Ricerca Operativa

G. Liuzzi[1]

Venerdí 6 Dicembre 2019

[1] Istituto di Analisi dei Sistemi ed Informatica IASI - CNR

Pre-appello

http://goo.gl/forms/swAieacbrl

Richiami

Il Metodo del Piano di Taglio di Gomory

Richiami

Il Metodo del Piano di Taglio di Gomory

(a) determina la soluzione x_i^* del rilassamento dato dalla formulazione P_i

Richiami

Il Metodo del Piano di Taglio di Gomory

(a) determina la soluzione x_i^* del rilassamento dato dalla formulazione P_i

(b) se $x_i^* \notin \mathbb{Z}$, genera uno o più tagli frazionari (di Gomory, mediante la procedura di Chvátal-Gomory)

Richiami

Il Metodo del Piano di Taglio di Gomory

(a) determina la soluzione x_i^* del rilassamento dato dalla formulazione P_i

(b) se $x_i^* \notin \mathbb{Z}$, genera uno o più tagli frazionari (di Gomory, mediante la procedura di Chvátal-Gomory)

(c) definisce P_{i+1} aggiungendo a P_i tutti i tagli generati al passo (b). Torna al passo (a)

Indichiamo con $L_i^* = c^\top x_i^*$

Richiami

Il Metodo del Piano di Taglio di Gomory

(a) determina la soluzione x_i^* del rilassamento dato dalla formulazione P_i

(b) se $x_i^* \notin \mathbb{Z}$, genera uno o più tagli frazionari (di Gomory, mediante la procedura di Chvátal-Gomory)

(c) definisce P_{i+1} aggiungendo a P_i tutti i tagli generati al passo (b). Torna al passo (a)

Indichiamo con $L_i^* = c^\top x_i^*$

Risulta, naturalmente: $L_0^* < L_1^* < L_2^* < \cdots < L_t^*$ ove $P_t = P_S$

Tailing-off

Problema:

Tailing-off

Problema: il valore dell'intero t potrebbe essere elevatissimo

In pratica:

Tailing-off

Problema: il valore dell'intero t potrebbe essere elevatissimo

In pratica: si manifesta il fenomeno del *tailing-off* cioè

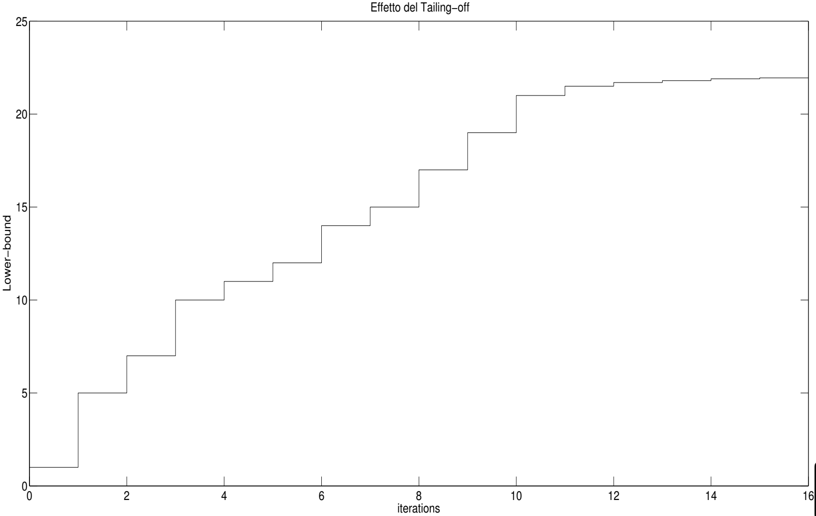
Tailing-off

Problema: il valore dell'intero t potrebbe essere elevatissimo

In pratica: si manifesta il fenomeno del *tailing-off* cioè

- nelle prime iterazioni, i lower-bound L_i^* migliorano (crescono) sensibilmente
- al crescere delle iterazioni, il miglioramento diventa sempre meno evidente

Tailing-off


Effetto del Tailing−off
Lower−bound
iterations
0
5
10
15
20
25
0
2
4
6
8
10
12
14
16

Introduzione

Problema: di PLI (c, S) cioè $\min\{c^\top x : \ x \in S\}$, S insieme finito di elementi

Introduzione

Problema: di PLI (c, S) cioè $\min\{c^\top x : \; x \in S\}$, S insieme finito di elementi

Enumerazione Totale: calcolare la f.ob. in ciascuna soluzione $x \in S$, scegliere la migliore

Introduzione

Problema: di PLI (c, S) cioè $\min\{c^\top x : \ x \in S\}$, S insieme finito di elementi

Enumerazione Totale: calcolare la f.ob. in ciascuna soluzione $x \in S$, scegliere la migliore

Più realisticamente possiamo pensare di effettuare una **Esplorazione Implicita** di tutto S

Passo Fondamentale

Indichiamo con $S_0 = S$ l'insieme delle soluzioni ammissibili **intere**

Passo Fondamentale

Indichiamo con $S_0 = S$ l'insieme delle soluzioni ammissibili **intere**

Decomporre S_0 in un famiglia di sottoinsiemi $(S_1, S_2, \ldots, S_r)$ con $r \geq 2$, tali che

- $S_i \cap S_j = \emptyset$, per ogni $1 \leq i < j \leq r$;
- $\bigcup_{i=1}^{r} S_i = S_0$;

ovvero individuare una *partizione* di S_0.

Passo Fondamentale

Indichiamo con $S_0 = S$ l'insieme delle soluzioni ammissibili **intere**

Decomporre S_0 in un famiglia di sottoinsiemi $(S_1, S_2, \dots, S_r)$ con $r \geq 2$, tali che

- $S_i \cap S_j = \emptyset$, per ogni $1 \leq i < j \leq r$;
- $\bigcup_{i=1}^{r} S_i = S_0$;

ovvero individuare una *partizione* di S_0.

Sia x_i^* la soluzione ottima del sottoproblema S_i e $z_i^* = c^\top x_i^*$.

Evidentemente, $x^* = x_\ell^*$, con ℓ tale che

$$c^\top x_\ell^* = \min_{i=1,\dots,r} \{c^\top x_i^*\}$$

Passo Fondamentale - *Divide et Impera*

Dato un sottoproblema S_i della partizione può succedere che:

- risolvere S_i (cioè calcolare x_i^*) sia **facile** oppure
- risolvere S_i è altrettanto **difficile** che risolvere S_0. In questo caso:

Passo Fondamentale - *Divide et Impera*

Dato un sottoproblema S_i della partizione può succedere che:

- risolvere S_i (cioè calcolare x_i^*) sia **facile** oppure
- risolvere S_i è altrettanto **difficile** che risolvere S_0. In questo caso:
    - si può determinare un (lower) bound L_i per la miglior sol. intera di S_i;
    - si partiziona ulteriormente S_i;

Passo Fondamentale - *Divide et Impera*

Dato un sottoproblema S_i della partizione può succedere che:

- risolvere S_i (cioè calcolare x_i^*) sia **facile** oppure
- risolvere S_i è altrettanto **difficile** che risolvere S_0. In questo caso:
  - si può determinare un (lower) bound L_i per la miglior sol. intera di S_i;
  - si partiziona ulteriormente S_i;

La procedura appena descritta risulterà computazionalmente efficiente solo se:

- il numero di sottoproblemi generati si mantiene **estremamente limitato**
- la strategia di soluzione di un sottoproblema è **efficace** ed evita il suo ulteriore partizionamento

Ingredienti Fondamentali

- **Strategia di Soluzione** (Bounding): come risolvere il sottoproblema oppure determinare un *lower-bound* della soluzione

$$L_i \leq z_i^*$$

- **Strategia di Separazione** (Branching): come partizionare l'insieme delle soluzioni ammissibili di un sottoproblema

[2] ($\bar{z} = +\infty$ se non si conosce alcuna sol. intera)

Ingredienti Fondamentali

- **Strategia di Soluzione** (Bounding): come risolvere il sottoproblema oppure determinare un *lower-bound* della soluzione

$$L_i \leq z_i^*$$

- **Strategia di Separazione** (Branching): come partizionare l'insieme delle soluzioni ammissibili di un sottoproblema

Sia $\bar{z}$ il valore corrispondente alla migliore soluzione intera nota ovvero l'**ottimo corrente** o **soluzione incombente**[2], allora

[2] ($\bar{z} = +\infty$ se non si conosce alcuna sol. intera)

Ingredienti Fondamentali

- **Strategia di Soluzione** (Bounding): come risolvere il sottoproblema oppure determinare un *lower-bound* della soluzione

$$L_i \leq z_i^*$$

- **Strategia di Separazione** (Branching): come partizionare l'insieme delle soluzioni ammissibili di un sottoproblema

Sia $\bar{z}$ il valore corrispondente alla migliore soluzione intera nota ovvero l'**ottimo corrente** o **soluzione incombente**[2], allora

Se $\bar{z} \leq L_i \leq z_i^*$, si può concludere che non esiste in S_i un vettore intero che abbia valore migliore di $\bar{z}$ e quindi si può sospendere l'esame del sottoproblema S_i.

[2]($\bar{z} = +\infty$ se non si conosce alcuna sol. intera)

Rilassamento Lineare

Sia S_i il sottoproblema per il quale vogliamo calcolare un bound L_i e sia

$$P_i = P(A^i, b^i) = \{x \in \Re^n;\ A^i x \geq b^i\}$$

una possibile formulazione di (c, S_i).

Il sottoproblema che vorremmo risolvere può quindi essere scritto come

$$\min\{c^\top x :\ A^i x \geq b^i,\ x \in \mathbb{Z}^n\}$$

Rilassamento Lineare

Sia S_i il sottoproblema per il quale vogliamo calcolare un bound L_i e sia

$$P_i = P(A^i, b^i) = \{x \in \Re^n;\ A^i x \geq b^i\}$$

una possibile formulazione di (c, S_i).

Il sottoproblema che vorremmo risolvere può quindi essere scritto come

$$\min\{c^\top x :\ A^i x \geq b^i,\ x \in \mathbb{Z}^n\}$$

Definiamo

$$L_i = \min\{c^\top x :\ A^i x \geq b^i\}$$

valore ottimo del **rilassamento lineare** del sottoproblema S_i

Nota: Se la sol. del problema rilassato è intera, allora essa è anche sol. ottima di S_i

Rilassamento della Formulazione

Sia S_i il sottoproblema per il quale vogliamo calcolare un bound L_i e sia

$$P_i = P(A^i, b^i) = \{x \in \Re^n;\ A^i x \geq b^i\}$$

una possibile formulazione di (c, S_i).

Rilassamento della Formulazione

Sia S_i il sottoproblema per il quale vogliamo calcolare un bound L_i e sia

$$P_i = P(A^i, b^i) = \{x \in \Re^n;\ A^i x \geq b^i\}$$

una possibile formulazione di (c, S_i).

Supponiamo che i vincoli $A^i x \geq b^i$ possano essere partizionati in $A_1^i x \geq b_1^i$ e $A_2^i x \geq b_2^i$ con la proprietà che

$$\min\{c^\top x :\ A_1^i x \geq b_1^i,\ x \in \mathbb{Z}^n\} \qquad (1)$$

è di **facile soluzione**.

Rilassamento della Formulazione

Sia S_i il sottoproblema per il quale vogliamo calcolare un bound L_i e sia

$$P_i = P(A^i, b^i) = \{x \in \Re^n;\ A^i x \geq b^i\}$$

una possibile formulazione di (c, S_i).

Supponiamo che i vincoli $A^i x \geq b^i$ possano essere partizionati in $A_1^i x \geq b_1^i$ e $A_2^i x \geq b_2^i$ con la proprietà che

$$\min\{c^\top x :\ A_1^i x \geq b_1^i,\ x \in \mathbb{Z}^n\} \tag{1}$$

è di **facile soluzione**.

Definiamo: L_i valore ottimo di (1)

Attenzione: La soluzione ottima di (1), anche se intera, è ottima anche per S_i solo se soddisfa anche i vincoli $A_2^i x \geq b_2^i$

Qualità del Bound

La **qualità** di un (lower) bound si misura in due modi:

- **Accuratezza** nell'approssimare la soluzione ottima
- **Facilità** (e quindi velocità) con la quale può essere calcolato il bound

Qualità del Bound

La **qualità** di un (lower) bound si misura in due modi:

- **Accuratezza** nell'approssimare la soluzione ottima
- **Facilità** (e quindi velocità) con la quale può essere calcolato il bound

Entrambi gli aspetti sono <u>molto critici</u>.

- Il calcolo di lower bound accurati (cioè elevati) può evitare di generare un eccessivo numero di sottoproblemi
- La soluzione di un problema intero (difficile) richiede usualmente il calcolo di un elevatissimo numero di lower bound. Pertanto i tempi di calcolo del singolo lower bound diventano critici per la soluzione del problema intero

Separazione Binaria

Sia $P_i = P(A^i, b^i) = \{x \in \Re^n : \; A^i x \geq b^i\}$ una possibile formulazione di S_i. Sia L_i il bound che abbiamo calcolato (nella fase di bounding) sul valore ottimo di (c, S_i), ovvero $L_i \leq c^\top x_i^*$.

Supponiamo che

- $L_i < \tilde{z}$ (valore della sol. intera incombente)
- $x_i^* \notin \mathbb{Z}^n$

allora il sottoproblema (c, S_i) potrebbe contenere una soluzione intera migliore di quella icombente $\tilde{x}$.

Separazione Binaria

Sia $P_i = P(A^i, b^i) = \{x \in \Re^n : \; A^i x \geq b^i\}$ una possibile formulazione di S_i. Sia L_i il bound che abbiamo calcolato (nella fase di bounding) sul valore ottimo di (c, S_i), ovvero $L_i \leq c^\top x_i^*$.

Supponiamo che

- $L_i < \tilde{z}$ (valore della sol. intera incombente)
- $x_i^* \notin \mathbb{Z}^n$

allora il sottoproblema (c, S_i) potrebbe contenere una soluzione intera migliore di quella icombente $\tilde{x}$.

(c, S_i) è quindi un problema **candidato** a contenere l'ottimo e va quindi ulteriormente esplorato, **partizionato**.

Separazione Binaria

Si divide (c, S_i) in due sottoproblemi, (c, S_{i1}) e (c, S_{i2}), tali che

- P_{i1} e P_{i2} sono **formulazioni** di S_{i1} e S_{i2}, rispettivamente

Separazione Binaria

Si divide (c, S_i) in due sottoproblemi, (c, S_{i1}) e (c, S_{i2}), tali che

- P_{i1} e P_{i2} sono **formulazioni** di S_{i1} e S_{i2}, rispettivamente
- $S_{i1} = P_{i1} \cap S_i$ e $S_{i2} = P_{i2} \cap S_i$ costituiscono una **partizione** di S_i

Separazione Binaria

Si divide (c, S_i) in due sottoproblemi, (c, S_{i1}) e (c, S_{i2}), tali che

- P_{i1} e P_{i2} sono **formulazioni** di S_{i1} e S_{i2}, rispettivamente
- $S_{i1} = P_{i1} \cap S_i$ e $S_{i2} = P_{i2} \cap S_i$ costituiscono una **partizione** di S_i
- ogni punto $\bar{x} \in S_i$ appartiene a P_{i1} oppure a P_{i2} ma **non ad entrambi**

Esempio di Separazione Binaria

Esempio: Consideriamo il caso di un sottoproblema S_i per cui il lower bound L_i è stato ottenuto in corrispondenza di una soluzione $\bar{x} \in \Re^n$ **non necessariamente** in P_i (p.es. nel caso di rilassamento della formulazione).

Sia $\bar{x}_h$ una componente di $\bar{x}$ (non necessariamente frazionaria) Consideriamo i poliedri:

$$\begin{aligned} P_{i1} &= \{x \in P_i : x_h \leq \alpha\} \\ P_{i2} &= \{x \in P_i : x_h \geq \alpha + 1\} \end{aligned}$$

Corrispondentemente, stiamo considerando la partizione di S_i in S_{i1} e S_{i2} con formulazioni P_{i1} e P_{i2}

Esempio di Separazione Binaria

Osservazione: In base a quanto visto precedentemente (Rilassamento della formulazione), il fatto che la soluzione approssimata $\bar{x}$ sia intera **non implica** che $\bar{x}$ sia soluzione di S_i.

Esempio di Separazione Binaria

Osservazione: In base a quanto visto precedentemente (Rilassamento della formulazione), il fatto che la soluzione approssimata $\bar{x}$ sia intera **non implica** che $\bar{x}$ sia soluzione di S_i.

Infatti, il problema rilassato sul quale viene determinato il bound L_i potrebbe contenere soluzioni **intere non ammissibili** per il problema S_i

Descrizione del Branch & Bound – passo 1

Sia dato un problema (c, S) di *minimizzazione*

Descrizione del Branch & Bound – passo 1

Sia dato un problema (c, S) di *minimizzazione*

1. $\boxed{\text{Inizializzazione}}$

Descrizione del Branch & Bound – passo 1

Sia dato un problema (c, S) di *minimizzazione*

1. $\boxed{\text{Inizializzazione}}$

   - Si pone $S_0 = S$
   - Si pone $\mathcal{Q} = \{(c, S_0)\}$ (lista dei problemi "candidati" o "aperti")
   - si calcola un **upper bound** $\tilde{z}$ per il problema (c, S) cioè il valore che si ottiene in corrispondenza ad una soluzione $\tilde{x} \in S$ (oppure si pone $\tilde{z} = +\infty$).

Descrizione del Branch & Bound – passo k

Passo k dell'algoritmo di B& B

Sia $\mathcal{Q}$ la lista di sottoproblemi *candidati*

Descrizione del Branch & Bound – passo k

Passo k dell'algoritmo di B& B

Sia $\mathcal{Q}$ la lista di sottoproblemi *candidati*

- Se $\mathcal{Q} = \emptyset$, STOP. $\tilde{x}$ è ottimo globale di (c, S). Altrimenti sia $\mathcal{Q} = \{(c, S_1), \ldots, (c, S_q)\}$ la lista di sottoproblemi candidati.

Descrizione del Branch & Bound – passo k

Passo k dell'algoritmo di B& B

Sia $\mathcal{Q}$ la lista di sottoproblemi *candidati*

- Se $\mathcal{Q} = \emptyset$, STOP. $\tilde{x}$ è ottimo globale di (c, S). Altrimenti sia $\mathcal{Q} = \{(c, S_1), \dots, (c, S_q)\}$ la lista di sottoproblemi candidati.
- **Selection** (Scelta di un sottoproblema candidato)
  Si seleziona un problema $(c, S_i) \in \mathcal{Q}$ e lo si elimina da $\mathcal{Q}$,
  $\mathcal{Q} := \mathcal{Q} \setminus \{(c, S_i)\}$

Descrizione del Branch & Bound – passo k

Passo k dell'algoritmo di B& B

Sia $\mathcal{Q}$ la lista di sottoproblemi *candidati*

- Se $\mathcal{Q} = \emptyset$, STOP. $\tilde{x}$ è ottimo globale di (c, S). Altrimenti sia $\mathcal{Q} = \{(c, S_1), \ldots, (c, S_q)\}$ la lista di sottoproblemi candidati.
- **Selection** (Scelta di un sottoproblema candidato)
  Si seleziona un problema $(c, S_i) \in \mathcal{Q}$ e lo si elimina da $\mathcal{Q}$,
  $\mathcal{Q} := \mathcal{Q} \setminus \{(c, S_i)\}$
- **Bounding**
  Si calcola un lower bound L_i applicando una *strategia di bounding* a (c, S_i).
  Se $\tilde{z} \leq L_i$, il problema (c, S_i) può essere chiuso e si ricomincia
  Se $\bar{x}_i \in S_i$ allora, siccome $L_i < \tilde{z}$, si pone $\tilde{x} := \bar{x}_i$ e $\tilde{z} := L_i$ (*aggiornamento dell'ottimo corrente*). Il problema (c, S_i) può essere chiuso e si ricomincia

Descrizione del Branch & Bound – passo k

Passo k dell'algoritmo di B& B

Sia $\mathcal{Q}$ la lista di sottoproblemi *candidati*

- Se $\mathcal{Q} = \emptyset$, STOP. $\tilde{x}$ è ottimo globale di (c, S). Altrimenti sia $\mathcal{Q} = \{(c, S_1), \dots, (c, S_q)\}$ la lista di sottoproblemi candidati.
- **Selection** (Scelta di un sottoproblema candidato)
  Si seleziona un problema $(c, S_i) \in \mathcal{Q}$ e lo si elimina da $\mathcal{Q}$,
  $\mathcal{Q} := \mathcal{Q} \setminus \{(c, S_i)\}$
- **Bounding**
  Si calcola un lower bound L_i applicando una *strategia di bounding* a (c, S_i).
  Se $\tilde{z} \leq L_i$, il problema (c, S_i) può essere chiuso e si ricomincia
  Se $\bar{x}_i \in S_i$ allora, siccome $L_i < \tilde{z}$, si pone $\tilde{x} := \bar{x}_i$ e $\tilde{z} := L_i$ (*aggiornamento dell'ottimo corrente*). Il problema (c, S_i) può essere chiuso e si ricomincia
- **Branching**
  Si applica una *strategia di separazione* a S_i, e si generano i sottoinsiemi $S_{i1}, \dots, S_{ir}$, Si pone
  $\mathcal{Q} := \mathcal{Q} \cup \{(c, S_{i1}), \dots, (c, S_{ir})\}$

Optimality Gap

Si consideri la k-esima iterazione dell'algoritmo di Branch & Bound per la soluzione di un problema (c, S) di **minimizzazione**.

Definizione

La differenza

$$\tilde{z} - \bar{L}$$

è detta Gap di ottimalità (optimality gap). $\bar{L}$ *indica il miglior valore teoricamente ottenibile.* $\bar{L}$ *è il più piccolo lower-bound tra tutti gli* L_i *nei nodi gia analizzati fino alla iterazione* k.

Optimality Gap

Si consideri la k-esima iterazione dell'algoritmo di Branch & Bound per la soluzione di un problema (c, S) di **minimizzazione**.

Definizione

La differenza

$$\tilde{z} - \bar{L}$$

è detta Gap di ottimalità (optimality gap). $\bar{L}$ *indica il miglior valore teoricamente ottenibile.* $\bar{L}$ *è il più piccolo lower-bound tra tutti gli* L_i *nei nodi gia analizzati fino alla iterazione* k.

relative optimality gap:

$$\frac{\tilde{z} - \bar{L}}{\max\{\epsilon, |\tilde{z}|\}}$$

Optimality Gap

Si consideri la k-esima iterazione dell'algoritmo di Branch & Bound per la soluzione di un problema (c, S) di **minimizzazione**.

Definizione

La differenza

$$\tilde{z} - \bar{L}$$

è detta Gap di ottimalità (optimality gap). $\bar{L}$ *indica il miglior valore teoricamente ottenibile.* $\bar{L}$ *è il più piccolo lower-bound tra tutti gli* L_i *nei nodi gia analizzati fino alla iterazione* k.

relative optimality gap:

$$\frac{\tilde{z} - \bar{L}}{\max\{\epsilon, |\tilde{z}|\}}$$

Misure che ci dicono quanto la miglior soluzione intera $(\tilde{x})$ è "lontana" dall'ottimo del problema.

Problema esempio

Sia dato il seguente problema di Programmazione Matematica

$$
\begin{aligned}
\min\ & -3x - y \\
s.t.\ & 7x + 2y \leq 22 \\
& -2x + 2y \leq 1 \\
& 1 \leq x \leq 4 \\
& 0 \leq y \leq 3 \\
& x, y \in \mathbb{Z}
\end{aligned}
\qquad (P_0)
$$

Inizializzazione del B&B

- $\mathcal{Q} := \{(c, S_0)\}$
- Soluzione incombente (best integer)

$$\tilde{x} = (2; 2)^\top \quad f(\tilde{x}) = -8 = \tilde{z} = UB$$

Albero di enumerazione (enumeration tree) costituito solo dal *root node*

$UB = -8, GAP = -, rGAP = -$

(c, S_0)

Iterazione 1

- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede

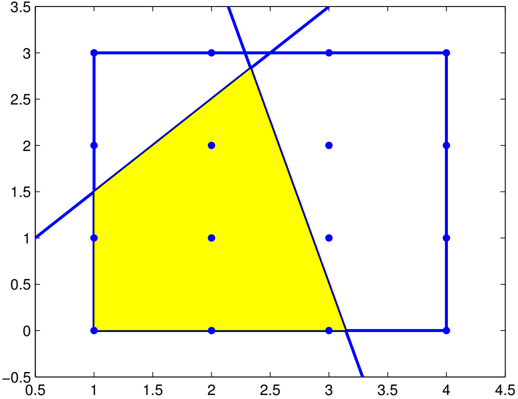
Iterazione 1

- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede
- (selection) $\mathcal{Q} = \{(c, S_0)\}$ quindi selezioniamo (c, S_0) e $\mathcal{Q} = \emptyset$

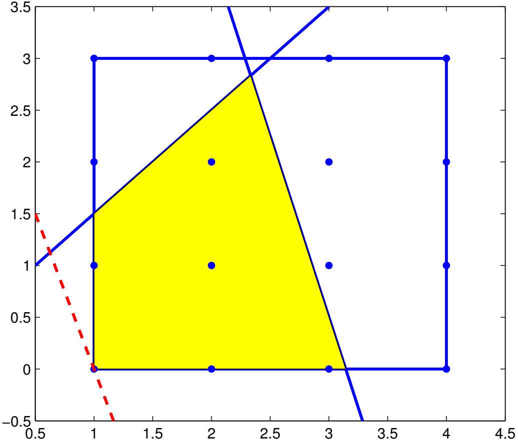
Iterazione 1

- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede
- (selection) $\mathcal{Q} = \{(c, S_0)\}$ quindi selezioniamo (c, S_0) e $\mathcal{Q} = \emptyset$
- (bounding) si calcola un lower bound per il problema (c, S_0)

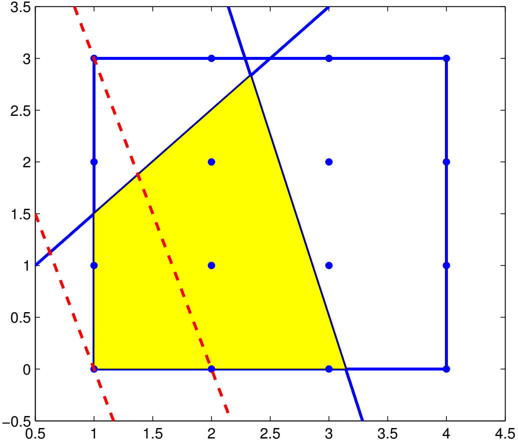
Soluzione grafica del rilassamento continuo



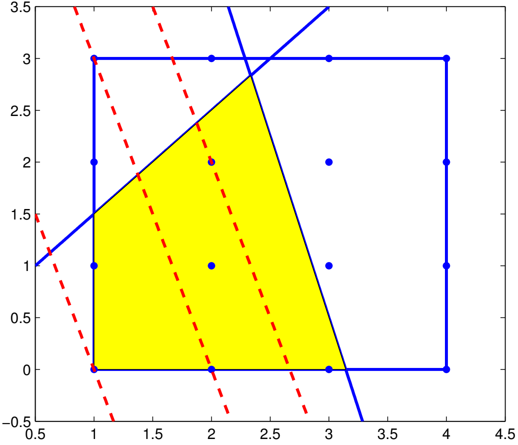
Soluzione grafica del rilassamento continuo



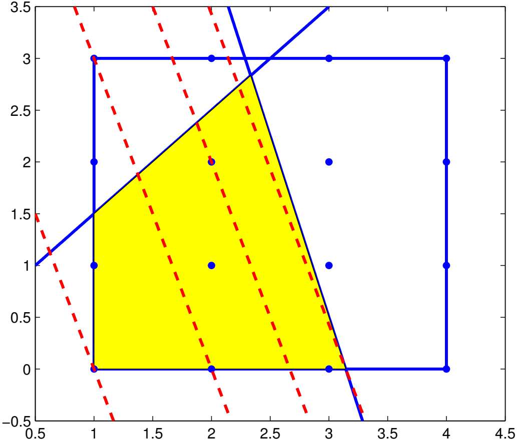
Soluzione grafica del rilassamento continuo



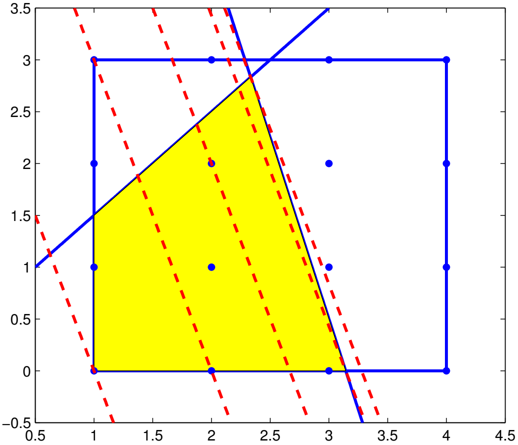
Soluzione grafica del rilassamento continuo



Soluzione grafica del rilassamento continuo



Soluzione grafica del rilassamento continuo

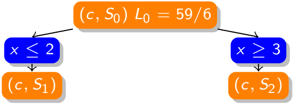


$$x^* = (7/3; 17/6)^\top \quad f(x^*) = -59/6 = -9.833 = L_0$$

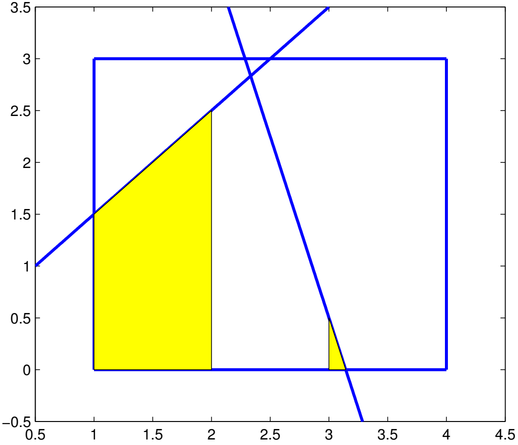
Iterazione 1

- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede
- (selection) $\mathcal{Q} = \{(c, S_0)\}$ quindi selezioniamo (c, S_0) e $\mathcal{Q} = \emptyset$
- (bounding) si calcola un lower bound L_0 per il problema (c, S_0)
$$x^* = (7/3; 17/6)^\top \quad L_0 = -9.833$$
x^* non è intero e $L_0 < UB$ quindi
  - il problema (c, S_0) non può essere chiuso
- (branching) (c, S_0) deve essere partizionato
separiamo rispetto alla prima componente frazionaria di x^0.
Generiamo (c, S_1) e (c, S_2) e li inseriamo nella lista
$\mathcal{Q} = \{(c, S_1), (c, S_2)\}$

$UB = -8$, $GAP = 1.833$, $rGAP = 22.91\%$



Rappresentazione grafica di P_1 e P_2



Iterazione 1

I problemi (c, S_1) e (c, S_2) sono:

$$\begin{aligned} \min\ & -3x - y \\ s.t.\ & 7x + 2y \leq 22 \\ & -2x + 2y \leq 1 \\ & 1 \leq x \leq 2 \\ & 0 \leq y \leq 3 \\ & x, y \in \mathbb{Z} \end{aligned} \qquad \begin{aligned} \min\ & -3x - y \\ s.t.\ & 7x + 2y \leq 22 \\ & -2x + 2y \leq 1 \\ & 3 \leq x \leq 4 \\ & 0 \leq y \leq 3 \\ & x, y \in \mathbb{Z} \end{aligned}$$

Iterazione 2

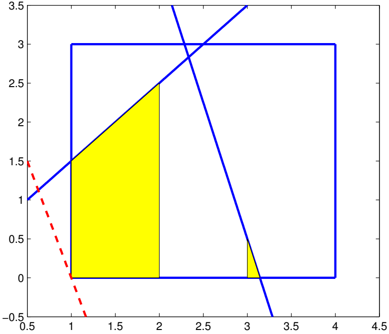
- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede

Iterazione 2

- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede
- (selection) $\mathcal{Q} = \{(c, S_1), (c, S_2)\}$, selezioniamo (c, S_1), quindi $\mathcal{Q} := \{(c, S_2)\}$

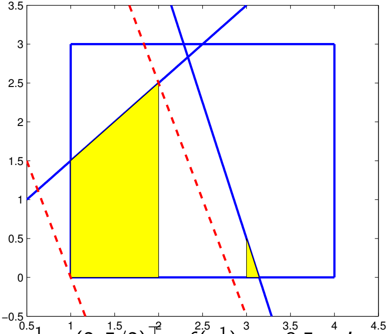
Iterazione 2

- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede
- (selection) $\mathcal{Q} = \{(c, S_1), (c, S_2)\}$, selezioniamo (c, S_1), quindi $\mathcal{Q} := \{(c, S_2)\}$
- (bounding) si calcola un lower bound L_1 per il problema (c, S_1)



Iterazione 2

- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede
- (selection) $\mathcal{Q} = \{(c, S_1), (c, S_2)\}$, selezioniamo (c, S_1), quindi $\mathcal{Q} := \{(c, S_2)\}$
- (bounding) si calcola un lower bound L_1 per il problema (c, S_1)



$x^1 = (2; 5/2)^\top \quad f(x^1) = -8.5 = L_1$

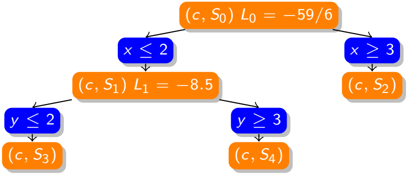
Iterazione 2

- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede
- (selection) $\mathcal{Q} = \{(c, S_1), (c, S_2)\}$, selezioniamo (c, S_1), quindi $\mathcal{Q} := \{(c, S_2)\}$
- (bounding) si calcola un lower bound L_1 per il problema (c, S_1)
  x^1 non è intero e $L_1 < UB$ quindi
  - il problema (c, S_1) non può essere chiuso
- (branching) (c, S_1) deve essere partizionato

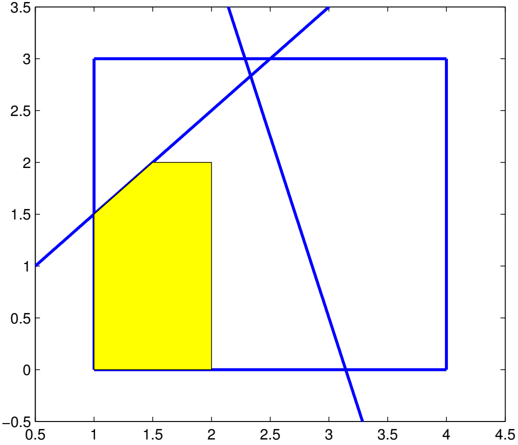
  separiamo rispetto alla prima componente frazionaria di x^1. Generiamo (c, S_3) e (c, S_4) e li inseriamo nella lista $\mathcal{Q} = \{(c, S_2), (c, S_3), (c, S_4)\}$

Iterazione 2

$UB = -8$, $GAP = 1.833$, $rGAP = 22.91\%$



Rappresentazione grafica di P_3 e P_4



Iterazione 2

I due problemi (c, S_3) e (c, S_4) sono:

$$\begin{aligned} \max\ & 3x + y \\ s.t.\ & 7x + 2y \leq 22 \\ & -2x + 2y \leq 1 \\ & 1 \leq x \leq 2 \\ & 0 \leq y \leq 2 \\ & x, y \in \mathbb{Z} \end{aligned} \qquad \begin{aligned} \max\ & 3x + y \\ s.t.\ & 7x + 2y \leq 22 \\ & -2x + 2y \leq 1 \\ & 1 \leq x \leq 2 \\ & 3 \leq y \leq 3 \\ & x, y \in \mathbb{Z} \end{aligned}$$

((c, S_4) è non ammissibile)

Iterazione 3

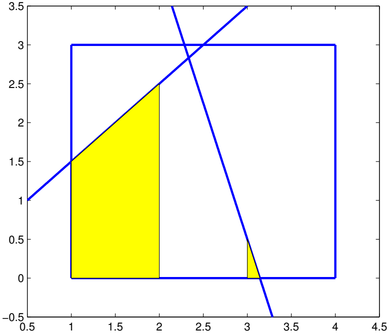
- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede

Iterazione 3

- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede
- (selection) $\mathcal{Q} = \{(c, S_2), (c, S_3), (c, S_4)\}$, selezioniamo (c, S_2), quindi $\mathcal{Q} := \{(c, S_3), (c, S_4)\}$

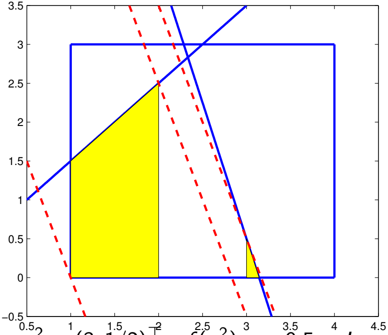
Iterazione 3

- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede
- (selection) $\mathcal{Q} = \{(c, S_2), (c, S_3), (c, S_4)\}$, selezioniamo (c, S_2), quindi $\mathcal{Q} := \{(c, S_3), (c, S_4)\}$
- (bounding) si calcola un lower bound L_2 per il problema (c, S_2)



Iterazione 3

- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede
- (selection) $\mathcal{Q} = \{(c, S_2), (c, S_3), (c, S_4)\}$, selezioniamo (c, S_2), quindi $\mathcal{Q} := \{(c, S_3), (c, S_4)\}$
- (bounding) si calcola un lower bound L_2 per il problema (c, S_2)



$x^2 = (3; 1/2)^\top \quad f(x^2) = -9.5 = L_2$

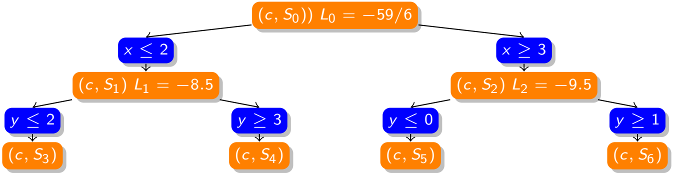
Iterazione 3

- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede
- (selection) $\mathcal{Q} = \{(c, S_2), (c, S_3), (c, S_4)\}$, selezioniamo (c, S_2), quindi $\mathcal{Q} := \{(c, S_3), (c, S_4)\}$
- (bounding) si calcola un lower bound L_2 per il problema (c, S_2) x^2 non è intero e $L_2 < UB$ quindi
    - il problema (c, S_2) non può essere chiuso
- (branching) (c, S_2) deve essere partizionato

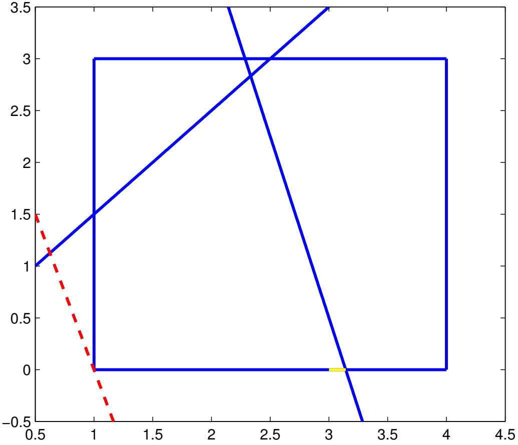
  separiamo rispetto alla prima componente frazionaria di x^2. Generiamo (c, S_5) e (c, S_6) e li inseriamo nella lista $\mathcal{Q} = \{(c, S_3), (c, S_4), (c, S_5), (c, S_6)\}$

Iterazione 3

$UB = -8$, $GAP = 1.5$, $rGAP = 18.75\%$



Rappresentazione grafica di P_5 e P_6



Iterazione 3

I due problemi (c, S_5) e (c, S_6) sono:

$$\begin{aligned} \max\ & 3x + y \\ s.t.\ & 7x + 2y \leq 22 \\ & -2x + 2y \leq 1 \\ & 3 \leq x \leq 4 \\ & 0 \leq y \leq 0 \\ & x, y \in \mathbb{Z} \end{aligned} \qquad \begin{aligned} \max\ & 3x + y \\ s.t.\ & 7x + 2y \leq 22 \\ & -2x + 2y \leq 1 \\ & 3 \leq x \leq 4 \\ & 1 \leq y \leq 3 \\ & x, y \in \mathbb{Z} \end{aligned}$$

((c, S_6) è non ammissibile)

Iterazione 4

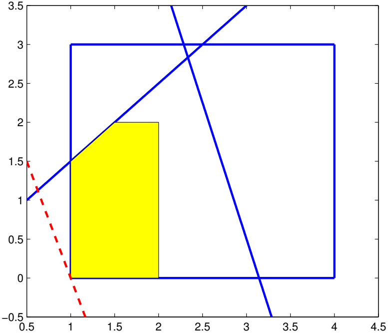
- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede

Iterazione 4

- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede
- (selection) $\mathcal{Q} = \{(c, S_3), (c, S_4), (c, S_5), (c, S_6)\}$, selezioniamo (c, S_3), quindi $\mathcal{Q} := \{(c, S_4), (c, S_5), (c, S_6)\}$

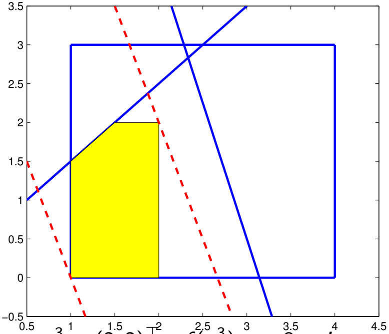
Iterazione 4

- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede
- (selection) $\mathcal{Q} = \{(c, S_3), (c, S_4), (c, S_5), (c, S_6)\}$, selezioniamo (c, S_3), quindi $\mathcal{Q} := \{(c, S_4), (c, S_5), (c, S_6)\}$
- (bounding) si calcola un lower bound L_3 per il problema (c, S_3)



Iterazione 4

- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede
- (selection) $\mathcal{Q} = \{(c, S_3), (c, S_4), (c, S_5), (c, S_6)\}$, selezioniamo (c, S_3), quindi $\mathcal{Q} := \{(c, S_4), (c, S_5), (c, S_6)\}$
- (bounding) si calcola un lower bound L_3 per il problema (c, S_3)



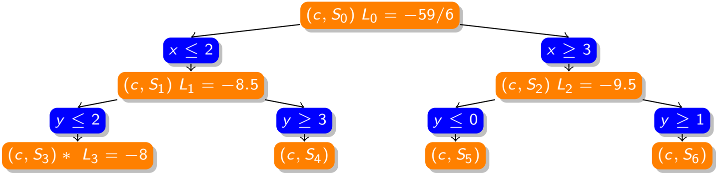
$x^3 = (2; 2)^\top \quad f(x^3) = -8 = L_3$

Iterazione 4

- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede
- (selection) $\mathcal{Q} = \{(c, S_3), (c, S_4), (c, S_5), (c, S_6)\}$, selezioniamo (c, S_3), quindi $\mathcal{Q} := \{(c, S_4), (c, S_5), (c, S_6)\}$
- (bounding) si calcola un lower bound L_3 per il problema (c, S_3)
  x^3 è intero e $L_3 \geq UB$ quindi
  - il problema (c, S_3) può essere chiuso

Iterazione 4

$UB = -8$, $GAP = 1.5$, $rGAP = 18.75\%$



Iterazione 5

- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede

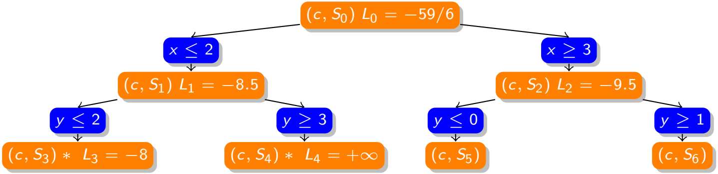
Iterazione 5

- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede
- (selection) $\mathcal{Q} = \{(c, S_4), (c, S_5), (c, S_6)\}$, selezioniamo (c, S_4), quindi $\mathcal{Q} := \{(c, S_5), (c, S_6)\}$

Iterazione 5

- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede
- (selection) $\mathcal{Q} = \{(c, S_4), (c, S_5), (c, S_6)\}$, selezioniamo (c, S_4), quindi $\mathcal{Q} := \{(c, S_5), (c, S_6)\}$
- (bounding) il problema (c, S_4) è non ammissibile. Si pone
  $L_4 = +\infty$ quindi
  $L_4 \geq UB$ e pertanto
  - il problema (c, S_4) può essere chiuso

$UB = -8$, $GAP = 1.5$, $rGAP = 18.75\%$



Iterazione 6

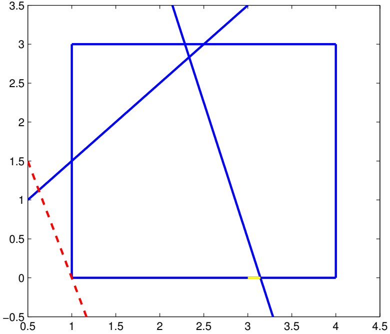
- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede

Iterazione 6

- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede
- (selection) $\mathcal{Q} = \{(c, S_5), (c, S_6)\}$, selezioniamo (c, S_5), quindi $\mathcal{Q} := \{(c, S_6)\}$

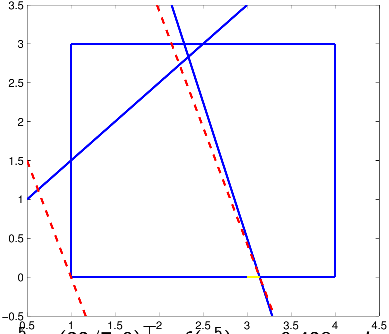
Iterazione 6

- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede
- (selection) $\mathcal{Q} = \{(c, S_5), (c, S_6)\}$, selezioniamo (c, S_5), quindi $\mathcal{Q} := \{(c, S_6)\}$
- (bounding) si calcola un lower bound L_5 per il problema (c, S_5)



Iterazione 6

- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede
- (selection) $\mathcal{Q} = \{(c, S_5), (c, S_6)\}$, selezioniamo (c, S_5), quindi $\mathcal{Q} := \{(c, S_6)\}$
- (bounding) si calcola un lower bound L_5 per il problema (c, S_5)



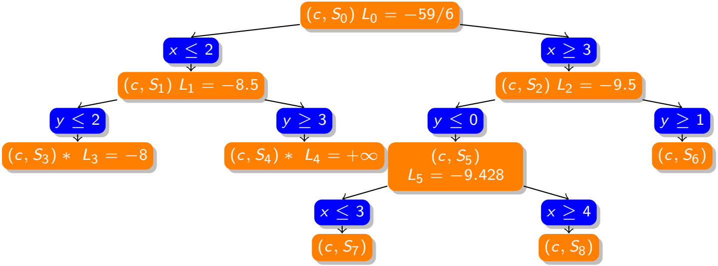
$x^5 = (22/7; 0)^\top \quad f(x^5) = -9.428 = L_5$

Iterazione 6

- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede
- (selection) $\mathcal{Q} = \{(c, S_5), (c, S_6)\}$, selezioniamo (c, S_5), quindi $\mathcal{Q} := \{(c, S_6)\}$
- (bounding) si calcola un lower bound L_5 per il problema (c, S_5)
  x^5 non è intero e $L_5 < UB$ quindi
  - il problema (c, S_5) non può essere chiuso
- (branching) (c, S_5) deve essere partizionato
  separiamo rispetto alla prima componente frazionaria di x^5. Generiamo (c, S_7) e (c, S_8) e li inseriamo nella lista $\mathcal{Q} = \{(c, S_6), (c, S_7), (c, S_8)\}$

Iterazione 6

$UB = -8$, $GAP = 1.5$, $rGAP = 18.75\%$



Iterazione 6

I due problemi (c, S_7) e (c, S_8) sono:

$$
\begin{aligned}
\max\ & 3x + y \\
s.t.\ & 7x + 2y \le 22 \\
& -2x + 2y \le 1 \\
& 3 \le x \le 3 \\
& 0 \le y \le 0 \\
& x, y \in \mathbb{Z}
\end{aligned}
\qquad\qquad
\begin{aligned}
\max\ & 3x + y \\
s.t.\ & 7x + 2y \le 22 \\
& -2x + 2y \le 1 \\
& 4 \le x \le 4 \\
& 0 \le y \le 0 \\
& x, y \in \mathbb{Z}
\end{aligned}
$$

(c, S_7) amette una sola soluzione $x^7 = (3, 0)^\top$, (c, S_8) è non ammissibile.

Iterazione 7

- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede

Iterazione 7

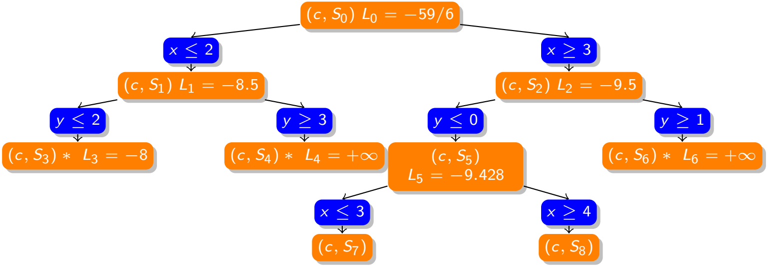
- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede
- (selection) $\mathcal{Q} = \{(c, S_6), (c, S_7), (c, S_8)\}$, selezioniamo (c, S_6), quindi $\mathcal{Q} := \{(c, S_7), (c, S_8)\}$

Iterazione 7

- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede
- (selection) $\mathcal{Q} = \{(c, S_6), (c, S_7), (c, S_8)\}$, selezioniamo (c, S_6), quindi $\mathcal{Q} := \{(c, S_7), (c, S_8)\}$
- (bounding) il problema (c, S_6) è non ammissibile. Si pone $L_6 = +\infty > UB$ e
  - il problema (c, S_6) può essere chiuso

Iterazione 6

$UB = -8$, $GAP = 1.428$, $rGAP = 17.85\%$



Iterazione 8

- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede

Iterazione 8

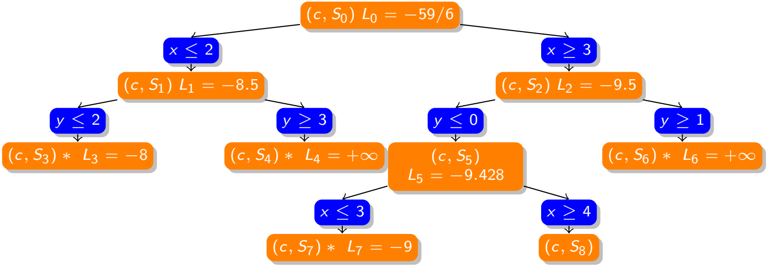
- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede
- (selection) $\mathcal{Q} = \{(c, S_7), (c, S_8)\}$, selezioniamo (c, S_7), quindi $\mathcal{Q} := \{(c, S_8)\}$

Iterazione 8

- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede
- (selection) $\mathcal{Q} = \{(c, S_7), (c, S_8)\}$, selezioniamo (c, S_7), quindi $\mathcal{Q} := \{(c, S_8)\}$
- (bounding) il problema (c, S_7) amette come unica soluzione $x^7 = (3, 0)^\top$ intera con $L_7 = -9 < UB$. Quindi
  - (aggiornamento ottimo corrente) $\tilde{x} := x^7$, $UB := L_7 = -9$
  - il problema (c, S_7) può essere chiuso

Iterazione 7

$UB = -9$, $GAP = 0.428$, $rGAP = 4.75\%$



Iterazione 9

- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede

Iterazione 9

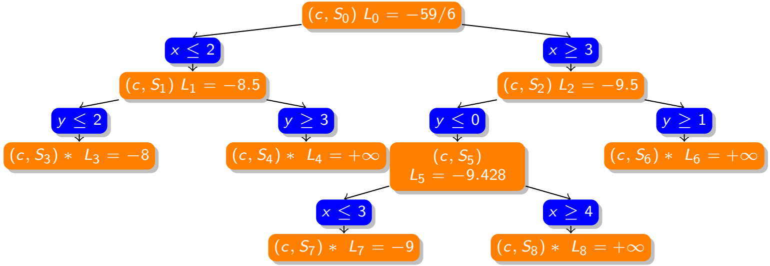
- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede
- (selection) $\mathcal{Q} = \{(c, S_8)\}$, selezioniamo (c, S_8), quindi $\mathcal{Q} := \emptyset$

Iterazione 9

- (criterio di aresto) $\mathcal{Q} \neq \emptyset$ quindi si procede
- (selection) $\mathcal{Q} = \{(c, S_8)\}$, selezioniamo (c, S_8), quindi $\mathcal{Q} := \emptyset$
- (bounding) il problema (c, S_8) è non ammissibile quindi $L_8 = +\infty$ e
  - il problema (c, S_8) può essere chiuso

Iterazione 8

$UB = -9$, $GAP = 0$, $rGAP = 0$



Iterazione 10

- (criterio di aresto) $\mathcal{Q} = \emptyset$ quindi STOP

Iterazione 10

- (criterio di aresto) $\mathcal{Q} = \emptyset$ quindi STOP

$$\tilde{x} = (3,0)^\top, \qquad c^\top \tilde{x} = UB = -9$$

Ulteriori Considerazioni sul Branch & Bound

Dalla definizione del metodo (come anche dall'esempio di esecuzione), appare evidente che l'algoritmo è notevolmente influenzato da:

Ulteriori Considerazioni sul Branch & Bound

Dalla definizione del metodo (come anche dall'esempio di esecuzione), appare evidente che l'algoritmo è notevolmente influenzato da:

- strategia di *bounding*
- strategia di *branching*

Ulteriori Considerazioni sul Branch & Bound

Dalla definizione del metodo (come anche dall'esempio di esecuzione), appare evidente che l'algoritmo è notevolmente influenzato da:

- strategia di *bounding*
- strategia di *branching*
- strategia di *selection* cioè di selezione del problema aperto da esaminare. Come si sceglie un problema $(c, S_i) \in \mathcal{Q}$?

Ulteriori Considerazioni sul Branch & Bound

Dalla definizione del metodo (come anche dall'esempio di esecuzione), appare evidente che l'algoritmo è notevolmente influenzato da:

- strategia di *bounding*
- strategia di *branching*
- strategia di *selection* cioè di selezione del problema aperto da esaminare. Come si sceglie un problema $(c, S_i) \in \mathcal{Q}$?
  - First-In-First-Out (FIFO), esplorazione dell'albero in ampiezza (come nell'esempio)

Ulteriori Considerazioni sul Branch & Bound

Dalla definizione del metodo (come anche dall'esempio di esecuzione), appare evidente che l'algoritmo è notevolmente influenzato da:

- strategia di *bounding*
- strategia di *branching*
- strategia di *selection* cioè di selezione del problema aperto da esaminare. Come si sceglie un problema $(c, S_i) \in \mathcal{Q}$?
  - First-In-First-Out (FIFO), esplorazione dell'albero in ampiezza (come nell'esempio)
  - Last-In-First-Out (LIFO), esplorazione dell'albero in profodità

Ulteriori Considerazioni sul Branch & Bound

Dalla definizione del metodo (come anche dall'esempio di esecuzione), appare evidente che l'algoritmo è notevolmente influenzato da:

- strategia di *bounding*
- strategia di *branching*
- strategia di *selection* cioè di selezione del problema aperto da esaminare. Come si sceglie un problema $(c, S_i) \in \mathcal{Q}$?
  - First-In-First-Out (FIFO), esplorazione dell'albero in ampiezza (come nell'esempio)
  - Last-In-First-Out (LIFO), esplorazione dell'albero in profodità
  - Best-Bound, si sceglie il problema con il lower bound più basso (esplorazione di problemi promettenti)

Ulteriori Considerazioni sul Branch & Bound

Dalla definizione del metodo (come anche dall'esempio di esecuzione), appare evidente che l'algoritmo è notevolmente influenzato da:

- strategia di *bounding*
- strategia di *branching*
- strategia di *selection* cioè di selezione del problema aperto da esaminare. Come si sceglie un problema $(c, S_i) \in \mathcal{Q}$?
  - First-In-First-Out (FIFO), esplorazione dell'albero in ampiezza (come nell'esempio)
  - Last-In-First-Out (LIFO), esplorazione dell'albero in profodità
  - Best-Bound, si sceglie il problema con il lower bound più basso (esplorazione di problemi promettenti)
  - Worst-Bound, si sceglie il problema con il lower bound più alto (tentativo di chiudere un maggior numero di problemi)

Ulteriori Considerazioni sul Branch & Bound

Dalla definizione del metodo (come anche dall'esempio di esecuzione), appare evidente che l'algoritmo è notevolmente influenzato da:

- strategia di *bounding*
- strategia di *branching*
- strategia di *selection* cioè di selezione del problema aperto da esaminare. Come si sceglie un problema $(c, S_i) \in \mathcal{Q}$?
  - First-In-First-Out (FIFO), esplorazione dell'albero in ampiezza (come nell'esempio)
  - Last-In-First-Out (LIFO), esplorazione dell'albero in profodità
  - Best-Bound, si sceglie il problema con il lower bound più basso (esplorazione di problemi promettenti)
  - Worst-Bound, si sceglie il problema con il lower bound più alto (tentativo di chiudere un maggior numero di problemi)
  - strategie euristiche miste