}$
- Makespan **(C_{max})** : $max(C_1, \ldots, C_n)$ tempo di completamento dell'ultimo job a essere completato
- Massima Lateness **(L_{max})**
- Tempo totale pesato di completamento **($\Sigma w_j C_j$)**
- Tardiness totale pesata **($\Sigma w_j T_j$)**
- Numero di tardy job **(ΣU_j)**

Il problema su singola macchina $1/-/\Sigma w_j C_j$

- Dati n job $1,\ldots,n$ e una sola macchina
- A ogni job è quindi associata un'unica operazione
- p_j: tempo di processamento (operazione associata al) job j

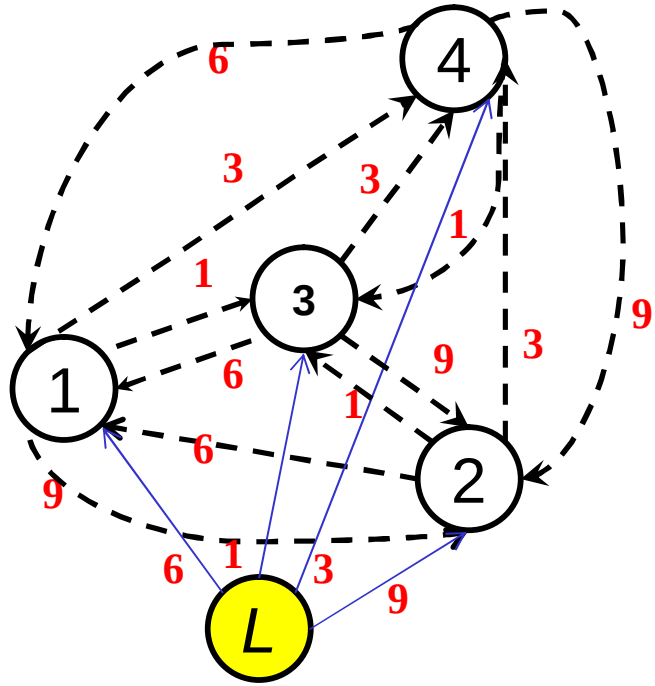
  Assunzione $\boldsymbol{p_j > 0}$ per ogni j
- w_j : peso del job j.
- $w_j \cdot C_j$: costo di completamento del job j
- s_j: istante d'inizio (variabile) del job j
- I job non possono essere interrotti (no-preemption) $\rightarrow C_j = s_j + p_j$

Problema : trovare uno scheduling s che minimizzi il tempo di completamento pesato $\Sigma_j w_j C_j = \Sigma_j w_j (s_j + p_j)$

- Nella classificazione a tre indici si chiama *Problema* $1/-/\Sigma w_j C_j$

Formulazione disgiuntiva

Job	1	2	3	4
p_j	6	9	1	3
w_j	2	5	9	6



Grafo disgiuntivo

Questa volta usiamo variabili che indicano istante di completamento del job: $C_j = s_j + p_j$

$$min \;\; 2C_1 + 5C_2 + 9C_3 + 6C_4$$

s.t.

$$C_j - C_L \geq p_j \qquad j = 1, \ldots, 4$$

$$(C_j - C_i \geq p_j) \vee (C_i - C_j \geq p_i) \qquad i \neq j \;,\; i, j = 1, \ldots, 4$$

$$C_L = 0, \qquad C_j \in R_+ \quad j = 1, \ldots, n$$

- Senza perdita di generalità si pone $C_L = 0$, che implica $C_j \geq p_j \quad \forall\, j$

Formulazione 0,1

Job	1	2	3	4
p_j	6	9	1	3
w_j	2	5	9	6

min 2C1 + 5C2 + 9C3 + 6C4

s.t.

C1 > 6

C2 > 9

C3 > 1

C4 > 3

C1 - C2 + 100 x12 > 6

C2 - C1 - 100 x12 > -91

C1 - C3 + 100 x13 > 6

C3 - C1 - 100 x13 > -99

C1 - C4 + 100 x14 > 6

C4 - C1 - 100 x14 > -97

C2 - C3 + 100 x23 > 9

C3 - C2 - 100 x23 > -99

C2 - C4 + 100 x24 > 9

C4 - C2 - 100 x24 > -97

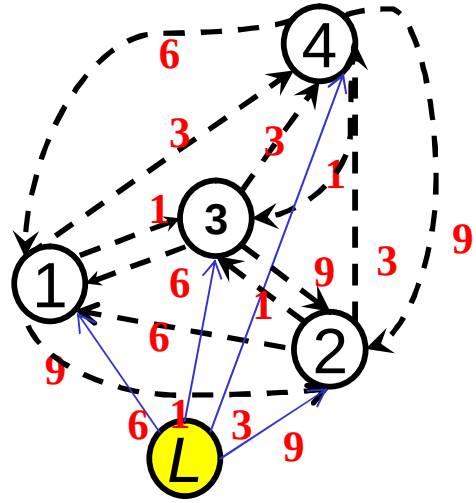
C3 - C4 + 100 x34 > 1

C4 - C3 - 100 x34 > -97

binary

x12 x13 x14 x23 x24 x34

end



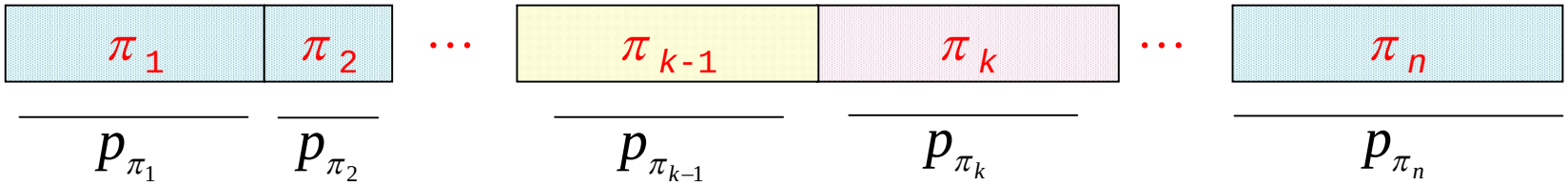
Grafo disgiuntivo

- Usiamo le variabili x_{uv} di alternativa tra: prima u poi v o prima v poi u
- File tipo “lp”, che può essere letto dai principali solutori commerciali
- Il segno > denota il $\geq$

Il problema $1/-/\Sigma w_j C_j$

- OSS: non conviene lasciare la macchina inattiva fra il completamento di un *job* e l`inizio del successivo
- In ogni soluzione ottima, un job comincia quindi esattamente quando il precedente termina. Chiamiamo questa *soluzione compatta*.

⇒ La sequenza (permutazione) $\pi_1, \ldots, \pi_n$ dei job sulla macchina determina univocamente la soluzione compatta



- Il tempo di completamento del k-esimo job vale $C_{\pi_k} = \sum_{r=1}^{k} p_{\pi_r}$

Problema $1/-/\Sigma w_j C_j$: trovare la sequenza dei job $\pi_1, \ldots, \pi_n$ che minimizza il tempo di completamento pesato $\Sigma_j w_j C_j$

La regola di Smith

Regola di Smith: ordina per valori non crescenti del rapporto w_j/p_j

Quindi, se $\pi_1, \ldots, \pi_n$ è una sequenza prodotta dalla regola di Smith:

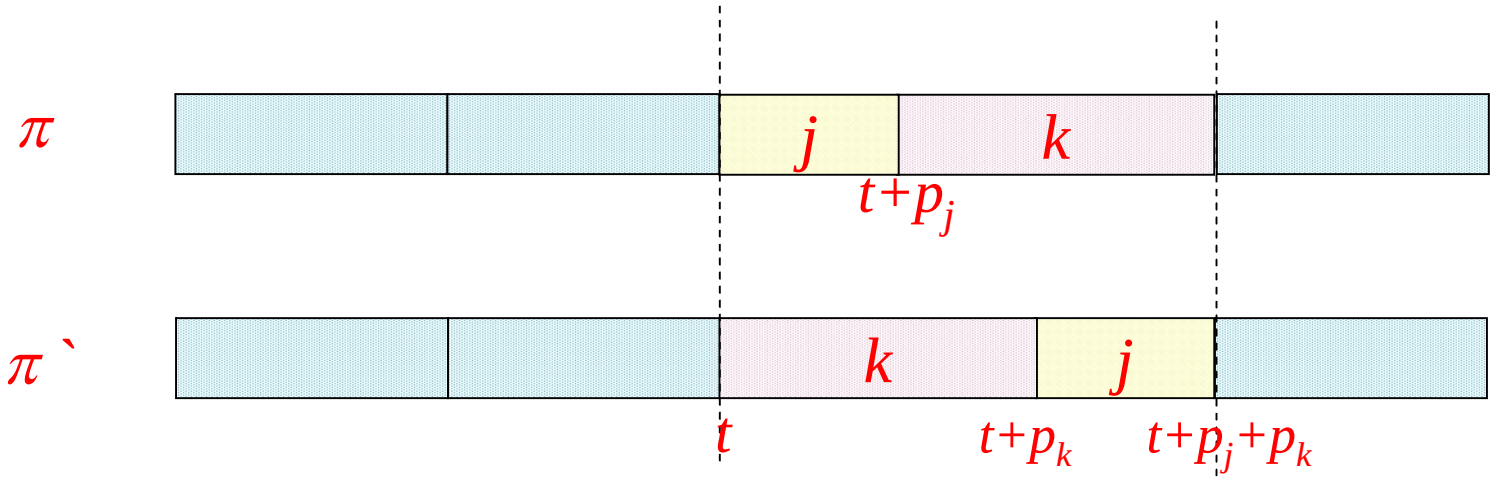
$$\frac{w_{\pi_1}}{p_{\pi_1}} \geq \frac{w_{\pi_2}}{p_{\pi_2}} \geq \ldots \geq \frac{w_{\pi_n}}{p_{\pi_n}}$$

- Se alcuni rapporti coincidono, possiamo scegliere in uno qualsiasi di essi (esistono più sequenze che soddisfano la regola di Smith)

Th. 6.1 (*Della regola di Smith*) La soluzione compatta associata una sequenza $\pi_1, \ldots, \pi_n$ prodotta dalla regola di Smith è ottima per il problema $1/\text{-}/\sum w_j C_j$

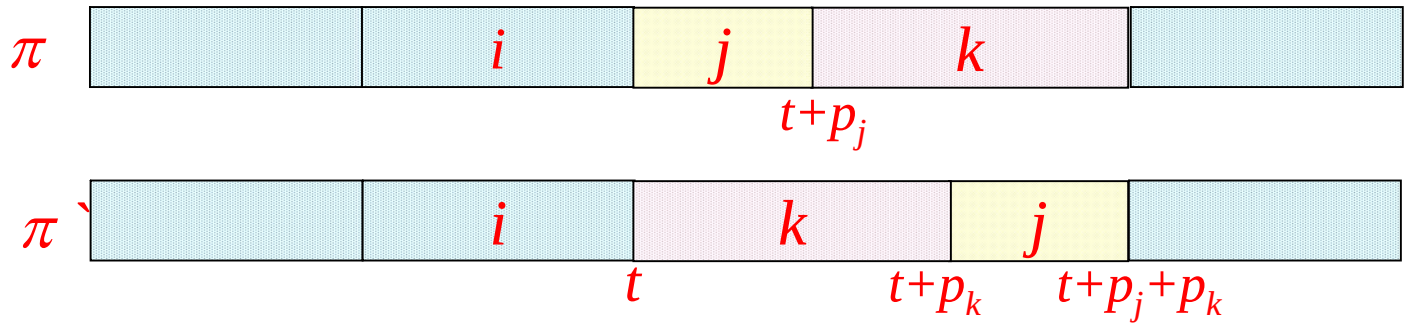
Dimostrazione Teorema Regola di Smith

- Sia π una sequenza ottima. Per assurdo supponiamo che π non rispetti la regola di Smith
- Esistono quindi due *job* adiacenti, j e k, tale che k viene immediatamente dopo j in π e si ha: $w_j/p_j < w_k/p_k$
- Consideriamo la sequenza π ` ottenuta da π invertendo j e k



- OSS. Tutti i tempi di completamento dei job rimangono invariati tranne quelli del job j e del job k

Dimostrazione Regola di Smith



- Sia t il tempo di completamento del *job* che precede j in π (e k in π `)
- Siano C e C` i tempi di completamento in $S$ e $S$`, rispettivamente

$C`_j = t+p_j+p_k$ $\quad$ $C`_k = t+p_k$ $\quad$ $C_j = t+p_j$ $\quad$ $C_k = t+p_j+p_k$

Differenza di costo fra C' e C: $\Delta = wC' - wC = w_j(C`_j - C_j) + w_k(C`_k - C_k)$

$C`_j \quad C_j \quad C`_k \quad C_k$

Sostituendo $\Delta = w_j(t+p_j+p_k - t-p_j) + w_k(t+p_k - t - p_j - p_k) = w_j(p_k) + w_k(-p_j)$

Da $w_j/p_j < w_k/p_k \Longrightarrow w_j p_k < w_k p_j \Longrightarrow \Delta = w_j p_k - w_k p_j < 0$

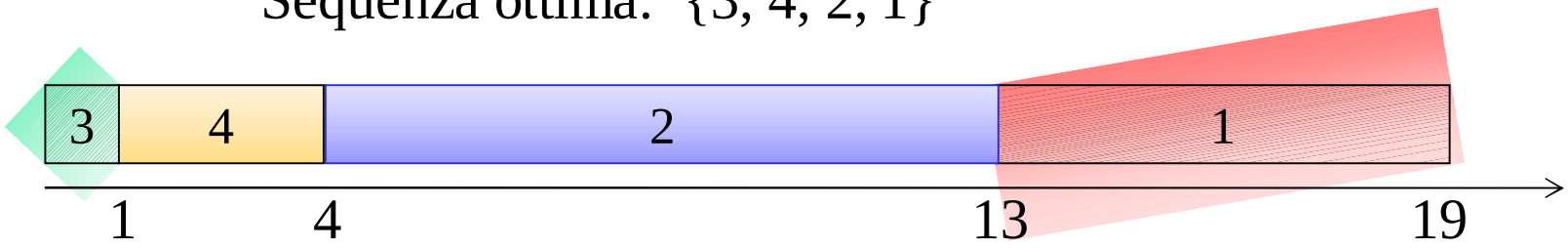
e π non è ottima, contraddizione ■

Esempio

Job	1	2	3	4
p_j	6	9	1	3
w_j	2	5	9	6

$w_1/p_1 = 1/3$, $w_2/p_2 = 5/9$, $w_3/p_3 = 9$, $w_4/p_4 = 2$

Sequenza ottima: {3, 4, 2, 1}



$C_1 = 19$ $C_2 = 13$, $C_3 = 1$, $C_4 = 4$

$w_1C_1 = 38$, $w_2C_2 = 65$, $w_3C_3 = 9$, $w_4C_4 = 24$

Costo totale $= w_1C_1 + w_2C_2 + w_3C_3 + w_4C_4 = 136$

- Da notare che il *makespan* è uguale per ogni sequenza e vale 19

Sequenze equivalenti

Lemma. 6.2 (*Sequenze Equivalenti*) Se ogni job $j \in N$ ha un peso w_j uguale al suo tempo di processamento p_j allora ogni sequenza è ottima

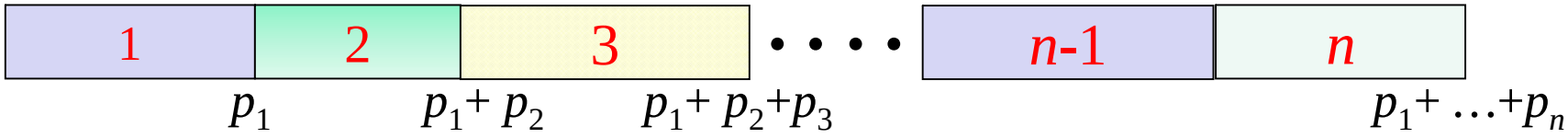
Infatti per ogni $j \in N$ si ha $w_j/p_j = 1$

Dal Teorema 6.1 (*della regola di Smith*) segue che ogni sequenza è ottima ■

Lemma. 6.3 (*Valore della soluzione* $w_j = p_j$) Se per ogni $j \in N$ si ha $w_j = p_j$ allora la soluzione ottima vale:

$$f(N) = \frac{1}{2}\left(\left(\sum_{j \in N} p_j\right)^2 + \sum_{j \in N} p_j^2\right)$$

- Per il Lemma 6.2 ogni sequenza π è ottima: quindi anche $\pi = \{1,2,\ldots,n\}$



$\Rightarrow$ $C_1 = p_1$ $C_2 = p_1 + p_2$ • • • • $C_n = p_1 + \ldots + p_n$

Dimostrazione Lemma 6.3

- Quindi lo schedule: $C_j = \sum_{r=1}^{j} p_r$ è ottimo

e il suo valore $f(N)$ è dato da

$$f(N) = \sum_{j \in N} w_j C_j = \sum_{j \in N} p_j C_j = \sum_{j \in N} \left(p_j \sum_{r=1}^{j} p_r \right) =$$

$$p_1 p_1 + p_2 (p_1 + p_2) + p_3 (p_1 + p_2 + p_3) + \ldots + p_n (p_1 + p_2 + \ldots + p_n)$$

$$\Rightarrow \quad \boxed{f(N) = \sum_{j \in N} p_j^2 + \sum_{i \in N, j \in N, i > j} p_i \, p_j}$$

Sapendo che $\left(\sum_{j \in N} p_j \right)^2 = (p_1 + p_2 + \ldots + p_n)^2 = \sum_{j \in N} p_j^2 + \sum_{i \in N, j \in N, i > j} 2 p_i \, p_j$

$$\Rightarrow \quad \left(\sum_{j \in N} p_j \right)^2 + \sum_{j \in N} p_j^2 = \sum_{j \in N} 2 p_j^2 + \sum_{i \in N, j \in N, i > j} 2 p_i \, p_j = 2 f(N)$$

■

Poliedro degli schedule ammissibili

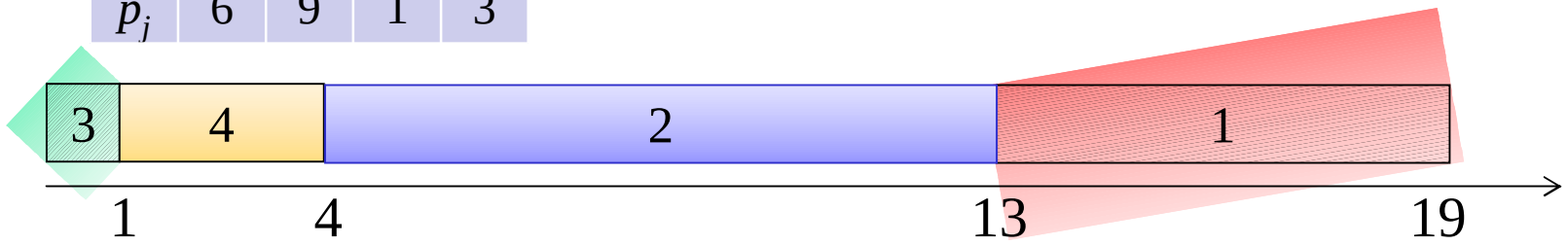
- Descriviamo l'insieme $S(p) \subseteq R^N$ degli scheduling ammissibili C
- In altri termini $C = (C_1, \ldots, C_n)^T \in S(p)$ se e solo se esiste una

sequenza $\pi = \{\pi_1, \ldots, \pi_n\}$ tale che

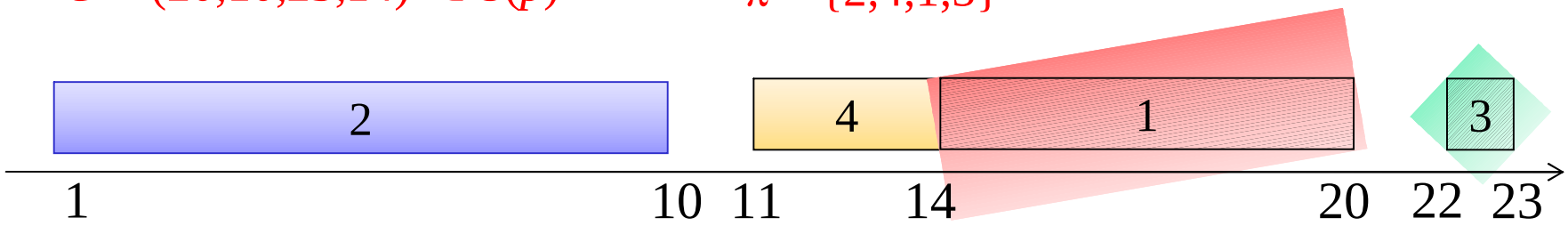
$$C_{\pi_1} \geq p_{\pi_1}$$
$$C_{\pi_k} \geq C_{\pi_{k-1}} + p_{\pi_k} \qquad k = 2, \ldots, n$$

J	1	2	3	4
p_j	6	9	1	3

$C = (19,13,1,4)^T \in S(p)$ $\pi = \{3,4,2,1\}$



$C = (20,10,23,14)^T \in S(p)$ $\pi = \{2,4,1,3\}$



Proiezione delle soluzioni

Sia $C = (C_1, \ldots, C_n)^T \in R^N$ uno schedule ammissibile, *i.e.* $C \in S(p)$

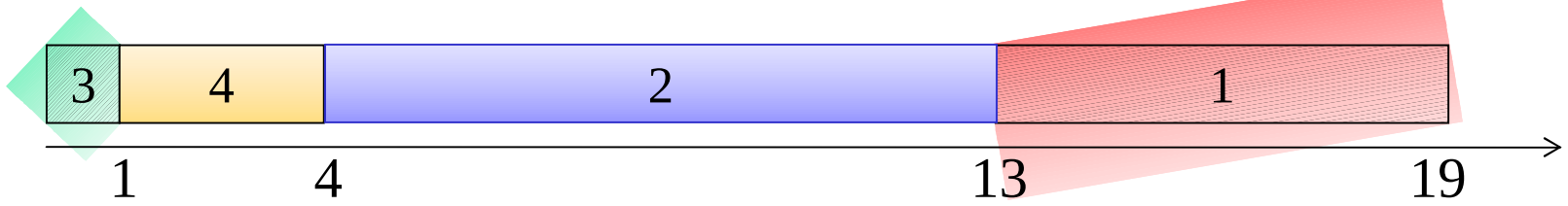
Sia $A \subseteq N$ un sottoinsieme di *job* e sia $S_A(p_A)$ l'insieme degli scheduling ammissibili per i soli job in A (p_A sono i tempi dei job in A)

Allora la proiezione C_A di C nello spazio R^A soddisfa $C_A \in S_A(p_A)$

Esempio $N = \{1,2,3,4\}$, $A = \{2,3\}$

J	1	2	3	4
p_j	6	9	1	3

$C = (19,\mathbf{13},\mathbf{1},4)^T \in S(p)$ $\quad$ $\pi = \{3,4,2,1\}$

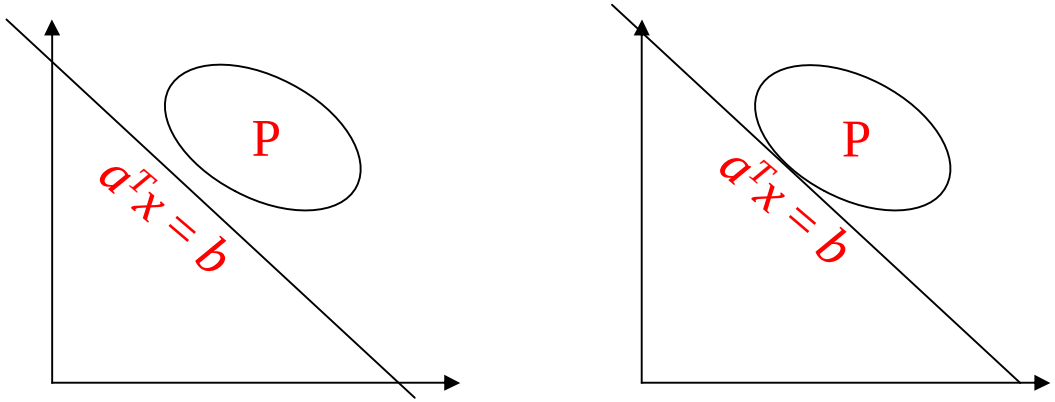


$C_A = (\mathbf{13},\mathbf{1})^T$ $\quad$ $\pi_A = \{3,2\}$

Naturalmente l'ottimalità di C rispetto a un sistema di pesi w non implica l'ottimalità di C_A in $S_A(p_A)$ rispetto w_A (C_A è in generale non compatta)

Vincoli validi

Def. Sia dato un insieme di punti $P \subseteq R^n$ e siano $a \in R^n$ e $b \in R$
Il vincolo $a^T x \geq b$ è detto *valido* per P se e solo se è soddisfatto da tutti i punti di P ovvero se e solo se non esiste $x' \in P$ tale che $a^T x' < b$



Lemma. Sia dato un insieme di punti $P \subseteq R^n$ e sia $a \in R^n$. Sia $b = min\{a^T x: x \in P\}$
Allora $a^T x \geq b$ è un vincolo *valido* per P

Infatti, se esistesse un punto $x' \in P$ tale che $a^T x' < b$ si avrebbe

$min\{a^T x: x \in P\} \leq a^T x' < b$ contraddizione

■

Proprietà degli schedule ammissibili

Teorema 6.4. Sia $A \subseteq N$ un sottoinsieme di *job* e sia

$$f(A) = \frac{1}{2}\left(\left(\sum_{j\in A} p_j\right)^2 + \sum_{j\in A} p_j^2\right)$$

Allora il vincolo $\sum_{j\in A} p_j C_j \geq f(A)$ è valido per $S(p)$

Dim. Qual è il valore minimo che può assumere la quantità $\sum_{j\in A} p_j C_j$, $C \in S(p)$?

Sia C^* la soluzione ottima del problema $\min\left\{\sum_{j\in A} p_j C_j : C \in S(p)\right\}$

C^*_A proiezione di C^* nello spazio R^A

$$C^*_A \in S_A(p_A) \quad \Rightarrow \quad \sum_{j\in A} p_j C^*_j \geq \min\left\{\sum_{j\in A} p_j Q_j : Q \in S_A(p_A)\right\}$$

dal Lemma 6.3 $\min\left\{\sum_{j\in A} p_j Q_j : Q \in S_A(p_A)\right\} = f(A)$ $\Rightarrow$

$$\left\{\sum_{j\in A} p_j C_j : C \in S(p)\right\} \geq \min\left\{\sum_{j\in A} p_j C_j : C \in S(p)\right\} \geq \min\left\{\sum_{j\in A} p_j Q_j : Q \in S_A(p_A)\right\} = f(A)$$

■

Esempio

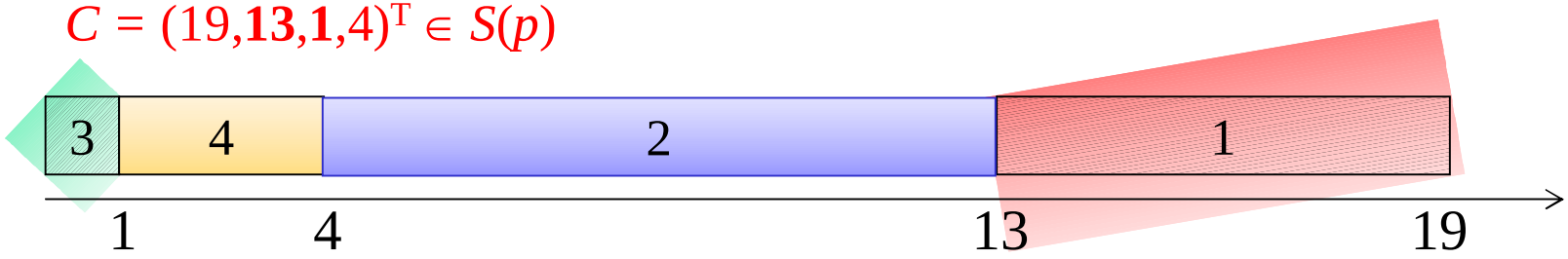
Esempio $N = \{1,2,3,4\}$, $A = \{2,3\}$

J	1	2	3	4
p_j	6	9	1	3

$$f(A) = \tfrac{1}{2}\ (1+9)^2 + \tfrac{1}{2}\ (1^2 + 9^2\ ) = 91$$

$$\sum_{j\in A}\ p_j C_j \geq f(A) \quad \Rightarrow \quad 9C_2 + C_3 \geq 91$$

$C = (19,\mathbf{13},\mathbf{1},4)^T \in S(p)$



C soddisfa il vincolo

$$9C_2 + C_3 = 9 \cdot 13 + 1 = 118 \geq 91$$

Il poliedro degli schedule ammissibili

- Per ogni $A \subseteq N$ il vincolo $\sum_{j \in A} p_j C_j \geq f(A)$ è valido per $S(p)$
- Chiamiamo $S_c(p)$ il *poliedro degli scheduling compatti*, e cioè l'involucro convesso degli scheduling compatti associati a p
- Si può far vedere che $S_c(p)$ coincide con

$S_c(p)$:

$$\sum_{j \in A} p_j C_j \geq f(A) \quad \forall A \subseteq N$$
$$\sum_{j \in N} p_j C_j = f(N)$$
$$C \in R^n$$

- Gli schedule compatti coincidono coi vertici $Ext(S_c(p))$ di $S_c(p)$

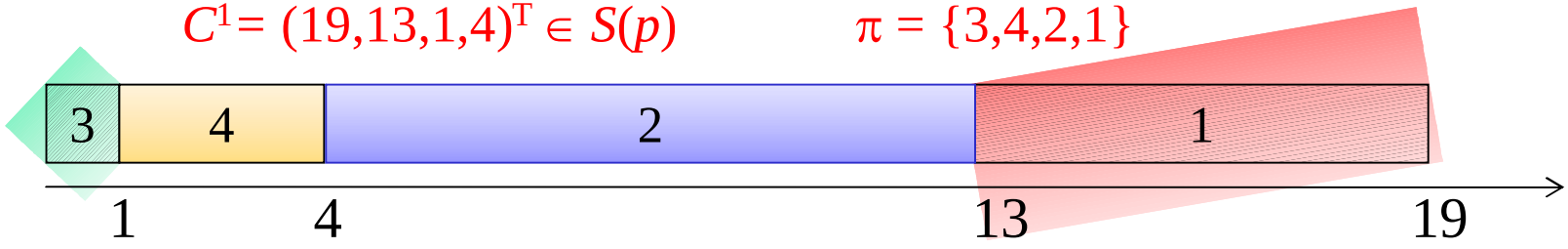
Esempio: vertici di $S_c(p)$

- I vertici $Ext(S_c(p))$ di $S_c(p)$ corrispondono agli scheduling compatti.

Esempio:

J	1	2	3	4
p_j	6	9	1	3

$C^1 = (19,13,1,4)^T \in S(p)$ $\pi = \{3,4,2,1\}$



$C^2 = (7,17,1,19)^T \in S(p)$ $\pi = \{3,4,2,1\}$

Le combinazioni convesse appartengono al poliedro ma non corrispondono a schedule ammissibili

Esempio: $\frac{1}{2} C^1 + \frac{1}{2} C^2 = (13,15,1,11.5)$

Il problema singola macchina
Tempo di completamento pesato
con vincoli di precedenza

Il problema $1/prec/\Sigma w_j C_j$

- I questa parte affronteremo una generalizzazione del problema singola macchina con tempo di completamento pesato
- La generalizzazione è la presenza di vincoli di precedenza fra job
- Non si può usare direttamente la Regola di Smith e il problema diviene sostanzialmente più difficile
- In linea di principio è possibile risolvere la formulazione 0,1 associata al programma disgiuntivo corrispondente
- Ma non è così semplice…

Il problema $1/prec/\Sigma w_j C_j$

- Dati n job $1,\ldots,n$ e una sola macchina
- p_j : tempo di processamento (operazione associata al) job j
- w_j : peso del job j.
- $w_j \cdot C_j$: costo di completamento del job j.
- I job non possono essere interrotti (no-preemption) $\rightarrow C_j = s_j + p_j$
- Alcuni job posso iniziare solo dopo il completamento di altri:

$$C_j \geq C_i + p_j \quad \text{per ogni } ij \in A$$

Problema : trovare uno scheduling compatto C che minimizzi il tempo di completamento pesato $\Sigma_j w_j C_j$ e soddisfi tutti i vincoli di precedenza

- Nella classificazione a tre indici: *Problema* $1/prec/\Sigma w_j C_j$

Esempio

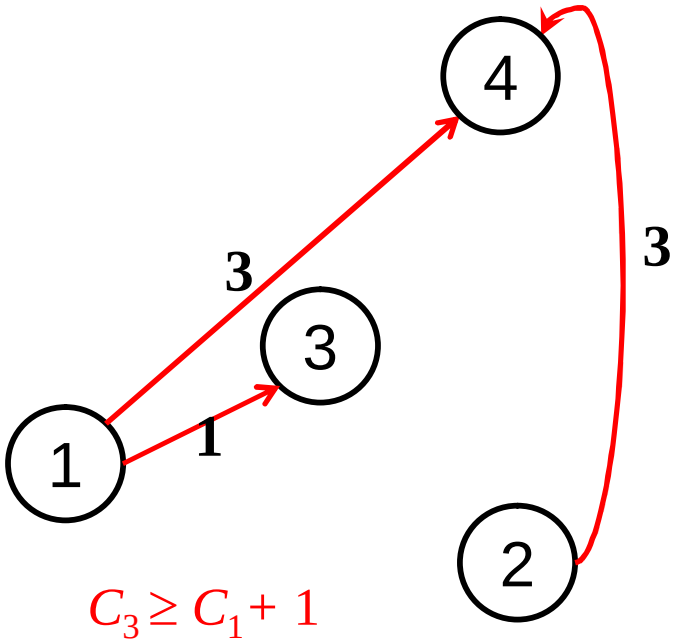
Job	1	2	3	4
p_j	6	9	1	3
w_j	2	5	9	6

Con i vincoli di precedenza:

Bisogna aver concluso 1 per iniziare 3

Bisogna aver concluso 1 per iniziare 4

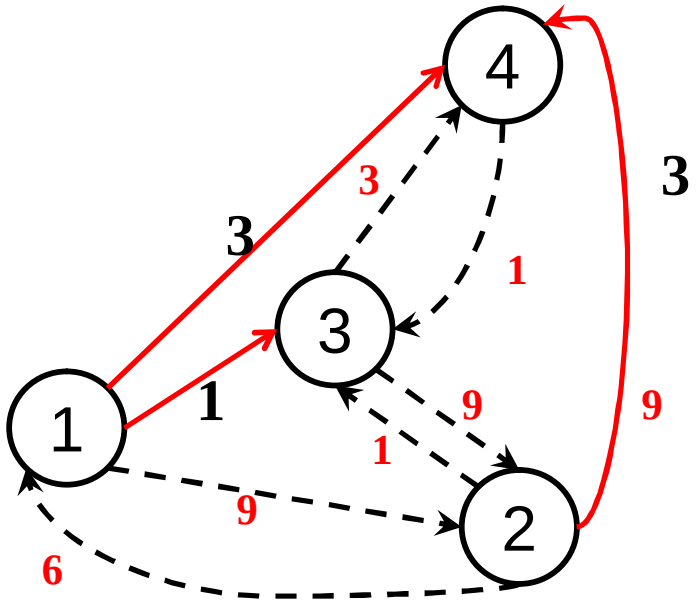
Bisogna aver concluso 2 per iniziare 4



$C_3 \geq C_1 + 1$

$C_4 \geq C_1 + 3$

$C_4 \geq C_2 + 3$



Grafo disgiuntivo

Formulazione 0,1

Job	1	2	3	4
p_j	6	9	1	3
w_j	2	5	9	6

min 2C1 + 5C2 + 9C3 + 6C4

s.t.

C1 > 6

C2 > 9

C3 > 1

C4 > 3

C1 - C2 + 100 x12 > 6

C2 - C1 - 100 x12 >= -91

C2 - C3 + 100 x23 > 9

C3 - C2 - 100 x23 > -99

C3 - C4 + 100 x34 > 1

C4 - C3 - 100 x34 > -97

C3 - C1 > 1

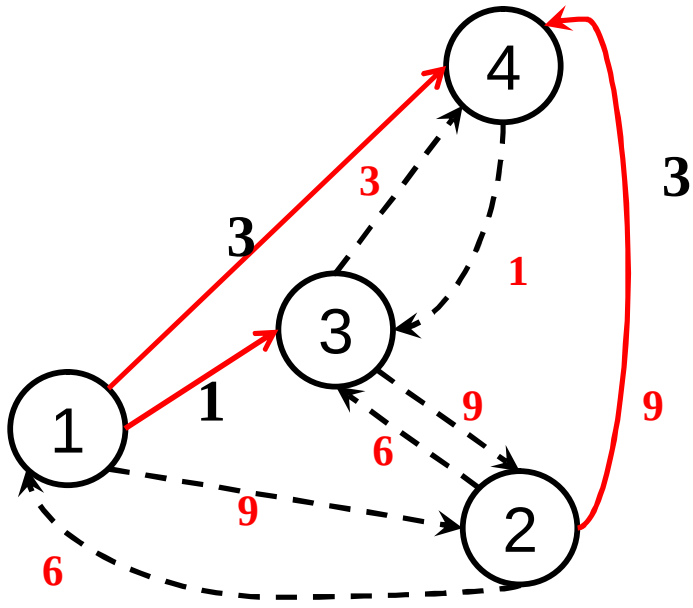
C4 - C1 > 3

C4 - C2 > 3

binary

x12 x23 x34

end



Grafo disgiuntivo

- *Sol Ottima*: C1 = 6, C2 = 16, C3 = 7, C4 = 19
- Valore: 269
- $x_{12}=x_{13}=x_{14}=x_{23}=x_{24}=x_{34}=1$

Complessità problema $1/prec/\sum w_j C_j$

- Il problema $1/prec/\sum w_j C_j$ è in genere difficile computazionalmente
- Per alcune classi di vincoli aggiuntivi il problema può essere risolto efficientemente
- In particolare, se il grafo associato ai vincoli aggiuntivi è *serie-parallelo* (ad esempio, un cammino o un insieme di cammini passanti per nodi distinti)
- Si possono usare tecniche di soluzione tipo *Branch&Bound*
- Per applicare queste tecniche occorre definire un rilassamento
- Tuttavia la presenza della *big M* rende questa formulazione molto debole: il rilassamento lineare non fornisce buoni bound
- Introduciamo quindi un'altra tecnica generale per costruire rilassamenti nota come *Rilassamento Lagrangiano* : si rilassano (=tolgono) alcuni vincoli, e la loro violazione viene sommata alla funzione obiettivo moltiplicata per un coefficiente

Teorema del Rilassamento Lagrangiano

- $P(Q)$: $z_Q = \min_x\{c^T x: Ax \geq b, x \in Q\}$, con $Q \subseteq R^n$ e A matrice $m \times n$
- $RL(u)$: $z_{RL}(u) = \min_x \{c^T x + u^T(b - Ax), x \in Q\}$, con $u \geq 0$

Teorema 6.5 (*del rilassamento lagrangiano*) Per ogni $u \in R_+^m$

$RL(u)$ è un rilassamento di $P(Q)$ e si ha $z_{RL}(u) \leq z_Q$

Dim . La regione ammissibile Q di $RL(u)$ è ottenuta rimuovendo vincoli da quella di $P(Q)$: $R = \{x: Ax \geq b, x \in Q\} \rightarrow R \subseteq Q$

x' ammissibile per $P(Q) \rightarrow Ax' \geq b \rightarrow b - Ax' \leq 0$

$u \geq 0,\ b - Ax' \leq 0 \rightarrow u^T(b - Ax') \leq 0 \rightarrow$ $c^T x' + u^T(b - Ax') \leq c^T x'$

(*per ogni $x' \in R$, la f.o.di $RL(u)$ è $\leq$ della f.o. di $P(Q)$*)

$\rightarrow RL(u)$ è un rilassamento di $P(Q)$ e $z_{RL}(u) \leq z_Q$

■

Esempio Rilassamento Lagrangiano

$$\begin{aligned} &\min\ 2C_1 + 5C_2 + 9C_3 + 6C_4 \\ &C_3 - C_1 \geq 1 \qquad u_1 = 2 \\ &C_4 - C_1 \geq 3 \qquad u_2 = 1 \\ &C_4 - C_2 \geq 3 \qquad u_3 = 2 \\ &C \in Ext(S_c(p)) \end{aligned}$$

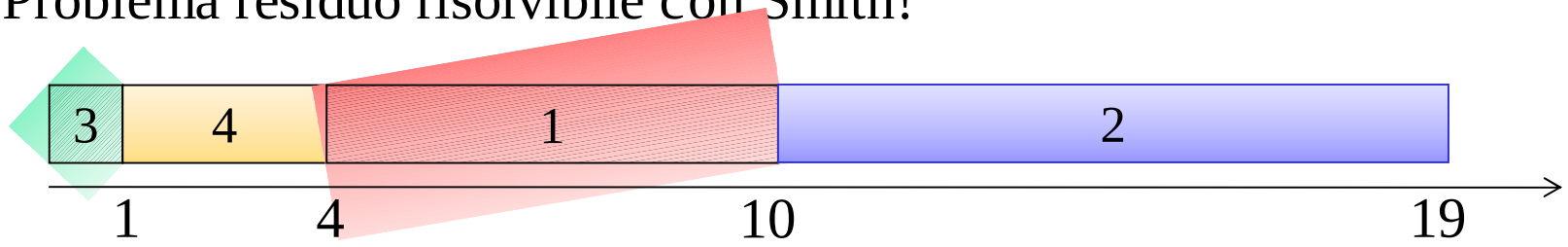
$$\begin{aligned} &\min\ 2C_1 + 5C_2 + 9C_3 + 6C_4 + 2(1 - C_3 + C_1) \\ &\quad + 1(3 - C_4 + C_1) + 2\,(3 - C_4 + C_2) \\ &C \in Ext(S_c(p)) \end{aligned}$$

$$11 + \min\ 5C_1 + 7C_2 + 7C_3 + 3C_4$$

$$C \in Ext(S_c(p))$$

J	1	2	3	4
p_j	6	9	1	3
w'_j	5	7	7	3
w_j/p_j	5/6	7/9	7	1

- Problema residuo risolvibile con Smith!



- Valore $z_{RL}(u) = z_{RL}((2,1,2)^T) = 11 + 202 = 213 < 269$ (valore ottimo $P(Q)$)

Ma cambia cambiando le u

$$\min\ 2C_1 + 5C_2 + 9C_3 + 6C_4$$

$C_3 - C_1 \geq 1 \qquad u_1 = 1$

$C_4 - C_1 \geq 3 \qquad u_2 = 2$

$C_4 - C_2 \geq 3 \qquad u_3 = 3$

$C \in Ext(S_c(p))$

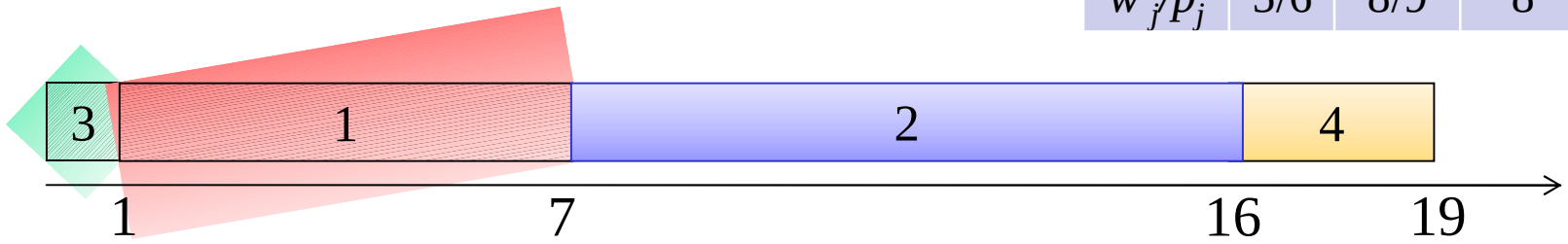
$$\min\ 2C_1 + 5C_2 + 9C_3 + 6C_4 + 1(1 - C_3 + C_1) + 2(3 - C_4 + C_1) + 3(3 - C_4 + C_2)$$

$C \in Ext(S_c(p))$

$$16 + \min\ 5C_1 + 8C_2 + 8C_3 + C_4$$

$C \in Ext(S_c(p))$

J	1	2	3	4
p_j	6	9	1	3
w'_j	5	8	8	1
w'_j/p_j	5/6	8/9	8	1/3



- Valore $z_{RL}(u) = z_{RL}((1,2,3)^T) = 16 + 202 = 218 < 269$ (valore ottimo $P(Q)$)
- Da notare che $z_{RL}((1,2,3)^T) = 218 > z_{RL}((2,1,2)^T)$

Duale Lagrangiano

- Il valore $z_{RL}(u) = \min_x \{c^T x + u^T (b - Ax), x \in Q\}, u \geq 0$ varia con u
- Per ottenere il "miglior" lower bound possibile, siamo interessati al massimo di $z_{RL}(u)$ al variare di u
- Il problema *DL*: $z_{RL} = \max_u \{z_{RL}(u)\}$ è detto *Duale Lagrangiano*
- Quindi *DL*: $\max_u \min_x \{c^T x + u^T (b - Ax), x \in Q, u \geq 0\}$
- *DL* è un probrema di programmazione non lineare e può essere risolto con varie tecniche (es. il *metodo del subgradiente*)
- Di seguito mostreremo un approccio basato sul metodo del simplesso dinamico: risolviamo senza considerare tutti i vincoli, poi ne generiamo uno violato, risolviamo nuovamente e così via

Risolvere il Duale Lagrangiano

DL: $\max_u \min_x \{ c^T x + u^T(b - Ax), x \in Q, u \geq 0\}$

Assunzione: $Q = \{x^1, \ldots, x^r\}$ è un insieme finito (non ho rilassato i vincoli di interezza!)

- Per fissato $u = u'$ il problema $\min_x \{c^T x + u'^T(b - Ax), x \in Q\}$ può essere risolto per enumerazione trovando il <u>minimo</u> delle seguenti quantità

$$v^1(u') = c^T x^1 + u'^T(b - A x^1)$$
$$v^2(u') = c^T x^2 + u'^T(b - A x^2)$$
$$\vdots$$
$$v^r(u') = c^T x^r + u'^T(b - A x^r)$$

- OSS: $min\ \{v^1(u'), \ldots, v^r(u')\} = max\ \{v: v \leq v^1(u'), \ldots, v \leq v^r(u'), v \in R\}$

DL può essere riscritto come

$$\max_{u \geq 0} \max_v \{v: v \leq v^1(u), \ldots, v \leq v^r(u), v \in R\}$$

Problema *DLv* con molti vincoli

Quindi $DL = \max_u \min_x \{c^T x + u^T(b - Ax), x \in Q , u \geq 0\}$ diventa

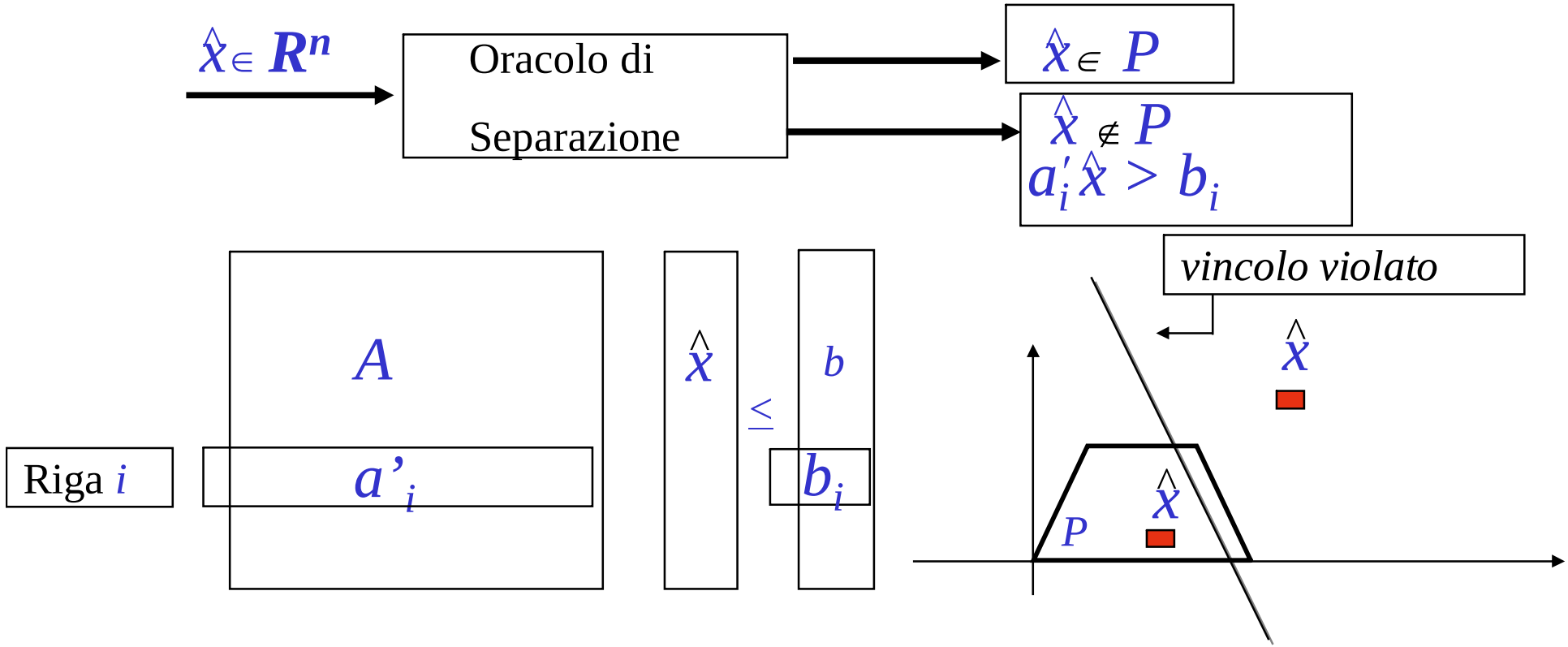
$$\max_u \; \max_v v = \max_{u,v} v$$

$$DLv \qquad \begin{aligned} v &\leq c^T x^1 + u^T(b - A x^1) \\ v &\leq c^T x^2 + u^T(b - A x^2) \\ &\vdots \\ v &\leq c^T x^r + u^T(b - A x^r) \\ v &\in R, \; u \in R^m_+ \end{aligned}$$

- Le variabili di *DLv* sono $\boldsymbol{v} \in R$ e $\boldsymbol{u} \in R^m$
- Ogni vincolo corrisponde a un punto di Q
- Se Q è grande si può (si deve!) applicare il metodo del simplesso dinamico
- Componente essenziale: individuare un oracolo di separazione

Oracolo di Separazione

Descrizione "implicita" di: $P = \{x \in \boldsymbol{R}^n : Ax \leq b\}$



- *Oracolo di separazione*: **algoritmo** che, dato un poliedro $P = \{x \in \boldsymbol{R}^n : Ax \leq b\}$ e un punto $x^* \in \boldsymbol{R}^n$ stabilisce se $x^* \in P$ oppure restituisce un vincolo della descrizione di P violato da x^*

Oracolo di separazione per *DLv*

Oracolo di Separazione: Data una soluzione (v^*, u^*) di *DLv* trova un vincolo tale che $v^* > c^T x^t + (u^*)^T(b - A x^t)$ (*vincolo violato*) o dimostra che tale vincolo non esiste.

I vincoli corrispondono ai punti di Q ⟹

Oracolo di Separazione

Trova $x^t \in Q$ che minimizza $c^T x^t + (u^*)^T (b - A x^t)$

Se $c^T x^t + u^{*T} (b - A x^t) < v^* \rightarrow$ vincolo violato

$$v - u^T (b - A x^t) \leq c x^t$$

Altrimenti $c^T x + u^{*T} (b - A x) \geq v^*$ per ogni $x \in Q$

e non esiste un vincolo violato

Oracolo di separazione per $1/prec/\Sigma w_j C_j$

$$(u^*)^T b + \min c^T x - u^{*T} A x$$

$$x \in Q$$

- Nel caso del problema $1/prec/\Sigma w_j C_j$, dopo aver rilassato i vincoli di precedenza $AC \geq b$, il problema residuo è un problema $1/-/\Sigma w_j C_j$
- Le soluzioni del rilassamento (*insieme* Q) sono quindi schedule compatti, corrispondenti ai vertici di $S_c(p)$
- Il problema di separazione per $(v^* \; u^*)$ diventa quindi

$$u^{*T} b + \min w^T C - u^{*T} A C$$

$$C \in Ext(S_c(p))$$

- Questo problema può essere risolto con la regola di Smith!

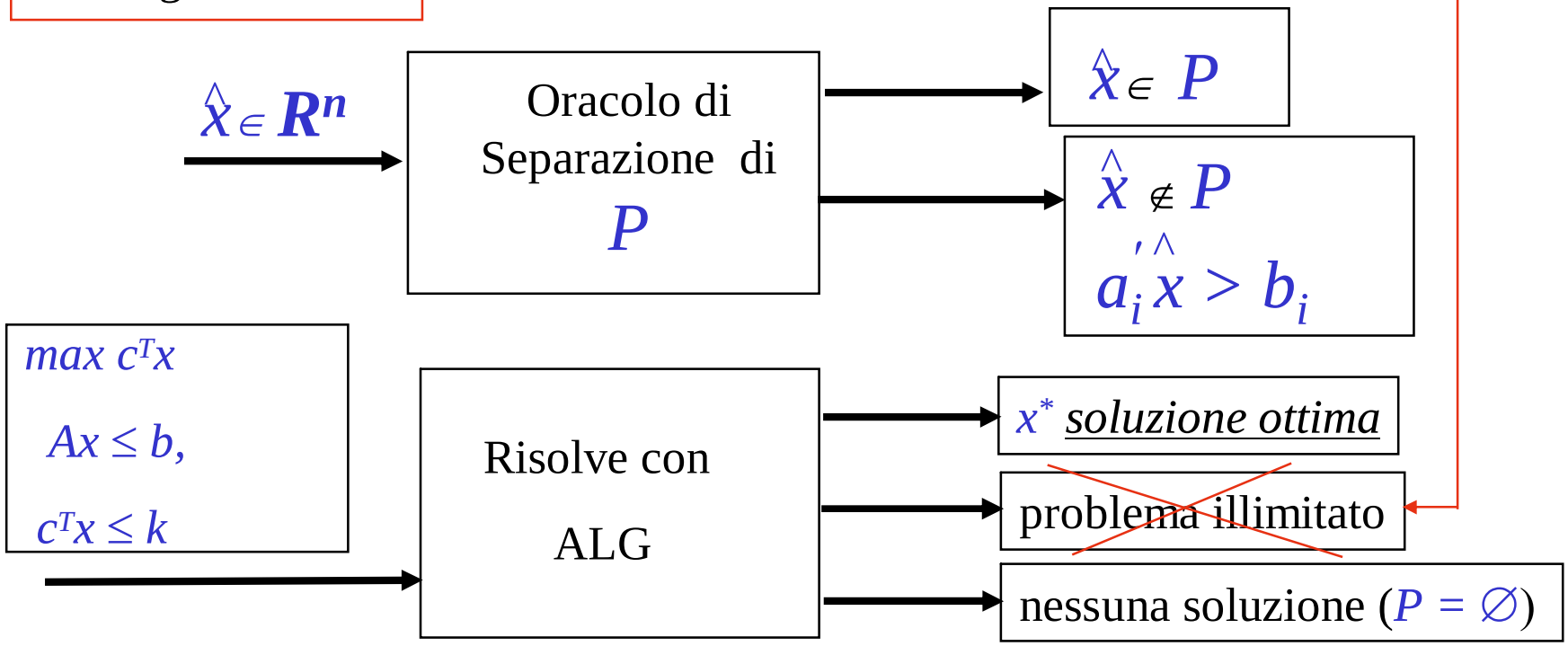
Metodo del Simplesso Dinamico

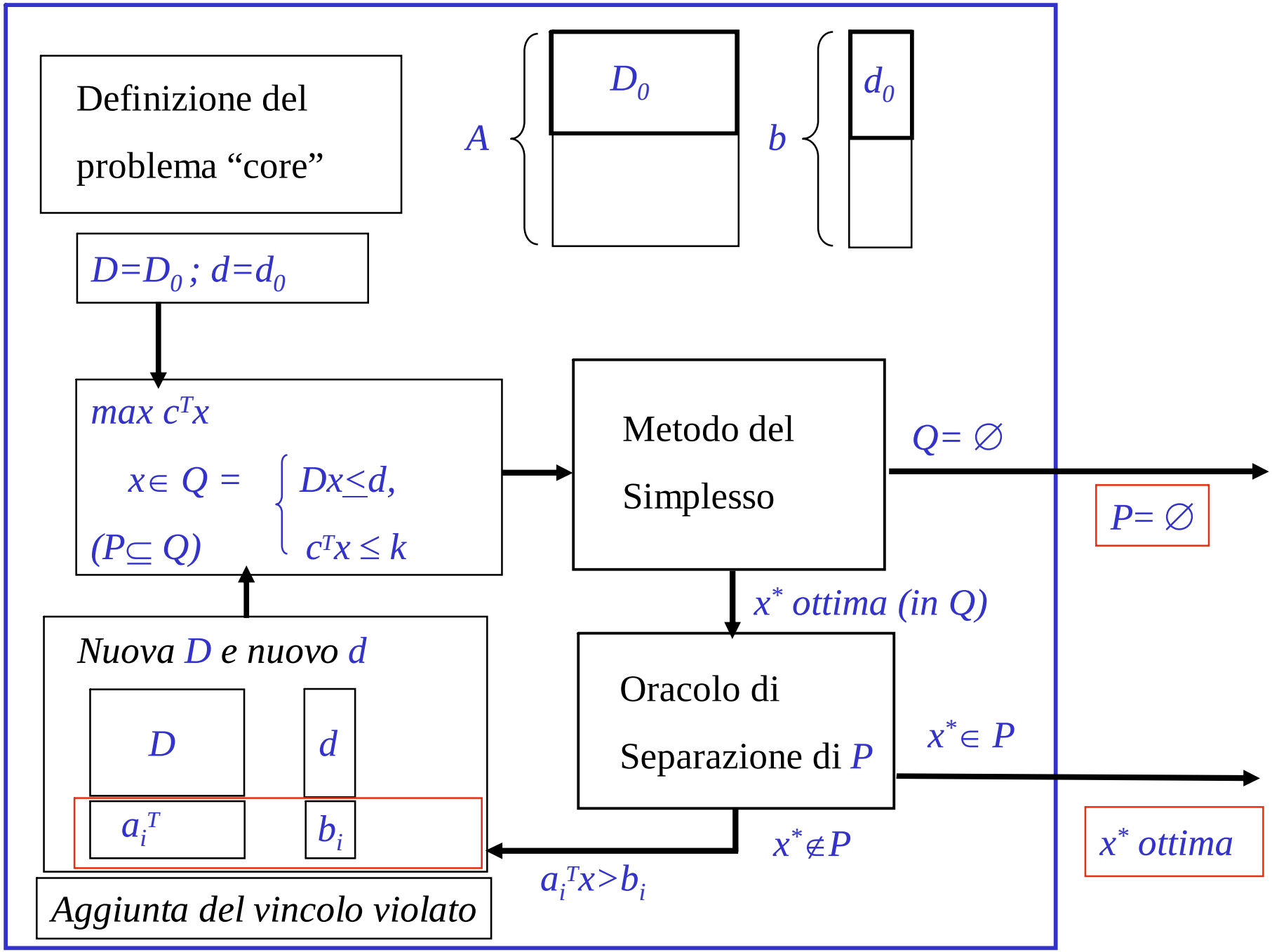
Risolve un problema per cui sia disponibile un algoritmo efficiente ALG (es. un problema di PL, e in tal caso ALG è il simplesso)

$$\max c^T x : \; x \in P = \{x \in \mathbf{R}^n : Ax \leq b, \; c^T x \leq k\}$$

Assunzione: problema non sia illimitato

Due ingredienti:




Definizione del problema “core”
A
D_0
b
d_0
$D=D_0$; $d=d_0$
$max\ c^Tx$
$x \in Q = \{ Dx \leq d, \; c^Tx \leq k$
$(P \subseteq Q)$
Metodo del Simplesso
$Q = \varnothing$
$P = \varnothing$
x^* ottima (in Q)
Oracolo di Separazione di P
$x^* \in P$
x^* ottima
$x^* \notin P$
$a_i^Tx > b_i$
Nuova D e nuovo d
D
d
a_i^T
b_i
Aggiunta del vincolo violato

Esempio

$$min \;\; 2C_1 + 5C_2 + 9C_3 + 6C_4$$
$$C_3 - C_1 \geq 1$$
$$C_4 - C_1 \geq 3$$
$$C_4 - C_2 \geq 3$$
$$C \in Ext(S_c(p))$$

J	1	2	3	4
p_j	6	9	1	3
w_j	2	5	9	6

- Risolviamo il duale lagrangiano applicando il simplesso dinamico
- Il problema core iniziale è costruito a partire da vincoli corrispondenti a un sottoinsieme di schedule compatti
- Senza perdita di generalità, possiamo aggiungere al problema (e al primo problema *core*) in vincolo $v \leq k$ con k opportuno, in modo da rendere non illimitati i problemi nella successione
- Essendo il valore finale di v un lower bound sul valore ottimo del problema originario, k può essere scelto come il valore di una soluzione ammissibile per il problema originario

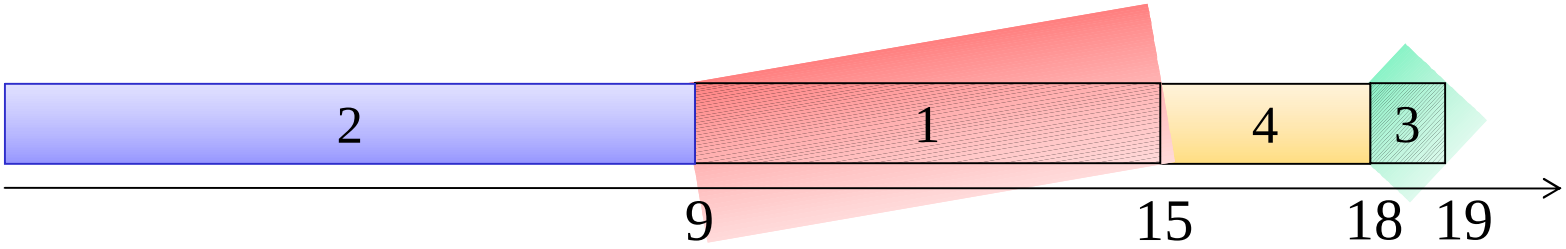
Esempio

$$\min\ 2C_1 + 5C_2 + 9C_3 + 6C_4$$
$$C_3 - C_1 \geq 1$$
$$C_4 - C_1 \geq 3$$
$$C_4 - C_2 \geq 3$$
$$C \in Ext(S_c(p))$$

J	1	2	3	4
p_j	6	9	1	3
w_j	2	5	9	6

$$A = \begin{pmatrix} -1 & 0 & 1 & 0 \\ -1 & 0 & 0 & 1 \\ 0 & -1 & 0 & 1 \end{pmatrix} \qquad b = \begin{pmatrix} 1 \\ 3 \\ 3 \end{pmatrix}$$

- La sequenza $\pi = \{2, 1, 4, 3\}$ è ammissibile. $C(\pi) = (15,9,19,18)^T$



$w^T C = 2\times15 + 5\times9 + 9\times19 + 6\times18 = 354$

Possiamo porre nel duale lagrangiano $v \leq 354$

Esempio

Duale Lagrangiano

$max \ \ v$

$v \leq 354$

$v \in R, u \in R^3_+$

J	1	2	3	4
p_j	6	9	1	3
w_j	2	5	9	6

Sol ottima: $v^* = 354$, $u^* = (0,0,0)$

Oracolo di separazione

$u^*b + min\ (w - (u^*)^T A) \cdot C$

$C \in Ext(S_c(p))$

$(u^*)^T b = 0 \qquad (u^*)^T A = (0, 0, 0, 0)$

$0 + min\ 2C_1 + 5C_2 + 9C_3 + 6C_4$

$x \in Ext(S_c(p))$

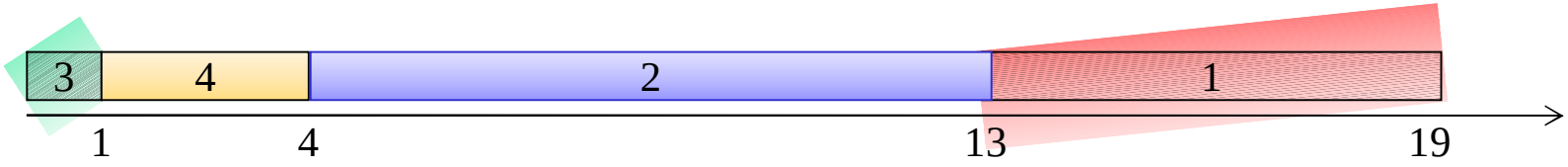
J	1	2	3	4
p_j	6	9	1	3
w_j	2	5	9	6
w_j/p_j	1/3	5/9	9	2

Regola di Smith

$\pi^1 = \{3, 4, 2, 1\}$

$C^1 = (19, 13, 1, 4)$

$wC^1 = 136$



Esempio

DLv

$max \; v$

$v \leq 354$

$v \in R, u \in R^3_+$

J	1	2	3	4
p_j	6	9	1	3
w_j	2	5	9	6

Sol ottima: $v^* = 354$, $u^* = (0,0,0)$

$C^1 = (19, 13, 1, 4)$ Test $v^* > w^T C^1 + (u^*)^T (b - A C^1)$ Viene 354 > 136

Si aggiunge al duale lagrangiano il vincolo $v \leq w C^1 + u^T (b - A C^1)$

$w C^1 = 136$

$A C^1 = (-18 , -15, -9)$

$b - A C^1 = (19, 18 , 12)$

$v \leq 136 + 19u_1 + 18u_2 + 12u_3$

$$A = \begin{pmatrix} -1 & 0 & 1 & 0 \\ -1 & 0 & 0 & 1 \\ 0 & -1 & 0 & 1 \end{pmatrix} \qquad b = \begin{pmatrix} 1 \\ 3 \\ 3 \end{pmatrix}$$

Esempio

DLv

$max\ \ v$

$v \le 354$

$v\ - 19u_1 - 18u_2 - 12u_3 \le 136$

$v \in R,\ u \in R^3_+$

J	1	2	3	4
p_j	6	9	1	3
w_j	2	5	9	6

Sol ottima: $v^* = 354$, $u^* = (11.5,0,0)$

Oracolo di separazione

$$(u^*)^T b + min\ (w - (u^*)^T A) \cdot C$$

$$x \in Ext(S_c(p))$$

$(u^*)^T b = 11.5 \quad (u^*)^T A = (-11.5, 0, 11.5, 0)$

$11{,}5 + min\ 13{,}5C_1 + 5C_2 - 2{,}5C_3 + 6C_4$

$x \in Ext(S_c(p))$

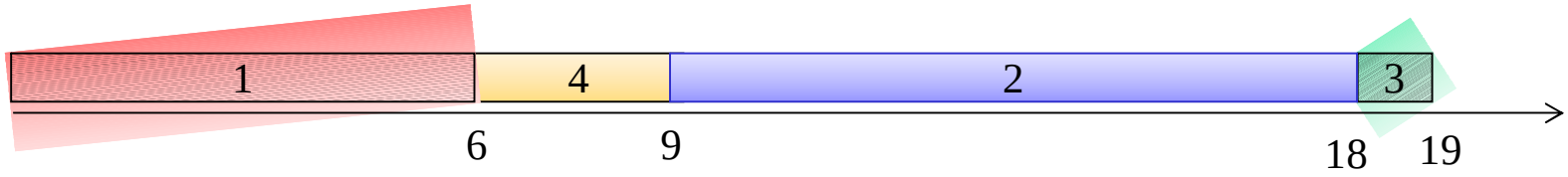
J	1	2	3	4
p_j	6	9	1	3
w_j	13,5	5	-2,5	6
w_j/p_j	2,2	5/9	-2,5	2

Regola di Smith

$\pi^2 = \{1, 4, 2, 3\}$

$C^2 = (6, 18, 19, 9)$

$wC^2 = 327$



Esempio

DLv

$max\ \ v$

$v \le 354$

$v\ - 19u_1 - 18u_2 - 12u_3 \le 136$

$v \in R,\ u \in R^3_+$

J	1	2	3	4
p_j	6	9	1	3
w_j	2	5	9	6

Sol ottima: $v^* = 354,\ u^* = (11.5,0,0)$

$C^2 = (6, 18, 19, 9)$ Test $v^* > w^T C^2 + (u^*)^T(b - A\,C^2)$ Viene 354 > 327-138

Si aggiunge al duale lagrangiano il vincolo $v \le w\,C^2 + u^T (b - A\,C^2)$

$w\,C^2 = 327$

$A\,C^2 = (13\ ,\ 3,\ -9)$

$b\ - A\,C^2 = (-12,\ 0\ ,\ 12)$

$v\ \le 327 - 12u_1 + 12u_3$

$$A = \begin{pmatrix} -1 & 0 & 1 & 0 \\ -1 & 0 & 0 & 1 \\ 0 & -1 & 0 & 1 \end{pmatrix} \qquad b = \begin{pmatrix} 1 \\ 3 \\ 3 \end{pmatrix}$$

Esempio

DLv

$max\ \ v$

$v \le 354$

$v - 19u_1 - 18u_2 - 12u_3 \le 136$

$v + 12u_1 - 12u_3 \le 327$

$v \in R,\ u \in R^3_+$

J	1	2	3	4
p_j	6	9	1	3
w_j	2	5	9	6

Sol ottima: $v^* = 354$, $u^* = (0, 10.61, 2.25)$

Oracolo di separazione

$$(u^*)^T b + min\ (w - (u^*)^T A) \cdot C$$

$$x \in Ext(S_c(p))$$

$(u^*)^T b = 38.58$

$(u^*)^T A = (-10.61, -2.25, 0, 12.86)$

$38{,}58 +$

$min\ 12{,}61C_1 + 7{,}25C_2 + 9C_3 - 6{,}86C_4$

$x \in Ext(S_c(p))$

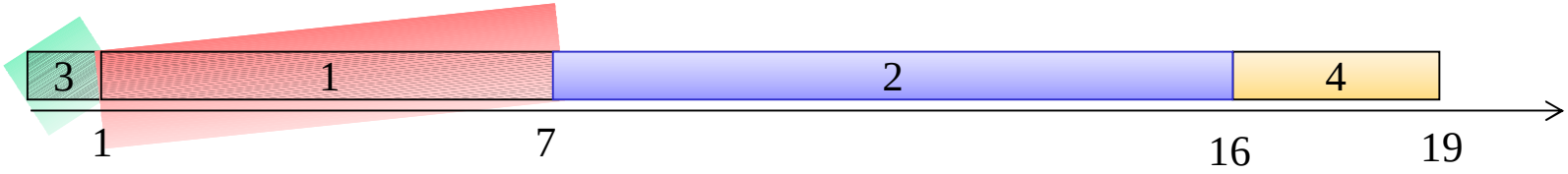
J	1	2	3	4
p_j	6	9	1	3
w_j	12,61	7,25	9	-6,86
w_j/p_j	2,1	0,8	9	-2,28

Regola di Smith

$\pi^3 = \{3, 1, 2, 4\}$

$C^3 = (7, 16, 1, 19)$

$wC^3 = 217$



Esempio

DLv

$max\ \ v$

$v \le 354$

$v - 19u_1 - 18u_2 - 12u_3 \le 136$

$v + 12u_1 - 12u_3 \le 327$

$v \in R,\ u \in R^3_+$

J	1	2	3	4
p_j	6	9	1	3
w_j	2	5	9	6

Sol ottima: $v^* = 354,\ u^* = (0, 10.61, 2.25)$

$C^3 = (7, 16, 1, 19)$ Test $v^* > w^T C^3 + (u^*)^T(b - A C^3)$ Viene 354 > 217-127,32

Si aggiunge al duale lagrangiano il vincolo $v \le w C^3 + u^T (b - A C^3)$

$w C^3 = 217$

$A C^3 = (-6, 12, 3)$

$b - A C^3 = (7, -12, 0)$

$v \le 217 + 7u_1 - 12u_2$

$$A = \begin{pmatrix} -1 & 0 & 1 & 0 \\ -1 & 0 & 0 & 1 \\ 0 & -1 & 0 & 1 \end{pmatrix} \qquad b = \begin{pmatrix} 1 \\ 3 \\ 3 \end{pmatrix}$$

Esempio

DLv

$max\ v$

$v \le 354$

$v - 19u_1 - 18u_2 - 12u_3 \le 136$

$v + 12u_1 - 12u_3 \le 327$

$v - 7u_1 + 12u_2 \le 217$

$v \in R,\ u \in R^3_+$

J	1	2	3	4
p_j	6	9	1	3
w_j	2	5	9	6

Sol ottima: $v^* = 354$, $u^* = (19.57, 0, 21.82)$

Oracolo di separazione

$$(u^*)^T b + min\ (w - (u^*)^T A) \cdot C$$

$$x \in Ext(S_c(p))$$

$(u^*)^T b = 85$

$(u^*)^T A = (-19.57, -21.82, 19.57, 21.82)$

$85 + min\ 21{,}57C_1 + 26{,}82C_2 - 10{,}57C_3 - 15{,}82C_4$

$x \in Ext(S_c(p))$

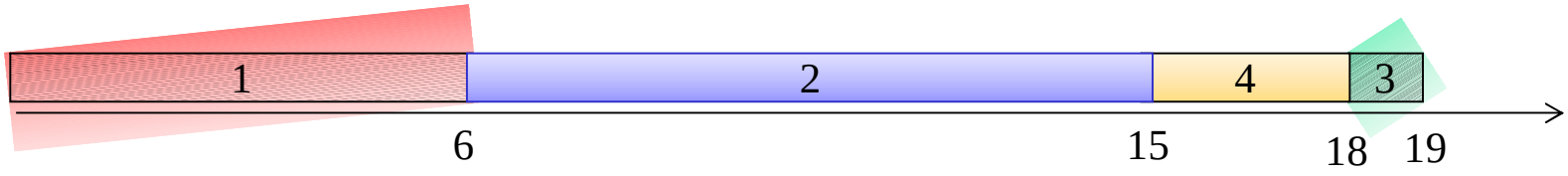
J	1	2	3	4
p_j	6	9	1	3
w_j	21,57	26,82	-10,57	-15,82
w_j/p_j	3,59	2,98	-10,57	-5,27

Regola di Smith

$\pi^4 = \{1, 2, 4, 3\}$

$C^4 = (6, 15, 19, 18)$

$wC^4 = 366$



Esempio

DLv

$max\ v$

$v \le 354$

$v - 19u_1 - 18u_2 - 12u_3 \le 136$

$v + 12u_1 - 12u_3 \le 327$

$v - 7u_1 + 12u_2 \le 217$

$v \in R,\ u \in R^3_+$

J	1	2	3	4
p_j	6	9	1	3
w_j	2	5	9	6

Sol ottima: $v^* = 354$, $u^* = (19.57, 0, 21.82)$

$C^4 = (6, 15, 19, 18)$ Test $v^* > w^T C^4 + (u^*)^T(b - A C^4)$ Viene 354 > 366-431,22

Si aggiunge al duale lagrangiano il vincolo $v \le w C^4 + u^T (b - A C^4)$

$w C^4 = 366$

$A C^4 = (13, 12, 3)$

$b - A C^4 = (-12, -9, 0)$

$v \le 366 - 12u_1 - 9u_2$

$$A = \begin{pmatrix} -1 & 0 & 1 & 0 \\ -1 & 0 & 0 & 1 \\ 0 & -1 & 0 & 1 \end{pmatrix} \qquad b = \begin{pmatrix} 1 \\ 3 \\ 3 \end{pmatrix}$$

Esempio

DLv

$max\ \ v$

$v \le 354$

$v - 19u_1 - 18u_2 - 12u_3 \le 136$

$v + 12u_1 - 12u_3 \le 327$

$v - 7u_1 + 12u_2 \le 217$

$v + 12u_1 + 9u_2 \le 366$

J	1	2	3	4
p_j	6	9	1	3
w_j	2	5	9	6

Sol ottima: $v^* = 271.9$, $u^* = (7.84, 0, 3.25)$

Oracolo di separazione

$$(u^*)^T b + min\ (w - (u^*)^T A) \cdot C$$

$$x \in Ext(S_c(p))$$

$(u^*)^T b = 17.6$

$(u^*)^T A = (-7.84, -3.25, 7.84, 3.25)$

$17.6 + min\ 9{,}84C_1 + 8{,}25C_2 + 1{,}16C_3 + 2{,}75C_4$

$x \in Ext(S_c(p))$

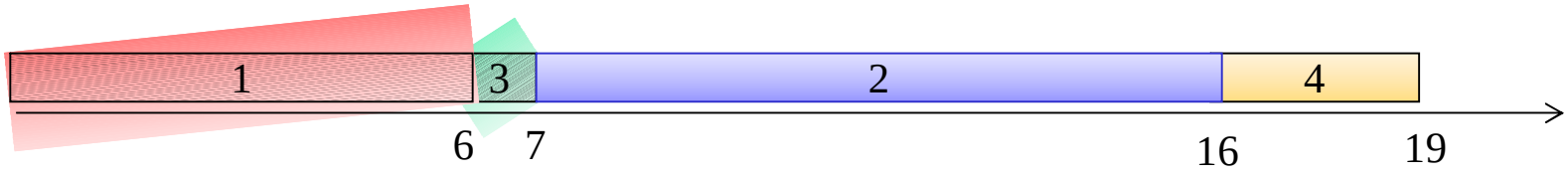
J	1	2	3	4
p_j	6	9	1	3
w_j	9,84	8,25	1,16	2,75
w_j/p_j	1,64	0,92	1,16	0,92

Regola di Smith

$\pi^5 = \{1, 3, 2, 4\}$

$C^5 = (6, 16, 7, 19)$

$wC^5 = 269$



Esempio

DLv

$max\ \ v$

$v \le 354$

$v - 19u_1 - 18u_2 - 12u_3 \le 136$

$v + 12u_1 - 12u_3 \le 327$

$v - 7u_1 + 12u_2 \le 217$

$v + 12u_1 + 9u_2 \le 366$

J	1	2	3	4
p_j	6	9	1	3
w_j	2	5	9	6

Sol ottima: $v^* = 271.9$, $u^* = (7.84, 0, 3.25)$

$C^5 = (6, 16, 7, 19)$ Test $v^* > w^T C^4 + (u^*)^T (b - A C^4)$ Viene 271.9 > 269 +0

Si aggiunge al duale lagrangiano il vincolo $v \le w C^5 + u^T (b - A C^5)$

$w C^5 = 269$

$A C^5 = (1, 13, 3)$

$b - A C^5 = (0, -10, 0)$

$v \le 269 - 10u_2$

$$A = \begin{pmatrix} -1 & 0 & 1 & 0 \\ -1 & 0 & 0 & 1 \\ 0 & -1 & 0 & 1 \end{pmatrix} \qquad b = \begin{pmatrix} 1 \\ 3 \\ 3 \end{pmatrix}$$

Esempio

$max\ v \qquad v \le 354$ DLv

$v - 19u_1 - 18u_2 - 12u_3 \le 136$

$v + 12u_1 - 12u_3 \le 327$

$v - 7u_1 + 12u_2 \le 217$

$v + 12u_1 + 9u_2 \le 366$

$v + 10u_2 \le 269$

J	1	2	3	4
p_j	6	9	1	3
w_j	2	5	9	6

Sol ottima: $v^* = 269$, $u^* = (8, 0, 3.25)$

Oracolo di separazione

$$(u^*)^T b + min\ (w - (u^*)^T A) \cdot C$$

$$x \in Ext(S_c(p))$$

$(u^*)^T b = 17.75$

$(u^*)^T A = (-8, -3.25, 8, 3.25)$

$17.75 +$

$min\ 10C_1 + 8{,}25C_2 + 1C_3 + 2{,}75C_4$

$x \in Ext(S_c(p))$

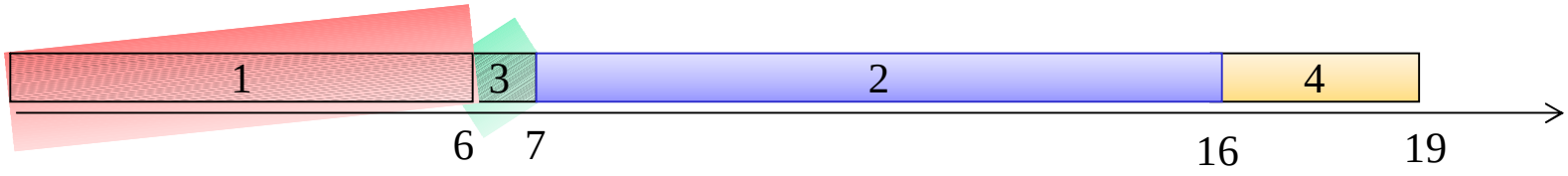
J	1	2	3	4
p_j	6	9	1	3
w_j	10	8,25	1	2,75
w_j/p_j	1,66	0,92	1	0,92

Regola di Smith

$\pi^6 = \{1, 3, 2, 4\}$

$C^6 = (6, 16, 7, 19)$

$wC^6 = 269$



Esempio

$max\ v \qquad v \le 354$ DLv

$v - 19u_1 - 18u_2 - 12u_3 \le 136$

$v + 12u_1 - 12u_3 \le 327$

$v - 7u_1 + 12u_2 \le 217$

$v + 12u_1 + 9u_2 \le 366$

$v + 10u_2 \le 269$

J	1	2	3	4
p_j	6	9	1	3
w_j	2	5	9	6

Sol ottima: $v^* = 269$, $u^* = (8, 0, 3.25)$

$C^6 = (6, 16, 7, 19)$ Test $v^* > w^T C^4 + (u^*)^T(b - A C^4)$ Viene 269 > 269 +0

Allora non si aggiunge più nessun vincolo!

$w C^6 = 269$

$A C^6 = (1, 13, 3)$

$b - A C^6 = (0, -10, 0)$

$$A = \begin{pmatrix} -1 & 0 & 1 & 0 \\ -1 & 0 & 0 & 1 \\ 0 & -1 & 0 & 1 \end{pmatrix} \qquad b = \begin{pmatrix} 1 \\ 3 \\ 3 \end{pmatrix}$$

Il vincolo $v \le 269 - 10u_2$ non è violato da $(v^*\ u^*)$, quindi non ci sono altri vincoli violati e $v^*=269$ è il valore della sol ottima del duale lagrangiano

Quindi è un LB (che posso usare in un Branch&Bound) per il problema misto intero scritto prima (il cui ottimo valeva proprio 269), ed è un LB migliore di quello del ril. lineare, che vale 192