Ricerca Operativa
Programmazione Lineare Intera

Marco A. Boschetti

Università degli Studi di Bologna
Dipartimento di Matematica
marco.boschetti@unibo.it

Outline

Outline (2)

Introduzione ai Metodi Branch and Bound

- Consideriamo il problema IP di programmazione lineare intera:

$$z_{IP} = \min \ \mathbf{cx} \tag{1}$$
$$s.t. \ \mathbf{Ax} \geq \mathbf{b} \tag{2}$$
$$\mathbf{x} \geq \mathbf{0} \text{ and integer} \tag{3}$$

- Il rilassamento lineare LP del problema IP è dato da:

$$z_{LP} = \min \ \mathbf{cx} \tag{4}$$
$$s.t. \ \mathbf{Ax} \geq \mathbf{b} \tag{5}$$
$$\mathbf{x} \geq \mathbf{0} \tag{6}$$

- Se la soluzione ottima $\mathbf{x}^*$ del problema LP è intera, allora è la soluzione ottima anche per il problema intero IP.
- Cosa accade se invece la soluzione ottima $\mathbf{x}^*$ del problema LP è frazionaria?

Selezione variabile frazionaria

- Una possibilità è l'applicazione del Metodo Branch and Bound che abbiamo già visto per il caso particolare del Knapsack Problem.
- Per il Knapsack Problem c'era al più una sola variabile frazionaria, mentre in generale potremo avere diverse variabili frazionarie.
- Quindi, in generale, è necessario stabilire un criterio per selezionare una variabile frazionaria (che deve essere intera nella soluzione ottima di IP), per fare un branch.
- Possibili esempi di criteri per selezionare la variabile frazionaria x_j:
  - la più frazionaria, i.e., quella con il valore $x_j - \lfloor x_j \rfloor$ più vicino a 0,5;
  - la più intera, i.e., quella con il valore $x_j - \lfloor x_j \rfloor$ più vicino a 0 o 1;
  - utilizzando un ordine prestabilito;
  - etc...

Regola di branching

- Data la soluzione ottima del problema LP e la variabile frazionaria x_j selezionata, una possibile regola di branching può prevedere la creazione di due sottoproblemi:

  $P(x_j \leq \lfloor x_j \rfloor)$: ottenuto da LP aggiungendo il vincolo $x_j \leq \lfloor x_j \rfloor$;

  $P(x_j \geq \lceil x_j \rceil)$: ottenuto da LP aggiungendo il vincolo $x_j \geq \lceil x_j \rceil$.

  La soluzione frazionaria x_j è esclusa in entrambe i sottoproblemi.

- Se consideriamo il problema LP originale come nodo radice di un albero di ricerca, i due sottopropblemi sono i suoi nodi figli.

- Il processo di branching descritto per il nodo radice è ripetuto per i nodi figli, per cui la regola di branching è applicata ad ogni nodo k dell'albero di ricerca (dove $k = 0$ è il nodo radice e $k = 1, 2$ i suoi nodi figli).

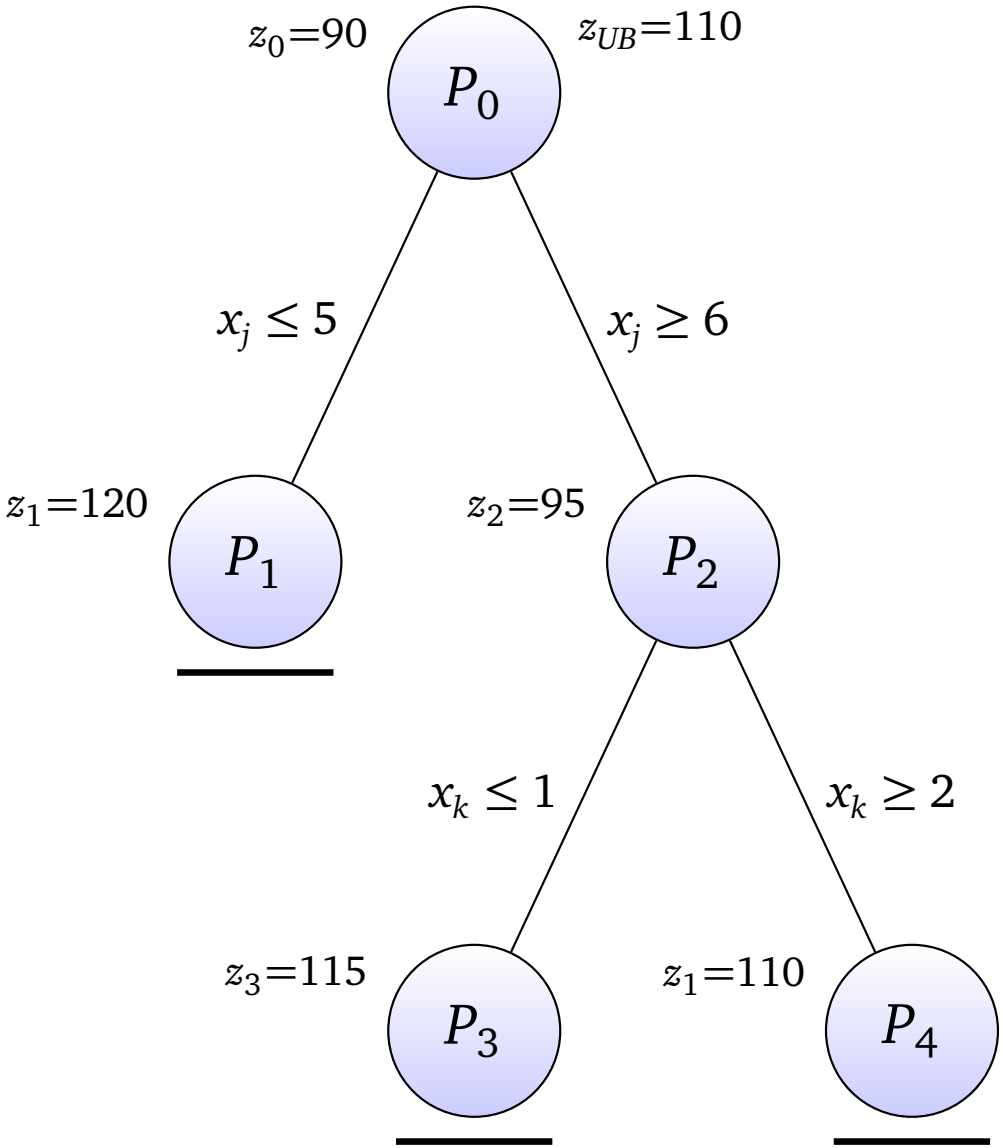
Lower e Upper Bound

- Per ogni nodo k dobbiamo risolvere il sottoproblema P_k ottenendo la soluzione $\mathbf{x}_k$ di costo z_k.
  - Se P_k non ha soluzione ammissibile poniamo $z_k = +\infty$ e non generiamo nodi figli (perché?).
  - Se la soluzione $\mathbf{x}_k$ di P_k è intera, allora è sicuramente un upper bound alla soluzione ottima del problema intero originario IP e non è necessario generare ulteriori nodi figli del nodo k (perché?).
  - Se la soluzione $\mathbf{x}_k$ di P_k è frazionaria, allora il costo z_k rappresenta un lower bound ($LB_k = z_k$) al valore della migliore soluzione intera che si può ottenere nel sottoalbero di cui il nodo k è il nodo radice.
- La miglior soluzione intera ammissibile finora trovata ha costo z_{UB} (i.e., è il miglior upper bound finora trovato).
- L'upper bound z_{UB} può essere inizializzato a $+\infty$ oppure con il valore della soluzione generata con un algoritmo euristico.

Lower e Upper Bound

- Oltre a ottenere un upper bound quando la soluzione $\mathbf{x}_k$ di P_k è intera, lo si può ottenere utilizzando un algoritmo euristico per ripristinare l'ammissibilità intera della soluzione $\mathbf{x}_k$ quando è frazionaria.
- L'upper bound z_{UB} è molto importante perché permette di eliminare in anticipo quei nodi che non consentiranno di ottenere la soluzione ottima, evitando di esplorare il corrispondente sotto-albero.
- Se per un nodo k la soluzione di P_k ha costo $z_k \geq z_{UB}$ allora il nodo non deve essere "espanso" (i.e., non devono essere generati i nodi figli) perché non consentiranno di ottenere una soluzione di costo inferiore a z_{UB} (conosciamo già una soluzione di costo z_{UB}).
- Migliore è il valore z_{UB} e tanto più velocemente lo otteniamo e più veloce sarà la convergenza e la quantità di memoria risparmiata.

Lower e Upper Bound



Metodo di visita dell'albero

- Quando a un nodo k viene svolto un branching vengono generati due nodi figli che vengono inseriti in un heap.
- L'ordine con cui i nodi sono estratti dalla heap per essere *risolti* determina il metodo di visita dell'albero di ricerca.
- I metodi di visita dell'albero di ricerca più noti sono i seguenti:
  - Depth-First Search: ricerca in profondità, esplorando i nodi figli prima di fare backtracking;
  - Breadth-First Search: esplora l'albero di ricerca un livello alla volta, i.e., estrae dalla heap prima i nodi dei livelli di indice più piccolo;
  - Best-First Search: ad ogni iterazione estrae dalla heap il nodo con il lower bound più basso.
- Non esite un metodo migliore di altri. La scelta dipende dal modello che si vuole risolvere e dalle performance a cui si è interessati.

Algoritmo Branch and Bound

Algoritmo Branch and Bound

Step 1. Poni $z_{UB} = +\infty$ e inserisci nella heap vuota il problema P_0: $z_0 = \min\{\mathbf{cx} : \mathbf{Ax} \geq \mathbf{b}, \mathbf{x} \geq \mathbf{0}\}$.

Step 2. Estrai il problema P_k dalla heap (se è vuota, STOP) e ottieni la soluzione $\mathbf{x}_k$ di costo z_k ($z_k = +\infty$ se non ha soluzione).

Step 3. Se $z_k \geq z_{UB}$ torna allo Step 2 (il nodo k non viene espanso).

Step 4. Se la soluzione $\mathbf{x}_k$ è intera, allora aggiorna l'upper bound: $z_{UB} = z_k$ e $\mathbf{x} = \mathbf{x}_k$ (avendo passato lo Step 3, $z_k < z_{UB}$).
Elimina dalla heap tutti i nodi P_k per cui $z_k \geq z_{UB}$.
Torna allo Step 2.

Step 5. Seleziona una variabile x_j frazionaria nella soluzione $\mathbf{x}_k$ e inserisci nella heap due nodi figli $P(x_j \leq \lfloor x_j \rfloor)$ e $P(x_j \geq \lceil x_j \rceil)$.
Torna allo Step 2.

Osservazioni e Varianti

- I problemi P_k possono essere risolti usando il metodo del simplesso.
- Per risolvere il problema P_k in modo efficiente sarebbe utile partire dalla base del nodo padre. Per cui, quando si salva in heap il nodo dovrebbe essere necessario salvare anche questa informazione.
- Per includere in P_k tutti i vincoli imposti dalla regola di branching è necessario individuare (o conservare) i vincoli definiti dal "cammino" dal nodo k fino alla radice dell'albero di ricerca.
- Rimarchiamo che la regola di branching proposta è solo una delle tante opzioni possibili. Per ogni problema specifico potrebbe essere proposta una regola diversa.
- Inoltre, i problemi P_k potrebbero essere risolti usando altri metodi (e.g., Rilassamento Lagrangiano) e/o rilassando ulteriormente i vincoli presenti nel modello (oltre ai vincoli di interezza).

Osservazioni e Varianti

- L'algoritmo Branch and Bound proposto potrebbe essere migliorato includendo altre opzioni, come ad esempio:
  - Algoritmi euristici di "repair" per generare un upper bound z_{UB} di buona qualità dalle soluzioni frazionarie di P_k;
  - Aggiunta di ulteriori vincoli ("cut") per eliminare delle soluzioni frazionarie. In questo caso parleremo di "Branch and Cut".
- Gli algoritmi Branch and Cut sono uno strumento molto potente per risolvere problemi difficili.
- Lo studio di quali vincoli/cut aggiungere e dei metodi per identificarli fa parte della "Poliedrale", un ambito di ricerca della programmazione matematica.
- I vincoli che eliminano le soluzioni frazionarie, ma non compromettono l'ammissibilità delle soluzioni intere, sono anche detti "valid inequalities" oppure "disuguaglianze valide".

Introduzione alle Valid Inequalities

- Consideriamo un problema P di programmazione lineare intera: $z_P = \min\{\mathbf{cx} : \mathbf{Ax} \geq \mathbf{b}, \mathbf{x} \geq \mathbf{0} \text{ e intera}\}$.
- Se il problma P ha una soluzione ammissibile e lo risolviamo con il Simplesso (Primale o Duale), potremo ottenere il seguente tableau ottimo (per semplicità abbiamo riordinato le colonne):

	z	$\mathbf{x_B}$					$\mathbf{x_N}$					RHS
z	1	0	…	0	…	0	$\mathbf{wa}_{m+1} - c_{m+1}$	…	$\mathbf{wa}_{m+j} - c_{m+j}$	…	$\mathbf{wa}_n - c_n$	$\mathbf{c_B B^{-1} b}$
	0	1	…	0	…	0	y_1^{m+1}	…	y_1^j	…	y_1^n	$\bar{b}_1$
	…	…	…	…	…	…	…	…	…	…	…	…
$\mathbf{x_B}$	0	0	…	1	…	0	y_i^{m+1}	…	y_i^j	…	y_i^n	$\bar{b}_i$
	…	…	…	…	…	…	…	…	…	…	…	…
	0	0	…	0	…	1	y_m^{m+1}	…	y_m^j	…	y_m^n	$\bar{b}_m$

- Ipotizziamo che la variabile in corrispondenza della riga i (in questo caso x_i) abbia un valore $\bar{b}_i$ frazionario.
- Quale vincolo possiamo aggiungere per "escludere" questa soluzione frazionaria dall'insieme delle soluzioni ammissibili senza escludere anche le soluzioni intere?

Tagli di Gomory

- Costruiamo delle disuguaglianze valide note come Tagli di Gomory.
- Se consideriamo il tableau ottimo, l'equazione corrispondente alla riga i può essere scritta come segue:

$$x_i + \sum_{j=m+1}^{n} y_i^j x_j = \bar{b}_i \tag{7}$$

- Possiamo decomporre le quantitä y_i^j e $\bar{b}_i$ nelle componenti intere e frazionarie:

$$y_i^j = I_i^j + F_i^j \quad \text{e} \quad \bar{b}_i = I_i + F_i$$

dove

$$I_i^j = \lfloor y_i^j \rfloor \quad \text{e} \quad F_i^j = y_i^j - I_i^j \quad (0 \leq F_i^j < 1)$$

$$I_i = \lfloor \bar{b}_i \rfloor \quad \text{e} \quad F_i = \bar{b}_i - I_i \quad (0 \leq F_i < 1)$$

Tagli di Gomory

- L'equazione corrispondente alla riga i può quindi essere riscritta come:

$$x_i + \sum_{j=m+1}^{n} y_i^j x_j = \bar{b}_i$$

$$x_i + \sum_{j=m+1}^{n} (I_i^j + F_i^j) x_j = I_i + F_i$$

$$\underbrace{x_i + \sum_{j=m+1}^{n} I_i^j x_j - I_i}_{\text{Integer for every integer } \mathbf{x}} = \underbrace{F_i - \sum_{j=m+1}^{n} F_i^j x_j}_{<1, \text{ for every } \mathbf{x} \geq \mathbf{0}}$$

- Siccome il termine a sinistra è sempre intero per ogni soluzione $\mathbf{x}$ intera, allora lo deve essere anche il termine a destra, che è minore strettamente di 1.

Tagli di Gomory

- Siccome

$$F_i - \sum_{j=m+1}^{n} F_i^j x_j < 1$$

ma $F_i - \sum_{j=m+1}^{n} F_i^j x_j$ deve essere anche intero, allora:

$$F_i - \sum_{j=m+1}^{n} F_i^j x_j \leq 0$$

che corrisponde al seguente vincolo, detto Taglio di Gomory:

$$-\sum_{j=m+1}^{n} F_i^j x_j \leq -F_i \tag{8}$$

che non è soddifatto dalla soluzione corrente, perché $x_j = 0$ per ogni $j = m+1, \ldots, n$ dato che sono non base, mentre $F_i > 0$ visto che abbiamo ipotizzato che $\bar{b}_i$ è frazionario.

Tagli di Gomory

- Per semplicità avevamo ipotizzato che la base fosse costituita dalle prime m colonne. Nel caso generale, il Taglio di Gomory relativo alla variabile in base frazionaria in corrispondenza della riga i può essere riscritto come:

$$-\sum_{j \in N} F_i^j x_j \leq -F_i \tag{9}$$

  dove N è l'insieme delle colonne non in base.

- Se risolviamo il problema con il Simplesso, quando aggiungiamo il nuovo vincolo al problema non è necessario ripetere l'algoritmo dall'inizio, ma possiamo ripartire dalla base corrente.

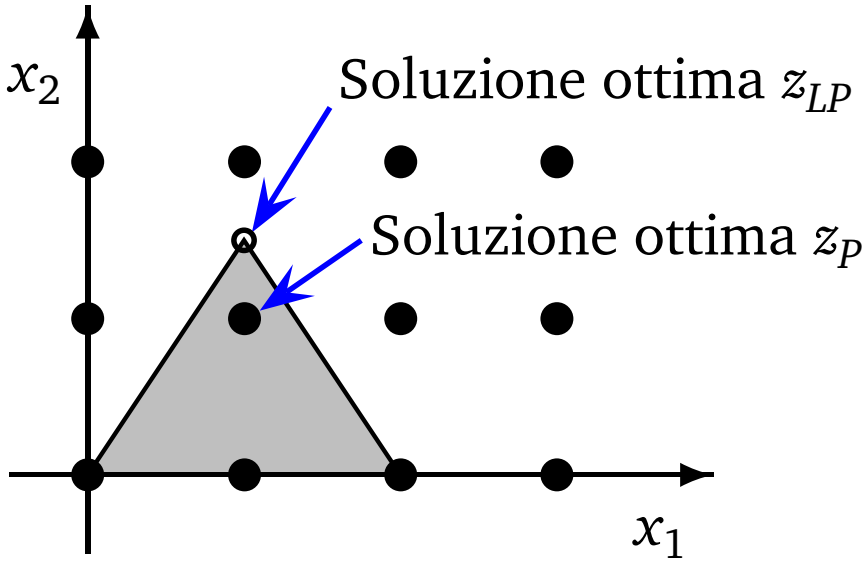
- Se le variabili attualmente incluse sono n, il taglio di Gomory corrisponde a una nuova riga da aggiungere al tableau:

$$-\sum_{j \in N} F_i^j x_j + x_{n+1} = -F_i \tag{10}$$

Esempio di Applicazione dei Tagli di Gomory

Si consideri il problema P:

$$
\begin{array}{rrrrrl}
\min z_P = & & -x_2 & & & \\
\text{s.t.} & +3x_1 & +2x_2 & \leq & 6 & \\
& -3x_1 & +2x_2 & \leq & 0 & \\
& x_1 & , \quad x_2 & \geq & 0 & \text{e intere}
\end{array}
$$



Esempio di Applicazione dei Tagli di Gomory

Se rilassiamo i vincoli di interezza (ottenendo il problema LP), possiamo risolvere il problema con il simplesso primale.
Il primo tableau è il seguente:

x_1	x_2	x_3	x_4	
0	1	0	0	0
3	2	1	0	6
-3	**2**	0	1	0

Dopo la prima iterazione abbiamo:

x_1	x_2	x_3	x_4	
3/2	0	0	-1/2	0
6	0	1	-1	6
-3/2	1	0	1/2	0

Esempio di Applicazione dei Tagli di Gomory

Dopo la seconda iterazione abbiamo ottenuto il seguente tableau ottimo:

x_1	x_2	x_3	x_4	
0	0	-1/4	-1/4	-3/2
1	0	1/6	-1/6	1
0	1	1/4	1/4	3/2

- La soluzione ottima del problema LP è $\mathbf{x} = (1, \frac{3}{2})$, che non è intera.
- Possiamo aggiungere un Taglio di Gomory per eliminare (si dice anche "separare") la soluzione frazionaria.
- Ricordiamo che dobbiamo scegliere una riga corrispondete a una variabile frazionaria. Nel nostro caso quella relativa a x_2:

$$x_2 + \frac{1}{4}x_3 + \frac{1}{4}x_4 = \frac{3}{2}$$

Esempio di Applicazione dei Tagli di Gomory

- Data la riga corrispondete a x_2 è:

$$x_2 + \frac{1}{4}x_3 + \frac{1}{4}x_4 = \frac{3}{2}$$

Ricordiamo che il Taglio di Gomory ha la seguente forma:

$$-\sum_{j=m+1}^{n} F_i^j x_j \leq -F_i$$

quindi nel nostro caso è:

$$-\frac{1}{4}x_3 - \frac{1}{4}x_4 \leq -\frac{1}{2}$$

Esempio di Applicazione dei Tagli di Gomory

Possiamo aggiungere il Taglio di Gomory inserendo una nuova riga al tableau, oltre alla colonna della nuova variabile di scarto x_5:

x_1	x_2	x_3	x_4	x_5	
0	0	-1/4	-1/4	0	-3/2
1	0	1/6	-1/6	0	1
0	1	1/4	1/4	0	3/2
0	0	-1/4	-1/4	1	-1/2

La base corrente che includeva le colonne delle variabili x_1 e x_2 può essere estesa aggiungendo la colonna corrispondente alla nuova variabile di scarto x_5.

Come risulta evidente, questa base non è primale ammissibile (il termine noto è negativo), ma duale ammissibile; quindi conviene continuare con il Simplesso Duale.

Esempio di Applicazione dei Tagli di Gomory

Applichiamo la prima iterazione del simplesso duale:

x_1	x_2	x_3	x_4	x_5	
0	0	-1/4	-1/4	0	-3/2
1	0	1/6	-1/6	0	1
0	1	1/4	1/4	0	3/2
0	0	**-1/4**	-1/4	1	-1/2

Il risultato ottenuto è:

x_1	x_2	x_3	x_4	x_5	
0	0	0	0	-1	-1
1	0	0	-1/3	2/3	2/3
0	1	0	0	1	1
0	0	1	1	-4	2

Esempio di Applicazione dei Tagli di Gomory

Dopo una sola iterazione abbiamo ottenuto il tableau ottimo:

x_1	x_2	x_3	x_4	x_5	
0	0	0	0	-1	-1
1	0	0	-1/3	2/3	2/3
0	1	0	0	1	1
0	0	1	1	-4	2

- La nuova soluzione ottima è $\mathbf{x} = (\frac{2}{3}, 1)$, che non è ancora intera.
- Possiamo aggiungere un nuovo Taglio di Gomory per eliminare la soluzione frazionaria.
- In questo caso la riga è quella relativa a x_1 e il Taglio di Gomory è:

$$-\frac{2}{3}x_4 - \frac{2}{3}x_5 + x_6 = -\frac{2}{3}$$

Esempio di Applicazione dei Tagli di Gomory

Applichiamo ancora il simplesso duale:

x_1	x_2	x_3	x_4	x_5	x_6	
0	0	0	0	-1	0	-1
1	0	0	-1/3	2/3	0	2/3
0	1	0	0	1	0	1
0	0	1	1	-4	0	2
0	0	0	**-2/3**	-2/3	1	-2/3

La soluzione ottima ottenuta è finalmente intera:

x_1	x_2	x_3	x_4	x_5	x_6	
0	0	0	0	-1	0	-1
1	0	0	0	1	-1/2	1
0	1	0	0	1	0	1
0	0	1	0	-5	3/2	1
0	0	0	1	1	-3/2	1

Osservazioni sui Tagli di Gomory

- Nell'esempio otteniamo la soluzione ottima intera $\mathbf{x} = (1, 1)$ aggiungendo solo due Tagli di Gomory.
- In teoria l'uso dei Tagli di Gomory dovrebbe consentire la convergenza a una soluzione intera (con ulteriori accorgimenti che non approfondiamo), ma nella pratica non consentono di ottenere la soluzione ottima intera.
- Esistono altre disuguaglianze valide, spesso studiate per risolvere specifici modelli, che forniscono più velocemente valori del lower bound migliori, mentre, come i Tagli di Gomory, raramente consentono di ottenere direttamente soluzioni intere.
- Migliorare i valori del lower bound consente a metodi come il branch and bound di convergere molto più velocemente alla soluzione ottima intera.

Introduzione ai Metodi di Decomposizione

- Alcuni problemi di programmazione lineare presentano un *elevato* numero di variabili e/o vincoli. Alcuni vincoli potrebbero rendere il problema di difficile soluzione.
- In alcuni casi la struttura della matrice dei vincoli permette di *decomporre* il problema originale in sottoproblemi di più facile soluzione (i.e., *Metodi di Decomposizione*).
- L'approccio più "semplice" prevede di risolvere tali problemi senza considerare esplicitamente tutte le variabili e/o i vincoli del problema.
- Il simplesso *dinamico* definisce un *core* iniziale che considera solo un sottoinsieme delle variabili e dei vincoli del problema. Dopodiché, aggiunge dinamicamente le variabili e i vincoli *mancanti* durante il processo di soluzione.

Introduzione ai Metodi di Decomposizione

- Una possibilità per decomporre facilmente un modello è rappresentata dai metodi di generazione di colonne, che generano dinamicamente le colonne di un problema (*Metodi Column Generation*).
- Metodi di decomposizione più sofisticati sono:
  - Il *Rilassamento Lagrangiano*;
  - Il *Metodo di Decomposizione di Dantzig-Wolfe*;
  - Il *Metodo di Decomposizione di Benders*.
- I metodi di decomposizione possono essere applicati a problemi di programmazione lineare continua, intera e mista intera.
- Per ciascun metodo di decomposizione ci possono essere diversi modi per decomporre un modello.

Introduzione ai Metodi di Decomposizione

- In generale un problema P può avere la seguente forma:

$$z_P = \min \mathbf{c}_1\mathbf{x} + \mathbf{c}_2\mathbf{y} \tag{11}$$
$$s.t.\ \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y} \geq \mathbf{b} \tag{12}$$
$$(\mathbf{x}, \mathbf{y}) \in D \tag{13}$$

dove l'insieme dei punti ammissibili D può essere definito, per esempio, come:

$$D = \{(\mathbf{x}, \mathbf{y}) : \mathbf{D}_1\mathbf{x} \geq \mathbf{d}_1, \mathbf{D}_2\mathbf{y} \geq \mathbf{d}_2, \mathbf{x} \geq \mathbf{0}, \mathbf{y} \geq \mathbf{0} \text{ and integer}\} \tag{14}$$

oppure:

$$D = \{(\mathbf{x}, \mathbf{y}) : \mathbf{D}_1\mathbf{x} + \mathbf{D}_2\mathbf{y} \geq \mathbf{d}, \mathbf{x} \geq \mathbf{0}, \mathbf{y} \geq \mathbf{0} \text{ and integer}\} \tag{15}$$

Introduzione ai Metodi di Decomposizione

- Se si considera il primo esempio il problema P è il seguente:

$$z_P = \min \ \mathbf{c}_1\mathbf{x} + \mathbf{c}_2\mathbf{y} \tag{16}$$
$$s.t. \quad \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y} \geq \mathbf{b} \tag{17}$$
$$\mathbf{D}_1\mathbf{x} \geq \mathbf{d}_1 \tag{18}$$
$$\mathbf{D}_2\mathbf{y} \geq \mathbf{d}_2 \tag{19}$$
$$\mathbf{x} \geq \mathbf{0} \tag{20}$$
$$\mathbf{y} \geq \mathbf{0} \text{ and integer} \tag{21}$$

- Se si rilassano i vincoli (17) allora il problema P si decompone in due problemi indipendenti, uno rispetto alle variabili $\mathbf{x}$, l'altro rispetto alle variabili $\mathbf{y}$.

- Però la soluzione $(\mathbf{x}, \mathbf{y})$ calcolata risolvendo i due problemi indipendenti separatamente potrebbe non soddisfare i vincoli (17) rilassati.

Introduzione ai Metodi di Decomposizione

- Se si considera il secondo esempio il problema P è il seguente:

$$z_P = \min \ \mathbf{c}_1\mathbf{x} + \mathbf{c}_2\mathbf{y} \tag{22}$$

$$s.t. \quad \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y} \geq \mathbf{b} \tag{23}$$

$$\mathbf{D}_1\mathbf{x} + \mathbf{D}_2\mathbf{y} \geq \mathbf{d} \tag{24}$$

$$\mathbf{x} \geq \mathbf{0} \tag{25}$$

$$\mathbf{y} \geq \mathbf{0} \text{ and integer} \tag{26}$$

- Anche se si rilassano i vincoli (23), il problema P non si decompone in due problemi indipendenti come nel primo caso.
- Nonostante questo potrebbe convenire perché i vincoli (23) rendono il problema particolarmente complesso da risolvere.
- Però anche in questo caso la soluzione $(\mathbf{x}, \mathbf{y})$ calcolata risolvendo il problema rilassato potrebbe non soddisfare i vincoli (23).

Introduzione ai Metodi di Decomposizione

- Si consideri, ora, il seguente problema P: LR(1) LR(2) LR(3)

$$z_P = \min \mathbf{cx} \tag{27}$$
$$s.t.\ \mathbf{Ax} \geq \mathbf{b} \tag{28}$$
$$\mathbf{Bx} \geq \mathbf{d} \tag{29}$$
$$\mathbf{x} \geq \mathbf{0} \text{ and integer} \tag{30}$$

  dove i vincoli (28) sono considerati complicati.

- Per semplicità si assume che la regione di ammissibilità sia non vuota e limitata.

- Si denota con LP il rilassamento lineare del problema P, ottenuto rilassando i vincoli di integralità (30), e con z_{LP} il valore della sua soluzione ottima.

Rilassamento Lagrangiano

- Il rilassamento Lagrangiano permette di ottenere un lower bound per il problema P nel seguente modo:
  - alcuni vincoli considerati *difficili* sono rimossi;
  - i vincoli rimossi sono inseriti nella funzione obiettivo per mezzo delle *penalità Lagrangiane*.
- Per esempio, se nel problema P (Vai al problema P), si rilassano i vincoli (28) usando il vettore non negativo di penalità Lagrangiane λ, si ottiene la seguente formulazione LR:

$$z_{LR}(\lambda) = \min \mathbf{cx} + \lambda(\mathbf{b} - \mathbf{Ax}) \quad (31)$$

$$s.t.\ \mathbf{Bx} \geq \mathbf{d} \quad (32)$$

$$\mathbf{x} \geq \mathbf{0} \text{ and integer} \quad (33)$$

Rilassamento Lagrangiano

Teorema (Dualità Lagrangiana Debole)

$z_{LR}(\lambda)$ è un valido lower bound al valore della soluzione ottima di P, i.e, $z_{LR}(\lambda) \leq z_P$, per ogni $\lambda \geq \mathbf{0}$.

Dimostrazione

- *Sia $\mathbf{x}^*$ la soluzione ottima di P.*
- *Siccome $\mathbf{x}^*$ è una soluzione ammissibile anche per $LR(\lambda)$, per ogni $\lambda \geq \mathbf{0}$, si ha che:*

$$z_{LR}(\lambda, \mathbf{x}^*) = \mathbf{c}\mathbf{x}^* + \lambda(\mathbf{b} - \mathbf{A}\mathbf{x}^*) \geq z_{LR}(\lambda)$$

- *Però $\lambda(\mathbf{b} - \mathbf{A}\mathbf{x}^*) \leq 0$, perché $\lambda \geq \mathbf{0}$ e $\mathbf{A}\mathbf{x}^* \geq \mathbf{b}$, quindi:*

$$z_{LR}(\lambda) \leq z_{LR}(\lambda, \mathbf{x}^*) \leq z_P, \quad \forall \lambda \geq \mathbf{0}.$$

Rilassamento Lagrangiano

- Il sottoproblema LR è detto *problema Lagrangiano*.
- Per identificare il vettore di penalità λ che massimizza il lower bound $z_{LR}(\lambda)$ si deve risolvere il cosiddetto *Lagrangiano Duale*, che può essere formulato come segue:

$$z_{LR} = \max \{z_{LR}(\lambda) : \lambda \geq \mathbf{0}\} \tag{34}$$

dove per ogni vettore di penalità fissato deve essere risolto un problema Lagrangiano LR, che può essere riscritto come:

$$z_{LR}(\lambda) = \min\ (\mathbf{c} - \lambda\mathbf{A})\mathbf{x} + \lambda\mathbf{b} \tag{35}$$

$$s.t.\ \mathbf{Bx} \geq \mathbf{d} \tag{36}$$

$$\mathbf{x} \geq \mathbf{0} \text{ and integer} \tag{37}$$

Rilassamento Lagrangiano

- Il lower bound z_{LR} fornito dal Lagrangiano duale è maggiore o uguale a quello fornito dalla soluzione ottima z_{LP} del rilassamento lineare del problema P, i.e. $z_{LR} \geq z_{LP}$.
- Se il lower bound z_{LR} è strettamente minore alla soluzione ottima del problema P, $z_{LR} < z_P$, allora si dice che c'è *Duality Gap*.
- Si noti che è possibile aggiungere al problema LR dei nuovi vincoli ridondanti nella formulazione originale P, ma che possono aiutare la convergenza ed eventualmente migliorare il lower bound z_{LR}.
- Per risolvere il problema LR il metodo più usato è il *subgradiente*.

Rilassamento Lagrangiano: esempio

Set Covering Problem

- Il Set Covering Problem (SC) è il problema di *coprire* le righe di una matrice A, con coefficienti $a_{ij} \in \{0,1\}$ e di dimensione $m \times n$, con un sottoinsieme S di colonne di costo minimo.
- Sia x_j una variabile binaria 0-1 uguale a 1 se la colonna j di costo c_j è in soluzione, 0 altrimenti. Un modello per il problema SC è:

$$z_{SC} = \min \sum_{j=1}^{n} c_j x_j \tag{38}$$

$$s.t. \sum_{j=1}^{n} a_{ij} x_j \geq 1, \quad i = 1, \ldots, m \tag{39}$$

$$x_j \in \{0,1\}, \qquad j = 1, \ldots, n \tag{40}$$

Rilassamento Lagrangiano: esempio

Esempio numerico:

- Si consideri l'istanza di set covering dove $m = 3$, $n = 6$ e il vettore dei costi è $\mathbf{c} = [1, 2, 3, 4, 5, 6]$
- Mentre, la matrice A dei coefficienti è la seguente:

$$A = \begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left[\begin{array}{cccccc} 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \end{array}\right] \end{array}$$

- Soluzione ottima di costo 5 è il sottoinsieme di colonne $S = \{2, 3\}$.

Rilassamento Lagrangiano: esempio

- Il Rilassamento Lagrangiano del problema SC rispetto ai vincoli (39) è il seguente:

$$z_{LR}(\lambda) = \min \sum_{j=1}^{n} c_j x_j + \sum_{i=1}^{m} \lambda_i (1 - \sum_{j=1}^{n} a_{ij} x_j)$$
$$s.t.\ x_j \in \{0,1\}, \qquad j = 1, \ldots, n$$

  dove $\lambda_i \geq 0$, $i = 1, \ldots, m$.

- Il problema $LR(\lambda)$ può essere riscritto come segue:

$$z_{LR}(\lambda) = \min \sum_{j=1}^{n} (c_j - \sum_{i=1}^{m} \lambda_i a_{ij}) x_j + \sum_{i=1}^{m} \lambda_i$$
$$s.t.\ x_j \in \{0,1\}, \qquad j = 1, \ldots, n$$

Rilassamento Lagrangiano: esempio

- Se si definisce il *costo penalizzato* $c'_j = c_j - \sum_{i=1}^{m} \lambda_i a_{ij}, j = 1, \ldots, n$, il problema $LR(\lambda)$ può essere riscritto come segue:

$$z_{LR}(\lambda) = \min \sum_{j=1}^{n} c'_j x_j + \sum_{i=1}^{m} \lambda_i$$
$$s.t.\ x_j \in \{0,1\}, \quad j = 1, \ldots, n$$

la cui soluzione ottima $\mathbf{x}^*$ può essere calcolata ponendo, per ogni $j = 1, \ldots, n$:

$$x_j^* = \begin{cases} 1 & \text{se } c'_j \leq 0; \\ 0 & \text{altrimenti.} \end{cases}$$

Rilassamento Lagrangiano: esempio

Esempio numerico:

- Si consideri il seguente problem di Set Covering:

$$
\begin{aligned}
z_{SC} = \min\ & 2x_1 + 3x_2 + 4x_3 + 5x_4 \\
s.t.\ & x_1 + x_3 \geq 1 \\
& x_1 + x_4 \geq 1 \\
& x_2 + x_3 + x_4 \geq 1 \\
& x_1, x_2, x_3, x_4 \in \{0, 1\}
\end{aligned}
$$

- La soluzione ottima di costo 5 è $x_1 = x_2 = 1$ e $x_3 = x_4 = 0$.
- Si rilassano mediante le penalità Lagrangiane λ_1, λ_2 e λ_3 i vincoli di set covering.

Rilassamento Lagrangiano: esempio

- Il problema Lagrangiano LR è il seguente:

$$z_{LR}(\lambda) = \min\ 2x_1 + 3x_2 + 4x_3 + 5x_4 + \lambda_1(1 - x_1 - x_3) + \lambda_2(1 - x_1 - x_4) + \lambda_3(1 - x_2 - x_3 - x_4)$$
$$s.t.\ x_1, x_2, x_3, x_4 \in \{0, 1\}$$

che equivale al problema:

$$z_{LR}(\lambda) = \min\ c'_1 x_1 + c'_2 x_2 + c'_3 x_3 + c'_4 x_4 + \lambda_1 + \lambda_2 + \lambda_3$$
$$s.t.\ x_1, x_2, x_3, x_4 \in \{0, 1\}$$

dove

$$c'_1 = 2 - \lambda_1 - \lambda_2$$
$$c'_2 = 3 - \lambda_3$$
$$c'_3 = 4 - \lambda_1 - \lambda_3$$
$$c'_4 = 5 - \lambda_2 - \lambda_3$$

Rilassamento Lagrangiano: esempio

- Se $\lambda_1 = 1.5$, $\lambda_2 = 1.6$ e $\lambda_3 = 2.2$ si ha:

$$
\begin{aligned}
c'_1 &= 2 - \lambda_1 - \lambda_2 = -1.1 \\
c'_2 &= 3 - \lambda_3 = 0.8 \\
c'_3 &= 4 - \lambda_1 - \lambda_3 = 0.3 \\
c'_4 &= 5 - \lambda_2 - \lambda_3 = 1.2
\end{aligned}
$$

- La soluzione ottima $\mathbf{x}^*$ di $LR(\lambda)$ è:

$$x^*_1 = 1, x^*_2 = x^*_3 = x^*_4 = 0$$

di costo $z_{LR}(\lambda) = -1.1 + 0 + 0 + 0 + 1.5 + 1.6 + 2.2 = 4.2$ (≤ 5).

Rilassamento Lagrangiano: esempio

- Se $\lambda_1 = 1$, $\lambda_2 = 1$ e $\lambda_3 = 3$ si ha:

$$\begin{aligned} c'_1 &= 2 - \lambda_1 - \lambda_2 = 0 \\ c'_2 &= 3 - \lambda_3 = 0 \\ c'_3 &= 4 - \lambda_1 - \lambda_3 = 0 \\ c'_4 &= 5 - \lambda_2 - \lambda_3 = 1 \end{aligned}$$

- La soluzione ottima $\mathbf{x}^*$ di $LR(\lambda)$ è:

$$x^*_1 = x^*_2 = x^*_3 = x^*_4 = 0$$

di costo $z_{LR}(\lambda) = 0 + 0 + 0 + 0 + 1 + 1 + 3 = 5$ (≤ 5).

- Anche se $z_{LR}(\lambda) = z_{SC} = 5$, si noti che $\mathbf{x}^*$ non è una soluzione ammissibile per SC.

- Esistono soluzioni ottime alternative di $LR(\lambda)$ di costo $z_{LR}(\lambda) = 5$ tra le quali c'è anche la soluzione ottima di SC.

Rilassamento Lagrangiano: esempio

Esempio Numerico:

- Si consideri il seguente problema:

$$z_P = \min\ 3x_1 + 7x_2 + 10x_3$$
$$s.t.\ x_1 + 3x_2 + 5x_3 \geq 7$$
$$x_1, x_2, x_3 \in \{0,1\}$$

- Il problema Lagrangiano è il seguente:

$$z_{LR}(\lambda) = \min\ 3x_1 + 7x_2 + 10x_3 + \lambda(7 - x_1 - 3x_2 - 5x_3)$$
$$s.t.\ x_1, x_2, x_3 \in \{0,1\}$$

che può essere riscritto come:

$$z_{LR}(\lambda) = \min\ (3-\lambda)x_1 + (7-3\lambda)x_2 + (10-5\lambda)x_3 + 7\lambda$$
$$s.t.\ x_1, x_2, x_3 \in \{0,1\}$$

Rilassamento Lagrangiano: esempio

- Calcoliamo $z_{LR}(\lambda)$ per $\lambda \geq 0$:
  - $\lambda = 0$, $z_{LR}(0) = 0$ e $\mathbf{x} = (0,0,0)$;
  - $\lambda = 1$, $z_{LR}(1) = 7$ e $\mathbf{x} = (0,0,0)$;
  - $\lambda = 2$, $z_{LR}(2) = 14$ e $\mathbf{x} = (0,0,0)$ oppure $\mathbf{x} = (0,0,1)$;
  - $\lambda = \frac{7}{3}$, $z_{LR}(\frac{7}{3}) = \frac{44}{3}$ e $\mathbf{x} = (0,0,1)$ oppure $\mathbf{x} = (0,1,1)$;
  - $\lambda = 3$, $z_{LR}(3) = 14$ e $\mathbf{x} = (0,0,1)$ oppure $\mathbf{x} = (1,1,1)$;
  - $\lambda > 3$, $z_{LR}(\lambda) = -2\lambda + 20$ e $\mathbf{x} = (1,1,1)$.
- Quindi esiste un gap di dualità in quanto $z_{LR} = \frac{44}{3}$ $(= z_{LR}(\frac{7}{3}))$, mentre la soluzione ottima $\mathbf{x}^* = (0,1,1)$ del problema P ha un costo pari a $z_P = 17$.
- Si noti che la soluzione ottima $\mathbf{x}^* = (0,1,1)$ corrisponde a una delle soluzioni di $z_{LR}(\frac{7}{3})$.

Dualità Lagrangiana Forte

Teorema (Dualità Lagrangiana Forte)

Dato il problema P (▸ Vai al problema P), sia $\bar{\mathbf{x}}$ la soluzione ottima di $z_{LR}(\bar{\lambda})$ per un dato $\bar{\lambda} \geq 0$. Se $\bar{\mathbf{x}}$ e $\bar{\lambda}$ soddisfano le seguenti condizioni:

- $\bar{\mathbf{x}}$ *è ammissibile per P (i.e.,* $\mathbf{A}\bar{\mathbf{x}} \geq \mathbf{b}$*)*
- $\bar{\lambda}(\mathbf{b} - \mathbf{A}\bar{\mathbf{x}}) = 0$,

allora $\bar{\mathbf{x}}$ è la soluzione ottima di P e $z_{LR} = z_{LR}(\bar{\lambda})$.

Dimostrazione

La dimostrazione è divisa in due parti:

- *Si dimostra che la soluzione $\bar{\mathbf{x}}$ è una soluzione ottima di P;*
- *Si dimostra che $z_{LR} = z_{LR}(\bar{\lambda})$.*

Dualità Lagrangiana Forte

Dimostrazione (la soluzione $\bar{\mathbf{x}}$ è una soluzione ottima di P)

Essendo $\bar{\mathbf{x}}$ *una soluzione ammissibile di P si ha:*

$$\mathbf{c}\bar{\mathbf{x}} \geq z_P. \tag{41}$$

Per il teorema della dualità Lagrangiana debole si ha:

$$z_P \geq z_{LR}(\bar{\lambda}) = \mathbf{c}\bar{\mathbf{x}} + \underbrace{\bar{\lambda}(\mathbf{b} - \mathbf{A}\bar{\mathbf{x}})}_{=0}. \tag{42}$$

Per cui, da (41) e (42) si ottiene:

$$\mathbf{c}\bar{\mathbf{x}} \geq z_P \geq \mathbf{c}\bar{\mathbf{x}} \quad \Longrightarrow \quad z_P = \mathbf{c}\bar{\mathbf{x}}\ (= z_{LR}(\bar{\lambda})). \tag{43}$$

Dualità Lagrangiana Forte

Dimostrazione ($z_{LR} = z_{LR}(\bar{\lambda})$)

Dalla definizione di Lagrangiano Duale z_{LR} si ha:

$$z_{LR} \geq z_{LR}(\bar{\lambda}) \tag{44}$$

Mentre, dalla dualità Lagrangiana debole si ha:

$$z_P \geq z_{LR} \tag{45}$$

Quindi, da (43), (44) e (45) si ottiene:

$$z_{LR} = z_{LR}(\bar{\lambda}). \tag{46}$$

Rilassamento Lineare vs Rilassamento Lagrangiano

- Per comodità il problema P (Vai al problema P) può essere riscritto nella seguente forma compatta:

$$z_P = \min \mathbf{cx} \tag{47}$$
$$s.t.\ \mathbf{Ax} \geq \mathbf{b} \tag{48}$$
$$\mathbf{x} \in X \tag{49}$$

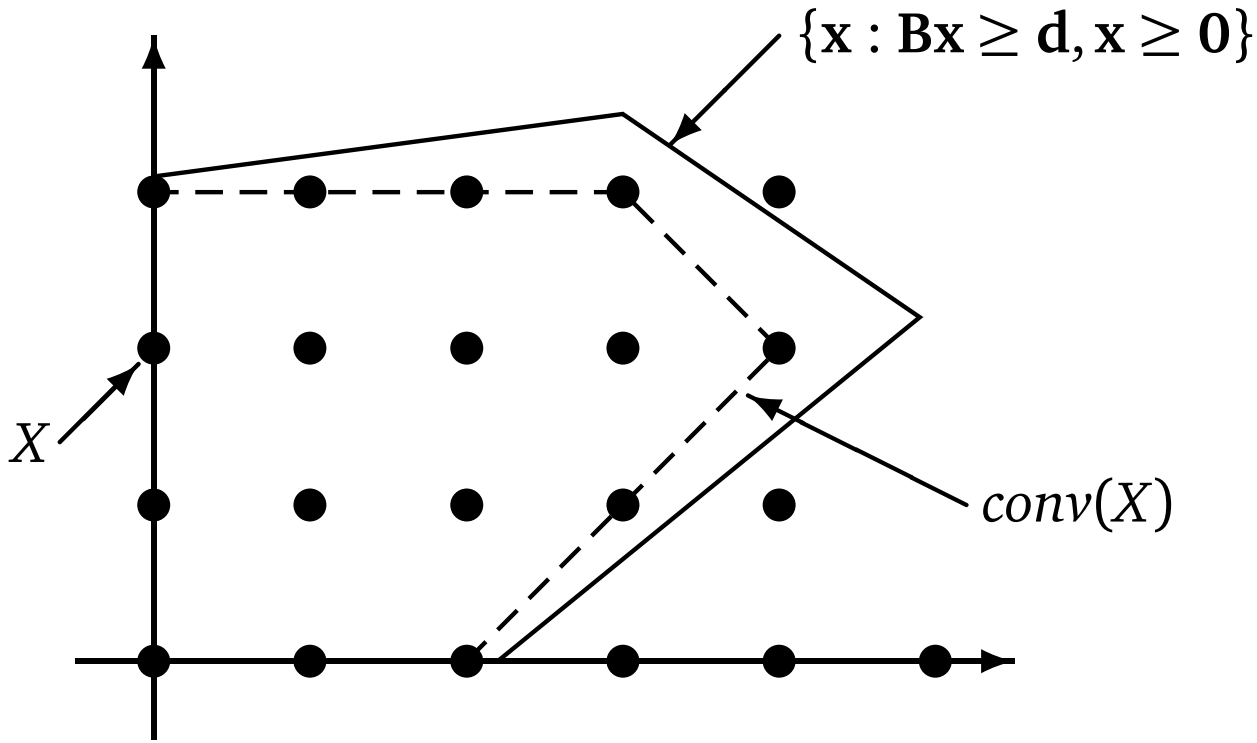
  dove $X = \{\mathbf{x} : \mathbf{Bx} \geq \mathbf{d}, \mathbf{x} \geq \mathbf{0} \text{ and integer}\}$.

- Il rilassamento lineare (LP) del problema P è ottenuto rilassando il vincolo di interezza delle variabili $\mathbf{x}$, i.e. $X' = \{\mathbf{x} : \mathbf{Bx} \geq \mathbf{d}, \mathbf{x} \geq \mathbf{0}\}$.

- Si ricorda che il valore ottimo z_{LP} del rilassamento lineare di P è un valido lower bound al valore z_P della soluzione ottima di P, i.e., $z_{LP} \leq z_P$.

Rilassamento Lineare vs Rilassamento Lagrangiano

- Si denota con $conv(X)$ l'inviluppo convesso di X, che è dato dall'intersezione di tutti gli insiemi convessi che contengono X.



Rilassamento Lineare vs Rilassamento Lagrangiano

Teorema (Caratterizzazione del Lagrangiano Duale)

Dato il problema $z_P = \min\{\mathbf{cx} : \mathbf{Ax} \geq \mathbf{b}, \mathbf{x} \in X\}$. *Se si rilassano in modo Lagrangiano i vincoli* $\mathbf{Ax} \geq \mathbf{b}$, *il costo della soluzione ottima* z_{LR} *del Lagrangiano Duale è uguale al costo della soluzione ottima del seguente problema di programmazione lineare:*

$$z_{LR} = \min \quad \mathbf{cx} \tag{50}$$

$$s.t. \quad \mathbf{Ax} \geq \mathbf{b} \tag{51}$$

$$\mathbf{x} \in conv(X) \tag{52}$$

Dimostrazione

Il Lagrangiano Duale è definito come:

$$z_{LR} = \max\{z_{LR}(\lambda) : \lambda \geq \mathbf{0}\} = \max_{\lambda \geq \mathbf{0}} \left\{ \min_{\mathbf{x} \in X} \{\mathbf{cx} + \lambda(\mathbf{b} - \mathbf{Ax})\} \right\} \tag{53}$$

Rilassamento Lineare vs Rilassamento Lagrangiano

Dimostrazione (Primo passo)

Se T è l'insieme dei punti estremi del poliedro X (che si è ipotizzato limitato e non vuoto) allora:

$$z_{LR} = \max_{\lambda \geq \mathbf{0}} \left\{ \min_{t \in T} \left\{ \mathbf{c}\mathbf{x}^t + \lambda(\mathbf{b} - \mathbf{A}\mathbf{x}^t) \right\} \right\} \tag{54}$$

che equivale al seguente problema:

$$z_{LR} = \max \ z \tag{55}$$

$$s.t. \ \ z \leq \mathbf{c}\mathbf{x}^t + \lambda(\mathbf{b} - \mathbf{A}\mathbf{x}^t), \quad t \in T \tag{56}$$

$$\lambda \geq \mathbf{0} \tag{57}$$

(NOTA: questo è il master della decomposizione di Benders)

Rilassamento Lineare vs Rilassamento Lagrangiano

Dimostrazione (Secondo passo)

Il duale del problema (55)–(57) è:

$$z_{LR} = \min \sum_{t \in T} (\mathbf{c}\mathbf{x}^t)\alpha_t \tag{58}$$

$$s.t. \sum_{t \in T} (\mathbf{b} - \mathbf{A}\mathbf{x}^t)\alpha_t \leq \mathbf{0} \tag{59}$$

$$\sum_{t \in T} \alpha_t = 1 \tag{60}$$

$$\alpha_t \geq 0, \qquad t \in T \tag{61}$$

(NOTA: questo è il master della decomposizione di Dantzig-Wolfe)

Rilassamento Lineare vs Rilassamento Lagrangiano

Dimostrazione (Conclusioni)

Definendo $\mathbf{x} = \sum_{t \in T} \mathbf{x}^t \alpha_t$, *dal problema (58)–(61) si ottiene il teorema:*

$$\begin{aligned} \min \quad & \mathbf{cx} \\ s.t. \quad & \mathbf{Ax} \geq \mathbf{b} \\ & \mathbf{x} \in conv(X) \end{aligned}$$

- Questo teorema è importante perché:
  - caratterizza il Lagrangiano Duale;
  - stabilisce l'equivalenza tra il rilassamento Lagrangiano e le decomposizioni di Dantzig-Wolfe e Benders.

Rilassamento Lineare vs Rilassamento Lagrangiano

Dettaglio dei passaggi chiave del teorema

Per calcolare il duale del problema (55)–(57) conviene riscriverlo come:

$$z_{LR} = -\min \; -z \tag{62}$$

$$s.t. \; -(\mathbf{b} - \mathbf{A}\mathbf{x}^t)\lambda + z \leq \mathbf{c}\mathbf{x}^t, \quad t \in T \tag{63}$$

$$\lambda \geq \mathbf{0} \tag{64}$$

Al seguente duale basterà poi cambiare il segno alle variabili α_t:

$$z_{LR} = -\max \sum_{t \in T} (\mathbf{c}\mathbf{x}^t)\alpha_t \tag{65}$$

$$s.t. \sum_{t \in T} -(\mathbf{b} - \mathbf{A}\mathbf{x}^t)\alpha_t \leq \mathbf{0} \tag{66}$$

$$\sum_{t \in T} \alpha_t = -1 \tag{67}$$

$$\alpha_t \leq 0, \qquad t \in T \tag{68}$$

Rilassamento Lineare vs Rilassamento Lagrangiano

Per l'ultmo passaggio che porta alle conclusioni, conviene riscrivere il problema (58)–(61) come:

$$z_{LR} = \min \mathbf{c} \sum_{t \in T} \mathbf{x}^t \alpha_t \tag{69}$$

$$s.t.\ \mathbf{b} \sum_{t \in T} \alpha_t - \mathbf{A} \sum_{t \in T} \mathbf{x}^t \alpha_t \leq \mathbf{0} \tag{70}$$

$$\sum_{t \in T} \alpha_t = 1 \tag{71}$$

$$\alpha_t \geq 0, \qquad t \in T \tag{72}$$

Dopodiché, basterà sostituire $\mathbf{x} = \sum_{t \in T} \mathbf{x}^t \alpha_t$ e osservare che il vincolo (70) si può riscrivere come:

$$-\mathbf{A}\mathbf{x} \leq -\mathbf{b} \tag{73}$$

Rilassamento Lineare vs Rilassamento Lagrangiano

Teorema (Relazione tra Rilassamento Lagrangiano e LP)

Il lower bound ottenuto dal rilassamento Lagrangiano del problema P è sempre maggiore o uguale del lower bound ottenuto dal Rilassamento Lineare di P, i.e., $z_{LP} \leq z_{LR}$.

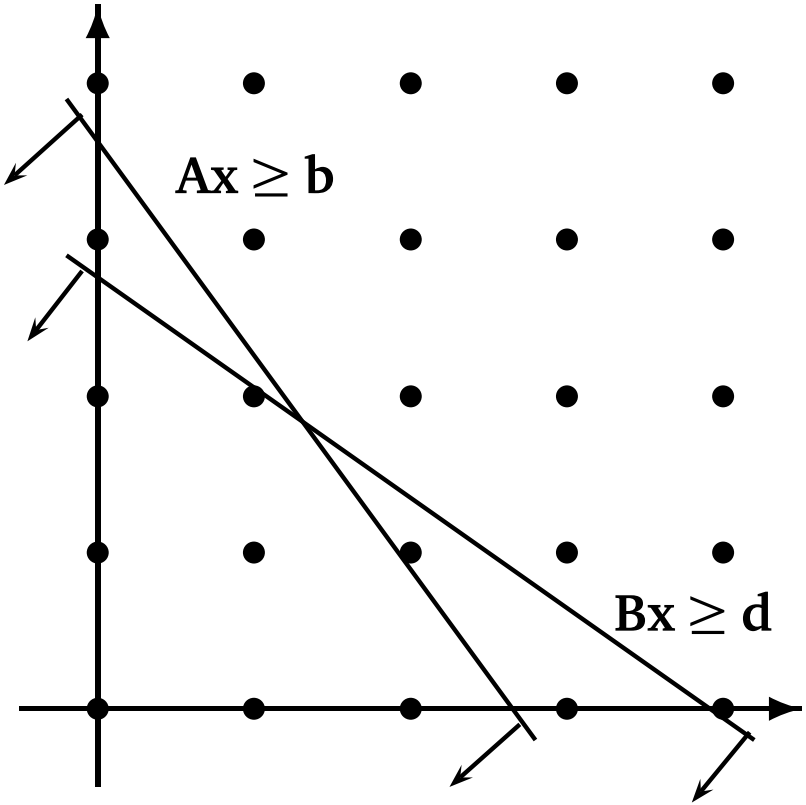
Dimostrazione

Siccome $conv(X) \subseteq \{\mathbf{x} : \mathbf{Bx} \geq \mathbf{d}, \mathbf{x} \geq \mathbf{0}\}$, *l'insieme delle soluzioni del Lagrangiano Duale dato da* $\{\mathbf{x} : \mathbf{Ax} \geq \mathbf{b}, \mathbf{x} \in conv(X)\}$ *è contenuto nell'insieme delle soluzioni di LP dato da* $\{\mathbf{x} : \mathbf{Ax} \geq \mathbf{b}, \mathbf{Bx} \geq \mathbf{d}, \mathbf{x} \geq \mathbf{0}\}$:

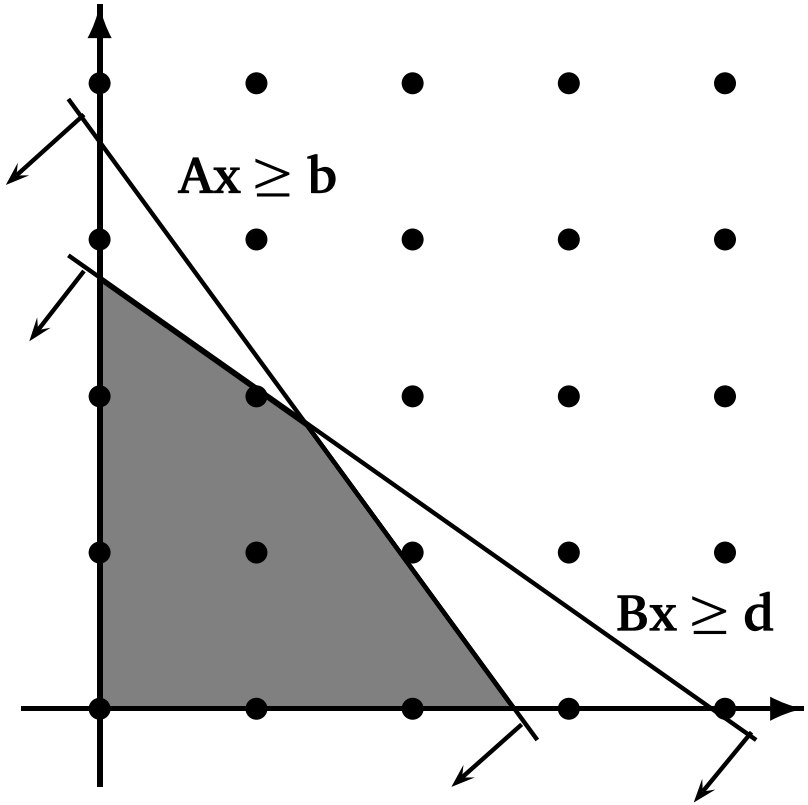
$$\{\mathbf{x} : \mathbf{Ax} \geq \mathbf{b}, \mathbf{x} \in conv(X)\} \subseteq \{\mathbf{x} : \mathbf{Ax} \geq \mathbf{b}, \mathbf{Bx} \geq \mathbf{d}, \mathbf{x} \geq \mathbf{0}\}. \quad (74)$$

da cui segue che $z_{LP} \leq z_{LR}$.

Rilassamento Lineare vs Rilassamento Lagrangiano

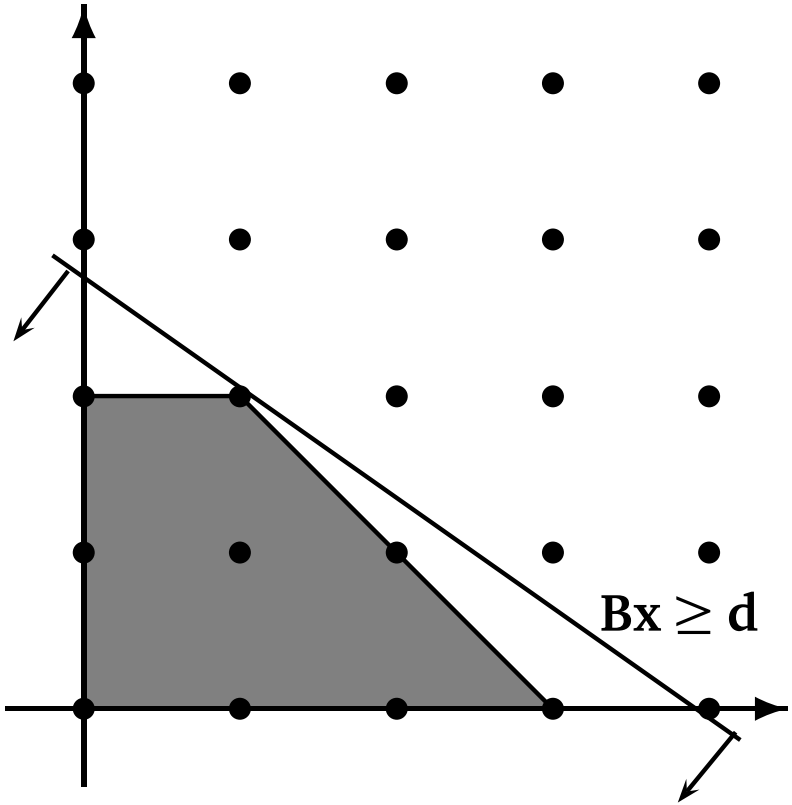


(a) $\{\mathbf{x} : \mathbf{Ax} \geq \mathbf{b}, \mathbf{Bx} \geq \mathbf{d}, \mathbf{x} \geq \mathbf{0}$ intero$\}$

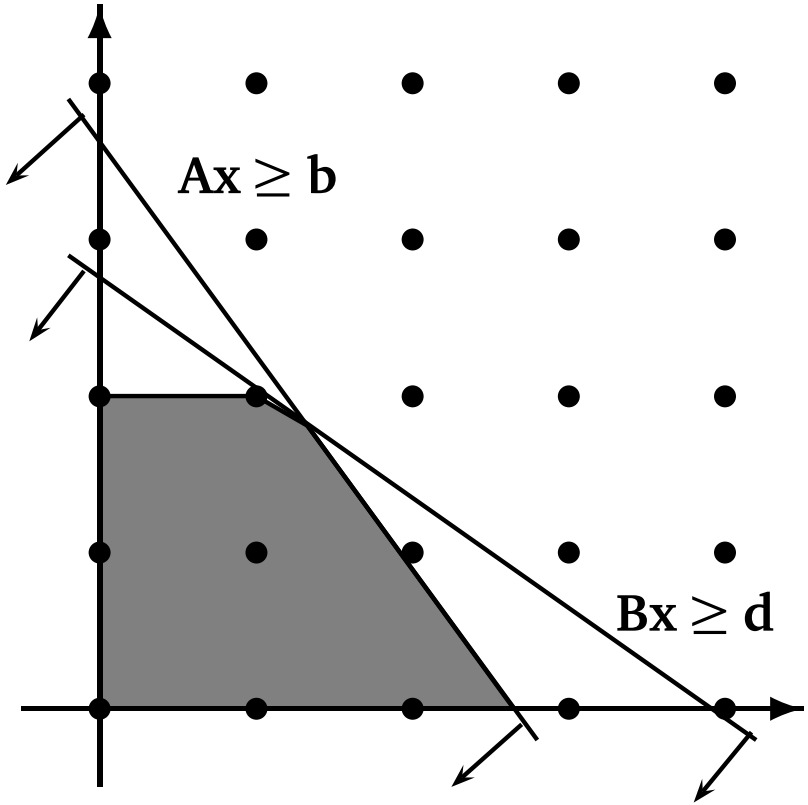


(b) $\{\mathbf{x} : \mathbf{Ax} \geq \mathbf{b}, \mathbf{Bx} \geq \mathbf{d}, \mathbf{x} \geq \mathbf{0}\}$

Rilassamento Lineare vs Rilassamento Lagrangiano



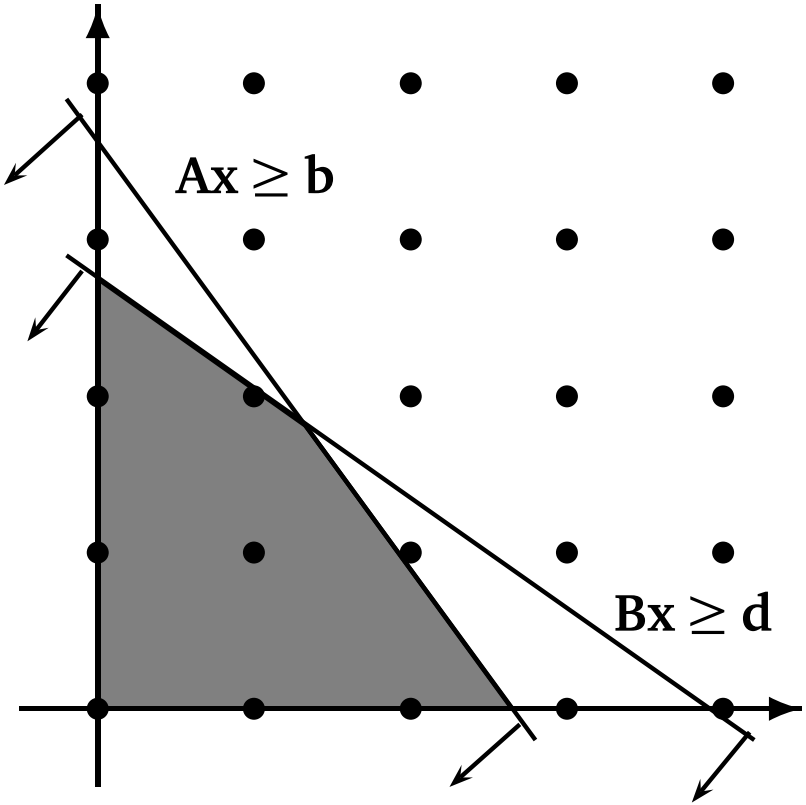
(a) $conv(X)$



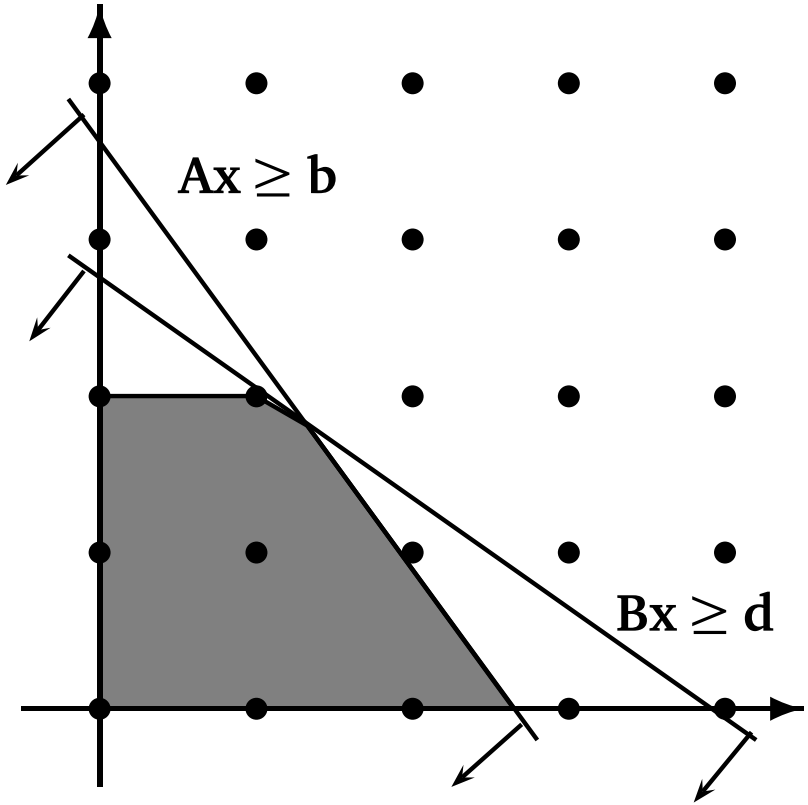
(b) $\{\mathbf{x} : \mathbf{Ax} \geq \mathbf{b}, \mathbf{x} \in conv(X)\}$

Nota: ricordiamo che $X = \{\mathbf{x} : \mathbf{Bx} \geq \mathbf{d}, \mathbf{x} \geq \mathbf{0} \text{ and integer}\}$

Rilassamento Lineare vs Rilassamento Lagrangiano



(a) $\{\mathbf{x} : \mathbf{Ax} \geq \mathbf{b}, \mathbf{Bx} \geq \mathbf{d}, \mathbf{x} \geq \mathbf{0}\}$



(b) $\{\mathbf{x} : \mathbf{Ax} \geq \mathbf{b}, \mathbf{x} \in conv(X)\}$

Rilassamento Lineare vs Rilassamento Lagrangiano

Teorema

Si hanno le seguenti proprietà:

- $z_P = z_{LR}$ *se e solo se:*

$$conv(X \cap \{\mathbf{x} : \mathbf{Ax} \geq \mathbf{b}, \mathbf{x} \geq \mathbf{0}\}) = conv(X) \cap conv(\{\mathbf{x} : \mathbf{Ax} \geq \mathbf{b}, \mathbf{x} \geq \mathbf{0}\});$$

- $z_{LP} = z_{LR}$ *se (Proprietà di Integralità):*

$$conv(X) = conv(\{\mathbf{x} : \mathbf{Bx} \geq \mathbf{d}, \mathbf{x} \geq \mathbf{0}\}).$$

Metodo del Subgradiente

- Il Lagrangiano Duale può essere risolto come problema di programmazione lineare, per esempio tramite le decomposizioni di Dantzig-Wolfe e Benders.
- Risolvere il Lagrangiano Duale come problema di Programmazione Lineare può essere *oneroso* dal punto di vista computazionale.
- Per risolvere il Lagrangiano Duale si possono utilizzare dei metodi *euristici*, come il metodo del *subgradiente*.
- Per introdurre il metodo del subgradiente è necessario dimostrare che la funzione Lagrangiana $z_{LR}(\lambda)$ è concava e definire cos'è un subgradiente.

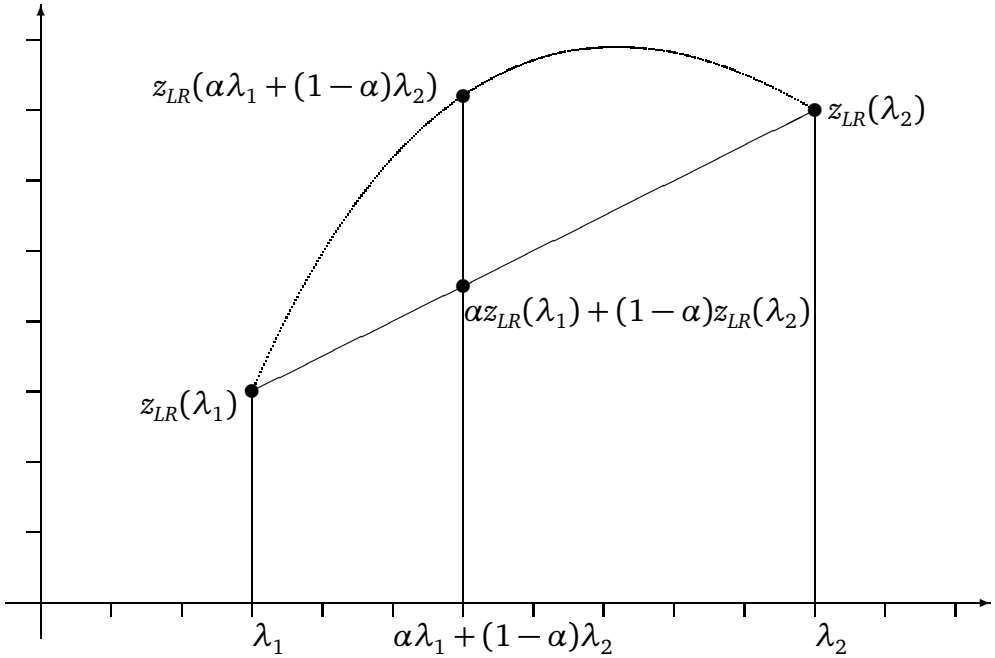
Metodo del Subgradiente

Teorema

La Funzione Lagrangiana $z_{LR}(\bar{\lambda}) = \mathbf{c}\bar{\mathbf{x}} + \bar{\lambda}(\mathbf{b} - \mathbf{A}\bar{\mathbf{x}})$ *è concava.*

Dimostrazione

Si consideri il seguente esempio:



Metodo del Subgradiente

Dimostrazione

Siano $\lambda_1, \lambda_2 \geq \mathbf{0}$ *e* $\bar{\lambda}$ *una combinazione convessa di* λ_1 *e* λ_2*, i.e.* $\bar{\lambda} = \alpha\lambda_1 + (1-\alpha)\lambda_2$ *con* $\alpha \in [0,1]$*. Sia* $\bar{\mathbf{x}}$ *la soluzione ottima di* $LR(\bar{\lambda})$*:*

$$z_{LR}(\bar{\lambda}) = \mathbf{c}\bar{\mathbf{x}} + \bar{\lambda}(\mathbf{b} - \mathbf{A}\bar{\mathbf{x}}) \tag{75}$$

$\bar{\mathbf{x}}$ *è una soluzione ammissibile di* $LR(\lambda_1)$ *e* $LR(\lambda_2)$*, quindi:*

$$z_{LR}(\lambda_1) \leq \mathbf{c}\bar{\mathbf{x}} + \lambda_1(\mathbf{b} - \mathbf{A}\bar{\mathbf{x}}) \tag{76}$$

$$z_{LR}(\lambda_2) \leq \mathbf{c}\bar{\mathbf{x}} + \lambda_2(\mathbf{b} - \mathbf{A}\bar{\mathbf{x}}) \tag{77}$$

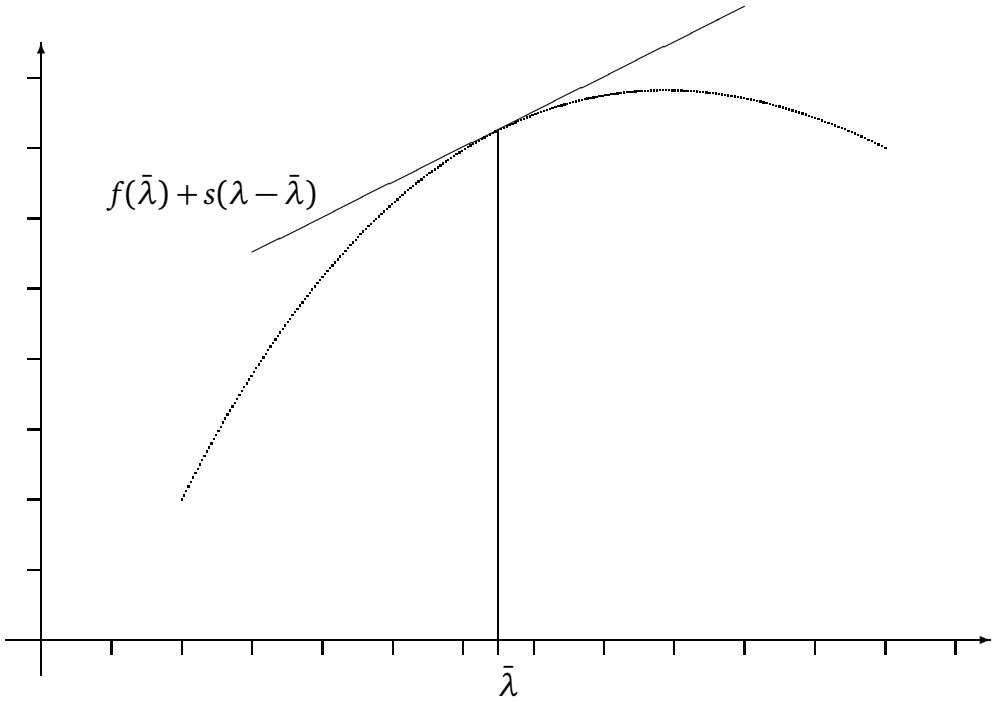
da cui si ottiene:

$$\alpha z_{LR}(\lambda_1) + (1-\alpha) z_{LR}(\lambda_2) \leq \mathbf{c}\bar{\mathbf{x}} + \underbrace{(\alpha\lambda_1 + (1-\alpha)\lambda_2)}_{=\bar{\lambda}}(\mathbf{b} - \mathbf{A}\bar{\mathbf{x}}) = z_{LR}(\bar{\lambda}).$$

Metodo del Subgradiente

- Un vettore s é detto *subgradiente* della funzione $f(\lambda)$ nel punto $\bar{\lambda}$ se soddisfa la seguente condizione:

$$f(\lambda) \leq f(\bar{\lambda}) + s(\lambda - \bar{\lambda}) \tag{78}$$



Metodo del Subgradiente

- Si vuole definire il subgradiente della funzione Lagrangiana:

$$z_{LR}(\bar{\lambda}) = \mathbf{c}\bar{\mathbf{x}} + \bar{\lambda}(\mathbf{b} - \mathbf{A}\bar{\mathbf{x}}) \tag{79}$$

- Per ogni $\lambda \geq \mathbf{0}$ si ha che:

$$z_{LR}(\lambda) \leq \mathbf{c}\bar{\mathbf{x}} + \lambda(\mathbf{b} - \mathbf{A}\bar{\mathbf{x}}) \tag{80}$$

- Sottraendo dalla (80) la (79) si ottiene:

$$z_{LR}(\lambda) - z_{LR}(\bar{\lambda}) \leq (\lambda - \bar{\lambda})(\mathbf{b} - \mathbf{A}\bar{\mathbf{x}}) \tag{81}$$

oppure

$$z_{LR}(\lambda) \leq z_{LR}(\bar{\lambda}) + (\lambda - \bar{\lambda})(\mathbf{b} - \mathbf{A}\bar{\mathbf{x}}) \tag{82}$$

Per cui $\mathbf{s} = (\mathbf{b} - \mathbf{A}\bar{\mathbf{x}})$ è un subgradiente della funzione Lagrangiana $z_{LR}(\lambda)$ calcolata nel punto $\bar{\lambda}$.

Metodo del Subgradiente

- Affinché $z_{LR}(\lambda)$ sia maggiore di $z_{LR}(\bar{\lambda})$ è necessario muoversi nella direzione del subgradiente, perché $(\lambda - \bar{\lambda})(\mathbf{b} - \mathbf{A}\bar{\mathbf{x}}) > 0$. Quindi, è necessario che:

$$\lambda = \bar{\lambda} + \theta \mathbf{s} \tag{83}$$

  dove $\theta > 0$ è lo spostamento lungo il subgradiente.

- Lo spostamento θ può essere calcolato in diversi modi:
  - in modo *esatto* risolvendo il problema:

$$\theta^* = \text{argmax}\left\{z_{LR}(\bar{\lambda} + \theta \mathbf{s}) : \theta > 0\right\} \tag{84}$$

    garantendo che $z_{LR}(\bar{\lambda} + \theta^* \mathbf{s}) \geq z_{LR}(\bar{\lambda})$, però potrebbe essere molto costoso.

  - in modo euristico, solitamente valutando una semplice equazione, ma non può essere garantito che $z_{LR}(\bar{\lambda} + \theta^* \mathbf{s}) \geq z_{LR}(\bar{\lambda})$.

Metodo del Subgradiente

- Ad ogni iterazione k del subgradiente lo spostamento θ^k può essere calcolato con uno dei seguenti approcci euristici:
  - *Polyak-type step size*:

$$\theta^k = \beta^k \frac{\bar{z} - z_{LR}(\boldsymbol{\lambda}^k)}{\|\mathbf{s}^k\|_2^2} \tag{85}$$

dove $\bar{z}$ è una stima per difetto della soluzione ottima del Lagrangiano duale z_{LR}, i.e., $\bar{z} \leq z_{LR}$. In questo caso, se $0 < \beta^k \leq 2$ allora la convergenza di $z_{LR}(\boldsymbol{\lambda}^k)$ a z_{LR} è garantita.
In alternativa si può sostituire $\bar{z}$ con un valido upper bound z_{UB}. Oppure si può sovrastimare $z_{LR}(\boldsymbol{\lambda}^k)$ come segue:

$$\theta^k = \beta^k \frac{0.01 \times z_{LR}(\boldsymbol{\lambda}^k)}{\|\mathbf{s}^k\|_2^2} \tag{86}$$

Il parametro β^k può essere diminuito (e.g., dimezzato) se dopo un certo numero di iterazioni $z_{LR}(\boldsymbol{\lambda}^k)$ non è migliorato.

Metodo del Subgradiente

- Altri approcci euristici per il calcolo dello spostamento θ^k sono:
  - *Constant step size*: $\theta^k = h$;
  - *Constant step length*: $\theta^k = h/\|\mathbf{s}^k\|_2$;
  - *Square summable but not summable*:

$$\sum_{k=1}^{\infty}(\theta^k)^2 < \infty \quad \text{and} \quad \sum_{k=1}^{\infty}\theta^k = \infty;$$

    Per esempio, $\alpha^k = a/(b+k)$, dove $a > 0$ e $b \geq 0$.

  - *Diminishing step size*:

$$\lim_{k\to\infty}\theta^k = 0 \quad \text{and} \quad \sum_{k=1}^{\infty}\theta^k = \infty.$$

    Per esempio, $\alpha^k = a/\sqrt{k}$, dove $a > 0$.

Metodo del Subgradiente

- Quando la penalità Lagrangiana λ_i è aggiornata si deve tenere conto di eventuali vincoli sul segno:
  - $\mathbf{a}_i\mathbf{x} = b_i$: la penalità λ_i può essere qualsiasi,

$$\lambda_i^{k+1} = \lambda_i^k + \theta^k s_i \tag{87}$$

  - $\mathbf{a}_i\mathbf{x} \geq b_i$: la penalità λ_i deve essere non negativa,

$$\lambda_i^{k+1} = \max\left\{0, \lambda_i^k + \theta^k s_i\right\} \tag{88}$$

  - $\mathbf{a}_i\mathbf{x} \leq b_i$: la penalità λ_i deve essere non positiva,

$$\lambda_i^{k+1} = \min\left\{0, \lambda_i^k + \theta^k s_i\right\} \tag{89}$$

Metodo del Subgradiente

Algoritmo del Subgradiente

Step 1. Sia $z_P = \min\{\mathbf{cx} : \mathbf{Ax} \geq \mathbf{b}, \mathbf{x} \in X\}$, dove $\mathbf{A} \in \mathbb{R}^{mn}$, $\mathbf{c} \in \mathbb{R}^n$ e $\mathbf{b} \in \mathbb{R}^m$. Poni $z_{LB} = -\infty$, $z_{UB} = +\infty$ e $\boldsymbol{\lambda} = \mathbf{0}$.

Step 2. Risolvi il problema Lagrangiano:

$$z_{LR}(\boldsymbol{\lambda}) = (\mathbf{c} - \boldsymbol{\lambda}\mathbf{A})\bar{\mathbf{x}} + \boldsymbol{\lambda}\mathbf{b} = \min\{(\mathbf{c} - \boldsymbol{\lambda}\mathbf{A})\mathbf{x} + \boldsymbol{\lambda}\mathbf{b} : \mathbf{x} \in X\}$$

e aggiorna il lower bound $z_{LB} = \max\{z_{LB}, z_{LR}(\boldsymbol{\lambda})\}$.

Step 3. Se $\bar{\mathbf{x}}$ è ammissibile $z_{UB} = \min\{z_{UB}, \mathbf{c}\bar{\mathbf{x}}\}$ e se ottima STOP.

Step 4. Aggiorna le penalità Lagrangiane:

$$\lambda_i = \max\{0, \lambda_i + \theta s_i\}, \quad i = 1, \ldots, m,$$

dove $s_i = b_i - \mathbf{a}_i\bar{\mathbf{x}}$, e vai allo Step 2.

Euristica Lagrangiana

Euristica Lagrangiana

Step 1. Calcola una soluzione euristica del problema P e inizializza il subgradiente;

Step 2. Calcola $z_{LR}(\boldsymbol{\lambda})$ ottenendo la soluzione $\mathbf{x}$;

Step 3. Verifica l'ammissibilità della soluzione $\mathbf{x}$;

Step 4. Costruisci una soluzione ammissibile per il problema P utilizzando $\mathbf{x}$ e/o $\boldsymbol{\lambda}$;

Step 5. Se si verificano le condizioni di arresto allora STOP;

Step 6. Aggiorna le penalità Lagrangiane $\boldsymbol{\lambda}$ e vai allo Step 2.

Rilassamento Lagrangiano: esempio

Generalized Assignment Problem (GAP)

- Il Generalized Assignment Problem (GAP) è il problema di *assegnare* un insieme di *server* I, a un insieme di *client* J, minimizzando i costi di assegnamento.
- Ogni server i ha una capacità massima Q_i.
- Ogni client j deve essere assegnato a esattamente un server.
- Se il server i è assegnato al client j si paga un costo c_{ij} e si consuma una quantità q_{ij} della capacità Q_i.
- Per costruire il modello matematico definiamo le variabili binarie x_{ij} che devono essere uguali a 1 se il server i è stato assegnato al client j, 0 altrimenti.

Rilassamento Lagrangiano: esempio

- Un modello per il problema GAP è:

$$z_{GAP} = \min \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij} \tag{90}$$

$$s.t. \sum_{i \in I} x_{ij} = 1, \qquad j \in J \tag{91}$$

$$\sum_{j \in J} q_{ij} x_{ij} \leq Q_i, \qquad i \in I \tag{92}$$

$$x_{ij} \in \{0,1\}, \qquad i \in I, j \in J \tag{93}$$

- Se vogliamo applicare il rilassamento Lagrangiano quali vincoli conviene rilassare?

- Proviamo prima a rilassare i vincoli (91) e poi i vincoli (92) e vediamo quali problemi Lagrangiani dobbiamo risolvere.

Rilassamento Lagrangiano: esempio

- Se rilassiamo i vincoli (91) in funzione obiettivo utilizzando le penalità Lagrangiane λ_j, il problema Langrangiano LR1 sarà:

$$z_{LR1} = \min \sum_{i \in I} \sum_{j \in J} (c_{ij} + \lambda_j) x_{ij} - \sum_{j \in J} \lambda_j \tag{94}$$

$$s.t. \sum_{j \in J} q_{ij} x_{ij} \leq Q_i, \qquad i \in I \tag{95}$$

$$x_{ij} \in \{0,1\}, \qquad i \in I, j \in J \tag{96}$$

- Siccome abbiamo rilassato delle equazioni, le penalità Lagrangiane non hanno vincoli di segno.
- Come si può notare, dobbiamo risolvere un insieme di Knapsack Problem indipendenti, uno per ogni server $i \in I$.
- In questo caso si può dimostrare che z_{LR1} può essere strettamente maggiore del rilassamento continuo del GAP.

Rilassamento Lagrangiano: esempio

- Se rilassiamo i vincoli (92) in funzione obiettivo utlizzando le penalità Lagrangiane $\lambda_i \geq 0$, il problema Langrangiano LR1 sarà un assegnamento:

$$z_{LR2} = \min \sum_{i \in I} \sum_{j \in J} (c_{ij} + \lambda_i q_{ij}) x_{ij} - \sum_{i \in I} \lambda_i Q_i \tag{97}$$

$$s.t. \sum_{i \in I} x_{ij} = 1, \qquad j \in J \tag{98}$$

$$x_{ij} \in \{0,1\}, \qquad i \in I, j \in J \tag{99}$$

- In questo caso si può dimostrare che z_{LR2} NON può essere strettamente maggiore del rilassamento continuo del GAP.

- Quindi, il rilassamento LR1 ha il potenziale per fornire un lower bound migliore del rilassamento LR2.

Introduzione

- I metodi di generazione di colonne risolvono il problema senza considerare esplicitamente tutte le variabili.
- Si definisce un *core* iniziale che considera solo un sottoinsieme delle variabili, dopodiché si aggiungono dinamicamente le variabili mancanti "necessarie" durante il processo di soluzione.
- Si consideri il seguente problema di programmazione lineare:

$$\begin{aligned} z_P = \quad & \min \quad cx \\ & s.t. \quad Ax = b \\ & \qquad x \geq 0 \end{aligned}$$

- Il problema duale è il seguente:

$$\begin{aligned} z_D = \quad & \max \quad wb \\ & s.t. \quad wA \leq c \end{aligned}$$

Introduzione

- Il simplesso primale esegue iterativamente le seguenti operazioni:
  - Selezione di una colonna s di A (*variabile entrante*);
  - Selezione di una riga r di A (*variabile uscente*);
  - Esecuzione di un'operazione di pivoting sull'elemento pivot a_{rs}.
- Sia B una base della matrice dei vincoli A.
- Possiamo definire le seguenti entità:

$$w = c_B B^{-1} : \text{ variabili duali associate alla base } B;$$
$$\bar{b} = B^{-1} b : \text{ valore delle variabili in base;}$$
$$a_j : \text{ colonna } j \text{ della matrice } A;$$
$$\bar{c}_j = w a_j - c_j : \text{ costo ridotto associato alla variabile } x_j;$$
$$\bar{y}_j = B^{-1} a_j : \text{ colonna } j \text{ della matrice } B^{-1} A.$$

Introduzione

- Le operazioni del metodo del simplesso primale possono essere riassunte come segue:

	Simplesso Primale
Test di ottimalità	$\bar{c}_j = wa_j - c_j \leq 0, j = 1, \ldots, m$
Variabile entrante	$s = argmax\{\bar{c}_j : j = 1, \ldots, n\}$
Variabile uscente	$r = argmin\left\{\frac{\bar{b}_i}{\bar{y}_{is}} : \bar{y}_{is} > 0, i = 1, \ldots, m\right\}$
Elemento di pivoting	$\bar{y}_{rs}$

Introduzione

- Simplesso in formato tableau:

z	$x_1 \quad \dots \quad x_m$	$x_{m+1} \quad \dots \quad x_n$	
1	$-c_B$	$-c_N$	0
0 $\vdots$ 0	B	N	b

$$z - c_B x_B - c_N x_N = 0$$

$$B x_B + N x_N = b$$

z	$x_1 \quad \dots \quad x_m$	$x_{m+1} \quad \dots \quad x_n$	
1	$0 \quad \dots \quad 0$	$\bar{c}_N = c_B B^{-1} N - c_N$	$\bar{c}_0 = c_B B^{-1} b$
0 $\vdots$ 0	I	$\bar{Y} = B^{-1} N$	$\bar{b} = B^{-1} b$

Algoritmo Column Generation

- Si consideri il seguente problema di programmazione lineare:

$$z(P) = \min \sum_{j=1}^{n} c_j x_j \tag{100}$$

$$s.t. \sum_{j=1}^{n} a_{ij} x_j = b_i, \quad i = 1, \ldots, m \tag{101}$$

$$x_j \geq 0, \qquad j = 1, \ldots, n \tag{102}$$

- Data una base B di A, la corrispondente soluzione base può essere migliorata se esiste una colonna h tale che $\bar{c}_h > 0$, dove:

$$\bar{c}_h = \max \left\{ \bar{c}_j = w a_j - c_j : j = 1, \ldots, n \right\} > 0$$

Algoritmo Column Generation

- Ad ogni iterazione è necessario determinare se esiste una colonna a_h di costo c_h tale che $\bar{c}_h = wa_h - c_h > 0$ risolvendo il *problema di pricing*:
$$\bar{c}_h = \max\left\{\bar{c}_j = wa_j - c_j : j = 1, \ldots, n\right\} > 0 \qquad (103)$$
- L'applicazione del simplesso può risultare proibitiva se il numero di colonne è elevato.
- In alcuni casi la struttura del problema di pricing (103) è tale da consentire la soluzione senza considerare esplicitamente tutte le variabili (colonne) del problema originario.
- Il metodo prende il nome di *generazione di colonne* perché nella soluzione del problema di pricing (103) le variabili (colonne) del problema vengono generate solo quando servono.

Algoritmo Column Generation

Algoritmo Column Generation

Step 1. Definisci una base ammissibile iniziale B;

Step 2. Calcola la soluzione base corrispondente a B:
$$x = (x_B, 0) = (B^{-1}b, 0);$$
e la corrispondente soluzione duale:
$$w = c_B B^{-1};$$

Step 3. Risolvi il problema di pricing, generando la variabile h di costo ridotto $\bar{c}_h = wa_h - c_h$ massimo;

Step 4. Se il costo ridotto $\bar{c}_h \leq 0$ allora STOP: la soluzione x è ottima e non è necessario generare altre colonne; altrimenti calcola la nuova base B e vai allo Step 2.

Algoritmo Column Generation

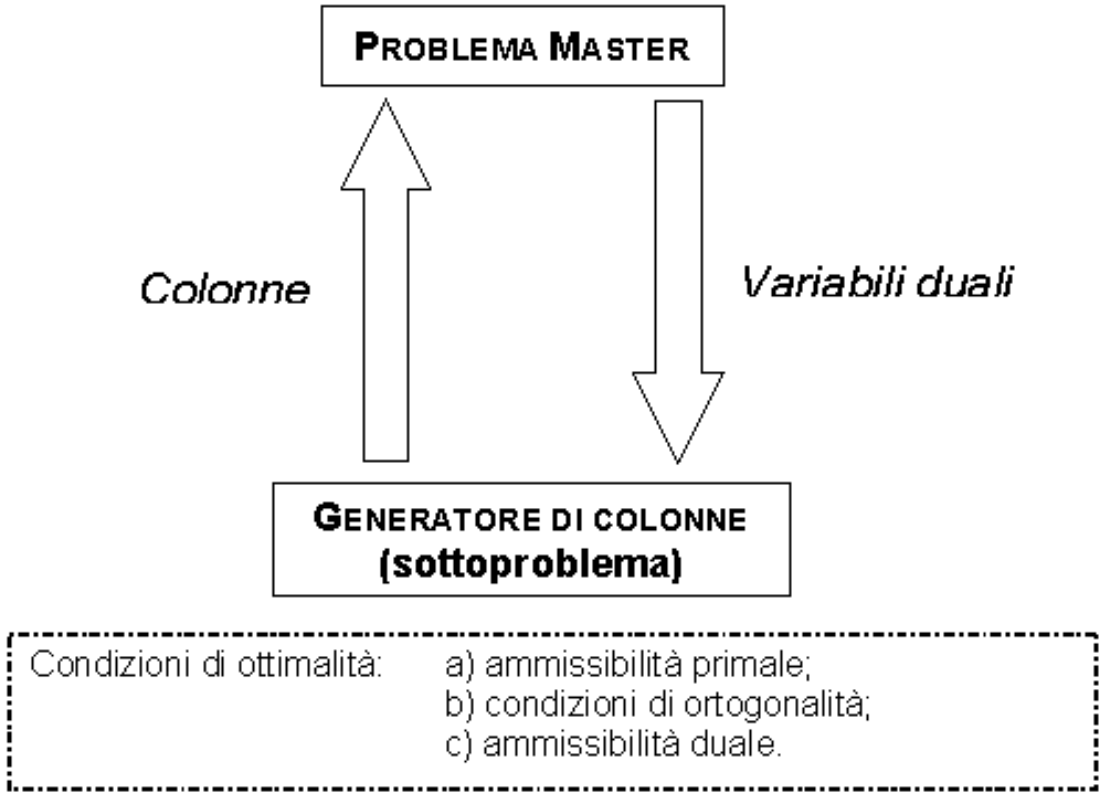


Figura: Schema di funzionamento dell'algoritmo Column Generation.

Cutting Stock Problem, Gilmore e Gomory (1960)

Descrizione del Problema

- Sia I un insieme di n oggetti ognuno dei quali di lunghezza pari a ℓ_i, $i = 1, \ldots, n$.
- Sia n_i il numero di pezzi che si devono produrre per ogni oggetto i, $i = 1, \ldots, n$.
- Sia L la lunghezza delle barre (*master*) dalle quali tagliare gli oggetti necessari.
- Una *configurazione di taglio* (*pattern*) k è una possibile configurazione di taglio di un sottoinsieme di oggetti $I_k \subseteq I$ da una barra master (i.e., $\sum_{i \in I_k} a_{ik} \ell_i \leq L$, dove a_{ik} è il numero di oggetti i nel pattern k).

Cutting Stock Problem: esempio

Esempio

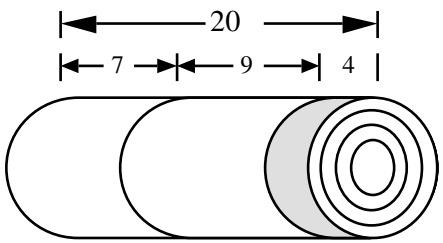
Si consideri il seguente esempio, in cui si devono produrre i seguenti 3 oggetti:

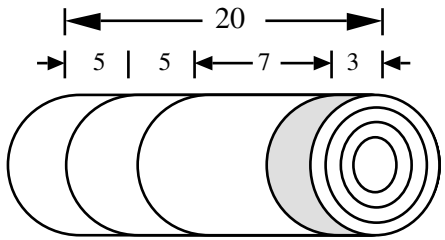
Oggetto	ℓ_i (m)	n_i
A	5	150
B	7	200
C	9	300

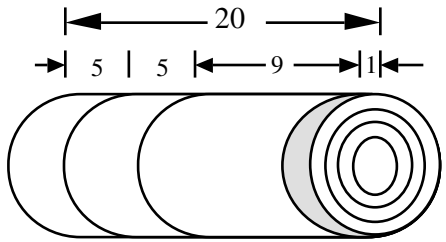
Le barre da cui tagliare gli oggetti richiesti ha lunghezza $L = 20$m

Cutting Stock Problem: esempio

Alcune configurazioni di taglio possibili sono le seguenti:







Altre configurazioni:

	Pattern 1	Pattern 2	Pattern 3	Pattern 4	Pattern 5
A (5 m)	0	2	2	4	1
B (7 m)	1	1	0	0	2
C (9 m)	1	0	1	0	0

Cutting Stock Problem: modello

Formulazione Matematica del Problema

- Sia J l'insieme degli indici di tutte le possibili configurazioni di taglio.
- Sia inoltre a_{ij} il numero di volte in cui l'oggetto i è tagliato nella configurazione j, $i \in I$, $j \in J$.
- L'obiettivo del problema è quello di tagliare dalle barre tutti i pezzi necessari minimizzando il numero di barre necessarie.
- Sia y_j una variabile intera non negativa che rappresenta il numero di volte in cui la configurazione $j \in J$ è usata per tagliare dei pezzi.

Cutting Stock Problem: modello

- La formulazione matematica del problema è la seguente:

$$z_P = \min \sum_{j \in J} y_j \tag{104}$$

$$s.t. \sum_{j \in J} a_{ij} y_j \geq n_i, \quad i \in I \tag{105}$$

$$y_j \geq 0 \text{ intero}, \quad j \in J \tag{106}$$

- Si noti che la cardinalità dell'insieme J può essere grande.
- Il rilassamento lineare (LP) del problema P lo si ottiene sostituendo i vincoli (106) con i seguenti:

$$y_j \geq 0, \quad j \in J \tag{107}$$

Cutting Stock Problem: Column Generation Algorithm

- Sia LP' il problema ottenuto da LP sostituendo l'insieme J delle configurazioni con il sottoinsieme $J' \subset J$ (si supponga che J' sia tale da garantire che LP' ammette una soluzione ammissibile).
- Si risolva il problema LP' e siano w_i, $i \in I$, le variabili duali associate ai vincoli di LP'.
- Il costo ridotto di ogni configurazione $j \in J$ è dato da:

$$\bar{c}_j = \sum_{i \in I} a_{ij} w_i - 1 \tag{108}$$

- Se esiste almeno una configurazione non contenuta in J' di costo ridotto positivo, i.e.:

$$\max_{j \in J \setminus J'} [\bar{c}_j] > 0 \tag{109}$$

allora la soluzione base corrente non è ottima per LP.

Cutting Stock Problem: Column Generation Algorithm

- Siccome i costi ridotti $\bar{c}_j$, $j \in J'$, sono non positivi:

$$\max_{j \in J \setminus J'} \left\{ \sum_{i \in I} a_{ij} w_i - 1 \right\} = \max_{j \in J} \left\{ \sum_{i \in I} a_{ij} w_i - 1 \right\} \tag{110}$$

che equivale a risolvere il seguente problema:

$$z_{SP} = \max \sum_{i \in I} w_i z_i - 1 \tag{111}$$

$$s.t. \sum_{i \in I} \ell_i z_i \leq L \tag{112}$$

$$z_i \geq 0 \text{ intero}, \quad i \in I \tag{113}$$

dove z_i, $i \in I$, è una variabile decisionale che rappresenta il numero di volte in cui l'oggetto i è tagliato nella *nuova configurazione*.

Cutting Stock Problem: Column Generation Algorithm

Algoritmo Column Generation

Step 1. Si generi un insieme iniziale J' di configurazioni di taglio in modo che il problema LP' abbia una soluzione ammissibile;

Step 2. Si risolva il problema LP' definito sull'insieme di configurazioni J'. Sia y^* la soluzione ottima primale e sia w^* la corrispondente soluzione duale;

Step 3. Si risolva il seguente problema di knapsack intero:

$$z_{SP} = \max \sum_{i \in I} w_i^* z_i \tag{114}$$

$$s.t. \sum_{i \in I} \ell_i z_i \leq L \tag{115}$$

$$z_i \geq 0 \text{ intero}, \quad i \in I \tag{116}$$

Sia $\mathbf{z}^*$ la soluzione ottima si SP;

Cutting Stock Problem: Column Generation Algorithm

Step 4. Se $\sum_{i \in I} w_i^* z_i^* \leq 1$ allora STOP: $\mathbf{x}^*$ è la soluzione ottima; altrimenti aggiungi a J' la colonna (nuova configurazione) data dalla soluzione $\mathbf{z}^*$ e vai allo Step 2.

NOTA:
L'Algoritmo Column Generation risolve all'ottimo il rilassamento lineare del problema originario. Quindi, la soluzione può essere frazionaria.

Considerazioni

- Nel caso di problemi di Programmazione Lineare, l'algoritmo column generation determina la soluzione ottima del problema.
- Nel caso di problemi di Programmazione Intera, come il Cutting Stock Problem, l'algoritmo column generation applicato al rilassamento lineare del problema può terminare con una soluzione che risulta frazionaria.
- Usualmente i problemi classificati come Cutting Stock Problem richiedono di tagliare poche tipologie di oggetti in grandi quantità. In questo caso si può ottenere una soluzione euristica di ottima qualità applicando un semplice "rounding".
- Per i problemi che richiedono di tagliare molte tipologie di oggetti in piccole quantità (*Bin Packing Problem*) il "rounding" può produrre soluzioni di qualità non adeguata.

Considerazioni

- In questo caso è necessario applicare un algoritmo di tipo branch and bound per determinare la soluzione ottima intera. Per il calcolo del lower bound si può sempre utilizzare una tecnica *column generation*.
- Questi metodi prendono il nome di metodi *Branch and Price*.

Generazione dei Vincoli

- Abbiamo visto che per migliorare il rilassamento continuo di un problema di programmazione lineare intera possiamo aggiungere delle disuguaglianze valide.
- Una disuguaglianza valida è un vincolo che elimina delle soluzioni frazionarie senza eliminare delle soluzioni intere, quindi è un vincolo ridondante nella formulazione originaria intera del problema.
- Ora vediamo che per alcuni modelli il numero di vincoli necessari (i.e., senza i quali il modello non è valido) può essere molto elevato, anche esponenziale.
- Consideriamo il Travelling Salesman Problem (TSP, Problema del Commesso Viaggiatore).

Travelling Salesman Problem

- Consideriamo il problema di un autista che deve fare le consegne visitanto una e una sola volta ciascun cliente.
- Possiamo rappresentare il problema utilizzando un grafo direzionato $G = (V, A)$, dove V è l'insieme dei vertici e A è l'insieme degli archi.
- I vertici $i \in V$ rappresentano i clienti da visitare e il deposito da cui partire e rientrare.
- Gli archi $(i, j) \in A$ rappresentano il tragitto dal vertice i al vertice j, a cui è associato un costo c_{ij} (e.g., i chilometri da percorrere, il tempo di guida, etc.). Per semplicità consideriamo il grafo completo.
- Si noti che in questo caso $c_{ij} \neq c_{ij}$, per cui parleremo di Asymmetric Travelling Salesman Problem (ATSP, Problema del Commesso Viaggiatore Asimmetrico).

Travelling Salesman Problem

- Siano x_{ij} delle variabili binarie uguali a 1 se l'arco (i,j) è nella soluzione ottima, 0 altrimenti.
- Un modello molto noto per il TSP asimmetrico è il seguente:

$$z_{TSP} = \min \sum_{i \in V} \sum_{j \in V} c_{ij} x_{ij} \tag{117}$$

$$s.t. \sum_{j \in V} x_{ij} = 1, \qquad i \in V \tag{118}$$

$$\sum_{j \in V} x_{ji} = 1, \qquad i \in V \tag{119}$$

$$\sum_{i \in S} \sum_{j \in \bar{S}} x_{ij} \geq 1, \quad \forall S \subset V, \bar{S} = V \setminus S \tag{120}$$

$$x_{ij} \in \{0,1\}, \qquad (i,j) \in A \tag{121}$$

Travelling Salesman Problem

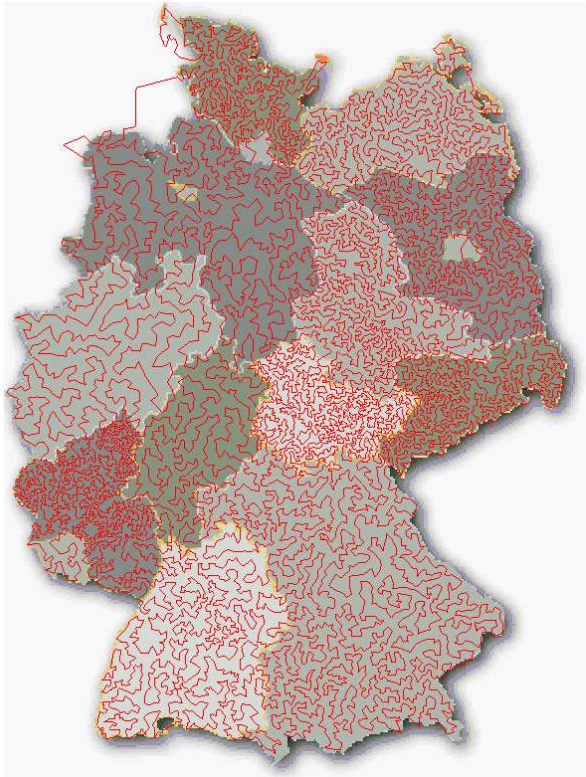
- I vincoli (120) sono detti *subtour elimination*, perché eliminano i sottocicli. Possono essere definiti anche in modo diverso.
- L'insieme degli archi $A(S,\bar{S}) = \{(i,j) \in A : i \in S, j \in \bar{S}\}$ è detto *taglio* determinato dagli insiemi S e $\bar{S}$.
- Siccome i vincoli (120) sono in un numero esponenziale, invece di generarli tutti, si può risolvere il problema dell'assegnamento ottenuto ignorando questi vincoli.
- Una volta risolto il problema dell'assegnamento, basta identificare se esiste un sottociclo. I vertici visitati dal sottociclo rappresentano un insieme S il cui corrispondente vincolo (120) è violato.
- Si aggiunge il vincolo e si riottimizza il problema, che ora non è più un semplice assegnamento e il suo rilassamento lineare può avere soluzioni frazionarie.

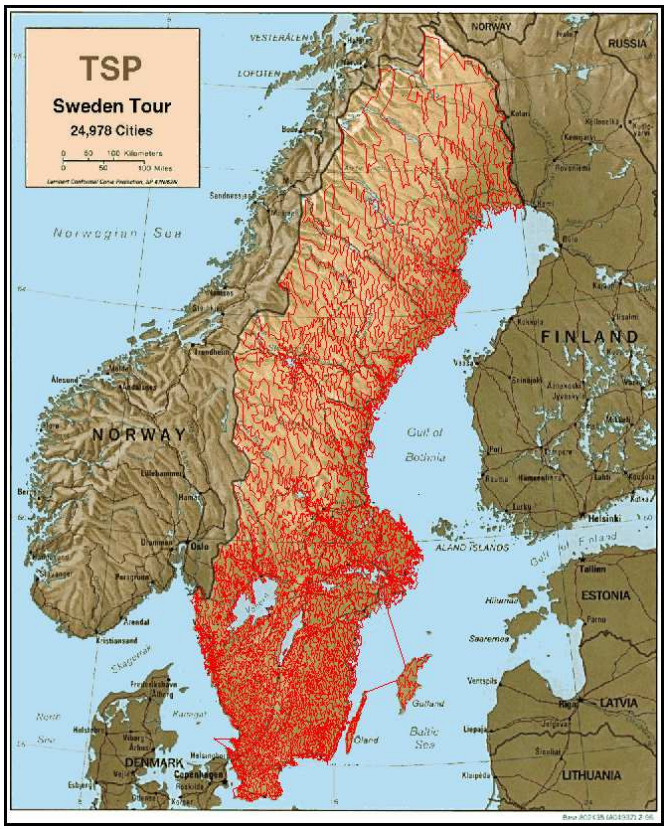
Travelling Salesman Problem

- Anche se le soluzioni sono frazionarie, è comunque possibile determinare se c'è un vicolo (120) violato e identificare il corrispondente insieme S.
- Purtroppo, anche per trovare l'ottimo del rilassamento lineare del modello (117)-(121) può essere necessario generare molti vincoli.
- Inoltre, la soluzione del rilassamneto continuo può essere molto frazionaria e può avere un valore molto distante dal valore della soluzione ottima intera.
- Per cui, è necessario fare ricorso anche alle disuguglianze valide e di un metodo branch and bound, che in questo caso chiameremo branch and cut.

Travelling Salesman Problem

- Gli strumenti matematici sviluppati hanno permesso la soluzione di TSP di enormi dimensioni:





Travelling Salesman Problem

Il World TSP Tour trovato da Keld Helsgaun nel dicembre 2003. Un lower bound fornita dal codice Concorde TSP ha dimostrato che la soluzione dista al massimo lo 0.076% tour ottimo che attraversa le 1,904,711 città.