



SAPIENZA
UNIVERSITÀ DI ROMA

Programmazione lineare intera

Volume 2 — Metodi esatti

L'algoritmo dei piani di taglio con i tagli frazionari di Gomory, il branch-and-bound basato sulla programmazione lineare e il branch-and-cut. Esempi svolti passo per passo ed esercizi con la soluzione completa.

Fabio Furini

fabiofurini.github.io

Dipartimento di Ingegneria informatica, automatica e gestionale — Sapienza Università di Roma

Versione online e materiale scaricabile:

fabiofurini.github.io/programmazione-lineare-intera/
github.com/fabiofurini/programmazione-lineare-intera

Anno accademico 2026–2027

Indice

III	Metodi esatti	2
6	L'algoritmo dei piani di taglio con i tagli frazionari di Gomory	3
6.1	Tagli frazionari di Gomory	4
6.2	Un algoritmo dei piani di taglio basato sui tagli frazionari di Gomory	14
6.3	L'esempio dell'auto	27
6.4	Errori tipici	28
6.5	In sintesi	28
7	Il branch-and-bound basato sulla programmazione lineare	30
7.1	L'idea in parole e in una figura	31
7.2	Definizioni e notazione	32
7.3	L'algoritmo	35
7.4	Breve riepilogo dell'algoritmo del simplesso duale	38
7.5	Strategie di selezione della variabile di branching	39
7.6	Strategie di selezione del nodo di branching	41
7.7	Esempi svolti sull'esempio dell'auto	43
7.8	Errori tipici	62
7.9	Estensioni	63
7.10	In sintesi	63
8	Il branch-and-cut	65
8.1	Algoritmi branch-and-cut	66
8.2	Disuguaglianze valide globali e locali	66
8.3	Cut-and-branch e branch-and-cut	67
8.4	Esempio svolto: cut-and-branch con tagli di Gomory	69
8.5	Tagli locali: un esempio	72
8.6	Aspetti pratici	74
8.7	Errori tipici	75
8.8	In sintesi	75

Parte III

Metodi esatti



L'algoritmo dei piani di taglio con i tagli frazionari di Gomory

Prerequisiti

- Dal modulo *Programmazione lineare*, [Capitolo 10 «Algoritmo del sempliceo duale»](#): dizionari del sempliceo, basi duali ammissibili e operazioni di pivot duali. Un breve richiamo è dato in questo capitolo.
- Da questo modulo, i capitoli «Programmazione lineare intera» e «Bound primali e bound duali»: problemi PLI, il rilassamento lineare e l'upper bound dato dal rilassamento lineare.
- Da questo modulo, il capitolo «Disuguaglianze valide»: disuguaglianze valide e tagli che eliminano punti frazionari.

Obiettivi

Alla fine di questo capitolo sarai in grado di:

- generare un taglio frazionario di Gomory da una riga di un dizionario del sempliceo, e dimostrare che è valido e che elimina la soluzione di base corrente;
- inserire un taglio nel dizionario e riottimizzare con l'algoritmo del sempliceo duale;
- eseguire a mano un algoritmo dei piani di taglio, con la strategia di selezione del termine noto più grande e le sue regole per i pareggi;
- calcolare l'upper bound a ogni iterazione e spiegare perché l'algoritmo può richiedere molti tagli;
- spiegare perché questi tagli sono validi solo per problemi PLI con dati interi.

6.1 Tagli frazionari di Gomory

Definizione 6.1: piano di taglio

Dato un problema PLI e una soluzione ottima frazionaria $\tilde{x}(\text{LP})$ del suo rilassamento lineare, un **piano di taglio** (o **taglio**) è una disuguaglianza valida per il problema PLI che elimina $\tilde{x}(\text{LP})$.

I tagli frazionari di Gomory sono disuguaglianze valide che eliminano soluzioni di base ammissibili frazionarie del rilassamento lineare di un problema PLI senza escludere alcuna soluzione intera ammissibile.

- **Consideriamo problemi PLI in forma canonica di massimo con dati interi.** Per questi problemi, anche le variabili di scarto introdotte nel rilassamento lineare in forma standard assumono valori interi in ogni soluzione intera ammissibile del problema PLI.
- I tagli frazionari di Gomory sono validi solo se tutte le variabili, comprese le variabili di scarto, sono intere. Per i problemi MIP con variabili continue possono eliminare soluzioni ammissibili, e serve una famiglia diversa di tagli.
- Ricordiamo che un dizionario del simplesso del rilassamento lineare in forma standard ha la forma seguente

$$\begin{cases} x_i = \bar{b}_i - \sum_{j \in \mathcal{N}} \bar{a}_{ij} x_j, & \forall i \in \mathcal{B}, \\ z = \bar{z} + \sum_{j \in \mathcal{N}} \bar{c}_j x_j. \end{cases}$$

dove $\mathcal{B}$ è l'insieme degli indici delle variabili in base e $\mathcal{N}$ è l'insieme degli indici delle variabili fuori base. Se $\bar{b}_i \geq 0$ per ogni $i \in \mathcal{B}$, il dizionario è associato alla seguente soluzione di base ammissibile:

$$x_i = \bar{b}_i, \quad \forall i \in \mathcal{B}, \quad x_j = 0, \quad \forall j \in \mathcal{N}.$$

Se, inoltre, almeno un $\bar{b}_i$ è frazionario, questa soluzione di base ammissibile è frazionaria.

- I tagli frazionari di Gomory si ottengono dalle righe di un dizionario del simplesso associato a una soluzione di base ammissibile frazionaria del rilassamento lineare in forma standard di un problema PLI.

Definizione 6.2: taglio frazionario di Gomory

Data una riga $i \in \mathcal{B}$ di un dizionario del simplesso associato a una soluzione di base ammissibile tale che $\bar{b}_i$ sia frazionario, la disuguaglianza lineare

$$\sum_{j \in \mathcal{N}} (\bar{a}_{ij} - \lfloor \bar{a}_{ij} \rfloor) x_j \geq \bar{b}_i - \lfloor \bar{b}_i \rfloor \quad (6.1)$$

si chiama **taglio frazionario di Gomory** generato dalla riga i .

- Dimostreremo che un taglio frazionario di Gomory non elimina alcuna soluzione intera ammissibile del problema PLI e che elimina la soluzione di base ammissibile frazionaria.

Esempio 6.1: Problema PLI in forma canonica, rilassamento lineare in forma standard e dizionari

Consideriamo il seguente problema PLI in forma canonica di massimo con 2 variabili intere e 2 vincoli lineari (detto [Esempio 1](#)):

$$\begin{aligned} \max \quad & 2x_1 + 4x_2 \\ & -x_1 + x_2 \leq 2 \\ & 8x_1 + 2x_2 \leq 19 \\ & x_1, x_2 \in \mathbb{Z}_{\geq 0} \end{aligned}$$

Il rilassamento lineare in forma standard, con 4 variabili non negative (2 variabili di scarto) e 2 vincoli lineari, è il seguente:

$$\begin{aligned} \max \quad & 2x_1 + 4x_2 \\ & -x_1 + x_2 + x_3 = 2 \\ & 8x_1 + 2x_2 + x_4 = 19 \\ & x_1, x_2, x_3, x_4 \geq 0 \end{aligned}$$

Come esempio, consideriamo il seguente dizionario del simpleso associato a una soluzione di base ammissibile frazionaria del rilassamento lineare in forma standard del problema PLI:

$$\left\{ \begin{array}{l} x_1 = \frac{3}{2} + \frac{1}{5}x_3 - \frac{1}{10}x_4 \\ x_2 = \frac{7}{2} - \frac{4}{5}x_3 - \frac{1}{10}x_4 \\ z = 17 - \frac{14}{5}x_3 - \frac{3}{5}x_4 \end{array} \right. \quad \text{dove} \quad \bar{A} = \begin{pmatrix} -\frac{1}{5} & \frac{1}{10} \\ \frac{4}{5} & \frac{1}{10} \end{pmatrix} \quad \bar{b} = \begin{pmatrix} \frac{3}{2} \\ \frac{7}{2} \end{pmatrix}$$

con $\mathcal{B} = \{1, 2\}$ e $\mathcal{N} = \{3, 4\}$. La soluzione di base ammissibile frazionaria associata è:

$$\bar{x}_1 = \frac{3}{2}, \quad \bar{x}_2 = \frac{7}{2}, \quad \bar{x}_3 = 0, \quad \bar{x}_4 = 0 \quad \text{di valore} \quad \bar{z} = 17$$

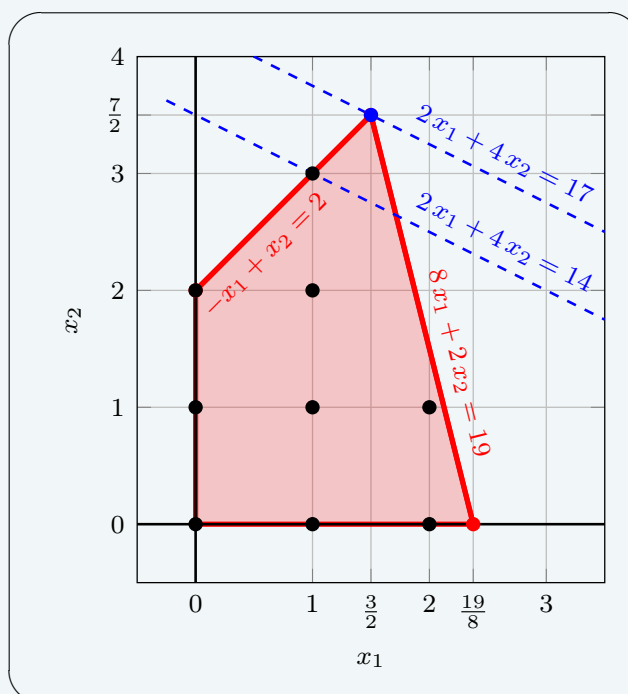
Esempio 6.2: Tagli frazionari di Gomory

Consideriamo di nuovo il problema PLI dell'[Esempio 1](#), in forma canonica di massimo con 2 variabili intere e 2 vincoli lineari:

$$\begin{aligned}
 \max \quad & 2x_1 + 4x_2 \\
 \text{s.t.} \quad & -x_1 + x_2 \leq 2 \\
 & 8x_1 + 2x_2 \leq 19 \\
 & x_1, x_2 \in \mathbb{Z}_{\geq 0}
 \end{aligned}$$

e il suo rilassamento lineare in forma canonica con 2 variabili non negative e 2 vincoli lineari:

$$\begin{aligned}
 \max \quad & 2x_1 + 4x_2 \\
 \text{s.t.} \quad & -x_1 + x_2 \leq 2 \\
 & 8x_1 + 2x_2 \leq 19 \\
 & x_1, x_2 \geq 0
 \end{aligned}$$



Una soluzione ottima $\tilde{\mathbf{x}}(\text{ILP})$ del problema PLI è

$$\tilde{x}_1 = 1, \quad \tilde{x}_2 = 3$$

Il valore ottimo è $z(\text{ILP}) = 14$. Una soluzione ottima $\tilde{\mathbf{x}}(\text{LP})$ del rilassamento lineare è

$$\tilde{x}_1 = \frac{3}{2}, \quad \tilde{x}_2 = \frac{7}{2}$$

Il valore ottimo è $z(\text{LP}) = 17$.

Consideriamo il seguente dizionario del simpleso associato a una soluzione di base ammissibile frazionaria del rilassamento lineare in forma standard del problema PLI dell'[Esempio 1](#):

$$\begin{cases} x_1 = \frac{3}{2} & + & \frac{1}{5} x_3 & - & \frac{1}{10} x_4 \\ x_2 = \frac{7}{2} & - & \frac{4}{5} x_3 & - & \frac{1}{10} x_4 \\ z = 17 & - & \frac{14}{5} x_3 & - & \frac{3}{5} x_4 \end{cases} \quad \text{dove} \quad \bar{A} = \begin{pmatrix} -\frac{1}{5} & \frac{1}{10} \\ \frac{4}{5} & \frac{1}{10} \end{pmatrix} \quad \bar{b} = \begin{pmatrix} \frac{3}{2} \\ \frac{7}{2} \end{pmatrix}$$

- Il taglio frazionario di Gomory generato dalla prima riga è:

$$\underbrace{\left(-\frac{1}{5} - \left\lfloor -\frac{1}{5} \right\rfloor \right)}_{-\frac{1}{5} - (-1) = \frac{4}{5}} x_3 + \underbrace{\left(\frac{1}{10} - \left\lfloor \frac{1}{10} \right\rfloor \right)}_{\frac{1}{10} - 0 = \frac{1}{10}} x_4 \geq \underbrace{\left(\frac{3}{2} - \left\lfloor \frac{3}{2} \right\rfloor \right)}_{\frac{3}{2} - 1 = \frac{1}{2}}$$

$$\frac{4}{5} x_3 + \frac{1}{10} x_4 \geq \frac{1}{2} \quad \Rightarrow \quad 8 x_3 + x_4 \geq 5$$

Dalla forma standard del rilassamento lineare abbiamo:

$$x_3 = 2 + x_1 - x_2 \quad \text{e} \quad x_4 = 19 - 8 x_1 - 2 x_2$$

Sostituendo nel taglio frazionario di Gomory, possiamo esprimerlo nelle variabili originali del problema PLI:

$$8 (2 + x_1 - x_2) + (19 - 8 x_1 - 2 x_2) \geq 5 \quad \Rightarrow \quad x_2 \leq 3$$

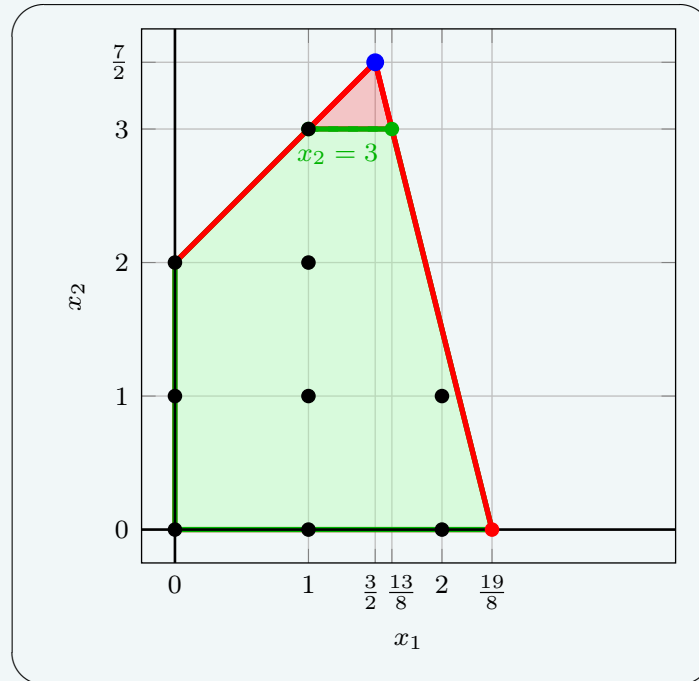
- Il taglio frazionario di Gomory generato dalla seconda riga è:

$$\underbrace{\left(\frac{4}{5} - \left\lfloor \frac{4}{5} \right\rfloor \right)}_{\frac{4}{5} - 0 = \frac{4}{5}} x_3 + \underbrace{\left(\frac{1}{10} - \left\lfloor \frac{1}{10} \right\rfloor \right)}_{\frac{1}{10} - 0 = \frac{1}{10}} x_4 \geq \underbrace{\left(\frac{7}{2} - \left\lfloor \frac{7}{2} \right\rfloor \right)}_{\frac{7}{2} - 3 = \frac{1}{2}}$$

$$\frac{4}{5} x_3 + \frac{1}{10} x_4 \geq \frac{1}{2} \quad \Rightarrow \quad 8 x_3 + x_4 \geq 5$$

Questo taglio frazionario di Gomory coincide con quello generato dalla prima riga.

Effetto del taglio frazionario di Gomory sulla regione ammissibile del rilassamento lineare in forma canonica.



Un taglio frazionario di Gomory si può generare anche dalla riga della funzione obiettivo del dizionario del semplice.

- Consideriamo la riga della funzione obiettivo di un dizionario del semplice:

$$z = \bar{z} + \sum_{j \in \mathcal{N}} \bar{c}_j x_j$$

Questa riga si può riscrivere nella stessa forma delle altre righe del dizionario:

$$z = \bar{z} - \sum_{j \in \mathcal{N}} (-\bar{c}_j) x_j$$

Il taglio frazionario di Gomory generato dalla riga della funzione obiettivo è

$$\sum_{j \in \mathcal{N}} (-\bar{c}_j - \lfloor -\bar{c}_j \rfloor) x_j \geq \bar{z} - \lfloor \bar{z} \rfloor \quad (6.2)$$

Esempio 6.3: Tagli frazionari di Gomory

Consideriamo il seguente dizionario del semplice associato a una soluzione di base ammissibile frazionaria del rilassamento lineare in forma standard del problema PLI dell'[Esempio 1](#):

$$\begin{cases} x_1 = \frac{19}{8} - \frac{1}{4} x_2 - \frac{1}{8} x_4 \\ x_3 = \frac{35}{8} - \frac{5}{4} x_2 - \frac{1}{8} x_4 \\ z = \frac{19}{4} + \frac{7}{2} x_2 - \frac{1}{4} x_4 \end{cases} \quad \text{dove} \quad \bar{A} = \begin{pmatrix} \frac{1}{4} & \frac{1}{8} \\ \frac{5}{4} & \frac{1}{8} \end{pmatrix} \quad \bar{b} = \begin{pmatrix} \frac{19}{8} \\ \frac{35}{8} \end{pmatrix}$$

- Questo dizionario non è ottimo ($\bar{c}_2 = \frac{7}{2} > 0$): un taglio frazionario di Gomory si può generare da qualsiasi dizionario con una soluzione di base frazionaria, ottimo o no. Nella figura la soluzione di base $(\frac{19}{8}, 0)$ è un vertice della regione ammissibile, non l'ottimo del rilassamento lineare.
- Il taglio frazionario di Gomory generato dalla prima riga è:

$$\underbrace{\left(\frac{1}{4} - \left\lfloor \frac{1}{4} \right\rfloor\right)}_{\frac{1}{4} - 0 = \frac{1}{4}} x_2 + \underbrace{\left(\frac{1}{8} - \left\lfloor \frac{1}{8} \right\rfloor\right)}_{\frac{1}{8} - 0 = \frac{1}{8}} x_4 \geq \underbrace{\left(\frac{19}{8} - \left\lfloor \frac{19}{8} \right\rfloor\right)}_{\frac{19}{8} - 2 = \frac{3}{8}}$$

$$\frac{1}{4} x_2 + \frac{1}{8} x_4 \geq \frac{3}{8} \quad \Rightarrow \quad 2 x_2 + x_4 \geq 3$$

Dalla forma standard del rilassamento lineare abbiamo:

$$x_4 = 19 - 8 x_1 - 2 x_2$$

Sostituendo nel taglio frazionario di Gomory, possiamo esprimerlo nelle variabili originali del problema PLI:

$$2 x_2 + (19 - 8 x_1 - 2 x_2) \geq 3 \quad \Rightarrow \quad x_1 \leq 2$$

- Il taglio frazionario di Gomory generato dalla seconda riga è:

$$\underbrace{\left(\frac{5}{4} - \left\lfloor \frac{5}{4} \right\rfloor\right)}_{\frac{5}{4} - 1 = \frac{1}{4}} x_2 + \underbrace{\left(\frac{1}{8} - \left\lfloor \frac{1}{8} \right\rfloor\right)}_{\frac{1}{8} - 0 = \frac{1}{8}} x_4 \geq \underbrace{\left(\frac{35}{8} - \left\lfloor \frac{35}{8} \right\rfloor\right)}_{\frac{35}{8} - 4 = \frac{3}{8}}$$

$$\frac{1}{4} x_2 + \frac{1}{8} x_4 \geq \frac{3}{8} \quad \Rightarrow \quad 2 x_2 + x_4 \geq 3$$

Questo taglio frazionario di Gomory coincide con quello generato dalla prima riga.

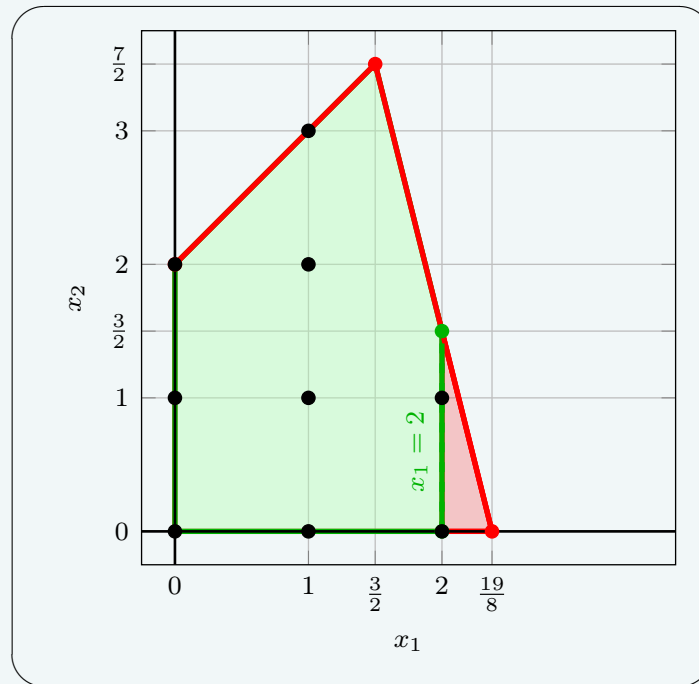
- Il taglio frazionario di Gomory generato dalla riga della funzione obiettivo è:

$$\underbrace{\left(-\frac{7}{2} - \left\lfloor -\frac{7}{2} \right\rfloor\right)}_{-\frac{7}{2} - (-4) = \frac{1}{2}} x_2 + \underbrace{\left(\frac{1}{4} - \left\lfloor \frac{1}{4} \right\rfloor\right)}_{\frac{1}{4} - 0 = \frac{1}{4}} x_4 \geq \underbrace{\left(\frac{19}{4} - \left\lfloor \frac{19}{4} \right\rfloor\right)}_{\frac{19}{4} - 4 = \frac{3}{4}}$$

$$\frac{1}{2} x_2 + \frac{1}{4} x_4 \geq \frac{3}{4} \quad \Rightarrow \quad 2 x_2 + x_4 \geq 3$$

Questo taglio frazionario di Gomory coincide con quello generato dalla prima riga.

Effetto del taglio frazionario di Gomory sulla regione ammissibile del rilassamento lineare in forma canonica.



6.1.1 Validità dei tagli frazionari di Gomory

Osservazione 6.3

Ogni taglio frazionario di Gomory elimina la soluzione di base ammissibile frazionaria associata al dizionario del semplice da cui è generato.

Dimostrazione. Nella soluzione di base ammissibile associata al dizionario tutte le variabili fuori base valgono zero:

$$x_j = 0, \quad \forall j \in \mathcal{N}.$$

Consideriamo ora la riga $i \in \mathcal{B}$ da cui è generato il taglio frazionario di Gomory. Per definizione, $\bar{b}_i$ è frazionario. Il corrispondente taglio frazionario di Gomory è

$$\sum_{j \in \mathcal{N}} (\bar{a}_{ij} - \lfloor \bar{a}_{ij} \rfloor) x_j \geq \bar{b}_i - \lfloor \bar{b}_i \rfloor.$$

Nella soluzione di base ammissibile il membro sinistro vale zero, perché $x_j = 0$ per ogni $j \in \mathcal{N}$. D'altra parte,

$$\bar{b}_i - \lfloor \bar{b}_i \rfloor > 0,$$

perché $\bar{b}_i$ è frazionario. Quindi, nella soluzione di base ammissibile corrente, il taglio diventa

$$0 \not\geq \bar{b}_i - \lfloor \bar{b}_i \rfloor.$$

Pertanto il taglio frazionario di Gomory elimina la soluzione di base ammissibile associata al dizionario da cui è generato. $\square$

Osservazione 6.4

Ogni soluzione intera ammissibile del problema PLI soddisfa ogni taglio frazionario di Gomory generato dalle righe di un dizionario del simplesso.

Dimostrazione. Consideriamo un taglio frazionario di Gomory generato dalla seguente riga $i \in \mathcal{B}$ di un dizionario del simplesso:

$$x_i = \bar{b}_i - \sum_{j \in \mathcal{N}} \bar{a}_{ij} x_j. \quad (A)$$

Dimostriamo che ogni soluzione intera ammissibile del problema PLI che soddisfa questa riga soddisfa anche il corrispondente taglio frazionario di Gomory.

Poiché $x_j \geq 0$ per ogni $j \in \mathcal{N}$ e $\lfloor \bar{a}_{ij} \rfloor \leq \bar{a}_{ij}$, da (A) otteniamo

$$x_i \leq \bar{b}_i - \sum_{j \in \mathcal{N}} \lfloor \bar{a}_{ij} \rfloor x_j.$$

Equivalentemente,

$$x_i + \sum_{j \in \mathcal{N}} \lfloor \bar{a}_{ij} \rfloor x_j \leq \bar{b}_i.$$

Il membro sinistro è intero, perché la soluzione è intera: le variabili che compaiono nella riga assumono valori interi e i coefficienti $\lfloor \bar{a}_{ij} \rfloor$ sono interi. Quindi,

$$x_i + \sum_{j \in \mathcal{N}} \lfloor \bar{a}_{ij} \rfloor x_j \leq \lfloor \bar{b}_i \rfloor.$$

Usando (A), sostituiamo

$$x_i = \bar{b}_i - \sum_{j \in \mathcal{N}} \bar{a}_{ij} x_j$$

nella disuguaglianza precedente e otteniamo

$$\bar{b}_i - \sum_{j \in \mathcal{N}} \bar{a}_{ij} x_j + \sum_{j \in \mathcal{N}} \lfloor \bar{a}_{ij} \rfloor x_j \leq \lfloor \bar{b}_i \rfloor.$$

Riordinando i termini si ottiene

$$\sum_{j \in \mathcal{N}} (\bar{a}_{ij} - \lfloor \bar{a}_{ij} \rfloor) x_j \geq \bar{b}_i - \lfloor \bar{b}_i \rfloor.$$

Questo è esattamente il taglio frazionario di Gomory generato dalla riga i . Pertanto la soluzione intera ammissibile soddisfa questo taglio.

Poiché sia la soluzione intera ammissibile sia il taglio frazionario di Gomory sono stati scelti arbitrariamente, ogni soluzione intera ammissibile del problema PLI soddisfa ogni taglio frazionario di Gomory generato dalle righe del dizionario. $\square$

- Lo stesso ragionamento vale per il taglio frazionario di Gomory generato dalla riga della funzione obiettivo. Infatti, la riga della funzione obiettivo si può scrivere come

$$z = \bar{z} + \sum_{j \in \mathcal{N}} \bar{c}_j x_j = \bar{z} - \sum_{j \in \mathcal{N}} (-\bar{c}_j) x_j.$$

Se $\bar{z}$ è frazionario, il corrispondente taglio frazionario di Gomory è

$$\sum_{j \in \mathcal{N}} (-\bar{c}_j - \lfloor -\bar{c}_j \rfloor) x_j \geq \bar{z} - \lfloor \bar{z} \rfloor.$$

- Questo taglio elimina la soluzione di base ammissibile corrente: in quella soluzione tutte le variabili fuori base valgono zero, e quindi il membro sinistro è zero, mentre

$$\bar{z} - \lfloor \bar{z} \rfloor > 0.$$

- Il taglio è valido per ogni soluzione intera ammissibile del problema PLI. Infatti, se la funzione obiettivo originale ha coefficienti interi, z assume valori interi in ogni soluzione intera ammissibile. Inoltre,

$$z + \sum_{j \in \mathcal{N}} \lfloor -\bar{c}_j \rfloor x_j \leq \bar{z}.$$

Poiché il membro sinistro è intero, segue che

$$z + \sum_{j \in \mathcal{N}} \lfloor -\bar{c}_j \rfloor x_j \leq \lfloor \bar{z} \rfloor.$$

Sostituendo

$$z = \bar{z} + \sum_{j \in \mathcal{N}} \bar{c}_j x_j$$

e riordinando i termini si ottiene esattamente

$$\sum_{j \in \mathcal{N}} (-\bar{c}_j - \lfloor -\bar{c}_j \rfloor) x_j \geq \bar{z} - \lfloor \bar{z} \rfloor.$$

6.1.2 Inserimento dei tagli frazionari di Gomory nel dizionario del semplice

- Un taglio frazionario di Gomory si ottiene da una riga del dizionario del semplice corrente. In generale, sia i suoi coefficienti sia il suo termine noto sono frazionari.
- Per inserire il taglio nel dizionario del semplice procediamo in tre passi:
 1. sostituendo le definizioni delle variabili di scarto, riscriviamo il taglio nelle variabili originali come una disuguaglianza $\sum_{j=1}^n a_j x_j \leq b$ con a_j e b interi;
 2. lo rendiamo **primitivo**: lo dividiamo per il massimo comun divisore g di $a_1, \dots, a_n$ e arrotondiamo per difetto il termine noto,

$$\sum_{j=1}^n \frac{a_j}{g} x_j \leq \left\lfloor \frac{b}{g} \right\rfloor;$$

3. solo allora aggiungiamo la variabile di scarto non negativa $s = \lfloor b/g \rfloor - \sum_{j=1}^n (a_j/g) x_j$, e sostituiamo le righe correnti del dizionario per ottenere la riga di s .
- L'arrotondamento è valido: il membro sinistro è intero per ogni $\mathbf{x}$ intero. La variabile di scarto s è intera in ogni soluzione intera ammissibile del problema PLI.

Prima il taglio si rende primitivo, poi si aggiunge la sua variabile di scarto. Una variabile di surplus aggiunta direttamente al taglio frazionario corrisponde alla disuguaglianza prima dell'arrotondamento, che è più debole quando g non divide b .

Esempio 6.4: Taglio frazionario di Gomory

Consideriamo il problema PLI dell'Esempio 1.

- Il primo taglio frazionario di Gomory nello spazio delle variabili originali è la seguente disuguaglianza, già primitiva ($g = 1$):

$$x_2 \leq 3.$$

Aggiungendo una variabile di scarto non negativa si ottiene

$$x_2 + x_5 = 3, \quad x_5 \geq 0.$$

Dal dizionario del simplesso corrente abbiamo

$$x_2 = \frac{7}{2} - \frac{4}{5} x_3 - \frac{1}{10} x_4.$$

Sostituendo e risolvendo rispetto alla nuova variabile x_5 , la riga corrispondente del dizionario è

$$x_5 = -\frac{1}{2} + \frac{4}{5} x_3 + \frac{1}{10} x_4.$$

- Il secondo taglio frazionario di Gomory nello spazio delle variabili originali è la seguente disuguaglianza, già primitiva ($g = 1$):

$$x_1 \leq 2.$$

Aggiungendo una variabile di scarto non negativa si ottiene

$$x_1 + x_5 = 2, \quad x_5 \geq 0.$$

Dal dizionario del simplesso corrente abbiamo

$$x_1 = \frac{19}{8} - \frac{1}{4} x_2 - \frac{1}{8} x_4.$$

Sostituendo e risolvendo rispetto alla nuova variabile x_5 , la riga corrispondente del dizionario è

$$x_5 = -\frac{3}{8} + \frac{1}{4} x_2 + \frac{1}{8} x_4.$$

- Nell'Esempio 1 i coefficienti non hanno divisori comuni e non si arrotonda nulla. Nell'esempio dell'auto delle sezioni finali il secondo taglio nelle variabili originali è $6x_1 + 14x_2 \leq 55$:

con $g = 2$ la disuguaglianza primitiva è $3x_1 + 7x_2 \leq \lfloor 55/2 \rfloor = 27$, e la sua variabile di scarto è $27 - 3x_1 - 7x_2$.

6.2 Un algoritmo dei piani di taglio basato sui tagli frazionari di Gomory

Definizione 6.5: algoritmo dei piani di taglio basato sui tagli frazionari di Gomory

Un **algoritmo dei piani di taglio basato sui tagli frazionari di Gomory** è un metodo esatto per risolvere problemi PLI basato sulla risoluzione ripetuta di rilassamenti lineari. Partendo dal rilassamento lineare del problema PLI, l'algoritmo aggiunge progressivamente tagli frazionari di Gomory con l'obiettivo di eliminare le soluzioni ottime frazionarie dei problemi LP mantenendo tutte le soluzioni intere ammissibili:

↔ **operazioni di generazione dei tagli**

Dopo ogni passo di generazione dei tagli, il nuovo rilassamento lineare, più stringente, viene risolto di nuovo:

↔ **operazioni di riottimizzazione LP**

- L'algoritmo alterna generazione dei tagli e riottimizzazione LP finché non trova una soluzione intera ottima, se esiste.
- Consideriamo problemi PLI in **forma canonica di massimo con dati interi**.

6.2.1 Operazioni di generazione dei tagli

- Le **operazioni di generazione dei tagli** producono una sequenza di rilassamenti lineari sempre più stringenti:

$$LP_1, LP_2, LP_3, \dots$$

Ogni problema LP si ottiene dal precedente aggiungendo uno o più *tagli frazionari di Gomory*.

- Per $k = 1$ abbiamo il primo problema LP, indicato con LP_1 , che è il rilassamento lineare del problema PLI originale. Per $k \geq 2$, il problema LP_k si ottiene da LP_{k-1} aggiungendo i tagli frazionari di Gomory generati all'iterazione $k - 1$.

Per ogni iterazione k , se il problema LP_k ha una soluzione ottima, il suo valore ottimo si indica con $z(LP_k)$ e una soluzione ottima con $\tilde{x}(LP_k)$.

- A ogni iterazione, il rilassamento lineare corrente contiene tutte le soluzioni intere ammissibili del problema PLI originale. Di conseguenza abbiamo il seguente upper bound UB sul valore ottimo del problema PLI:

$$z(ILP) \leq \underbrace{\lfloor z(LP_k) \rfloor}_{UB}$$

Quando si aggiunge un taglio frazionario di Gomory, la regione ammissibile del rilassamento lineare si riduce o resta invariata.

- L'algoritmo può eliminare punti ammissibili per il rilassamento lineare, ma non elimina alcuna soluzione intera ammissibile del problema PLI originale. In questo senso, ogni nuovo taglio rende più stringente il rilassamento lineare corrente senza escludere alcuna soluzione intera ammissibile del problema PLI originale.

6.2.2 Pseudocodice ad alto livello

Algoritmo

Input: Un problema PLI in forma canonica di massimo con dati interi.

Output: Una soluzione ottima del problema PLI (se esiste) oppure un certificato di inammissibilità/illimitatezza.

Passo 0: Inizializzazione

Rilassare i vincoli di integralità e risolvere il problema LP_1 . Porre il contatore delle iterazioni $k \leftarrow 1$.

1. se $\tilde{x}(LP_1)$ è **intera**, è una soluzione ottima e $z(LP_1)$ è il valore ottimo del problema PLI. **STOP**.
2. se il problema LP_1 è **inammissibile**, il problema PLI è inammissibile. **STOP**.
3. se il problema LP_1 è **illimitato**, il problema PLI è inammissibile oppure illimitato. **STOP**.

Passo 1: Generazione dei tagli

- (a) Selezionare una o più righe $i \in \mathcal{B}$ del dizionario del semplice ottimo del problema LP_k con $\bar{b}_i$ frazionario e generare i corrispondenti **tagli frazionari di Gomory**:

$$\sum_{j \in \mathcal{N}} (\bar{a}_{ij} - \lfloor \bar{a}_{ij} \rfloor) x_j \geq \bar{b}_i - \lfloor \bar{b}_i \rfloor.$$

- (b) Riscrivere ogni taglio nelle variabili originali in forma primitiva, aggiungere la sua variabile di scarto e aggiungerlo al problema LP_k : si ottiene il nuovo rilassamento lineare LP_{k+1} . Aggiornare il contatore delle iterazioni $k \leftarrow k + 1$.

Passo 2: Test di ottimalità

- (a) Risolvere il problema LP_k .
- (b) Se il problema LP_k è **inammissibile**, il problema PLI è inammissibile. **STOP**.
- (c) Sia $\tilde{x}(LP_k)$ una soluzione ottima del problema LP_k e sia $z(LP_k)$ il suo valore ottimo. Se $\tilde{x}(LP_k)$ è **intera**, è una soluzione ottima del problema PLI e $z(LP_k)$ è il valore ottimo del problema PLI. **STOP**. Altrimenti, andare al Passo 1.

- Nel passo di generazione dei tagli si può generare più di un taglio frazionario di Gomory: selezionando più righe del dizionario del simplesso ottimo con $\bar{b}_i$ frazionario, oppure la riga della funzione obiettivo.
- Lo pseudocodice non dice come si riottimizzano i problemi LP dopo l'inserimento dei tagli: la sezione seguente spiega l'uso dell'algoritmo del simplesso duale.

6.2.3 Riottimizzazione con l'algoritmo del simplesso duale

- Il nuovo problema differisce dal precedente problema LP_k per un vincolo, quindi non lo risolviamo da zero: partiamo dal **dizionario ottimo** del problema LP_k e lo riottimizziamo con l'**algoritmo del simplesso duale**.
- Il taglio fornisce una nuova riga, con la variabile di scarto x_s della disuguaglianza primitiva in base. Il suo valore nella soluzione di base corrente è negativo, perché il taglio elimina questa soluzione. I costi ridotti non cambiano.

Dopo l'inserimento di un taglio il dizionario è **duale ammissibile** ma non primale ammissibile: questo è esattamente il punto di partenza dell'algoritmo del simplesso duale.

Operazione di pivot duale su un dizionario $x_i = \bar{b}_i - \sum_{j \in \mathcal{N}} \bar{a}_{ij} x_j$, per $i \in \mathcal{B}$, e $z = \bar{z} + \sum_{j \in \mathcal{N}} \bar{c}_j x_j$, con $\bar{c}_j \leq 0$ per ogni $j \in \mathcal{N}$:

- a. la **riga pivot** è quella della variabile in base con valore negativo di indice minimo:

$$r = \min \{ i \in \mathcal{B} : \bar{b}_i < 0 \};$$

- b. se $\bar{a}_{rj} \geq 0$ per ogni $j \in \mathcal{N}$, il problema LP è **inammissibile**;

- c. altrimenti la **colonna pivot** x_p realizza il rapporto minimo (pareggi: indice minimo)

$$\frac{\bar{c}_p}{\bar{a}_{rp}} = s_m = \min \left\{ \frac{\bar{c}_j}{\bar{a}_{rj}} : j \in \mathcal{N}, \bar{a}_{rj} < 0 \right\};$$

- d. dopo l'operazione di pivot il valore dell'obiettivo è $\bar{z} + s_m \bar{b}_r \leq \bar{z}$.

L'algoritmo si ferma quando tutte le variabili in base sono non negative: il dizionario è allora ottimo.

- Se il nuovo problema LP_{k+1} è inammissibile, l'algoritmo del simplesso duale lo rileva al passo b, e il problema PLI è inammissibile. Di solito bastano una o due operazioni di pivot.

Esempio 6.5: algoritmo dei piani di taglio

Descriviamo ora l'esecuzione dell'algoritmo dei piani di taglio per il problema PLI dell'[Esempio 1](#) usando:

1. un taglio frazionario di Gomory a ogni iterazione;

2. il termine noto più grande come strategia di selezione dei tagli, con le regole per i pareggi della sezione sulla selezione dei tagli (indice minimo della variabile in base, riga della funzione obiettivo per ultima);
3. ogni problema LP_k viene riottimizzato con l'algoritmo del simplesso duale.

Inizializzazione

Per il problema LP_1 in forma canonica abbiamo:

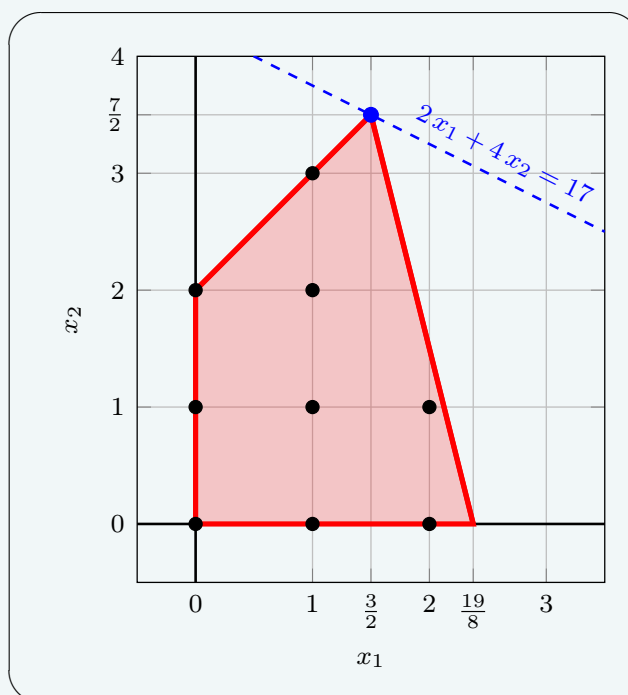
$$\begin{array}{rcll} \max & 2 & x_1 & + & 4 & x_2 \\ & - & x_1 & + & & x_2 & \leq & 2 \\ & 8 & x_1 & + & 2 & x_2 & \leq & 19 \\ & & x_1, & & x_2 & \geq & 0 \end{array}$$

Una soluzione ottima $\tilde{\mathbf{x}}(LP_1)$ è:

$$\tilde{x}_1 = \frac{3}{2}, \quad \tilde{x}_2 = \frac{7}{2}$$

Il valore ottimo è:

$$z(LP_1) = 17$$



La soluzione ottima $\tilde{\mathbf{x}}(LP_1)$ è **frazionaria** e il problema LP_1 è **ammissibile**.

Abbiamo l'upper bound sul valore ottimo del problema PLI:

$$z(\text{ILP}) \leq 17$$

Iterazione 1

a) Fase di generazione dei tagli

Consideriamo il seguente dizionario del simpleso associato a una soluzione di base ammissibile frazionaria del problema LP_1 in forma standard:

$$\begin{cases} x_1 = \frac{3}{2} + \frac{1}{5} x_3 - \frac{1}{10} x_4 \\ x_2 = \frac{7}{2} - \frac{4}{5} x_3 - \frac{1}{10} x_4 \\ z = 17 - \frac{14}{5} x_3 - \frac{3}{5} x_4 \end{cases} \quad \text{dove} \quad \bar{A} = \begin{pmatrix} -\frac{1}{5} & \frac{1}{10} \\ \frac{4}{5} & \frac{1}{10} \end{pmatrix} \quad \bar{b} = \begin{pmatrix} \frac{3}{2} \\ \frac{7}{2} \end{pmatrix}$$

- Il taglio frazionario di Gomory generato dalla prima riga è:

$$\underbrace{\left(-\frac{1}{5} - \left\lfloor -\frac{1}{5} \right\rfloor\right)}_{-\frac{1}{5} - (-1) = \frac{4}{5}} x_3 + \underbrace{\left(\frac{1}{10} - \left\lfloor \frac{1}{10} \right\rfloor\right)}_{\frac{1}{10} - 0 = \frac{1}{10}} x_4 \geq \underbrace{\left(\frac{3}{2} - \left\lfloor \frac{3}{2} \right\rfloor\right)}_{\frac{3}{2} - 1 = \frac{1}{2}}$$

$$\frac{4}{5} x_3 + \frac{1}{10} x_4 \geq \frac{1}{2} \Rightarrow 8 x_3 + x_4 \geq 5$$

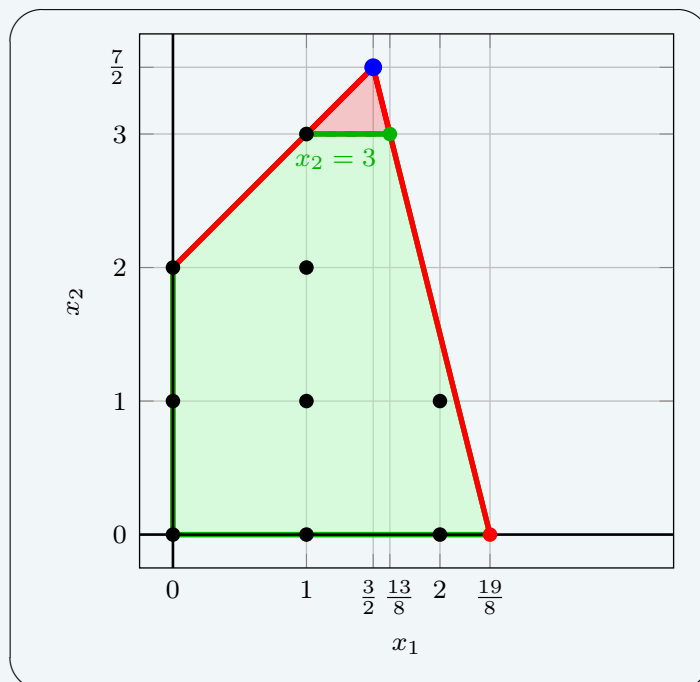
Dalla forma standard del problema LP_1 abbiamo:

$$x_3 = 2 + x_1 - x_2 \quad \text{e} \quad x_4 = 19 - 8 x_1 - 2 x_2$$

Sostituendo nel taglio frazionario di Gomory, possiamo esprimerlo nelle variabili originali del problema PLI:

$$8 (2 + x_1 - x_2) + (19 - 8 x_1 - 2 x_2) \geq 5 \Rightarrow x_2 \leq 3$$

Effetto del taglio frazionario di Gomory generato dalla prima riga sulla regione ammissibile del problema LP_1 in forma canonica:



- Il taglio frazionario di Gomory generato dalla seconda riga è:

$$\underbrace{\left(\frac{4}{5} - \left\lfloor \frac{4}{5} \right\rfloor\right)}_{\frac{4}{5} - 0 = \frac{4}{5}} x_3 + \underbrace{\left(\frac{1}{10} - \left\lfloor \frac{1}{10} \right\rfloor\right)}_{\frac{1}{10} - 0 = \frac{1}{10}} x_4 \geq \underbrace{\left(\frac{7}{2} - \left\lfloor \frac{7}{2} \right\rfloor\right)}_{\frac{7}{2} - 3 = \frac{1}{2}}$$

$$\frac{4}{5} x_3 + \frac{1}{10} x_4 \geq \frac{1}{2} \Rightarrow 8 x_3 + x_4 \geq 5$$

Questo taglio frazionario di Gomory coincide con quello generato dalla prima riga.

I due tagli coincidono e hanno il termine noto più grande, $\frac{1}{2}$. Selezioniamo quello generato dalla riga di x_1 (indice minimo) e otteniamo il problema LP_2 aggiungendolo al problema LP_1 .

b) Fase di riottimizzazione LP

Per il problema LP_2 in forma canonica abbiamo:

$$\begin{array}{rclcl} \max & 2 x_1 & + & 4 x_2 & \\ & - x_1 & + & x_2 & \leq 2 \\ & 8 x_1 & + & 2 x_2 & \leq 19 \\ & & & x_2 & \leq 3 \\ & x_1, & & x_2 & \geq 0 \end{array}$$

- Nelle variabili originali il taglio è $x_2 \leq 3$, già primitivo (i suoi coefficienti non hanno divisori comuni).
- Aggiungiamo la variabile di scarto $x_5 = 3 - x_2 \geq 0$ della disuguaglianza primitiva. Sostituendo le righe correnti del dizionario, la nuova riga è

$$x_5 = -\frac{1}{2} + \frac{4}{5}x_3 + \frac{1}{10}x_4.$$

- Il dizionario seguente è **duale ammissibile** (i costi ridotti non cambiano) ma non primale ammissibile, poiché $\bar{b}_5 = -\frac{1}{2} < 0$:

$$\left\{ \begin{array}{rclcl} x_1 & = & \frac{3}{2} & + & \frac{1}{5}x_3 & - & \frac{1}{10}x_4 \\ x_2 & = & \frac{7}{2} & - & \frac{4}{5}x_3 & - & \frac{1}{10}x_4 \\ x_5 & = & -\frac{1}{2} & + & \frac{4}{5}x_3 & + & \frac{1}{10}x_4 \\ z & = & 17 & - & \frac{14}{5}x_3 & - & \frac{3}{5}x_4 \end{array} \right.$$

- **Operazione di pivot duale.** La riga pivot è x_5 , l'unica variabile in base con valore negativo. I rapporti $\bar{c}_j/\bar{a}_{5j}$ delle variabili fuori base con $\bar{a}_{5j} < 0$ sono

$$\frac{\bar{c}_3}{\bar{a}_{5,3}} = \frac{-\frac{14}{5}}{-\frac{4}{5}} = \frac{7}{2}, \quad \frac{\bar{c}_4}{\bar{a}_{5,4}} = \frac{-\frac{3}{5}}{-\frac{1}{10}} = 6.$$

- La colonna pivot è x_3 , con $s_m = \frac{7}{2}$, e il valore dell'obiettivo diventa $\bar{z} + s_m \bar{b}_5 = 17 + \frac{7}{2} \cdot \left(-\frac{1}{2}\right) = \frac{61}{4}$. Il nuovo dizionario è primale e duale ammissibile, quindi ottimo:

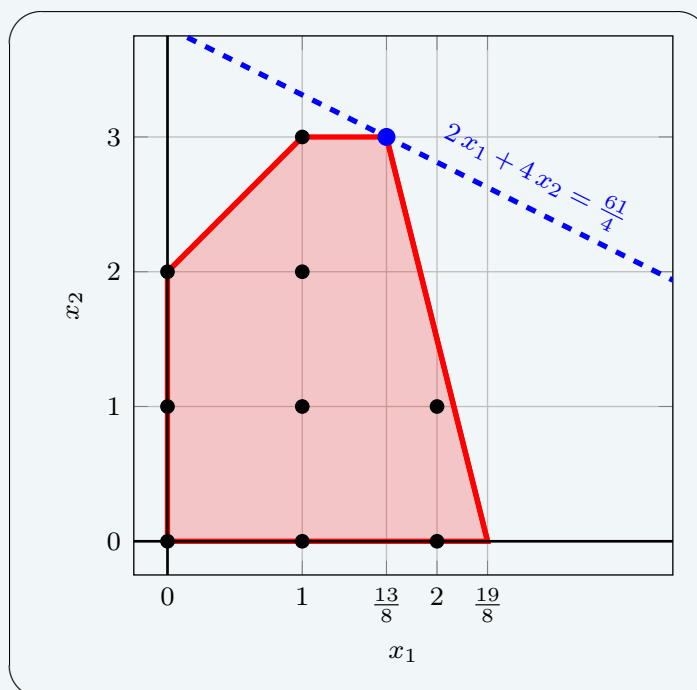
$$\left\{ \begin{array}{rclcl} x_1 & = & \frac{13}{8} & - & \frac{1}{8}x_4 & + & \frac{1}{4}x_5 \\ x_2 & = & 3 & & & - & x_5 \\ x_3 & = & \frac{5}{8} & - & \frac{1}{8}x_4 & + & \frac{5}{4}x_5 \\ z & = & \frac{61}{4} & - & \frac{1}{4}x_4 & - & \frac{7}{2}x_5 \end{array} \right.$$

Una soluzione ottima $\tilde{\mathbf{x}}(\text{LP}_2)$ è:

$$\tilde{x}_1 = \frac{13}{8}, \quad \tilde{x}_2 = 3$$

Il valore ottimo è:

$$z(\text{LP}_2) = \frac{61}{4}$$



Abbiamo l'upper bound sul valore ottimo del problema PLI:

$$z(\text{ILP}) \leq \underbrace{\left\lfloor \frac{61}{4} \right\rfloor}_{=15}$$

Iterazione 2

a) Fase di generazione dei tagli

Abbiamo il seguente dizionario del simpleso associato a una soluzione di base ammissibile frazionaria del problema LP_2 in forma standard:

$$\begin{cases} x_1 = \frac{13}{8} - \frac{1}{8}x_4 + \frac{1}{4}x_5 \\ x_2 = 3 - x_5 \\ x_3 = \frac{5}{8} - \frac{1}{8}x_4 + \frac{5}{4}x_5 \\ z = \frac{61}{4} - \frac{1}{4}x_4 - \frac{7}{2}x_5 \end{cases} \quad \text{dove} \quad \bar{A} = \begin{pmatrix} \frac{1}{8} & -\frac{1}{4} \\ 0 & 1 \\ \frac{1}{8} & -\frac{5}{4} \end{pmatrix} \quad \bar{b} = \begin{pmatrix} \frac{13}{8} \\ 3 \\ \frac{5}{8} \end{pmatrix}$$

- Il taglio frazionario di Gomory generato dalla prima riga è:

$$\underbrace{\left(\frac{1}{8} - \left\lfloor \frac{1}{8} \right\rfloor \right)}_{\frac{1}{8} - 0 = \frac{1}{8}} x_4 + \underbrace{\left(-\frac{1}{4} - \left\lfloor -\frac{1}{4} \right\rfloor \right)}_{-\frac{1}{4} - (-1) = \frac{3}{4}} x_5 \geq \underbrace{\left(\frac{13}{8} - \left\lfloor \frac{13}{8} \right\rfloor \right)}_{\frac{13}{8} - 1 = \frac{5}{8}}$$

$$\frac{1}{8} x_4 + \frac{3}{4} x_5 \geq \frac{5}{8} \Rightarrow x_4 + 6 x_5 \geq 5$$

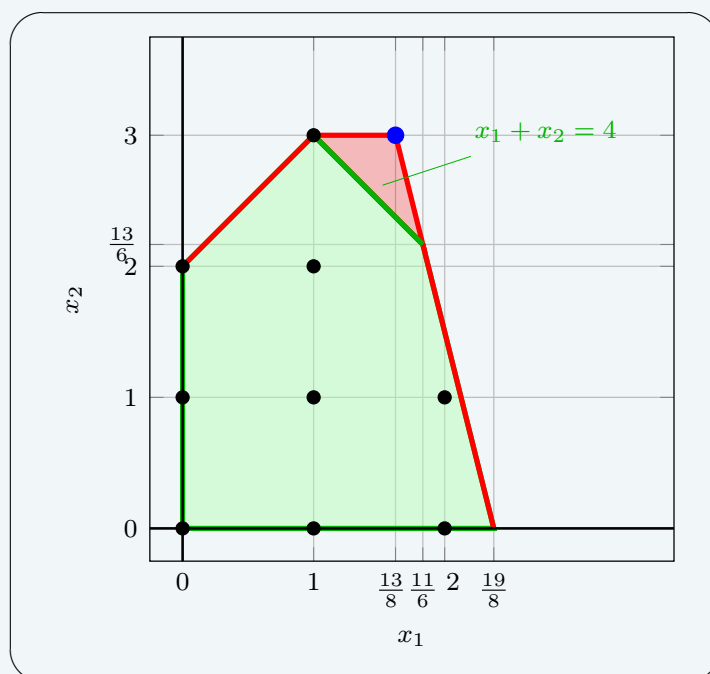
Dalla forma standard del problema LP_2 abbiamo:

$$x_4 = 19 - 8x_1 - 2x_2 \quad \text{e} \quad x_5 = 3 - x_2.$$

Sostituendo nel taglio frazionario di Gomory, possiamo esprimerlo nelle variabili originali del problema PLI:

$$(19 - 8x_1 - 2x_2) + 6(3 - x_2) \geq 5 \Rightarrow x_1 + x_2 \leq 4.$$

Effetto del taglio frazionario di Gomory generato dalla prima riga sulla regione ammissibile del problema LP_2 in forma canonica:



- Il taglio frazionario di Gomory generato dalla terza riga è:

$$\underbrace{\left(\frac{1}{8} - \left\lfloor \frac{1}{8} \right\rfloor\right)}_{\frac{1}{8} - 0 = \frac{1}{8}} x_4 + \underbrace{\left(-\frac{5}{4} - \left\lfloor -\frac{5}{4} \right\rfloor\right)}_{-\frac{5}{4} - (-2) = \frac{3}{4}} x_5 \geq \underbrace{\left(\frac{5}{8} - \left\lfloor \frac{5}{8} \right\rfloor\right)}_{\frac{5}{8} - 0 = \frac{5}{8}}$$

$$\frac{1}{8} x_4 + \frac{3}{4} x_5 \geq \frac{5}{8} \Rightarrow x_4 + 6 x_5 \geq 5$$

Questo taglio frazionario di Gomory coincide con quello generato dalla prima riga.

- Il taglio frazionario di Gomory generato dalla riga della funzione obiettivo è:

$$\underbrace{\left(\frac{1}{4} - \left\lfloor \frac{1}{4} \right\rfloor\right)}_{\frac{1}{4} - 0 = \frac{1}{4}} x_4 + \underbrace{\left(\frac{7}{2} - \left\lfloor \frac{7}{2} \right\rfloor\right)}_{\frac{7}{2} - 3 = \frac{1}{2}} x_5 \geq \underbrace{\left(\frac{61}{4} - \left\lfloor \frac{61}{4} \right\rfloor\right)}_{\frac{61}{4} - 15 = \frac{1}{4}}$$

$$\frac{1}{4} x_4 + \frac{1}{2} x_5 \geq \frac{1}{4} \Rightarrow x_4 + 2 x_5 \geq 1$$

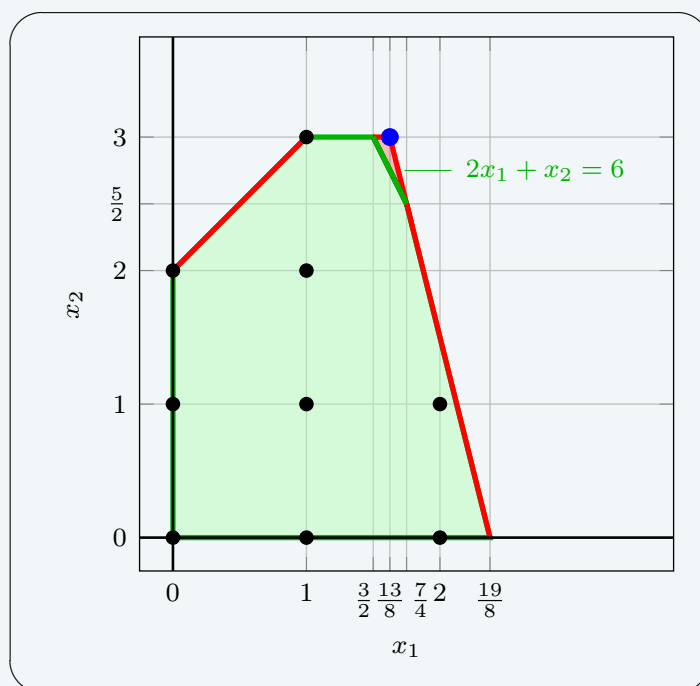
Dalla forma standard del problema LP₂ abbiamo:

$$x_4 = 19 - 8x_1 - 2x_2 \quad \text{e} \quad x_5 = 3 - x_2.$$

Sostituendo nel taglio frazionario di Gomory, possiamo esprimerlo nelle variabili originali del problema PLI:

$$(19 - 8x_1 - 2x_2) + 2(3 - x_2) \geq 1 \Rightarrow 2x_1 + x_2 \leq 6.$$

Effetto del taglio frazionario di Gomory generato dalla riga della funzione obiettivo sulla regione ammissibile del problema LP₂ in forma canonica:



I tagli delle righe di x_1 e x_3 coincidono e hanno il termine noto più grande, $\frac{5}{8}$ (la riga della funzione obiettivo dà $\frac{1}{4}$). Selezioniamo quello generato dalla riga di x_1 (indice minimo) e otteniamo il problema LP₃ aggiungendolo al problema LP₂.

b) Fase di riottimizzazione LP

Per il problema LP_3 in forma canonica abbiamo:

$$\begin{array}{rcll} \max & 2x_1 & + & 4x_2 \\ & -x_1 & + & x_2 \leq 2 \\ & 8x_1 & + & 2x_2 \leq 19 \\ & & & x_2 \leq 3 \\ & x_1 & + & x_2 \leq 4 \\ & x_1, & & x_2 \geq 0 \end{array}$$

- Nelle variabili originali il taglio è $x_1 + x_2 \leq 4$, già primitivo (i suoi coefficienti non hanno divisori comuni).
- Aggiungiamo la variabile di scarto $x_6 = 4 - (x_1 + x_2) \geq 0$ della disuguaglianza primitiva. Sostituendo le righe correnti del dizionario, la nuova riga è

$$x_6 = -\frac{5}{8} + \frac{1}{8}x_4 + \frac{3}{4}x_5.$$

- Il dizionario seguente è **duale ammissibile** (i costi ridotti non cambiano) ma non primale ammissibile, poiché $\bar{b}_6 = -\frac{5}{8} < 0$:

$$\left\{ \begin{array}{rcll} x_1 & = & \frac{13}{8} & - \frac{1}{8}x_4 + \frac{1}{4}x_5 \\ x_2 & = & 3 & - x_5 \\ x_3 & = & \frac{5}{8} & - \frac{1}{8}x_4 + \frac{5}{4}x_5 \\ x_6 & = & -\frac{5}{8} & + \frac{1}{8}x_4 + \frac{3}{4}x_5 \\ z & = & \frac{61}{4} & - \frac{1}{4}x_4 - \frac{7}{2}x_5 \end{array} \right.$$

- **Operazione di pivot duale.** La riga pivot è x_6 , l'unica variabile in base con valore negativo. I rapporti $\bar{c}_j/\bar{a}_{6j}$ delle variabili fuori base con $\bar{a}_{6j} < 0$ sono

$$\frac{\bar{c}_4}{\bar{a}_{6,4}} = \frac{-\frac{1}{4}}{-\frac{1}{8}} = 2, \quad \frac{\bar{c}_5}{\bar{a}_{6,5}} = \frac{-\frac{7}{2}}{-\frac{3}{4}} = \frac{14}{3}.$$

- La colonna pivot è x_4 , con $s_m = 2$, e il valore dell'obiettivo diventa $\bar{z} + s_m \bar{b}_6 = \frac{61}{4} + 2 \cdot \left(-\frac{5}{8}\right) = 14$. Il nuovo dizionario è primale e duale ammissibile, quindi ottimo:

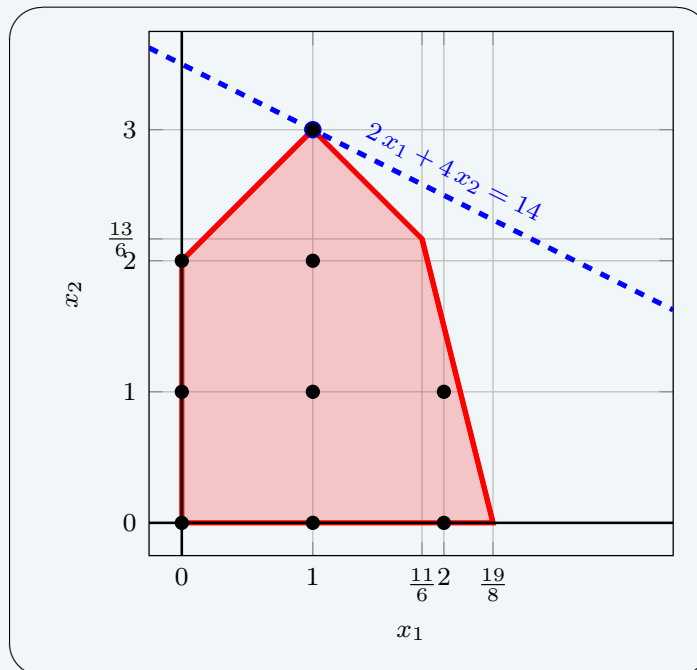
$$\begin{cases} x_1 = 1 + x_5 - x_6 \\ x_2 = 3 - x_5 \\ x_3 = 0 + 2x_5 - x_6 \\ x_4 = 5 - 6x_5 + 8x_6 \\ z = 14 - 2x_5 - 2x_6 \end{cases}$$

Una soluzione ottima $\tilde{\mathbf{x}}(\text{LP}_3)$ è:

$$\tilde{x}_1 = 1, \quad \tilde{x}_2 = 3$$

Il valore ottimo è:

$$z(\text{LP}_3) = 14$$



La soluzione ottima $\tilde{\mathbf{x}}(\text{LP}_3)$ è **intera**. Di conseguenza, una soluzione ottima $\tilde{\mathbf{x}}(\text{ILP})$ è:

$$\tilde{x}_1 = 1, \quad \tilde{x}_2 = 3$$

Il valore ottimo $z(\text{ILP})$ è 14.

6.2.4 Strategie di selezione dei tagli

- Nella fase di generazione dei tagli si possono generare più tagli frazionari di Gomory da aggiungere al rilassamento lineare. La **strategia di selezione dei tagli** decide quali.

La **strategia di selezione del termine noto più grande** seleziona il taglio frazionario di Gomory il cui termine noto frazionario è massimo fra tutti i tagli candidati.

- I tagli candidati sono generati dalle righe $i \in \mathcal{B}$ del dizionario con $\bar{b}_i$ frazionario; il termine noto del taglio della riga i è

$$\bar{b}_i - \lfloor \bar{b}_i \rfloor.$$

- Se il valore dell'obiettivo $\bar{z}$ è frazionario, anche la riga della funzione obiettivo è candidata, con termine noto $\bar{z} - \lfloor \bar{z} \rfloor$.
- **Pareggi.** Tagli con lo stesso termine noto possono essere diversi, quindi la scelta va fissata per rendere l'esecuzione riproducibile: selezioniamo la riga della variabile in base di indice minimo, e la riga della funzione obiettivo si considera per ultima.

6.2.5 Inizializzazione euristica

Prima di eseguire un algoritmo dei piani di taglio si può eseguire un **algoritmo euristico** con l'obiettivo di trovare una soluzione ammissibile di buona qualità del problema PLI. Questa soluzione euristica fornisce un lower bound iniziale LB sul valore ottimo del problema PLI.

- A ogni iterazione il valore ottimo del rilassamento lineare corrente fornisce un **upper bound**

$$UB = \lfloor z(LP_k) \rfloor$$

sul valore ottimo del problema PLI. Man mano che si aggiungono tagli validi, questo upper bound migliora progressivamente.

- Disporre di una soluzione intera ammissibile permette di monitorare il gap fra upper bound e lower bound:

$$\frac{UB - LB}{|LB|}$$

L'algoritmo dei piani di taglio si può fermare quando questo gap è zero: la soluzione euristica è allora dimostrata ottima.

6.2.6 Osservazioni aggiuntive

- La terminazione finita di un algoritmo dei piani di taglio con tagli frazionari di Gomory è garantita solo con regole precise (un algoritmo del simplesso duale lessicografico e una scelta opportuna delle righe). Con regole arbitrarie l'algoritmo può richiedere un numero molto grande di iterazioni.
- I tagli frazionari di Gomory sono di solito molto efficaci nelle prime iterazioni, ma la loro efficacia diminuisce man mano che l'algoritmo procede (**tailing off**); la sezione seguente ne mostra un esempio.
- Per questo motivo i tagli frazionari di Gomory si usano spesso per rafforzare i rilassamenti lineari risolti nei nodi di un albero di branch-and-bound: il metodo che si ottiene è un **algoritmo di branch-and-cut**.

6.3 L'esempio dell'auto

- Risolviamo l'esempio dell'auto con l'algoritmo dei piani di taglio e le stesse regole dell'Esempio 1: un taglio per iterazione, termine noto più grande, pareggi risolti con l'indice minimo della variabile in base e la riga della funzione obiettivo per ultima.
- La tabella riporta, per ogni taglio, la disuguaglianza nelle variabili originali prima dell'arrotondamento (quando $g > 1$) e in forma primitiva, la soluzione e il valore del problema LP successivo, e l'upper bound $UB = \lfloor z(LP_{k+1}) \rfloor$.

Taglio	Prima dell'arrotondamento	Taglio primitivo	$\tilde{x}(LP_{k+1})$	$z(LP_{k+1})$	UB
1	—	$5x_1 + 14x_2 \leq 52$	$(\frac{211}{60}, \frac{59}{24})$	$\frac{232}{15}$	15
2	$6x_1 + 14x_2 \leq 55$	$3x_1 + 7x_2 \leq 27$	$(\frac{207}{58}, \frac{135}{58})$	$\frac{891}{58}$	15
3	—	$4x_1 + 7x_2 \leq 30$	$(\frac{65}{18}, \frac{20}{9})$	$\frac{275}{18}$	15
4	$7x_1 + 7x_2 \leq 40$	$x_1 + x_2 \leq 5$	$(\frac{25}{6}, \frac{5}{6})$	$\frac{85}{6}$	14
5	—	$9x_1 + 4x_2 \leq 40$	$(4, 1)$	14	14

- L'algoritmo richiede 5 tagli. L'upper bound resta a 15 per i primi 3 tagli, e la soluzione intera si trova solo all'ultimo taglio.
- Nei tagli 2 e 4 i coefficienti hanno un divisore comune: il taglio primitivo è più forte della disuguaglianza prima dell'arrotondamento.
- La soluzione ottima è $\tilde{x} = (4, 1)$, di valore $z(ILP) = 14$.

6.4 Errori tipici

Attenzione

- **Usare il taglio per problemi con variabili continue.** Il taglio frazionario è valido solo se tutte le variabili e i dati sono interi: con una variabile continua può eliminare soluzioni ammissibili.
- **Prendere la parte intera inferiore di un coefficiente o di un termine noto.** Il taglio usa le parti frazionarie: $\bar{a}_{ij} - \lfloor \bar{a}_{ij} \rfloor$ è sempre in $[0, 1)$, anche per un $\bar{a}_{ij}$ negativo.
- **Dimenticare la riga della funzione obiettivo.** Dà un taglio candidato se $\bar{z}$ è frazionario; con la strategia del termine noto più grande può essere quello selezionato.
- **Aggiungere la variabile di surplus prima di rendere primitivo il taglio.** Il taglio nelle variabili originali va prima diviso per il massimo comun divisore dei suoi coefficienti, arrotondando per difetto il termine noto; solo allora si aggiunge la sua variabile di scarto. Altrimenti il taglio è più debole.
- **Scegliere fra termini noti uguali senza una regola.** Scelte diverse danno esecuzioni diverse; si usa l'indice minimo della variabile in base, e la riga della funzione obiettivo per ultima.
- **Risolvere ogni problema LP_k da zero.** Il risultato è lo stesso, ma molto più lento: il dizionario con la nuova riga è duale ammissibile, e l'algoritmo del simplesso duale parte da lì.
- **Fermarsi quando l'upper bound smette di migliorare.** L'algoritmo si ferma solo quando la soluzione del problema LP_k è intera.

6.5 In sintesi

In sintesi

- Un taglio frazionario di Gomory si genera da una riga del dizionario con $\bar{b}_i$ frazionario; elimina la soluzione di base corrente e nessuna soluzione intera ammissibile del problema PLI.
- Il taglio è valido solo per problemi PLI con dati interi, in cui le variabili di scarto e di surplus sono intere.
- Dopo ogni taglio il problema LP viene riottimizzato con l'algoritmo del simplesso duale, partendo dal dizionario del problema precedente.
- L'upper bound è $\lfloor z(LP_k) \rfloor$; l'algoritmo termina quando la soluzione del problema LP è intera.
- Ogni taglio si riscrive nelle variabili originali, si rende primitivo (diviso per il massimo comun divisore dei suoi coefficienti, termine noto arrotondato per difetto) e solo allora si aggiunge la sua variabile di scarto.

- I tagli funzionano bene all'inizio e poi perdono efficacia: nell'esempio dell'auto ne servono 5.

Cruscotto dell'esempio dell'auto. I lower bound e gli upper bound su $z(\text{ILP})$ trovati finora in questo modulo:

Capitolo	Lower bound	Upper bound	Da
Bound primali e bound duali	14	15	soluzione euristica $(4, 1)$, e $\lfloor z(\text{LP}) \rfloor$ con $z(\text{LP}) = \frac{31}{2}$
Questo capitolo	14	14	5 tagli frazionari di Gomory

Il branch-and-bound basato sulla programmazione lineare

Prerequisiti

- Dal modulo *Programmazione lineare*, [Capitolo 10 «Algoritmo del simplesso duale»](#): dizionari del simplesso, basi duali ammissibili e operazioni di pivot duali. Un breve riepilogo è dato in questo capitolo.
- Da questo modulo, il capitolo «Programmazione lineare intera»: problemi PLI, soluzioni ammissibili e ottime, e l'esempio dell'auto.
- Da questo modulo, il capitolo «Bound primali e bound duali»: il rilassamento lineare, i lower bound dalle soluzioni ammissibili, gli upper bound dal rilassamento lineare e il gap di ottimalità.

Obiettivi

Alla fine di questo capitolo sarai in grado di:

- spiegare come le operazioni di branching dividono un problema PLI in problemi LP più piccoli, e perché nessuna soluzione ammissibile viene persa;
 - applicare i tre criteri di potatura, e decidere quando il bound di un nodo si può arrotondare per difetto;
 - eseguire a mano un algoritmo branch-and-bound basato sul rilassamento lineare, riottimizzando ogni nodo con l'algoritmo del simplesso duale;
 - calcolare il lower bound, l'upper bound e il gap di ottimalità a ogni iterazione;
 - confrontare diverse strategie di selezione della variabile di branching e del nodo di branching sulla stessa istanza.
- Questo capitolo definisce l'algoritmo branch-and-bound basato sul rilassamento lineare per i problemi PLI in forma canonica di massimo con dati interi, con le sue operazioni di branching e di bounding.

- L'istanza guida è l'esempio dell'auto, $\max\{3x_1 + 2x_2 : 10x_1 + 4x_2 \leq 45, 4x_1 + 14x_2 \leq 49, x_1, x_2 \in \mathbb{Z}_{\geq 0}\}$, il cui rilassamento lineare ha la soluzione ottima frazionaria $(\frac{7}{2}, \frac{5}{2})$, con $z(\text{LP}) = \frac{31}{2}$.

7.1 L'idea in parole e in una figura

- L'algoritmo **branch-and-bound** basato sul rilassamento lineare (in breve, **algoritmo B&B**) è un metodo esatto per i problemi PLI. Si dice **basato sul rilassamento lineare** perché risolve rilassamenti lineari per guidare la ricerca di una soluzione ottima.
- Nella soluzione ottima del rilassamento lineare dell'esempio dell'auto la variabile x_1 assume un valore frazionario compreso tra 3 e 4. Ogni soluzione ammissibile del problema PLI soddisfa $x_1 \leq 3$ oppure $x_1 \geq 4$, quindi possiamo dividere il problema in due problemi LP più piccoli, uno per ciascun lato, ed eliminare la striscia intermedia, che non contiene punti interi:

↔ operazioni di branching.

- Ogni nuovo problema LP viene risolto. Il suo valore ottimo limita dall'alto il valore di ogni soluzione nella sua regione, quindi ci dice se la regione va esplorata ancora o si può scartare:

↔ operazioni di bounding.

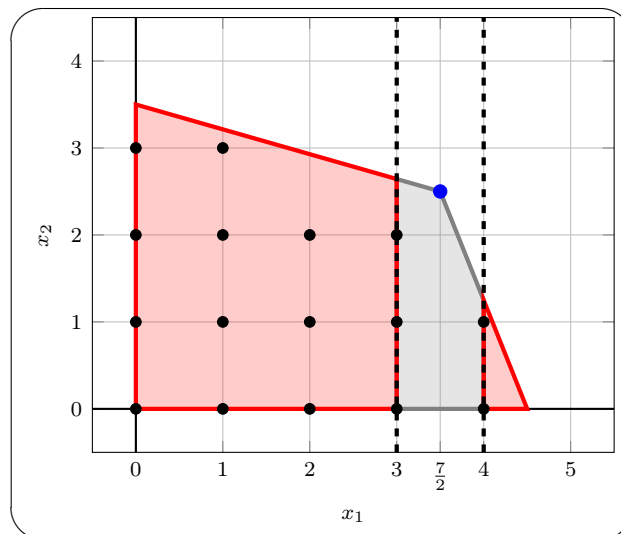


Figura 7.1: Branching su x_1 alla radice dell'esempio dell'auto: le due regioni rosse sono le regioni ammissibili dei figli ($x_1 \leq 3$ e $x_1 \geq 4$); la striscia grigia $3 < x_1 < 4$ viene tagliata via, e non contiene punti interi.

Un algoritmo branch-and-bound esplora implicitamente tutte le soluzioni ammissibili: divide la regione ammissibile in regioni più piccole (branching) e scarta ogni regione che non può contenere una soluzione migliore della migliore trovata finora (bounding).

7.2 Definizioni e notazione

7.2.1 Problemi PLI in forma canonica di massimo

- Dati n valori $c_j \in \mathbb{Z}$, $m \cdot n$ valori $a_{ij} \in \mathbb{Z}$ e m valori $b_i \in \mathbb{Z}$, un problema di **programmazione lineare intera** (PLI) in forma canonica di massimo, con n variabili intere e dati interi, si scrive:

$$\max \sum_{j=1}^n c_j x_j \quad (7.1a)$$

$$\text{soggetto a } \sum_{j=1}^n a_{ij} x_j \leq b_i, \quad \forall i \in \{1, 2, \dots, m\}, \quad (7.1b)$$

$$x_j \in \mathbb{Z}_{\geq 0}, \quad \forall j \in \{1, 2, \dots, n\}. \quad (7.1c)$$

- L'esempio dell'auto è un problema PLI con $n = 2$ e $m = 2$.
- Indichiamo con $z(\text{ILP})$ il valore ottimo del problema (7.1), e con $\tilde{\mathbf{x}}(\text{ILP})$ una soluzione ottima.
- Il **rilassamento lineare** del problema (7.1), detto problema LP_1 , si ottiene sostituendo i vincoli di integralità (7.1c) con i vincoli di non negatività $x_j \geq 0$, per ogni $j \in \{1, 2, \dots, n\}$.

7.2.2 Albero branch-and-bound, soluzione incumbent e lower bound

Le **operazioni di branching** generano un **albero branch-and-bound** che contiene problemi LP. Ogni problema LP si ottiene dal rilassamento lineare aggiungendo opportuni **vincoli di branching**.

- La struttura ricorsiva di un algoritmo B&B induce l'albero. Ogni **nodo di branching** dell'albero corrisponde a un problema LP; la radice è il problema LP_1 .

Definizione 7.1: indice del nodo di branching

Per ogni nodo di branching k , il problema LP associato si indica con LP_k . Se LP_k ha una soluzione ottima, questa si indica con $\tilde{\mathbf{x}}(\text{LP}_k)$ e il suo valore ottimo con $z(\text{LP}_k)$. L'intero k si chiama **indice del nodo di branching**: identifica il nodo in modo univoco, e i nodi sono numerati nell'ordine in cui vengono risolti i loro problemi LP.

- Un algoritmo B&B conserva la migliore soluzione ammissibile del problema (7.1) trovata durante la ricerca. Questa soluzione si chiama **soluzione incumbent** e si indica con $\bar{\mathbf{x}}$; il suo valore è un **lower bound** LB su $z(\text{ILP})$. Prima di trovare una soluzione ammissibile poniamo $\text{LB} = -\infty$.
- Il lower bound e la soluzione incumbent vengono aggiornati ogni volta che si trova una soluzione ammissibile migliore. Alla terminazione abbiamo un **certificato algoritmico** che

$$z(\text{ILP}) = \text{LB} \quad \text{e} \quad \tilde{\mathbf{x}}(\text{ILP}) = \bar{\mathbf{x}}. \quad (7.2)$$

7.2.3 Operazioni di branching

- L'operazione di branching più comune è il **branching binario**: agisce su una sola **variabile di branching** e produce due nodi di branching figli, uno con un **upper bound** aggiuntivo e uno con un **lower bound** aggiuntivo sulla variabile di branching.
- Più precisamente, sia $\tilde{x}(\text{LP}_k)$ una soluzione ottima del problema LP_k , e sia x_j , con $j \in \{1, 2, \dots, p\}$, una variabile intera che assume un valore frazionario $\tilde{x}_j$. Il branching binario su x_j genera due problemi LP figli:
 1. il problema LP_k con il **vincolo di branching verso il basso** aggiuntivo

$$x_j \leq \lfloor \tilde{x}_j \rfloor; \quad (7.3)$$

2. il problema LP_k con il **vincolo di branching verso l'alto** aggiuntivo

$$x_j \geq \lfloor \tilde{x}_j \rfloor + 1. \quad (7.4)$$

- I due nodi figli si chiamano **nodo di branching verso il basso** e **nodo di branching verso l'alto**. In entrambi $\tilde{x}(\text{LP}_k)$ non è più ammissibile.

Ogni soluzione ammissibile del problema PLI contenuta nella regione di LP_k soddisfa (7.3) oppure (7.4), quindi è contenuta nella regione di uno dei due figli. **Questo garantisce la correttezza dell'algoritmo B&B.**

7.2.4 Operazioni di bounding e bound di un nodo

- Il valore $z(\text{LP}_k)$ è un upper bound sul valore di ogni soluzione ammissibile del problema PLI nella regione di LP_k . Quando tutti i valori sono interi, questo bound si può arrotondare per difetto.

Osservazione 7.2: arrotondamento del bound di un nodo

Poiché tutte le variabili e tutti i costi sono interi, ogni soluzione ammissibile nella regione di LP_k ha valore al più $\lfloor z(\text{LP}_k) \rfloor$.

Dimostrazione. Poiché $c_j \in \mathbb{Z}$ per ogni j , il valore $\sum_{j=1}^n c_j \bar{x}_j$ di una soluzione ammissibile $\bar{x}$ è una somma di prodotti di interi, quindi un intero. Poiché è al più $z(\text{LP}_k)$, è al più il massimo intero non superiore a $z(\text{LP}_k)$, cioè $\lfloor z(\text{LP}_k) \rfloor$. □

- Di conseguenza, chiamiamo **bound del nodo k** il valore

$$\text{UB}(\text{LP}_k) = \begin{cases} \lfloor z(\text{LP}_k) \rfloor & \text{se } p = n \text{ e } c_j \in \mathbb{Z} \text{ per ogni } j \in \{1, 2, \dots, n\}, \\ z(\text{LP}_k) & \text{altrimenti.} \end{cases} \quad (7.5)$$

- Dato un nodo di branching k , su di esso non si esegue alcuna operazione di branching se vale uno dei seguenti **criteri di potatura**. In questi casi si dice che il nodo è **potato**.

1. **Potatura per inammissibilità:** il problema LP_k è inammissibile, quindi la sua regione non contiene soluzioni ammissibili del problema PLI.
2. **Potatura per integralità:** le variabili intere hanno valori interi in $\tilde{x}(LP_k)$, quindi $\tilde{x}(LP_k)$ è ammissibile per il problema PLI, ed è ottima nella regione di LP_k . Se $z(LP_k) > LB$, il lower bound e la soluzione incumbent vengono aggiornati.
3. **Potatura per bound:** il bound del nodo non supera il lower bound, $UB(LP_k) \leq LB$, quindi la regione di LP_k non contiene soluzioni migliori di quella incumbent.

I criteri si verificano in quest'ordine: inammissibilità, integralità, bound. Quando valgono sia l'integralità sia il bound, il nodo è potato per integralità.

7.2.5 La lista dei problemi LP

- Un nodo di branching che non viene potato è inserito in una **lista** L di problemi LP in attesa di un'operazione di branching.
- L'ordine in cui i problemi LP vengono estratti dalla lista determina la **strategia di selezione del nodo di branching**; la scelta della variabile di branching determina la **strategia di selezione della variabile di branching**. Entrambe sono discusse dopo l'algoritmo.

7.2.6 Upper bound e gap di ottimalità

- Se la lista L non è vuota, ogni soluzione ammissibile migliore di quella incumbent si trova nella regione di qualche nodo della lista. Di conseguenza, l'algoritmo B&B ha il **upper bound**

$$z(ILP) \leq UB = \max_{k \in L} \{UB(LP_k)\}. \quad (7.6)$$

Quando la lista è vuota, la soluzione incumbent è ottima e $UB = LB$.

- L'upper bound si calcola dopo la soluzione della radice e si aggiorna alla fine di ogni iterazione. All'algoritmo non serve, ma misura i progressi attraverso il **gap di ottimalità**:

$$\text{gap di ottimalità} = \frac{UB - LB}{|LB|}, \quad \text{per } LB \neq 0. \quad (7.7)$$

7.2.7 Inizializzazione euristica

Prima dell'algoritmo B&B si può eseguire un **algoritmo euristico** per trovare una buona soluzione ammissibile. Il lower bound LB e la soluzione incumbent $\bar{x}$ vengono allora inizializzati con questa soluzione.

- Un buon lower bound iniziale permette all'algoritmo di potare prima i nodi per bound, e può ridurre di molto la dimensione dell'albero branch-and-bound.
- Se il lower bound iniziale soddisfa già $UB(LP_1) \leq LB$, la radice è potata per bound: la soluzione euristica è ottima e non serve alcun branching.

7.3 L'algoritmo

7.3.1 Pseudocodice

Algoritmo

Input: Un problema PLI in forma canonica di massimo con dati interi.

Output: Una soluzione ottima del problema PLI (se esiste) oppure un certificato di inammissibilità/illimitatezza.

Passo 0: Inizializzazione

- (a) Rilassare i vincoli di integralità e risolvere il problema LP_1 .
 1. se $\tilde{x}(LP_1)$ è **intera**, allora è una soluzione ottima e $z(LP_1)$ è il valore ottimo del problema PLI. **STOP**.
 2. se il problema LP_1 è **inammissibile**, allora il problema PLI è inammissibile. **STOP**.
 3. se il problema LP_1 è **illimitato**, allora il problema PLI è inammissibile oppure illimitato. **STOP**.
- (b) Inserire il problema LP_1 in una **lista dei problemi LP per il branching** (indicata con L). Porre il **lower bound** $LB \leftarrow -\infty$ (oppure il valore di una soluzione euristica), e porre un contatore dei nodi di branching $c \leftarrow 1$.

Passo 1: Test di ottimalità

- (a) se la lista L è vuota e $LB \neq -\infty$, allora LB è il valore ottimo e la **soluzione incumbent** $\bar{x}(ILP)$ è una soluzione ottima del problema PLI. **STOP**.
- (b) se la lista L è vuota e $LB = -\infty$, allora il problema PLI è inammissibile. **STOP**.

Passo 2: Strategia di selezione del nodo di branching

Selezionare e rimuovere un problema LP_k dalla lista L .

Passo 3: Strategia di selezione della variabile di branching

Selezionare una variabile x_j con valore frazionario $\tilde{x}_j$ in $\tilde{x}(LP_k)$ e generare due nuovi problemi LP:

- (a) il problema $LP_{c+1} \rightsquigarrow$ il problema LP_k con il **vincolo di branching verso il basso** aggiuntivo: $x_j \leq \lfloor \tilde{x}_j \rfloor$
- (b) il problema $LP_{c+2} \rightsquigarrow$ il problema LP_k con il **vincolo di branching verso l'alto** aggiuntivo: $x_j \geq \lfloor \tilde{x}_j \rfloor + 1$

Eseguire la **Sottoprocedura di soluzione del nodo di branching** per il problema LP_{c+1} e per il problema LP_{c+2} ; la seconda si salta se, dopo la prima, $\lfloor z(LP_k) \rfloor \leq LB$. Incrementare il contatore dei nodi di branching $c \leftarrow c + 2$ e andare al Passo 1.

- Questo è lo pseudocodice della **Sottoprocedura di soluzione del nodo di branching**.

Algoritmo

Input: Un problema LP_k associato a un nodo di branching.

Passo 1: Operazioni di bounding

Risolvere il problema LP_k .

- (a) se il problema LP_k è inammissibile, allora **STOP**.
- (b) se $\lfloor z(LP_k) \rfloor \leq LB$, allora **STOP**.
- (c) se $\tilde{x}(LP_k)$ è intera, allora aggiornare il **lower bound** e la **soluzione incumbent**:

$$LB \leftarrow z(LP_k) \quad \text{e} \quad \bar{x}(ILP) \leftarrow \tilde{x}(LP_k)$$

Rimuovere dalla lista L tutti i problemi LP con $\lfloor z(LP) \rfloor \leq LB$ e **STOP**.

Passo 2: Inserimento nella lista

Inserire il problema LP_k nella lista L e **STOP**.

- Il problema LP con il vincolo di branching verso il basso si chiama **nodo di branching verso il basso**. Il problema LP con il vincolo di branching verso l'alto si chiama **nodo di branching verso l'alto**. L'ordine in cui vengono risolti determina l'ordine in cui i problemi LP entrano nella lista L .
- Lo pseudocodice non dice come si risolvono i problemi LP : si può usare qualunque metodo per i problemi LP .
- Se si usa l'**algoritmo del simplesso duale**, ogni figlio si risolve a partire dal dizionario ottimo del padre: dopo l'aggiunta del vincolo di branching questo dizionario è ancora duale ammissibile, quindi in generale bastano poche operazioni di pivot (Sezione 7.4).

7.3.2 La regola di salto e la numerazione dei nodi

- Il bound di un figlio non supera mai il bound del padre, perché il figlio ha un vincolo in più. Di conseguenza, se dopo la soluzione del primo figlio abbiamo $UB(LP_k) \leq LB$, il secondo figlio non può contenere una soluzione migliore, e **non viene risolto**. Questa è la **regola di salto**.
- Un figlio saltato non riceve un indice di nodo: i nodi sono numerati nell'ordine in cui vengono risolti i loro problemi LP, quindi gli indici dei nodi risolti sono sempre $1, 2, \dots, c$.
- L'ordine in cui vengono risolti i due figli determina anche l'ordine in cui entrano nella lista L . In queste note si risolve per primo il nodo di branching verso il basso, salvo indicazione contraria.

7.3.3 Numero di nodi di branching

- Un albero branch-and-bound binario **completo** con $n \geq 1$ livelli ha 2^{n-1} nodi di branching foglia, e lungo ogni cammino dalla radice a una foglia vengono imposti esattamente $n - 1$ vincoli di branching.

Osservazione 7.3

Il numero totale di nodi di branching in un albero branch-and-bound binario completo con $n \geq 1$ livelli è:

$$\sum_{j=0}^{n-1} 2^j = 2^n - 1. \quad (7.8)$$

Dimostrazione. Per induzione su n .

- **Caso base.** Per $n = 1$ l'enunciato diventa $\sum_{j=0}^0 2^j = 2^1 - 1$, cioè $1 = 1$, che è vero.
- **Passo induttivo.** Supponiamo che l'enunciato valga per n ; lo dimostriamo per $n + 1$. Abbiamo:

$$\begin{aligned} \sum_{j=0}^n 2^j &= \sum_{j=0}^{n-1} 2^j + 2^n = 2^n - 1 + 2^n \quad (\text{per l'ipotesi induttiva}) \\ &= 2^{n+1} - 1, \end{aligned}$$

che è l'enunciato per $n + 1$. □

- Il numero totale di nodi è la somma dei primi n termini di una progressione geometrica di ragione $r = 2$:

$$\sum_{j=0}^{n-1} r^j = \begin{cases} \frac{r^n - 1}{r - 1} & \text{se } r \neq 1, \\ n & \text{altrimenti.} \end{cases} \quad (7.9)$$

- Con p variabili binarie l'albero completo ha $p + 1$ livelli e $2^{p+1} - 1$ nodi: sono i criteri di potatura a rendere l'algoritmo utilizzabile in pratica.

7.4 Breve riepilogo dell'algoritmo del simplesso duale

- Un figlio differisce dal padre per un vincolo. Risolverlo da zero sprecherebbe il lavoro fatto per il padre: l'algoritmo B&B parte invece dal **dizionario ottimo del padre** e lo riottimizza con l'**algoritmo del simplesso duale**.
- Sia x_j la variabile di branching. È in base nel dizionario ottimo di LP_k , perché il suo valore è frazionario, e la sua riga è

$$x_j = \bar{b}_j - \sum_{q \in \mathcal{N}} \bar{a}_{jq} x_q, \quad \text{con } \bar{b}_j = \tilde{x}_j.$$

- Alla profondità d dell'albero, il vincolo di branching riceve la variabile di scarto (o di surplus) x_s , con $s = n + m + d$.

Osservazione 7.4: la riga del vincolo di branching

Sostituendo la riga di x_j , i vincoli di branching diventano nuove righe in base del dizionario:

$$x_j \leq \lfloor \tilde{x}_j \rfloor \implies x_s = \lfloor \tilde{x}_j \rfloor - x_j = (\lfloor \tilde{x}_j \rfloor - \tilde{x}_j) + \sum_{q \in \mathcal{N}} \bar{a}_{jq} x_q,$$

$$x_j \geq \lfloor \tilde{x}_j \rfloor + 1 \implies x_s = x_j - \lfloor \tilde{x}_j \rfloor - 1 = (\tilde{x}_j - \lfloor \tilde{x}_j \rfloor - 1) - \sum_{q \in \mathcal{N}} \bar{a}_{jq} x_q.$$

In entrambi i casi il valore di x_s è negativo, mentre i costi ridotti non cambiano: il nuovo dizionario è **duale ammissibile** ma non primale ammissibile.

- Questo è esattamente il punto di partenza dell'algoritmo del simplesso duale. In queste note usiamo le stesse regole di pivot del modulo *Programmazione lineare*.

Operazione di pivot duale su un dizionario $x_i = \bar{b}_i - \sum_{j \in \mathcal{N}} \bar{a}_{ij} x_j$, per $i \in \mathcal{B}$, e $z = \bar{z} + \sum_{j \in \mathcal{N}} \bar{c}_j x_j$, con $\bar{c}_j \leq 0$ per ogni $j \in \mathcal{N}$:

- la **riga pivot** è la variabile in base con valore negativo di indice minimo:

$$r = \min \{ i \in \mathcal{B} : \bar{b}_i < 0 \};$$

- se $\bar{a}_{rj} \geq 0$ per ogni $j \in \mathcal{N}$, il problema LP è **inammissibile**;
- altrimenti la **colonna pivot** x_p realizza il rapporto minimo (pareggi: indice minimo)

$$\frac{\bar{c}_p}{\bar{a}_{rp}} = s_m = \min \left\{ \frac{\bar{c}_j}{\bar{a}_{rj}} : j \in \mathcal{N}, \bar{a}_{rj} < 0 \right\};$$

- dopo l'operazione di pivot il valore obiettivo è $\bar{z} + s_m \bar{b}_r \leq \bar{z}$.

L'algoritmo si ferma quando tutte le variabili in base sono non negative: il dizionario è allora ottimo.

- In un nodo figlio bastano spesso una o due operazioni di pivot duali, mentre l'algoritmo del simplesso a partire dalla base delle variabili di scarto richiederebbe diverse operazioni di pivot per ogni nodo.

7.5 Strategie di selezione della variabile di branching

- La scelta della variabile di branching ha un grande impatto sulla dimensione dell'albero branch-and-bound.

Idealmente, la variabile di branching dà la massima diminuzione dei bound dei figli e porta rapidamente a soluzioni ammissibili. Tuttavia conta anche il tempo speso per sceglierla: una regola costosa può rallentare l'intero algoritmo.

- In tutte le strategie che seguono, i pareggi si risolvono con l'**indice minimo**.

7.5.1 Lessicografica

La strategia **lessicografica** seleziona, tra le variabili intere con valore frazionario, quella di indice minimo (o massimo).

- Non richiede calcoli, ma non tiene conto né della diminuzione dei bound né della ricerca di soluzioni ammissibili. È la regola usata negli esercizi, perché rende l'esecuzione facile da riprodurre.

7.5.2 Minima e massima inammissibilità

La strategia di **minima** (rispettivamente, **massima**) **inammissibilità** seleziona la variabile il cui valore frazionario è il più vicino (rispettivamente, il più lontano) a un valore intero.

- Dato un nodo di branching k , per ogni variabile intera x_j con valore frazionario $\tilde{x}_j$ in $\tilde{x}(\text{LP}_k)$, il **valore frazionario** della variabile è

$$f_j = \min \{ \tilde{x}_j - \lfloor \tilde{x}_j \rfloor, \lceil \tilde{x}_j \rceil - \tilde{x}_j \}. \quad (7.10)$$

- La minima inammissibilità cerca di ridurre il numero di variabili frazionarie nei figli, per raggiungere più in fretta soluzioni ammissibili.
- La massima inammissibilità elimina per prime le variabili che contribuiscono di più all'inammissibilità di $\tilde{x}(\text{LP}_k)$. Nessuna delle due regole guarda i bound dei figli.

Esempio 7.1: valori frazionari

Al nodo radice dell'esempio dell'auto abbiamo $\tilde{x}_1 = \frac{7}{2}$, $\tilde{x}_2 = \frac{5}{2}$, e i valori frazionari sono:

$$f_1 = \min \left\{ \frac{7}{2} - 3, 4 - \frac{7}{2} \right\} = \frac{1}{2} \quad f_2 = \min \left\{ \frac{5}{2} - 2, 3 - \frac{5}{2} \right\} = \frac{1}{2}$$

I due valori sono uguali: sia la minima sia la massima inammissibilità hanno un pareggio, che si risolve con l'indice minimo, quindi entrambe le strategie selezionano x_1 .

7.5.3 Strong branching

La strategia di **strong branching** seleziona la variabile che dà la massima diminuzione dei bound dei figli.

- Dato un nodo di branching k , per ogni variabile intera x_j con valore frazionario risolviamo i due figli e calcoliamo le diminuzioni del valore ottimo

$$\delta_k^d(x_j) = z(\text{LP}_k) - z(\text{figlio verso il basso}), \quad \delta_k^u(x_j) = z(\text{LP}_k) - z(\text{figlio verso l'alto}),$$

dove la diminuzione vale $+\infty$ per un figlio inammissibile.

- La variabile di branching è x_s , con

$$s = \arg \max \left\{ \min \left\{ \delta_k^d(x_j), \delta_k^u(x_j) \right\} : j \in \{1, 2, \dots, p\}, \tilde{x}_j \text{ frazionario} \right\}. \quad (7.11)$$

Entrambi i figli della variabile selezionata hanno una grande diminuzione, quindi entrambi hanno più probabilità di essere potati per bound.

- Lo strong branching è costoso, perché risolve due problemi LP per ogni variabile candidata. Per questo di solito si applica solo ai livelli più alti dell'albero.

Esempio 7.2: strong branching

Al nodo radice dell'esempio dell'auto abbiamo $\tilde{x}_1 = \frac{7}{2}$, $\tilde{x}_2 = \frac{5}{2}$ e $z(\text{LP}_1) = \frac{31}{2}$. Per ogni variabile frazionaria risolviamo i due problemi LP figli (con l'algoritmo del simplesso duale, a partire dal dizionario ottimo di LP_1):

Variabile	Verso il basso	z	$\delta_1^d(x_j)$	Verso l'alto	z	$\delta_1^u(x_j)$	$\min\{\delta_1^d, \delta_1^u\}$
x_1	$x_1 \leq 3$	$\frac{100}{7}$	$\frac{17}{14}$	$x_1 \geq 4$	$\frac{29}{2}$	1	1
x_2	$x_2 \leq 2$	$\frac{151}{10}$	$\frac{2}{5}$	$x_2 \geq 3$	$\frac{45}{4}$	$\frac{17}{4}$	$\frac{2}{5}$

La massima diminuzione minima è 1: lo strong branching seleziona x_1 , la stessa variabile scelta dalla regola dell'indice minimo.

7.5.4 Altre strategie

- I solver stimano le diminuzioni δ_k^d e δ_k^u dalla storia della ricerca, senza risolvere i figli, per mezzo degli **pseudocosti**; le stesse stime danno una regola di selezione del nodo di branching detta **best estimate**. Queste strategie sono una possibile estensione di questo capitolo.

7.6 Strategie di selezione del nodo di branching

- L'ordine in cui si elaborano i nodi della lista L è la **strategia di selezione del nodo di branching**: decide dove prosegue la ricerca.

7.6.1 Best bound

La strategia **best bound** seleziona il nodo della lista L con il massimo valore ottimo $z(LP_k)$. I pareggi si risolvono con l'indice di nodo minimo.

- Il nodo selezionato dà l'upper bound (7.6): elaborarlo è l'unico modo per far scendere UB. La regola tende a esplorare pochi nodi, ma la lista può diventare lunga.

7.6.2 LIFO e FIFO

La strategia **Last-In-First-Out** (LIFO) elabora i nodi nell'ordine inverso di inserimento nella lista L . È equivalente a una **ricerca in profondità** (DFS).

- La lista è realizzata come una **pila**. La ricerca scende in profondità nell'albero, dove le foglie contengono spesso soluzioni ammissibili: lo scopo è aggiornare rapidamente il lower bound e la soluzione incumbent. La lista resta corta.

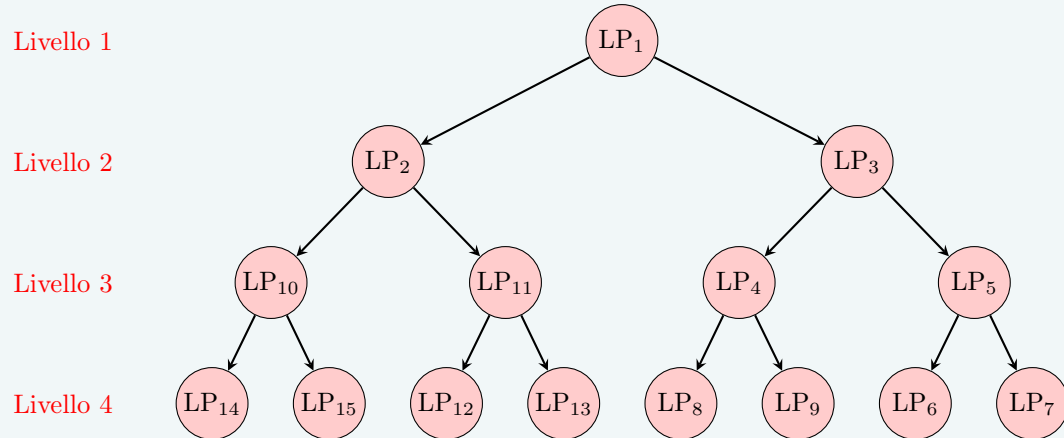
La strategia **First-In-First-Out** (FIFO) elabora i nodi nell'ordine di inserimento nella lista L . È equivalente a una **ricerca in ampiezza** (BFS).

- La lista è realizzata come una **coda**. L'albero viene esplorato livello per livello: questo migliora l'upper bound, ma di solito non trova rapidamente buoni lower bound.

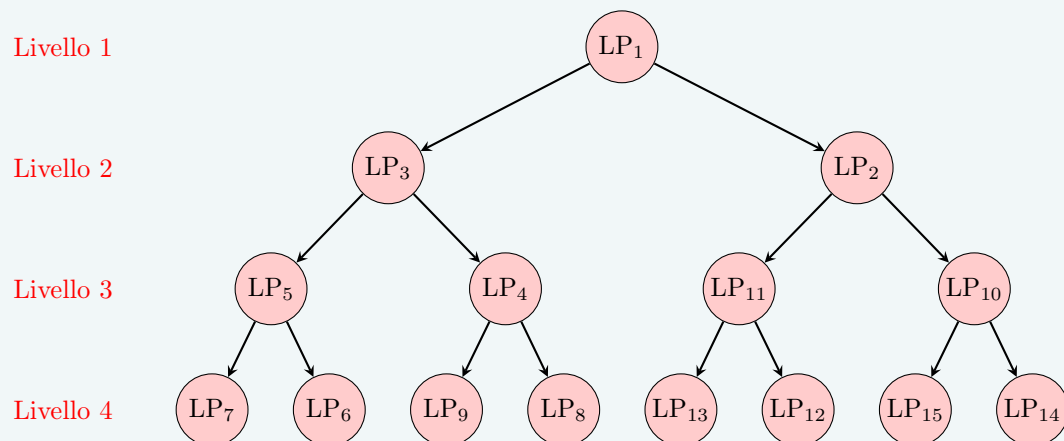
Esempio 7.3: strategia LIFO

Ordine in cui un albero branch-and-bound binario completo di 4 livelli viene esplorato con la strategia LIFO (nessun nodo viene potato).

1. Se si risolve per primo il nodo di branching verso il basso, abbiamo:



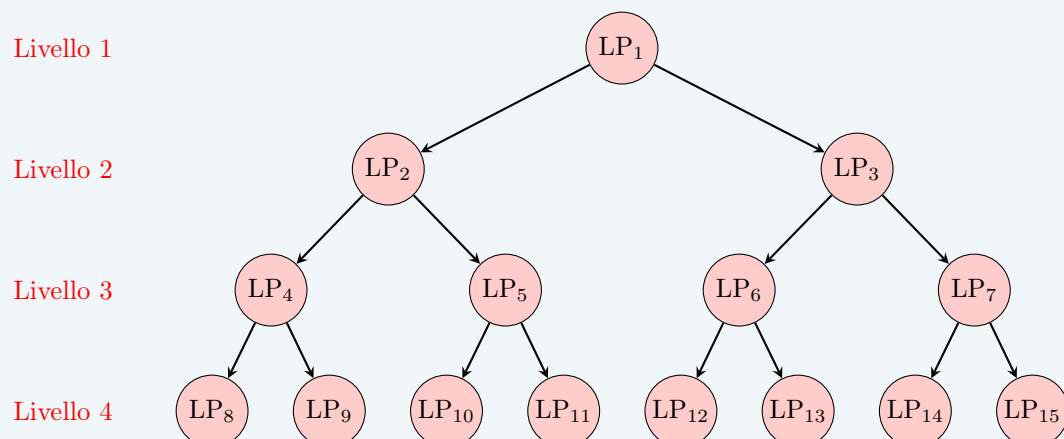
2. Se si risolve per primo il nodo di branching verso l'alto, abbiamo:



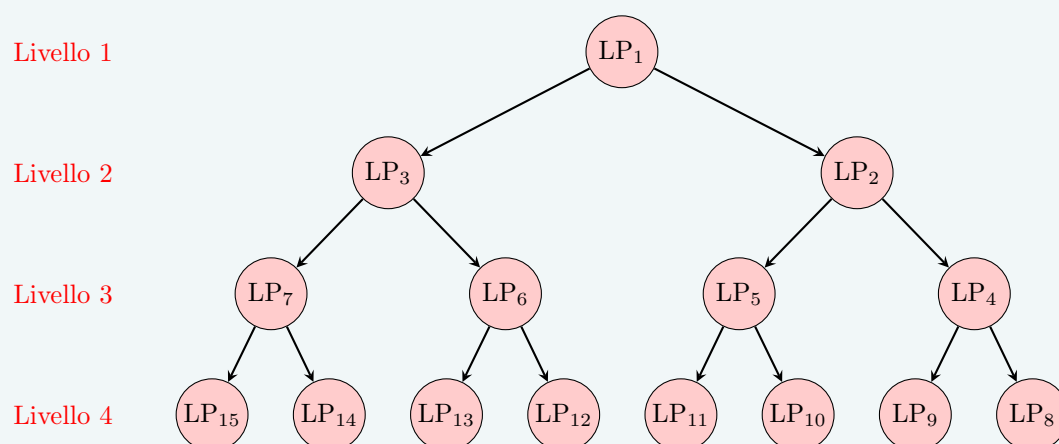
Esempio 7.4: strategia FIFO

Ordine in cui un albero branch-and-bound binario completo di 4 livelli viene esplorato con la strategia FIFO (nessun nodo viene potato).

1. Se si risolve per primo il nodo di branching verso il basso, abbiamo:



2. Se si risolve per primo il nodo di branching verso l'alto, abbiamo:



7.7 Esempi svolti sull'esempio dell'auto

- Risolviamo ora l'esempio dell'auto con due esecuzioni dell'algoritmo B&B, che differiscono solo per le strategie di selezione. Poiché tutte le variabili e tutti i costi sono interi, il bound di un nodo è $UB(LP_k) = \lfloor z(LP_k) \rfloor$.
- Nella prima esecuzione ogni nodo è riottimizzato con l'algoritmo del semplice duale, e riportiamo tutti i dizionari. Per ogni nodo una figura mostra la sua regione ammissibile.
- Prima di leggere la soluzione, prova a prevedere quale variabile viene selezionata alla radice, e quale nodo viene estratto alla seconda iterazione.

Esempio 7.5: esecuzione A: indice minimo e LIFO

Eseguiamo l'algoritmo B&B sull'esempio dell'auto con:

1. la regola dell'indice minimo (lessicografica) come strategia di selezione della variabile di branching;
2. la regola LIFO come strategia di selezione del nodo di branching;
3. si risolve prima il nodo di branching verso il basso, poi il nodo di branching verso l'alto;
4. il secondo figlio non viene risolto quando si applica la regola di salto;
5. il lower bound è inizializzato a $LB = -\infty$ (non si esegue alcun algoritmo euristico).

• Inizializzazione

Il problema LP_1 è il rilassamento lineare dell'esempio dell'auto. L'algoritmo del semplice con la regola di Bland, a partire dalla base delle variabili di scarto, esegue 2 operazioni di pivot (x_1 entra in base e x_3 esce; x_2 entra in base e x_4 esce) e termina con il dizionario ottimo:

$$\begin{cases} x_1 = \frac{7}{2} - \frac{7}{62}x_3 + \frac{1}{31}x_4 \\ x_2 = \frac{5}{2} + \frac{1}{31}x_3 - \frac{5}{62}x_4 \\ z = \frac{31}{2} - \frac{17}{62}x_3 - \frac{2}{31}x_4 \end{cases}$$

Una soluzione ottima $\tilde{\mathbf{x}}(\text{LP}_1)$ è:

$$\tilde{x}_1 = \frac{7}{2}, \quad \tilde{x}_2 = \frac{5}{2}, \quad z(\text{LP}_1) = \frac{31}{2}.$$

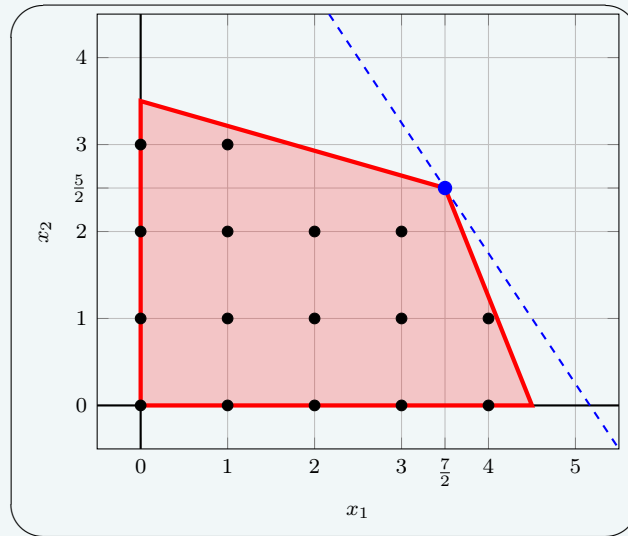


Figura 7.2: Nodo 1: la regione ammissibile di LP_1 , i punti interi, la soluzione ottima di LP_1 (in blu) e la retta dell'obiettivo che passa per essa (tratteggiata).

La soluzione è frazionaria e $\lfloor z(\text{LP}_1) \rfloor = \left\lfloor \frac{31}{2} \right\rfloor = 15 > \text{LB} = -\infty$: il problema LP_1 è inserito nella lista L .

La lista è $L = \{\text{LP}_1\}$ e l'upper bound è il massimo dei bound dei nodi della lista:

$$\underbrace{-\infty}_{\text{LB}} \leq z(\text{ILP}) \leq \underbrace{15}_{\text{UB}} \quad \left(\text{UB} = \max_{k \in L} \text{UB}(\text{LP}_k) = 15 \right).$$

• Iterazione 1

La lista è $L = \{\text{LP}_1\}$: il problema LP_1 viene estratto e rimosso da L .

In $\tilde{\mathbf{x}}(\text{LP}_1)$ abbiamo $\tilde{x}_1 = \frac{7}{2}$ e $\tilde{x}_2 = \frac{5}{2}$, entrambe frazionarie; per la regola dell'indice minimo la variabile di branching è x_1 . I vincoli di branching sono:

$$x_1 \leq \underbrace{\left\lfloor \frac{7}{2} \right\rfloor}_{=3} \quad \text{e} \quad x_1 \geq \underbrace{\left\lfloor \frac{7}{2} \right\rfloor + 1}_{=4}.$$

1. **Nodo 2: il figlio verso il basso** ($x_1 \leq 3$).

Il vincolo di branching $x_1 \leq 3$ si scrive $x_1 + x_5 = 3$, con la variabile di scarto $x_5 \geq 0$. Usando la riga di x_1 nel dizionario ottimo di LP_1 , abbiamo:

$$x_5 = 3 - x_1 = 3 - \left(\frac{7}{2} - \frac{7}{62}x_3 + \frac{1}{31}x_4 \right) = -\frac{1}{2} + \frac{7}{62}x_3 - \frac{1}{31}x_4.$$

Aggiungiamo questa riga al dizionario ottimo di LP_1 . I costi ridotti non cambiano, quindi il nuovo dizionario è **duale ammissibile**; non è primale ammissibile, perché $\bar{b}_5 = -\frac{1}{2} < 0$:

$$\begin{cases} x_1 &= \frac{7}{2} - \frac{7}{62}x_3 + \frac{1}{31}x_4 \\ x_2 &= \frac{5}{2} + \frac{1}{31}x_3 - \frac{5}{62}x_4 \\ x_5 &= -\frac{1}{2} + \frac{7}{62}x_3 - \frac{1}{31}x_4 \\ z &= \frac{31}{2} - \frac{17}{62}x_3 - \frac{2}{31}x_4 \end{cases}$$

Operazione di pivot duale. La riga pivot è x_5 , l'unica variabile in base con valore negativo ($\bar{b}_5 = -\frac{1}{2}$).

Le candidate per la colonna pivot sono le variabili fuori base con $\bar{a}_{5,j} < 0$:

$$\frac{\bar{c}_3}{\bar{a}_{5,3}} = \frac{-\frac{17}{62}}{-\frac{7}{62}} = \frac{17}{7}$$

La colonna pivot è x_3 , con $s_m = \frac{17}{7}$, e il valore obiettivo diventa

$$\bar{z} + s_m \bar{b}_5 = \frac{31}{2} + \frac{17}{7} \cdot \left(-\frac{1}{2} \right) = \frac{100}{7}.$$

Dopo l'operazione di pivot otteniamo il seguente dizionario, che è primale e duale ammissibile, quindi **ottimo**:

$$\begin{cases} x_1 &= 3 & & - & x_5 \\ x_2 &= \frac{37}{14} - \frac{1}{14}x_4 + \frac{2}{7}x_5 \\ x_3 &= \frac{31}{7} + \frac{2}{7}x_4 + \frac{62}{7}x_5 \\ z &= \frac{100}{7} - \frac{1}{7}x_4 - \frac{17}{7}x_5 \end{cases}$$

Una soluzione ottima $\tilde{\mathbf{x}}(LP_2)$ è:

$$\tilde{x}_1 = 3, \quad \tilde{x}_2 = \frac{37}{14}, \quad z(LP_2) = \frac{100}{7}.$$

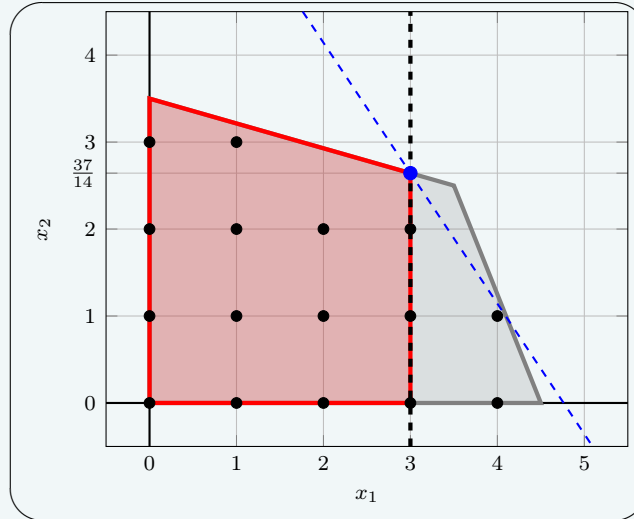


Figura 7.3: Nodo 2: la regione ammissibile di LP_2 (in rosso) dentro la regione del padre LP_1 (in grigio), e la soluzione ottima di LP_2 (in blu) con la retta dell'obiettivo che passa per essa (tratteggiata).

La soluzione è **frazionaria** e

$$\lfloor z(LP_2) \rfloor = \left\lfloor \frac{100}{7} \right\rfloor = 14 > LB = -\infty,$$

quindi il problema LP_2 è **inserito nella lista L** .

2. **Nodo 3: il figlio verso l'alto** ($x_1 \geq 4$).

Il vincolo di branching $x_1 \geq 4$ si scrive $x_1 - x_5 = 4$, con la variabile di surplus $x_5 \geq 0$. Usando la riga di x_1 nel dizionario ottimo di LP_1 , abbiamo:

$$x_5 = x_1 - 4 = \left(\frac{7}{2} - \frac{7}{62}x_3 + \frac{1}{31}x_4 \right) - 4 = -\frac{1}{2} - \frac{7}{62}x_3 + \frac{1}{31}x_4.$$

Aggiungiamo questa riga al dizionario ottimo di LP_1 . I costi ridotti non cambiano, quindi il nuovo dizionario è **duale ammissibile**; non è primale ammissibile, perché $\bar{b}_5 = -\frac{1}{2} < 0$:

$$\begin{cases} x_1 &= \frac{7}{2} - \frac{7}{62}x_3 + \frac{1}{31}x_4 \\ x_2 &= \frac{5}{2} + \frac{1}{31}x_3 - \frac{5}{62}x_4 \\ x_5 &= -\frac{1}{2} - \frac{7}{62}x_3 + \frac{1}{31}x_4 \\ z &= \frac{31}{2} - \frac{17}{62}x_3 - \frac{2}{31}x_4 \end{cases}$$

Operazione di pivot duale. La riga pivot è x_5 , l'unica variabile in base con valore negativo ($\bar{b}_5 = -\frac{1}{2}$).

Le candidate per la colonna pivot sono le variabili fuori base con $\bar{a}_{5,j} < 0$:

$$\frac{\bar{c}_4}{\bar{a}_{5,4}} = \frac{-\frac{2}{31}}{-\frac{1}{31}} = 2$$

La colonna pivot è x_4 , con $s_m = 2$, e il valore obiettivo diventa

$$\bar{z} + s_m \bar{b}_5 = \frac{31}{2} + 2 \cdot \left(-\frac{1}{2}\right) = \frac{29}{2}.$$

Dopo l'operazione di pivot otteniamo il seguente dizionario, che è primale e duale ammissibile, quindi **ottimo**:

$$\begin{cases} x_1 = 4 & + & x_5 \\ x_2 = \frac{5}{4} - \frac{1}{4}x_3 - \frac{5}{2}x_5 \\ x_4 = \frac{31}{2} + \frac{7}{2}x_3 + 31x_5 \\ z = \frac{29}{2} - \frac{1}{2}x_3 - 2x_5 \end{cases}$$

Una soluzione ottima $\tilde{\mathbf{x}}(\text{LP}_3)$ è:

$$\tilde{x}_1 = 4, \quad \tilde{x}_2 = \frac{5}{4}, \quad z(\text{LP}_3) = \frac{29}{2}.$$

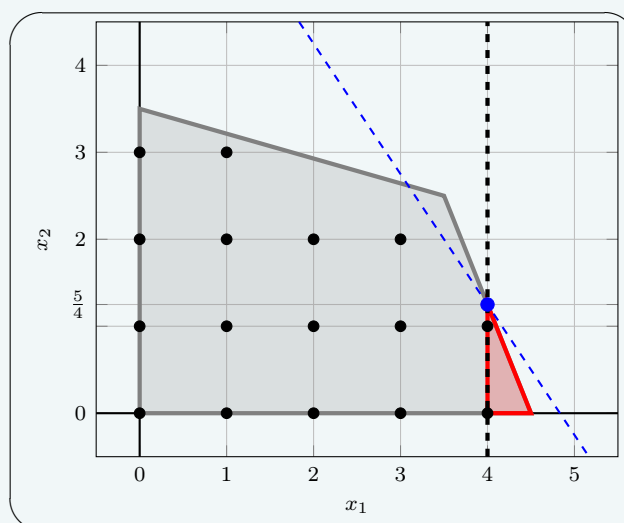


Figura 7.4: Nodo 3: la regione ammissibile di LP_3 (in rosso) dentro la regione del padre LP_1 (in grigio), e la soluzione ottima di LP_3 (in blu) con la retta dell'obiettivo che passa per essa (tratteggiata).

La soluzione è **frazionaria** e

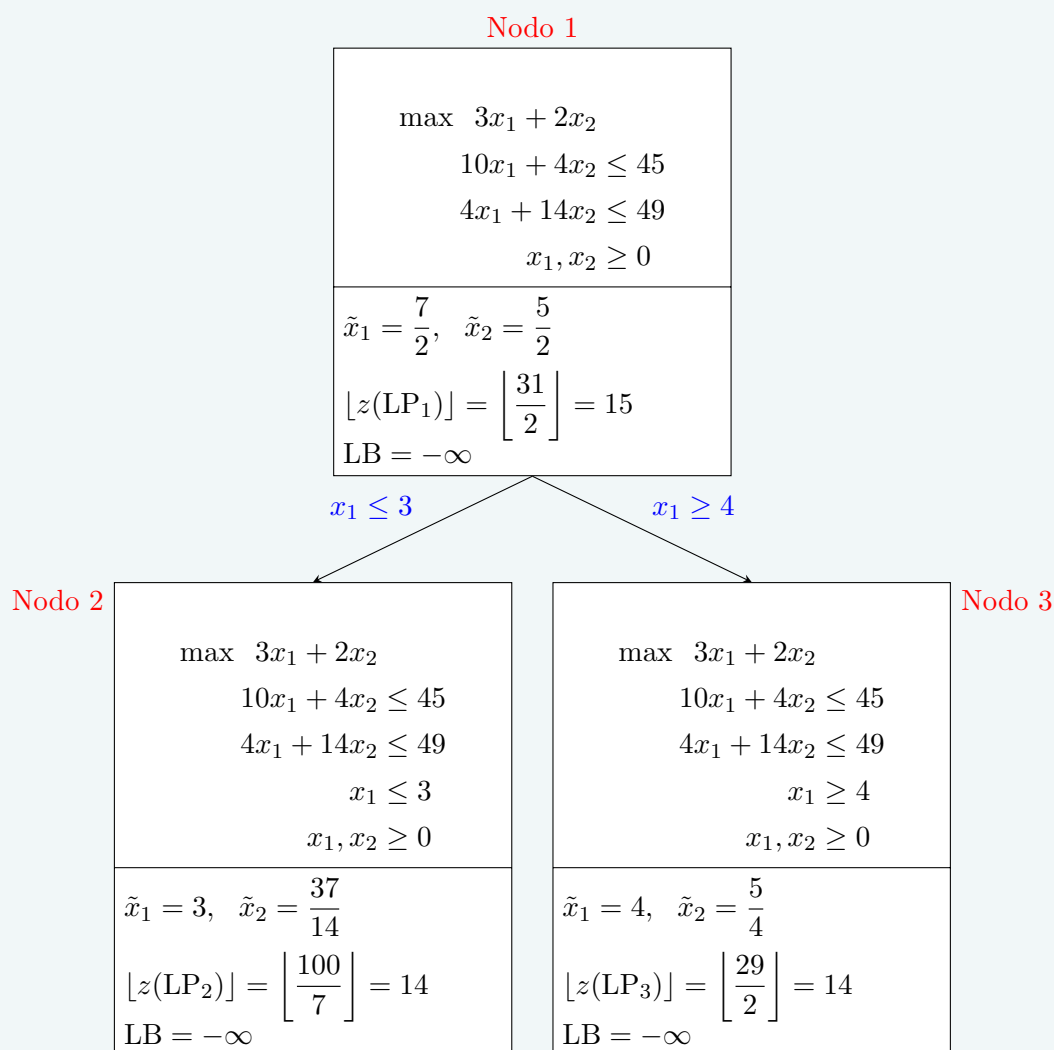
$$\lfloor z(\text{LP}_3) \rfloor = \left\lfloor \frac{29}{2} \right\rfloor = 14 > \text{LB} = -\infty,$$

quindi il problema LP_3 è **inserito nella lista L** .

La lista è $L = \{LP_2, LP_3\}$ e l'upper bound è il massimo dei bound dei nodi della lista:

$$\underbrace{-\infty}_{LB} \leq z(ILP) \leq \underbrace{14}_{UB} \quad \left(UB = \max_{k \in L} UB(LP_k) = 14 \right).$$

Questa è la parte dell'albero branch-and-bound costruita nella prima iterazione, con i problemi LP dei nodi di branching:



• Iterazione 2

La lista è $L = \{LP_2, LP_3\}$; seguendo la regola LIFO, LP_3 (l'ultimo problema inserito) viene estratto e rimosso da L .

In $\tilde{x}(LP_3)$ l'unica variabile intera con valore frazionario è x_2 , che è la variabile di branching. I vincoli di branching sono:

$$x_2 \leq \underbrace{\left\lfloor \frac{5}{4} \right\rfloor}_{=1} \quad \text{e} \quad x_2 \geq \underbrace{\left\lfloor \frac{5}{4} \right\rfloor + 1}_{=2}.$$

1. **Nodo 4: il figlio verso il basso** ($x_2 \leq 1$).

Il vincolo di branching $x_2 \leq 1$ si scrive $x_2 + x_6 = 1$, con la variabile di scarto $x_6 \geq 0$. Usando la riga di x_2 nel dizionario ottimo di LP_3 , abbiamo:

$$x_6 = 1 - x_2 = 1 - \left(\frac{5}{4} - \frac{1}{4} x_3 - \frac{5}{2} x_5 \right) = -\frac{1}{4} + \frac{1}{4} x_3 + \frac{5}{2} x_5.$$

Aggiungiamo questa riga al dizionario ottimo di LP_3 . I costi ridotti non cambiano, quindi il nuovo dizionario è **duale ammissibile**; non è primale ammissibile, perché $\bar{b}_6 = -\frac{1}{4} < 0$:

$$\left\{ \begin{array}{rclcl} x_1 & = & 4 & + & x_5 \\ x_2 & = & \frac{5}{4} & - & \frac{1}{4} x_3 - \frac{5}{2} x_5 \\ x_4 & = & \frac{31}{2} & + & \frac{7}{2} x_3 + 31 x_5 \\ x_6 & = & -\frac{1}{4} & + & \frac{1}{4} x_3 + \frac{5}{2} x_5 \\ z & = & \frac{29}{2} & - & \frac{1}{2} x_3 - 2 x_5 \end{array} \right.$$

Operazione di pivot duale. La riga pivot è x_6 , l'unica variabile in base con valore negativo ($\bar{b}_6 = -\frac{1}{4}$).

Le candidate per la colonna pivot sono le variabili fuori base con $\bar{a}_{6,j} < 0$:

$$\frac{\bar{c}_3}{\bar{a}_{6,3}} = \frac{-\frac{1}{2}}{-\frac{1}{4}} = 2, \quad \frac{\bar{c}_5}{\bar{a}_{6,5}} = \frac{-2}{-\frac{5}{2}} = \frac{4}{5}$$

$$s_m = \min \left\{ 2, \frac{4}{5} \right\} = \frac{4}{5}.$$

La colonna pivot è x_5 , con $s_m = \frac{4}{5}$, e il valore obiettivo diventa

$$\bar{z} + s_m \bar{b}_6 = \frac{29}{2} + \frac{4}{5} \cdot \left(-\frac{1}{4} \right) = \frac{143}{10}.$$

Dopo l'operazione di pivot otteniamo il seguente dizionario, che è primale e duale ammissibile, quindi **ottimo**:

$$\left\{ \begin{array}{rclcl} x_1 & = & \frac{41}{10} & - & \frac{1}{10} x_3 + \frac{2}{5} x_6 \\ x_2 & = & 1 & & - x_6 \\ x_4 & = & \frac{93}{5} & + & \frac{2}{5} x_3 + \frac{62}{5} x_6 \\ x_5 & = & \frac{1}{10} & - & \frac{1}{10} x_3 + \frac{2}{5} x_6 \\ z & = & \frac{143}{10} & - & \frac{3}{10} x_3 - \frac{4}{5} x_6 \end{array} \right.$$

Una soluzione ottima $\tilde{x}(\text{LP}_4)$ è:

$$\tilde{x}_1 = \frac{41}{10}, \quad \tilde{x}_2 = 1, \quad z(\text{LP}_4) = \frac{143}{10}.$$

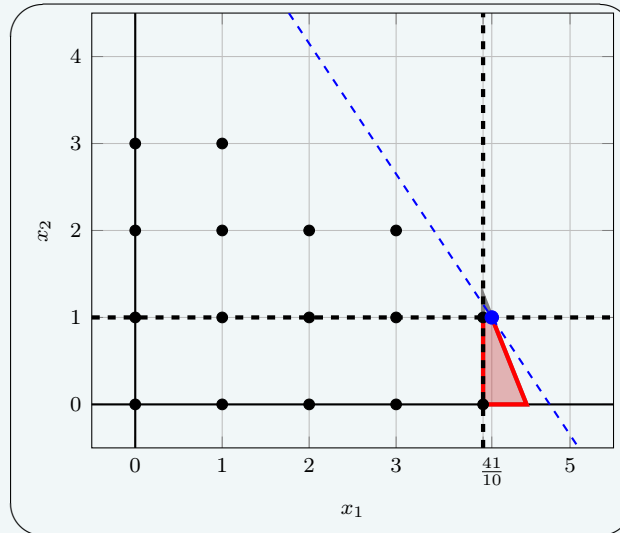


Figura 7.5: Nodo 4: la regione ammissibile di LP_4 (in rosso) dentro la regione del padre LP_3 (in grigio), e la soluzione ottima di LP_4 (in blu) con la retta dell'obiettivo che passa per essa (tratteggiata).

La soluzione è **frazionaria** e

$$\lfloor z(\text{LP}_4) \rfloor = \left\lfloor \frac{143}{10} \right\rfloor = 14 > \text{LB} = -\infty,$$

quindi il problema LP_4 è **inserito nella lista L** .

2. **Nodo 5: il figlio verso l'alto** ($x_2 \geq 2$).

Il vincolo di branching $x_2 \geq 2$ si scrive $x_2 - x_6 = 2$, con la variabile di surplus $x_6 \geq 0$. Usando la riga di x_2 nel dizionario ottimo di LP_3 , abbiamo:

$$x_6 = x_2 - 2 = \left(\frac{5}{4} - \frac{1}{4}x_3 - \frac{5}{2}x_5 \right) - 2 = -\frac{3}{4} - \frac{1}{4}x_3 - \frac{5}{2}x_5.$$

Aggiungiamo questa riga al dizionario ottimo di LP_3 . I costi ridotti non cambiano, quindi il nuovo dizionario è **duale ammissibile**; non è primale ammissibile, perché $\bar{b}_6 = -\frac{3}{4} < 0$:

$$\left\{ \begin{array}{lclcl} x_1 & = & 4 & + & x_5 \\ x_2 & = & \frac{5}{4} - \frac{1}{4}x_3 - \frac{5}{2}x_5 \\ x_4 & = & \frac{31}{2} + \frac{7}{2}x_3 + 31x_5 \\ x_6 & = & -\frac{3}{4} - \frac{1}{4}x_3 - \frac{5}{2}x_5 \\ z & = & \frac{29}{2} - \frac{1}{2}x_3 - 2x_5 \end{array} \right.$$

Operazione di pivot duale. La riga pivot è x_6 , l'unica variabile in base con valore negativo ($\bar{b}_6 = -\frac{3}{4}$).

Tutti i coefficienti della riga pivot sono non negativi ($\bar{a}_{6,3} = \frac{1}{4}$, $\bar{a}_{6,5} = \frac{5}{2}$), cioè la riga è

$$x_6 = -\frac{3}{4} - \frac{1}{4}x_3 - \frac{5}{2}x_5 \leq -\frac{3}{4} < 0$$

per ogni valore non negativo delle variabili fuori base. Per l'algoritmo del simplesso duale, il problema LP_5 è **inammissibile**.

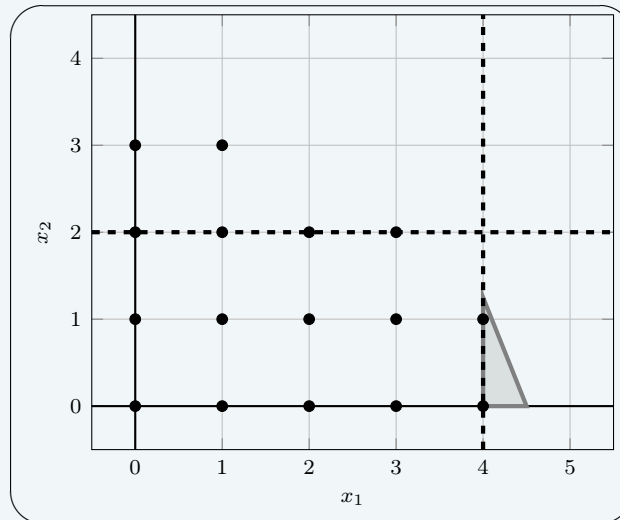


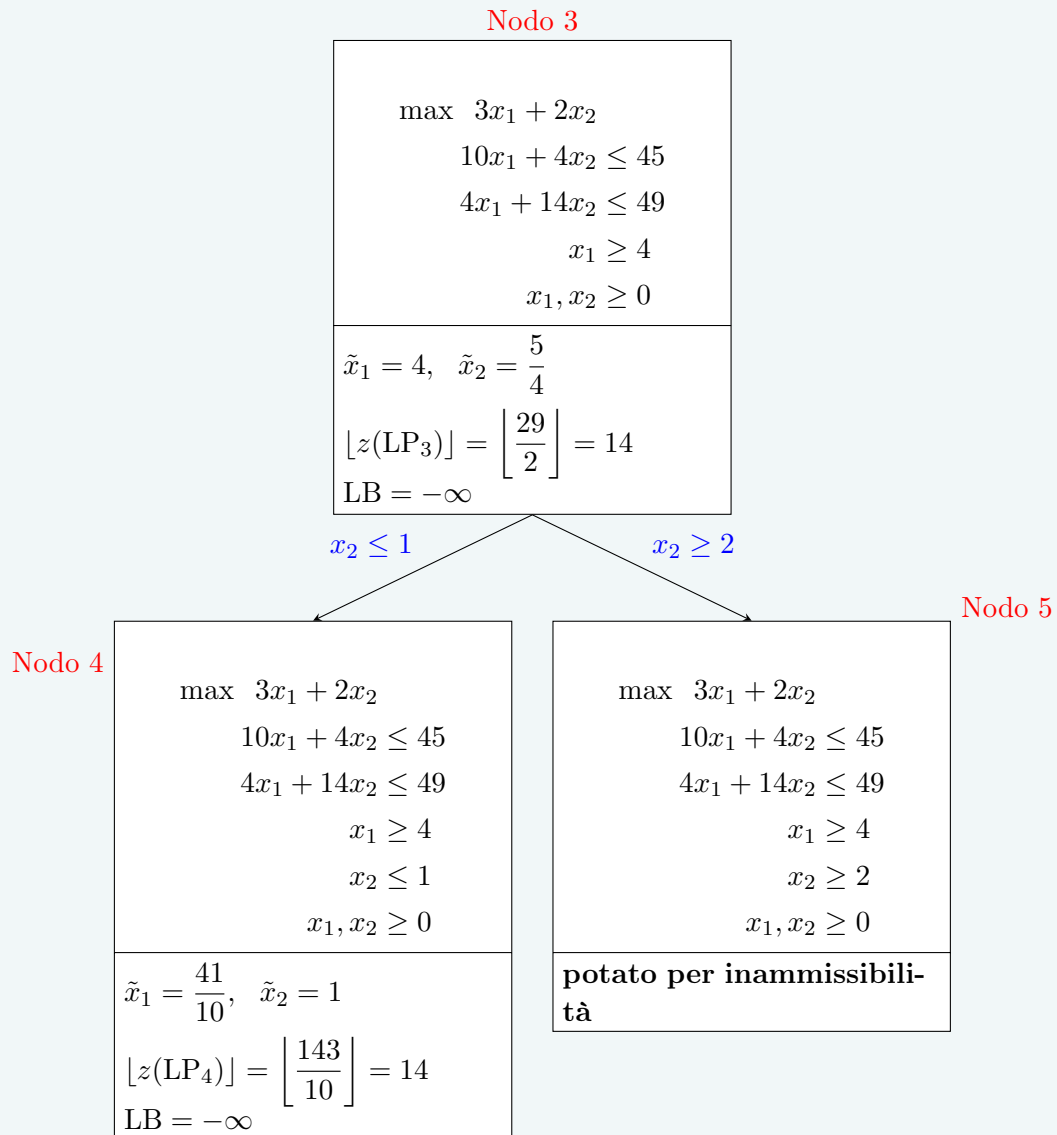
Figura 7.6: Nodo 5: il vincolo $x_2 \geq 2$ non lascia alcun punto della regione del padre LP_3 (in grigio): LP_5 è inammissibile.

Di conseguenza, il Nodo 5 è potato per inammissibilità.

La lista è $L = \{LP_2, LP_4\}$ e l'upper bound è il massimo dei bound dei nodi della lista:

$$\underbrace{-\infty}_{LB} \leq z(ILP) \leq \underbrace{14}_{UB} \quad \left(UB = \max_{k \in L} UB(LP_k) = 14 \right).$$

Questa è la parte dell'albero branch-and-bound costruita nella seconda iterazione, con i problemi LP dei nodi di branching:



• **Iterazione 3**

La lista è $L = \{\text{LP}_2, \text{LP}_4\}$; seguendo la regola LIFO, LP_4 (l'ultimo problema inserito) viene estratto e rimosso da L .

In $\tilde{x}(\text{LP}_4)$ l'unica variabile intera con valore frazionario è x_1 , che è la variabile di branching. I vincoli di branching sono:

$$x_1 \leq \underbrace{\left\lfloor \frac{41}{10} \right\rfloor}_{=4} \quad \text{e} \quad x_1 \geq \underbrace{\left\lfloor \frac{41}{10} \right\rfloor + 1}_{=5}.$$

1. **Nodo 6: il figlio verso il basso ($x_1 \leq 4$).**

Il vincolo di branching $x_1 \leq 4$ si scrive $x_1 + x_7 = 4$, con la variabile di scarto $x_7 \geq 0$. Usando la riga di x_1 nel dizionario ottimo di LP_4 , abbiamo:

$$x_7 = 4 - x_1 = 4 - \left(\frac{41}{10} - \frac{1}{10} x_3 + \frac{2}{5} x_6 \right) = -\frac{1}{10} + \frac{1}{10} x_3 - \frac{2}{5} x_6.$$

Aggiungiamo questa riga al dizionario ottimo di LP_4 . I costi ridotti non cambiano, quindi il nuovo dizionario è **duale ammissibile**; non è primale ammissibile, perché $\bar{b}_7 = -\frac{1}{10} < 0$:

$$\left\{ \begin{array}{rclcl} x_1 & = & \frac{41}{10} & - & \frac{1}{10} x_3 & + & \frac{2}{5} x_6 \\ x_2 & = & 1 & & & - & x_6 \\ x_4 & = & \frac{93}{5} & + & \frac{2}{5} x_3 & + & \frac{62}{5} x_6 \\ x_5 & = & \frac{1}{10} & - & \frac{1}{10} x_3 & + & \frac{2}{5} x_6 \\ x_7 & = & -\frac{1}{10} & + & \frac{1}{10} x_3 & - & \frac{2}{5} x_6 \\ z & = & \frac{143}{10} & - & \frac{3}{10} x_3 & - & \frac{4}{5} x_6 \end{array} \right.$$

Operazione di pivot duale. La riga pivot è x_7 , l'unica variabile in base con valore negativo ($\bar{b}_7 = -\frac{1}{10}$).

Le candidate per la colonna pivot sono le variabili fuori base con $\bar{a}_{7,j} < 0$:

$$\frac{\bar{c}_3}{\bar{a}_{7,3}} = \frac{-\frac{3}{10}}{-\frac{1}{10}} = 3$$

La colonna pivot è x_3 , con $s_m = 3$, e il valore obiettivo diventa

$$\bar{z} + s_m \bar{b}_7 = \frac{143}{10} + 3 \cdot \left(-\frac{1}{10} \right) = 14.$$

Dopo l'operazione di pivot otteniamo il seguente dizionario, che è primale e duale ammissibile, quindi **ottimo**:

$$\left\{ \begin{array}{rclcl} x_1 & = & 4 & & - & x_7 \\ x_2 & = & 1 & - & & x_6 \\ x_3 & = & 1 & + & 4 x_6 & + & 10 x_7 \\ x_4 & = & 19 & + & 14 x_6 & + & 4 x_7 \\ x_5 & = & 0 & & & - & x_7 \\ z & = & 14 & - & 2 x_6 & - & 3 x_7 \end{array} \right.$$

Una soluzione ottima $\tilde{\mathbf{x}}(LP_6)$ è:

$$\tilde{x}_1 = 4, \quad \tilde{x}_2 = 1, \quad z(LP_6) = 14.$$

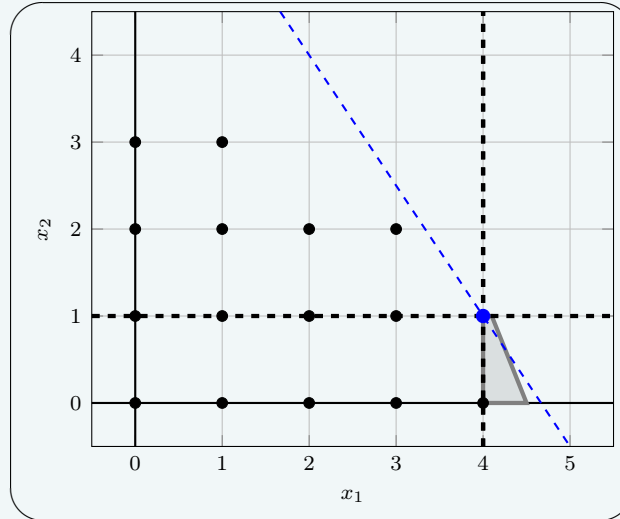


Figura 7.7: Nodo 6: la regione ammissibile di LP_6 (in rosso) dentro la regione del padre LP_4 (in grigio), e la soluzione ottima di LP_6 (in blu) con la retta dell'obiettivo che passa per essa (tratteggiata).

La soluzione $\tilde{\mathbf{x}}(LP_6)$ è **intera**.

Poiché $z(LP_6) = 14 > LB = -\infty$, il lower bound e la soluzione incumbent vengono aggiornati:

$$LB \leftarrow 14, \quad \bar{\mathbf{x}} \leftarrow (4, 1).$$

Il Nodo 6 è potato per integralità.

Inoltre, LP_2 viene rimosso dalla lista L (potato per bound), perché ora

$$\lfloor z(LP_2) \rfloor = \left\lfloor \frac{100}{7} \right\rfloor = 14 \leq LB = 14.$$

2. **Regola di salto.** Dopo il Nodo 6 il lower bound soddisfa

$$\lfloor z(LP_4) \rfloor = \left\lfloor \frac{143}{10} \right\rfloor = 14 \leq LB = 14,$$

quindi nessuna soluzione nella regione di LP_4 può migliorare la soluzione incumbent: il figlio verso l'alto ($x_1 \geq 5$) **non viene risolto** e non riceve un indice di nodo.

La lista è vuota, quindi l'upper bound è uguale al lower bound:

$$\underbrace{14}_{LB} \leq z(ILP) \leq \underbrace{14}_{UB} \quad (UB = LB = 14).$$

Questa è la parte dell'albero branch-and-bound costruita nella terza iterazione, con i problemi LP dei nodi di branching:

Nodo 4

$$\begin{aligned}
 \max \quad & 3x_1 + 2x_2 \\
 & 10x_1 + 4x_2 \leq 45 \\
 & 4x_1 + 14x_2 \leq 49 \\
 & x_1 \geq 4 \\
 & x_2 \leq 1 \\
 & x_1, x_2 \geq 0
 \end{aligned}$$

$$\tilde{x}_1 = \frac{41}{10}, \quad \tilde{x}_2 = 1$$

$$\lfloor z(\text{LP}_4) \rfloor = \left\lfloor \frac{143}{10} \right\rfloor = 14$$

$$\text{LB} = -\infty$$

$$x_1 \leq 4$$

Nodo 6

$$\begin{aligned}
 \max \quad & 3x_1 + 2x_2 \\
 & 10x_1 + 4x_2 \leq 45 \\
 & 4x_1 + 14x_2 \leq 49 \\
 & x_1 \geq 4 \\
 & x_2 \leq 1 \\
 & x_1 \leq 4 \\
 & x_1, x_2 \geq 0
 \end{aligned}$$

$$\tilde{x}_1 = 4, \quad \tilde{x}_2 = 1$$

$$z(\text{LP}_6) = 14$$

$$\text{LB} = 14$$

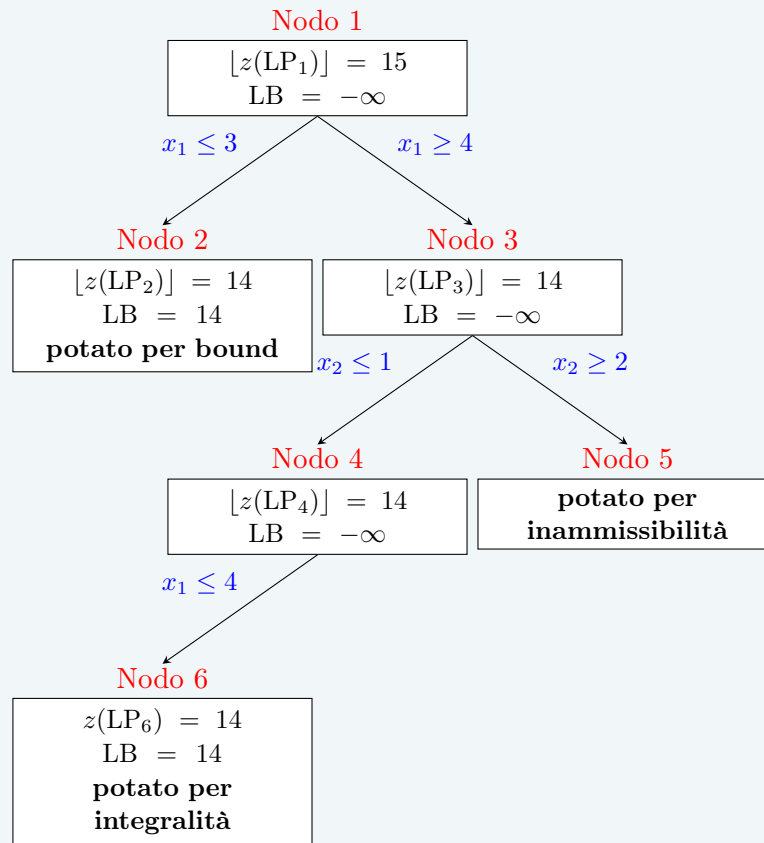
potato per integralità

- Terminazione**

La lista L è vuota: $\text{LB} = \text{UB} = 14$ e la soluzione incumbent è ottima:

$$\tilde{x}_1 = 4, \quad \tilde{x}_2 = 1, \quad z(\text{ILP}) = 14.$$

Questo è l'albero branch-and-bound finale. Per ogni nodo di branching mostra il bound del nodo, il lower bound LB al momento in cui il nodo è stato risolto (o rimosso dalla lista) e il criterio di potatura.



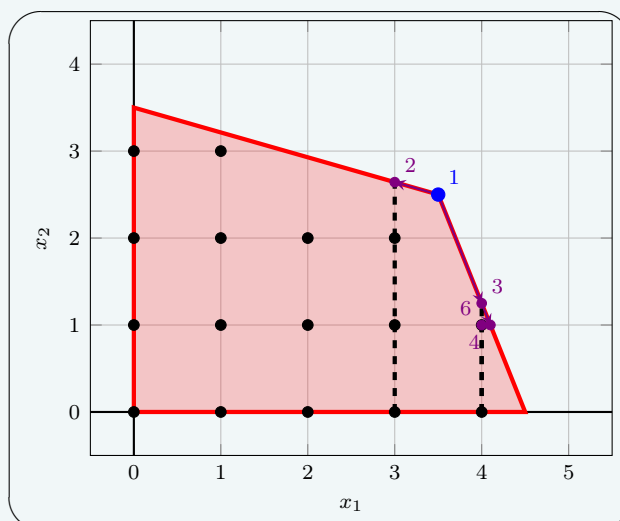


Figura 7.8: L'esecuzione A nel piano: i vincoli di branching (in nero, tratteggiati) dentro le regioni dei nodi padre, e le soluzioni ottime dei nodi risolti (in viola), a partire da LP_1 (in blu).

Esempio 7.6: esecuzione B: indice massimo e best bound

Eseguiamo di nuovo l'algoritmo B&B sull'esempio dell'auto, ora con:

1. la regola dell'indice massimo (lessicografica) come strategia di selezione della variabile di branching;
2. la regola best bound come strategia di selezione del nodo di branching (pareggi: indice di nodo minimo);
3. si risolve prima il nodo di branching verso il basso, poi il nodo di branching verso l'alto;
4. il secondo figlio non viene risolto quando si applica la regola di salto;
5. il lower bound è inizializzato a $LB = -\infty$ (non si esegue alcun algoritmo euristico).

Ogni figlio è riottimizzato con l'algoritmo del simplesso duale a partire dal dizionario ottimo del padre, esattamente come nell'Esempio 7.5; qui riportiamo solo i risultati.

• Inizializzazione

Come prima, $\tilde{\mathbf{x}}(LP_1) = (\frac{7}{2}, \frac{5}{2})$ con $z(LP_1) = \frac{31}{2}$, e LP_1 è inserito nella lista L .

La lista è $L = \{LP_1\}$ e l'upper bound è il massimo dei bound dei nodi della lista:

$$\underbrace{-\infty}_{LB} \leq z(ILP) \leq \underbrace{15}_{UB} \quad \left(UB = \max_{k \in L} UB(LP_k) = 15 \right).$$

• Iterazione 1

La lista è $L = \{LP_1\}$: il problema LP_1 viene estratto e rimosso da L .

In $\tilde{\mathbf{x}}(\text{LP}_1)$ abbiamo $\tilde{x}_1 = \frac{7}{2}$ e $\tilde{x}_2 = \frac{5}{2}$, entrambe frazionarie; per la regola dell'indice massimo la variabile di branching è x_2 . I vincoli di branching sono:

$$x_2 \leq \underbrace{\left\lfloor \frac{5}{2} \right\rfloor}_{=2} \quad \text{e} \quad x_2 \geq \underbrace{\left\lfloor \frac{5}{2} \right\rfloor + 1}_{=3}.$$

- Nodo 2 ($x_2 \leq 2$): $\tilde{\mathbf{x}}(\text{LP}_2) = (\frac{37}{10}, 2)$ con $z(\text{LP}_2) = \frac{151}{10}$. La soluzione è **frazionaria** e

$$\lfloor z(\text{LP}_2) \rfloor = \left\lfloor \frac{151}{10} \right\rfloor = 15 > \text{LB} = -\infty,$$

quindi il problema LP_2 è **inserito nella lista L** .

- Nodo 3 ($x_2 \geq 3$): $\tilde{\mathbf{x}}(\text{LP}_3) = (\frac{7}{4}, 3)$ con $z(\text{LP}_3) = \frac{45}{4}$. La soluzione è **frazionaria** e

$$\lfloor z(\text{LP}_3) \rfloor = \left\lfloor \frac{45}{4} \right\rfloor = 11 > \text{LB} = -\infty,$$

quindi il problema LP_3 è **inserito nella lista L** .

La lista è $L = \{\text{LP}_2, \text{LP}_3\}$ e l'upper bound è il massimo dei bound dei nodi della lista:

$$\underbrace{-\infty}_{\text{LB}} \leq z(\text{ILP}) \leq \underbrace{15}_{\text{UB}} \quad \left(\text{UB} = \max_{k \in L} \text{UB}(\text{LP}_k) = 15 \right).$$

• Iterazione 2

La lista è $L = \{\text{LP}_2, \text{LP}_3\}$; seguendo la regola best bound, LP_2 viene estratto e rimosso da L , perché ha il massimo valore ottimo:

$$z(\text{LP}_2) = \frac{151}{10}, \quad z(\text{LP}_3) = \frac{45}{4}.$$

In $\tilde{\mathbf{x}}(\text{LP}_2)$ l'unica variabile intera con valore frazionario è x_1 , che è la variabile di branching. I vincoli di branching sono:

$$x_1 \leq \underbrace{\left\lfloor \frac{37}{10} \right\rfloor}_{=3} \quad \text{e} \quad x_1 \geq \underbrace{\left\lfloor \frac{37}{10} \right\rfloor + 1}_{=4}.$$

- Nodo 4 ($x_1 \leq 3$): $\tilde{\mathbf{x}}(\text{LP}_4) = (3, 2)$ con $z(\text{LP}_4) = 13$. La soluzione $\tilde{\mathbf{x}}(\text{LP}_4)$ è **intera**. Poiché $z(\text{LP}_4) = 13 > \text{LB} = -\infty$, il lower bound e la soluzione incumbent vengono aggiornati:

$$\text{LB} \leftarrow 13, \quad \bar{\mathbf{x}} \leftarrow (3, 2).$$

Il Nodo 4 è potato per integralità. Inoltre, LP_3 viene rimosso dalla lista L (potato per bound), perché ora

$$\lfloor z(\text{LP}_3) \rfloor = \left\lfloor \frac{45}{4} \right\rfloor = 11 \leq \text{LB} = 13.$$

- Nodo 5 ($x_1 \geq 4$): $\tilde{\mathbf{x}}(\text{LP}_5) = (4, \frac{5}{4})$ con $z(\text{LP}_5) = \frac{29}{2}$. La soluzione è **frazionaria** e

$$\lfloor z(\text{LP}_5) \rfloor = \left\lfloor \frac{29}{2} \right\rfloor = 14 > \text{LB} = 13,$$

quindi il problema LP_5 è **inserito nella lista L** .

La lista è $L = \{\text{LP}_5\}$ e l'upper bound è il massimo dei bound dei nodi della lista:

$$\underbrace{13}_{\text{LB}} \leq z(\text{ILP}) \leq \underbrace{14}_{\text{UB}} \quad \left(\text{UB} = \max_{k \in L} \text{UB}(\text{LP}_k) = 14 \right).$$

• Iterazione 3

La lista è $L = \{\text{LP}_5\}$: il problema LP_5 viene estratto e rimosso da L .

In $\tilde{\mathbf{x}}(\text{LP}_5)$ l'unica variabile intera con valore frazionario è x_2 , che è la variabile di branching. I vincoli di branching sono:

$$x_2 \leq \underbrace{\left\lfloor \frac{5}{4} \right\rfloor}_{=1} \quad \text{e} \quad x_2 \geq \underbrace{\left\lfloor \frac{5}{4} \right\rfloor + 1}_{=2}.$$

- Nodo 6 ($x_2 \leq 1$): $\tilde{\mathbf{x}}(\text{LP}_6) = (\frac{41}{10}, 1)$ con $z(\text{LP}_6) = \frac{143}{10}$. La soluzione è **frazionaria** e

$$\lfloor z(\text{LP}_6) \rfloor = \left\lfloor \frac{143}{10} \right\rfloor = 14 > \text{LB} = 13,$$

quindi il problema LP_6 è **inserito nella lista L** .

- Nodo 7 ($x_2 \geq 2$): il problema LP_7 è inammissibile: potato per inammissibilità.

La lista è $L = \{\text{LP}_6\}$ e l'upper bound è il massimo dei bound dei nodi della lista:

$$\underbrace{13}_{\text{LB}} \leq z(\text{ILP}) \leq \underbrace{14}_{\text{UB}} \quad \left(\text{UB} = \max_{k \in L} \text{UB}(\text{LP}_k) = 14 \right).$$

• Iterazione 4

La lista è $L = \{\text{LP}_6\}$: il problema LP_6 viene estratto e rimosso da L .

In $\tilde{\mathbf{x}}(\text{LP}_6)$ l'unica variabile intera con valore frazionario è x_1 , che è la variabile di branching. I vincoli di branching sono:

$$x_1 \leq \underbrace{\left\lfloor \frac{41}{10} \right\rfloor}_{=4} \quad \text{e} \quad x_1 \geq \underbrace{\left\lfloor \frac{41}{10} \right\rfloor + 1}_{=5}.$$

- Nodo 8 ($x_1 \leq 4$): $\tilde{\mathbf{x}}(\text{LP}_8) = (4, 1)$ con $z(\text{LP}_8) = 14$. La soluzione $\tilde{\mathbf{x}}(\text{LP}_8)$ è **intera**. Poiché $z(\text{LP}_8) = 14 > \text{LB} = 13$, il lower bound e la soluzione incumbent vengono aggiornati:

$$\text{LB} \leftarrow 14, \quad \bar{\mathbf{x}} \leftarrow (4, 1).$$

Il Nodo 8 è potato per integralità.

– **Regola di salto.** Dopo il Nodo 8 il lower bound soddisfa

$$\lfloor z(\text{LP}_6) \rfloor = \left\lfloor \frac{143}{10} \right\rfloor = 14 \leq \text{LB} = 14,$$

quindi nessuna soluzione nella regione di LP_6 può migliorare la soluzione incumbent: il figlio verso l'alto ($x_1 \geq 5$) **non viene risolto** e non riceve un indice di nodo.

La lista è vuota, quindi l'upper bound è uguale al lower bound:

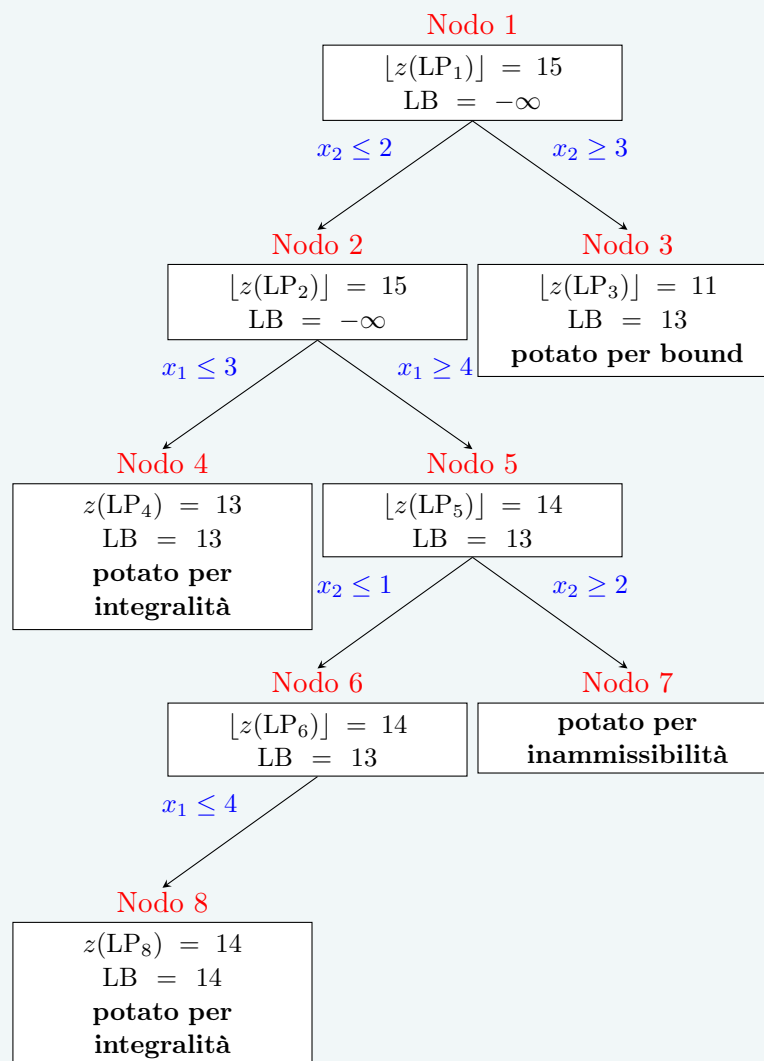
$$\underbrace{14}_{\text{LB}} \leq z(\text{ILP}) \leq \underbrace{14}_{\text{UB}} \quad (\text{UB} = \text{LB} = 14).$$

- **Terminazione**

La lista L è vuota: $\text{LB} = \text{UB} = 14$ e la soluzione incumbent è ottima:

$$\tilde{x}_1 = 4, \quad \tilde{x}_2 = 1, \quad z(\text{ILP}) = 14.$$

Questo è l'albero branch-and-bound finale:



Nodo	Padre	It.	Branching	$\tilde{\mathbf{x}}(\text{LP}_k)$	$z(\text{LP}_k)$	LB	Esito
1	–	0	–	$(\frac{7}{2}, \frac{5}{2})$	$\frac{31}{2}$	$-\infty$	branching su x_2
2	1	1	$x_2 \leq 2$	$(\frac{37}{10}, 2)$	$\frac{151}{10}$	$-\infty$	branching su x_1
3	1	1	$x_2 \geq 3$	$(\frac{7}{4}, 3)$	$\frac{45}{4}$	13	potato per bound (rimosso da L)
4	2	2	$x_1 \leq 3$	$(3, 2)$	13	13	potato per integralità
5	2	2	$x_1 \geq 4$	$(4, \frac{5}{4})$	$\frac{29}{2}$	13	branching su x_2
6	5	3	$x_2 \leq 1$	$(\frac{41}{10}, 1)$	$\frac{143}{10}$	13	branching su x_1
7	5	3	$x_2 \geq 2$	–	–	13	potato per inammissibilità
8	6	4	$x_1 \leq 4$	$(4, 1)$	14	14	potato per integralità

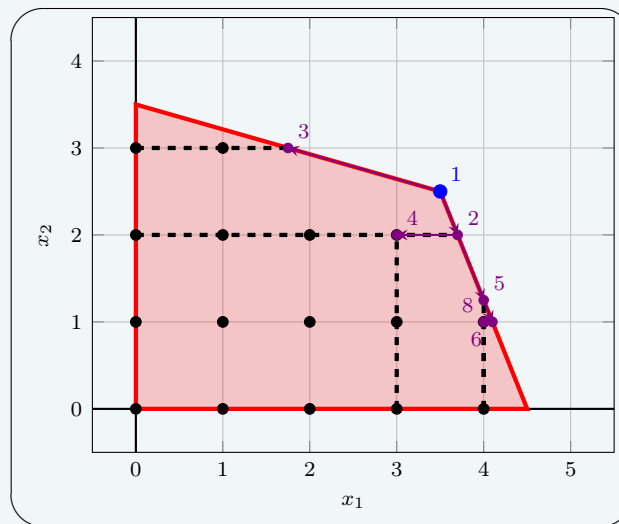


Figura 7.9: L'esecuzione B nel piano: vincoli di branching e soluzioni ottime dei nodi risolti.

- L'esecuzione A risolve 6 problemi LP, l'esecuzione B ne risolve 8. Entrambe certificano la stessa soluzione ottima $\tilde{x}_1 = 4, \tilde{x}_2 = 1$ di valore 14, che è unica.
- L'esecuzione B trova prima una soluzione incumbent di valore 13, ma fare branching prima su x_2 porta, su questa istanza, a un albero più grande.

Esempio 7.7: inizializzazione euristica

Eseguiamo l'algoritmo B&B dell'Esempio 7.5 (indice minimo, LIFO), partendo da una soluzione euristica dell'esempio dell'auto.

- L'euristica a una variabile del Capitolo 1 fissa x_1 al valore più grande possibile e $x_2 = 0$: dà $\bar{\mathbf{x}} = (4, 0)$ di valore 12. Avviando l'esecuzione A con $\text{LB} = 12$, l'algoritmo risolve 6 problemi LP, tanti quanti senza l'euristica: questo lower bound è troppo debole per potare prima qualche nodo.
- L'euristica che fissa prima x_1 e poi x_2 al valore più grande possibile dà $\bar{\mathbf{x}} = (4, 1)$ di

valore 14. Partendo da $LB = 14$, la radice non è potata, perché $\lfloor z(LP_1) \rfloor = 15 > 14$, ma entrambi i figli sono potati per bound e l'algoritmo si ferma dopo 3 problemi LP:

Nodo	Padre	It.	Branching	$\tilde{x}(LP_k)$	$z(LP_k)$	LB	Esito
1	–	0	–	$(\frac{7}{2}, \frac{5}{2})$	$\frac{31}{2}$	14	branching su x_1
2	1	1	$x_1 \leq 3$	$(3, \frac{37}{14})$	$\frac{100}{7}$	14	potato per bound
3	1	1	$x_1 \geq 4$	$(4, \frac{5}{4})$	$\frac{29}{2}$	14	potato per bound

La soluzione incumbent data dall'euristica è ottima, e l'algoritmo branch-and-bound ne **certifica** l'ottimalità.

7.8 Errori tipici

Attenzione

- **Arrotondare la soluzione LP.** Arrotondare una soluzione ottima del rilassamento lineare dà al più una soluzione ammissibile, in generale non ottima. Nell'esempio dell'auto, arrotondando per difetto $\tilde{x}(LP) = (\frac{7}{2}, \frac{5}{2})$ si ottiene $(3, 2)$ di valore 13, mentre $z(ILP) = 14$ in $(4, 1)$.
- **Fare branching sulla variabile sbagliata.** La variabile di branching deve essere una variabile intera con valore frazionario: una variabile con valore intero non esclude $\tilde{x}(LP_k)$.
- **Prendere l'upper bound dall'ultimo nodo.** L'upper bound è il massimo dei bound dei nodi della lista L , non il bound dell'ultimo nodo risolto.
- **Potare per bound solo con la disuguaglianza stretta.** Un nodo con $UB(LP_k) = LB$ è potato: la sua regione può contenere soluzioni buone quanto quella incumbent, ma non migliori.
- **Numerare i figli saltati.** Un figlio che non viene risolto non riceve un indice di nodo.
- **Risolvere ogni nodo da zero.** La risposta è corretta, ma molto più lenta: ogni figlio parte dal dizionario ottimo del padre, che è duale ammissibile.

7.9 Estensioni

- I problemi LP dei nodi si possono risolvere anche con l'**algoritmo del simplesso con variabili limitate**, che tratta i vincoli di branching come limiti sulle variabili invece che come nuovi vincoli.
- In un problema **misto**, dove solo alcune variabili sono intere, si fa branching solo sulle variabili intere e il bound di un nodo è $z(LP_k)$, senza arrotondamento.

7.10 In sintesi

In sintesi

- Un algoritmo B&B divide la regione ammissibile con vincoli di branching su variabili intere con valori frazionari, e non perde mai una soluzione ammissibile.
- Ogni nodo è potato per inammissibilità, per integralità o per bound, oppure attende nella lista; il bound di un nodo è $\lfloor z(LP_k) \rfloor$ solo se tutte le variabili e tutti i costi sono interi.
- Il lower bound viene dalla soluzione incumbent, l'upper bound dai nodi della lista; quando la lista è vuota coincidono, e la soluzione incumbent è certificata ottima.
- Ogni figlio è riottimizzato con l'algoritmo del simplesso duale a partire dal dizionario del padre.
- Le strategie di selezione e una buona soluzione euristica cambiano la dimensione dell'albero, non il valore ottimo.

Cruscotto dell'esempio dell'auto. I lower bound e gli upper bound su $z(ILP)$ trovati finora in questo modulo:

Capitolo	Lower bound	Upper bound	Da
Bound primali e bound duali	14	15	soluzione euristica $(4, 1)$, e $\lfloor z(LP) \rfloor$ con $z(LP) = \frac{31}{2}$
Algoritmo dei piani di taglio	14	14	5 tagli frazionari di Gomory
Questo capitolo, esecuzione A	14	14	6 problemi LP risolti
Questo capitolo, esecuzione B	14	14	8 problemi LP risolti
Questo capitolo, con $LB = 14$	14	14	3 problemi LP risolti

Nell'esempio dell'auto l'algoritmo chiude il gap tra $LB = 13$ (soluzione LP arrotondata) e $UB = 15$ (rilassamento lineare) risolvendo 6 problemi LP.

Il branch-and-cut

Prerequisiti

- Da questo modulo, i capitoli «Disuguaglianze valide» e «Disuguaglianze di copertura»: le disuguaglianze valide e la separazione di una disuguaglianza violata di una famiglia.
- Da questo modulo, il capitolo «Algoritmo dei piani di taglio»: i tagli frazionari di Gomory e la riottimizzazione del problema LP con l'algoritmo del simplesso duale dopo l'inserimento di un taglio.
- Da questo modulo, il capitolo «Branch-and-bound basato sul rilassamento lineare»: la lista dei problemi LP, i criteri di potatura e le strategie di selezione della variabile e del nodo di branching.

Obiettivi

Alla fine di questo capitolo sarai in grado di:

- spiegare come i piani di taglio e le operazioni di branching si completano a vicenda;
- distinguere le disuguaglianze valide globali e locali, e dire dove si può usare un taglio;
- descrivere gli algoritmi cut-and-branch e branch-and-cut, e il ruolo del pool di tagli;
- eseguire a mano un algoritmo cut-and-branch con tagli di Gomory su un piccolo problema PLI e confrontare la dimensione dell'albero con e senza tagli;
- generare un taglio locale in un nodo di branching e riottimizzare con l'algoritmo del simplesso duale.

8.1 Algoritmi branch-and-cut

Definizione 8.1: algoritmo branch-and-cut

Dato un problema PLI in forma canonica di massimo con dati interi, un **algoritmo branch-and-cut** è un algoritmo branch-and-bound in cui i problemi LP dei nodi di branching sono rafforzati con piani di taglio prima del branching.

- Un **algoritmo dei piani di taglio** avvicina il rilassamento lineare all'involuppo convesso delle soluzioni intere, ma i tagli frazionari di Gomory perdono efficacia dopo le prime iterazioni (**tailing off**).
- Un **algoritmo branch-and-bound** suddivide la regione ammissibile in problemi LP più piccoli, ma partendo da un rilassamento lineare debole ha bisogno di un albero grande.

I tagli abbassano i bound dei nodi, così più nodi vengono potati per bound; il branching garantisce la terminazione quando i tagli non sono più efficaci.

- Tutti i solver MIP moderni si basano su algoritmi branch-and-cut (si veda il capitolo «Solver MIP: cosa fanno»).

8.2 Disuguaglianze valide globali e locali

- In un algoritmo branch-and-bound, il problema LP_k di un nodo di branching k è il rilassamento lineare con i vincoli di branching del cammino dalla radice al nodo k .
- Indichiamo con F l'insieme delle soluzioni intere ammissibili del problema PLI, e con $F_k \subseteq F$ l'insieme di quelle che soddisfano i vincoli di branching del nodo k . Nel nodo radice $F_1 = F$.

Definizione 8.2: disuguaglianze valide globali e locali

Una disuguaglianza lineare $\alpha'x \leq \beta$ si chiama:

1. **disuguaglianza globalmente valida** (o **taglio globale**) se è soddisfatta da tutte le soluzioni in F ;
2. **disuguaglianza localmente valida** per il nodo di branching k (o **taglio locale**) se è soddisfatta da tutte le soluzioni in F_k .

- Poiché $F_k \subseteq F$, ogni taglio globale è anche un taglio locale di ogni nodo. Il viceversa è falso: un taglio locale può eliminare soluzioni intere ammissibili che appartengono ad altri rami dell'albero.
- Un taglio locale del nodo k è valido per tutti i nodi del sottoalbero con radice in k , e non deve essere usato fuori da questo sottoalbero.

Un taglio derivato solo dai vincoli originali (e da altri tagli globali) è **globale**. Un taglio la cui derivazione usa un vincolo di branching è, in generale, solo **locale**.

- Un taglio frazionario di Gomory generato nel nodo radice è globale: deriva dai soli vincoli originali, e lo stesso vale per una disuguaglianza di una famiglia (come le disuguaglianze di copertura di un vincolo a coefficienti positivi) separata a partire da essi. Un taglio frazionario di Gomory generato da una riga che contiene la variabile di scarto di un vincolo di branching con coefficiente non nullo si tratta come locale.

8.3 Cut-and-branch e branch-and-cut

Definizione 8.3: algoritmo cut-and-branch

Un **algoritmo cut-and-branch** esegue un algoritmo dei piani di taglio nel nodo radice per un numero limitato di **round di tagli**, aggiunge al problema PLI i tagli trovati, e poi esegue un algoritmo branch-and-bound sul problema rafforzato. Negli altri nodi non si generano tagli.

- Tutti i tagli della radice sono globali: nel nodo radice non c'è alcun vincolo di branching.
- In un **algoritmo branch-and-cut** i tagli si generano anche negli altri nodi (in tutti, o in alcuni, per esempio ogni dieci livelli dell'albero). I tagli possono essere globali o locali.
- In ogni nodo la generazione dei tagli si ferma quando la soluzione del problema LP_k è intera o il nodo può essere potato; quando non si trova alcun taglio violato; quando si raggiunge un numero massimo di round; o quando la diminuzione di $z(LP_k)$ negli ultimi round è piccola (tailing off).

8.3.1 Il pool di tagli

- I tagli si memorizzano in una struttura dati chiamata **pool di tagli** (cut pool). Prima di chiamare le procedure (più costose) di generazione dei tagli, l'algoritmo controlla se un taglio del pool è violato da $\tilde{x}(LP_k)$.
- I tagli globali si memorizzano in un **pool globale**, utilizzabile in ogni nodo. I tagli locali si memorizzano con il loro nodo e sono ereditati solo dai suoi discendenti.
- Ogni taglio aggiunge un vincolo ai problemi LP. I tagli che restano inattivi per molte iterazioni si tolgono dal problema LP e si conservano solo nel pool.

8.3.2 Pseudocodice ad alto livello

- L'algoritmo branch-and-cut ha lo pseudocodice dell'algoritmo branch-and-bound (stesse operazioni di branching, stessi criteri di potatura, stessa lista L): cambia solo la sottoprocedura di soluzione del nodo di branching.

Algoritmo

Input: Un problema LP_k associato a un nodo di branching, il pool di tagli e un numero massimo R di round di tagli.

Passo 1: Operazioni di bounding

Risolvere il problema LP_k .

- (a) se il problema LP_k è inammissibile, allora **STOP**.
- (b) se $\lfloor z(LP_k) \rfloor \leq LB$, allora **STOP**.
- (c) se $\tilde{x}(LP_k)$ è intera, allora aggiornare il **lower bound** e la **soluzione incumbent**, togliere dalla lista L tutti i problemi LP con $\lfloor z(LP) \rfloor \leq LB$ e **STOP**.

Passo 2: Generazione dei tagli

Se sono stati eseguiti R round di tagli nel nodo k , andare al Passo 3. Altrimenti:

- (a) cercare nel pool di tagli i tagli violati da $\tilde{x}(LP_k)$; se non se ne trova nessuno, generare nuovi tagli violati e memorizzarli nel pool (i tagli globali nel pool globale, quelli locali con il nodo k);
- (b) se non esiste alcun taglio violato, andare al Passo 3. Altrimenti aggiungere i tagli al problema LP_k , riottimizzarlo e andare al Passo 1.

Passo 3: Inserimento nella lista

Inserire il problema LP_k (con i suoi tagli) nella lista L e **STOP**.

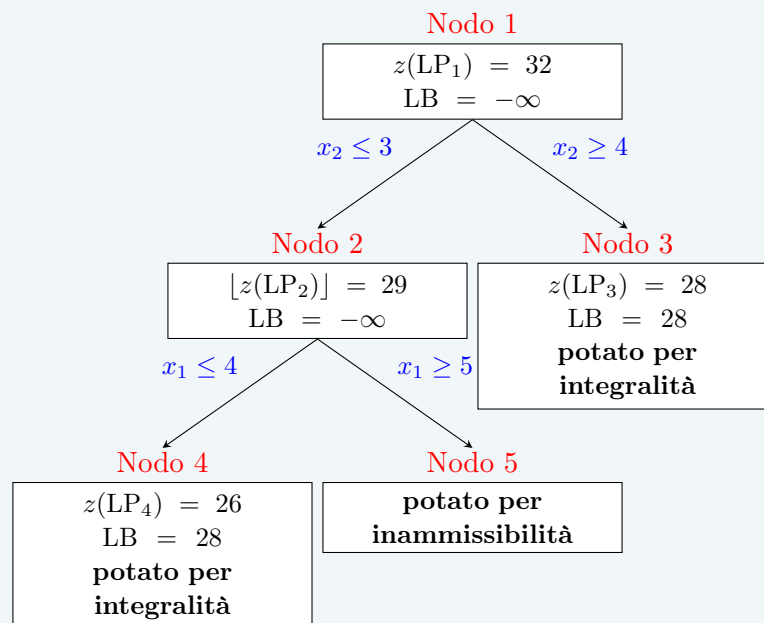
- Con $R = 0$ in ogni nodo tranne la radice otteniamo un algoritmo cut-and-branch; con $R = 0$ in ogni nodo otteniamo l'algoritmo branch-and-bound.
- Lo pseudocodice non dice come si generano i tagli né come si riottimizza il problema LP . Dopo l'inserimento di un taglio il dizionario è duale ammissibile, quindi l'**algoritmo del simplesso duale** parte dal dizionario ottimo del problema precedente, come dopo un vincolo di branching.

8.4 Esempio svolto: cut-and-branch con tagli di Gomory

- Usiamo il problema $P10$ del catalogo dei problemi MIP: massimizzare $5x_1 + 2x_2$ soggetto a $-x_1 + 3x_2 \leq 12$, $x_1 + 2x_2 \leq 12$, $3x_1 - 2x_2 \leq 8$, $x_1, x_2 \in \mathbb{Z}_{\geq 0}$. La sua unica soluzione ottima è $\tilde{x}(\text{ILP}) = (4, 4)$, con $z(\text{ILP}) = 28$.
- I tagli sono i tagli frazionari di Gomory del capitolo sull'algoritmo dei piani di taglio, con la stessa regola: la riga con il termine noto frazionario più grande e, in caso di pareggio, l'indice minimo.
- Poiché i dati sono interi, il bound di un nodo è $\lfloor z(\text{LP}_k) \rfloor$. Tutte le esecuzioni usano le stesse regole: la regola dell'indice minimo come strategia di selezione della variabile di branching, la regola LIFO come strategia di selezione del nodo di branching, il nodo di branching verso il basso risolto per primo, e $\text{LB} = -\infty$ all'inizio.

Esempio 8.1: branch-and-bound senza tagli

Il problema LP_1 è il rilassamento lineare di $P10$. Una soluzione ottima è $\tilde{x}(\text{LP}_1) = (5, \frac{7}{2})$, con $z(\text{LP}_1) = 32$ e $\text{UB} = 32$. L'algoritmo branch-and-bound risolve 5 problemi LP:



Nodo	Padre	Branching	$\tilde{x}(\text{LP}_k)$	$z(\text{LP}_k)$	LB	Esito
1	–	–	$(5, \frac{7}{2})$	32	$-\infty$	branching su x_2
2	1	$x_2 \leq 3$	$(\frac{14}{3}, 3)$	$\frac{88}{3}$	$-\infty$	branching su x_1
3	1	$x_2 \geq 4$	$(4, 4)$	28	28	potato per integralità
4	2	$x_1 \leq 4$	$(4, 3)$	26	28	potato per integralità
5	2	$x_1 \geq 5$	–	–	28	potato per inammissibilità

Esempio 8.2: cut-and-branch con un taglio di Gomory

- **Radice: generazione dei tagli.** Le due righe del dizionario ottimo di LP_1 con termine noto frazionario, le righe di x_2 e di x_3 , hanno lo stesso termine noto $\frac{1}{2}$; usiamo la riga di x_2 , che ha l'indice minore. Con le variabili di scarto $x_4 = 12 - x_1 - 2x_2$ e $x_5 = 8 - 3x_1 + 2x_2$ il taglio è

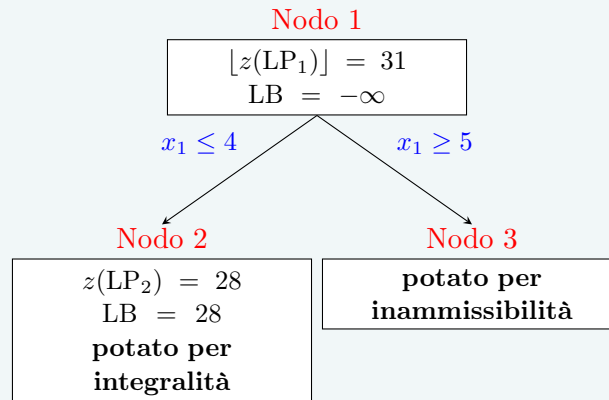
$$\frac{3}{8}x_4 + \frac{7}{8}x_5 \geq \frac{1}{2} \quad \Longleftrightarrow \quad 3x_1 - x_2 \leq 11.$$

È un taglio globale, ed esclude $\tilde{x}(LP_1)$: $3 \cdot 5 - \frac{7}{2} = \frac{23}{2} > 11$.

- **Radice: riottimizzazione.** Aggiungiamo il taglio al problema LP_1 e lo riottimizziamo con l'algoritmo del simplesso duale. Una soluzione ottima è

$$\tilde{x}(LP_1) = \left(\frac{34}{7}, \frac{25}{7}\right), \quad z(LP_1) = \frac{220}{7}, \quad UB = 31.$$

- **Branch-and-bound.** L'algoritmo branch-and-bound parte ora da questo problema LP_1 . Ogni nodo eredita il taglio. Risolve 3 problemi LP invece di 5:



Nodo	Padre	Branching	$\tilde{x}(LP_k)$	$z(LP_k)$	LB	Esito
1	–	–	$\left(\frac{34}{7}, \frac{25}{7}\right)$	$\frac{220}{7}$	$-\infty$	branching su x_1
2	1	$x_1 \leq 4$	$(4, 4)$	28	28	potato per integralità
3	1	$x_1 \geq 5$	–	–	28	potato per inammissibilità

- Un solo taglio di Gomory riduce il numero di problemi LP da 5 a 3. Il taglio non trova la soluzione ottima: sposta la soluzione ottima del problema LP della radice, così che l'algoritmo fa branching su x_1 e il nodo $x_1 \leq 4$ dà subito la soluzione ottima.
- Un secondo round di tagli, nello stesso algoritmo cut-and-branch, dà una soluzione intera nella radice. La riga di x_1 ha il termine noto più grande, $\frac{6}{7}$; con la variabile di scarto $x_6 = 11 - 3x_1 + x_2$ del primo taglio il nuovo taglio è

$$\frac{1}{7}x_4 + \frac{2}{7}x_6 \geq \frac{6}{7} \quad \Longleftrightarrow \quad x_1 \leq 4.$$

- La soluzione ottima del nuovo problema LP_1 è $(4, 4)$, con $z(LP_1) = 28$. Il nodo è potato per interezza, e l'algoritmo si ferma con $LB = UB = 28$ dopo aver risolto un solo problema LP.

Tagli aggiunti nella radice	$z(LP_1)$	UB nella radice	Problemi LP risolti
nessuno (branch-and-bound)	32	32	5
$3x_1 - x_2 \leq 11$	$\frac{220}{7}$	31	3
$3x_1 - x_2 \leq 11$ e $x_1 \leq 4$	28	28	1

- Più i tagli sono forti, più l'albero è piccolo, fino a ridursi al solo nodo radice.
- Non è la regola. La tabella seguente riporta lo stesso algoritmo sul problema $P2$ del catalogo (massimizzare $x_1 + 3x_2$ soggetto a $-4x_1 + 10x_2 \leq 35$, $3x_1 + x_2 \leq 12$, $x_1, x_2 \in \mathbb{Z}_{\geq 0}$, con $z(ILP) = 14$) per un numero crescente di round di tagli di Gomory.

Round	Ultimo taglio aggiunto	$z(LP_1)$	UB nella radice	Problemi LP risolti
0	–	16	16	4
1	$-3x_1 + 10x_2 \leq 37$	$\frac{524}{33}$	15	4
2	$x_1 + 3x_2 \leq 15$	15	15	4
3	$-x_1 + 5x_2 \leq 19$	15	15	4
4	$x_2 \leq 4$	$\frac{44}{3}$	14	2
5	$x_1 + x_2 \leq 6$	14	14	1

- I primi tre tagli abbassano $z(LP_1)$ da 16 a 15, ma l'albero ha sempre 4 nodi; servono 5 round per arrivare al solo nodo radice. L'upper bound migliora lentamente, come nel **tailing off** dell'algoritmo dei piani di taglio.
- Un algoritmo cut-and-branch deve decidere quando smettere di tagliare e cominciare il branching: il costo di ogni round si paga in ogni nodo dell'albero.

8.5 Tagli locali: un esempio

- Usiamo l'esempio delle automobili (x_1 auto sportive e x_2 auto familiari): massimizzare $3x_1 + 2x_2$ soggetto a $10x_1 + 4x_2 \leq 45$, $4x_1 + 14x_2 \leq 49$, $x_1, x_2 \in \mathbb{Z}_{\geq 0}$. Nell'algoritmo branch-and-bound senza tagli, il nodo 3 è il figlio verso l'alto della radice, con il vincolo di branching $x_1 \geq 4$ e la variabile di scarto $x_5 = x_1 - 4$.
- Il dizionario ottimo del problema LP_3 , con $\tilde{x}(LP_3) = (4, \frac{5}{4})$ e $z(LP_3) = \frac{29}{2}$, è:

$$\begin{cases} x_1 &= 4 & & + & x_5 \\ x_2 &= \frac{5}{4} - \frac{1}{4}x_3 - \frac{5}{2}x_5 \\ x_4 &= \frac{31}{2} + \frac{7}{2}x_3 + 31x_5 \\ z &= \frac{29}{2} - \frac{1}{2}x_3 - 2x_5 \end{cases}$$

Esempio 8.3: un taglio locale nel nodo 3

- **Un taglio locale semplice.** Da $10x_1 + 4x_2 \leq 45$ e dal vincolo di branching $x_1 \geq 4$ abbiamo

$$4x_2 \leq 45 - 10x_1 \leq 45 - 10 \cdot 4 = 5 \implies x_2 \leq \frac{5}{4} \implies x_2 \leq 1.$$

- La disuguaglianza $x_2 \leq 1$ è valida per tutte le soluzioni in F_3 , ma **non è globalmente valida**: la soluzione intera ammissibile $(3, 2)$, di valore 13, la viola. È un taglio locale del nodo 3.
- La aggiungiamo al problema LP_3 , con la variabile di scarto $x_6 = 1 - x_2$, e riottimizziamo.
- Il taglio si inserisce con la variabile di scarto $x_6 \geq 0$. La nuova riga è

$$x_6 = -\frac{1}{4} + \frac{1}{4}x_3 + \frac{5}{2}x_5.$$

- Il dizionario seguente è **duale ammissibile** ma non primale ammissibile, perché $\bar{b}_6 = -\frac{1}{4} < 0$:

$$\begin{cases} x_1 &= 4 & & + & x_5 \\ x_2 &= \frac{5}{4} - \frac{1}{4}x_3 - \frac{5}{2}x_5 \\ x_4 &= \frac{31}{2} + \frac{7}{2}x_3 + 31x_5 \\ x_6 &= -\frac{1}{4} + \frac{1}{4}x_3 + \frac{5}{2}x_5 \\ z &= \frac{29}{2} - \frac{1}{2}x_3 - 2x_5 \end{cases}$$

- **Operazione di pivot duale.** La riga pivot è quella di x_6 , l'unica variabile in base con valore negativo. I rapporti delle variabili fuori base con $\bar{a}_{6j} < 0$ sono

$$\frac{\bar{c}_3}{\bar{a}_{6,3}} = \frac{-\frac{1}{2}}{-\frac{1}{4}} = 2, \quad \frac{\bar{c}_5}{\bar{a}_{6,5}} = \frac{-2}{-\frac{5}{2}} = \frac{4}{5}.$$

- La colonna pivot è quella di x_5 , con $s_m = \frac{4}{5}$, e il valore obiettivo diventa $\bar{z} + s_m \bar{b}_6 = \frac{29}{2} + \frac{4}{5} \cdot \left(-\frac{1}{4}\right) = \frac{143}{10}$. Il nuovo dizionario è primale e duale ammissibile, quindi ottimo:

$$\left\{ \begin{array}{lclcl} x_1 & = & \frac{41}{10} & - & \frac{1}{10} x_3 & + & \frac{2}{5} x_6 \\ x_2 & = & 1 & & & - & x_6 \\ x_4 & = & \frac{93}{5} & + & \frac{2}{5} x_3 & + & \frac{62}{5} x_6 \\ x_5 & = & \frac{1}{10} & - & \frac{1}{10} x_3 & + & \frac{2}{5} x_6 \\ z & = & \frac{143}{10} & - & \frac{3}{10} x_3 & - & \frac{4}{5} x_6 \end{array} \right.$$

- La soluzione $\tilde{x}(\text{LP}_3) = (\frac{41}{10}, 1)$, di valore $\frac{143}{10}$, è la stessa che dà il figlio verso il basso del nodo 3: qui il taglio locale fa da vincolo di branching, senza creare un nuovo nodo.

Esempio 8.4: un taglio frazionario di Gomory nel nodo 3

- La riga di x_2 del dizionario del nodo 3 dà il taglio frazionario di Gomory

$$\frac{1}{4} x_3 + \frac{1}{2} x_5 \geq \frac{1}{4}$$

- Contiene la variabile di scarto x_5 del vincolo di branching $x_1 \geq 4$, quindi la sua derivazione usa questo vincolo e l'algoritmo lo tratta come un **taglio locale**.
- Nello spazio delle variabili originali ($x_3 = 45 - 10x_1 - 4x_2$, $x_5 = x_1 - 4$, moltiplicando per 4) si scrive $2x_1 + x_2 \leq 9$. Enumerando le 16 soluzioni intere ammissibili si vede che questa disuguaglianza è valida per tutte.
- L'algoritmo non può fare questo controllo in un nodo, quindi tiene il taglio come locale.

8.6 Aspetti pratici

8.6.1 Selezione dei tagli

- A ogni round possono essere disponibili molti tagli violati; aggiungerli tutti rende i problemi LP grandi e lenti da riottimizzare. Un algoritmo branch-and-cut ne seleziona un sottoinsieme.
- L'**efficacia** di un taglio $\alpha'x \leq \beta$ nella soluzione frazionaria $\tilde{x}(\text{LP}_k)$ è la distanza euclidea di $\tilde{x}(\text{LP}_k)$ dall'iperpiano $\alpha'x = \beta$:

$$\text{efficacia} = \frac{\alpha' \tilde{x}(\text{LP}_k) - \beta}{\|\alpha\|_2}.$$

- A differenza della violazione $\alpha' \tilde{x}(\text{LP}_k) - \beta$, l'efficacia non cambia se il taglio si moltiplica per una costante positiva. Le regole di selezione combinano l'efficacia con una misura del **parallelismo** dei tagli.
- Per esempio, in $\tilde{x}(\text{LP}_1) = (5, \frac{7}{2})$ di P10 il taglio $3x_1 - x_2 \leq 11$ ha violazione $\frac{1}{2}$ ed efficacia $\frac{1}{2\sqrt{10}} \approx 0.158$, mentre lo stesso taglio moltiplicato per 2, $6x_1 - 2x_2 \leq 22$, ha violazione 1 e la stessa efficacia $\frac{1}{\sqrt{40}} \approx 0.158$.

8.6.2 Quali tagli

- Si può usare qualunque famiglia di disuguaglianze valide con una procedura di separazione: i tagli frazionari di Gomory (per problemi PLI con dati interi), le disuguaglianze di copertura per i vincoli $\sum_j a_j x_j \leq b$ a coefficienti positivi di un problema MIP con variabili binarie, e famiglie che sfruttano la struttura di un modello specifico.
- I tagli frazionari di Gomory vengono dalle righe del dizionario ottimo, quindi sono sempre disponibili. In aritmetica floating point i loro coefficienti risentono degli errori di arrotondamento, e i solver scartano i tagli con coefficienti molto grandi o molto piccoli.

8.6.3 Upper bound e gap

- Come nell'algoritmo branch-and-bound, se la lista L non è vuota abbiamo

$$z(\text{ILP}) \leq \text{UB} = \max_{k \in L} \{ \text{UB}(\text{LP}_k) \},$$

dove ora $z(\text{LP}_k)$ è il valore ottimo del problema LP_k dopo i tagli.

- I tagli abbassano i valori $z(\text{LP}_k)$ e, di conseguenza, l'upper bound UB e il gap di ottimalità $\frac{\text{UB}-\text{LB}}{|\text{LB}|}$.

8.7 Errori tipici

Attenzione

- **Usare un taglio locale fuori dal suo sottoalbero.** Un taglio derivato con un vincolo di branching del nodo k è valido solo nel sottoalbero del nodo k : in un altro ramo può eliminare una soluzione ottima.
- **Trattare un taglio come globale senza una dimostrazione.** Un taglio frazionario di Gomory generato in un nodo può essere globale per caso (Esempio 8.4); l'algoritmo lo tiene locale, a meno che la sua derivazione non usi vincoli di branching.
- **Aggiungere tutti i tagli violati.** Troppi tagli rallentano i problemi LP; meglio pochi tagli efficaci per round, togliendo quelli inattivi.
- **Credere che ogni taglio riduca l'albero.** Su $P2$ i primi tre tagli di Gomory lasciano l'albero a 4 nodi (tabella della Sezione 8.4); conta la qualità dei tagli.
- **Scegliere come più forte il taglio di efficacia massima.** L'efficacia si misura solo nel punto corrente: un taglio dominato può avere in quel punto un'efficacia maggiore.
- **Dimenticare che i tagli abbassano il bound, non il valore ottimo.** Il valore ottimo $z(\text{ILP})$ non cambia mai: cambiano solo $z(\text{LP}_k)$ e l'upper bound.

8.8 In sintesi

In sintesi

- Un algoritmo branch-and-cut rafforza con tagli il problema LP di ogni nodo prima del branching.
- Un taglio globale è valido per tutte le soluzioni intere; un taglio locale solo per quelle di un sottoalbero, e si memorizza con il suo nodo.
- Un algoritmo cut-and-branch genera tagli solo nella radice; i tagli sono quindi globali.
- Dopo un taglio, il problema LP si riottimizza con l'algoritmo del simplesso duale a partire dal dizionario precedente.
- I tagli riducono la dimensione dell'albero, ma solo se sono efficaci: su $P10$ un taglio di Gomory riduce l'albero da 5 a 3 problemi LP, e due tagli a 1.

Cruscotto di $P10$. I lower e upper bound su $z(\text{ILP}) = 28$ trovati in questo capitolo:

Fase	Lower bound	Upper bound	Da
Rilassamento lineare di P_{10}	—	32	$z(\text{LP}) = 32, \tilde{\mathbf{x}}(\text{LP}) = (5, \frac{7}{2})$
Branch-and-bound	28	28	5 problemi LP risolti
Dopo il primo taglio di Gomory	—	31	$z(\text{LP}_1) = \frac{220}{7}$
Cut-and-branch con un solo taglio	28	28	3 problemi LP risolti
Dopo il secondo taglio di Gomory	28	28	$\tilde{\mathbf{x}}(\text{LP}_1) = (4, 4)$ è intera, 1 problema LP risolto

Come i solver MIP combinano queste idee è l'argomento del capitolo «Solver MIP».