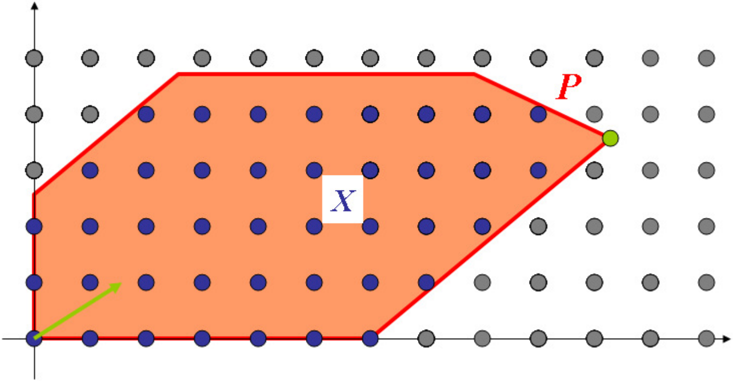
Branch-and-bound
per problemi di programmazione lineare intera

Luigi De Giovanni

Dipartimento di Matematica, Università di Padova

Programmazione Lineare Intera

$$\begin{array}{ll} \min / \max & c^T x \\ \text{s.t.} & Ax = b \\ & x \in \mathbb{Z}_+^n \end{array}$$



Algoritmo universale per ottimizzazione combinatoria

1. generare tutte le possibili soluzioni x;
2. verificare l'ammissibilità della soluzione $x \in X$;
3. valutare $f(x)$
4. scegliere la x ammissibile cui corrisponde la migliore $f(x)$.

- Come *generare* lo spazio delle soluzioni (ammissibili)?
- Come *esplorare* efficientemente lo spazio delle soluzioni?

Generazione delle soluzioni: **Branch**

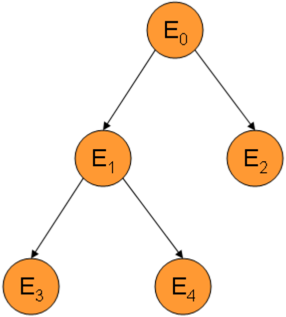
$z = \text{opt}\{f(x) : x \in X\}$ $\quad\quad X = \bigcup_{i=1}^{n} X_i = X$

$$z^{(k)} = \text{opt}\{f(x) : x \in X_k\}$$

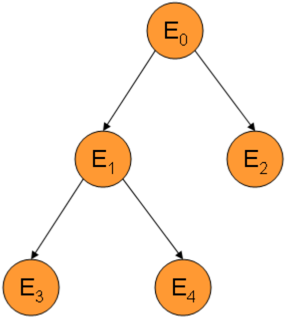
$$z = \text{opt} \; \{ \; z^{(k)}, \; k = 1, \ldots, n \; \}$$

- *divide et impera* ($\sim$ top-down)
- divisione (e soluzione) ricorsiva
- albero delle soluzioni (ammissibili)

- **operazione di branch**



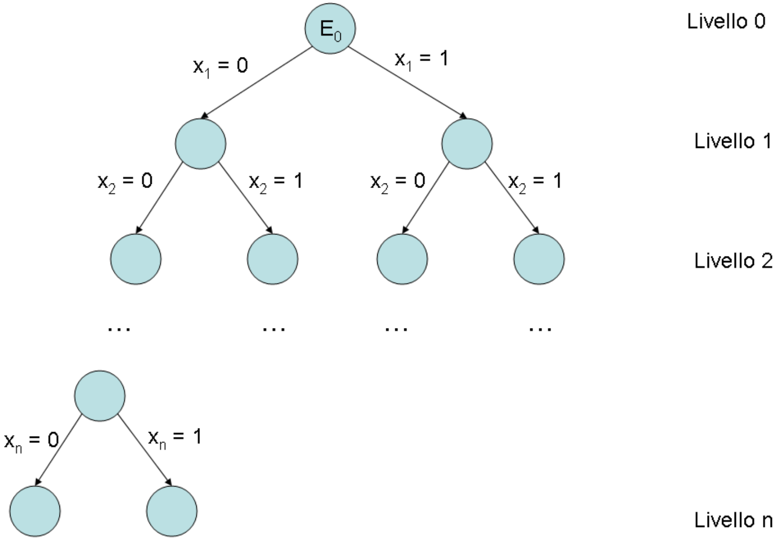
Branching: regole base



- $E_0 = X$
- $\boxed{E_i = \bigcup_{j \text{ figlio di } i} E_j}$
- preferibilmente
  $E_j \cap E_k = \emptyset, \ \forall\, j, k$ figli di i

Esempio: branching binario

$x_i \in \{0, 1\}, i = 1..n$



Esempio: branching "naturale" per path-finding

Ad ogni passo, esplora le diverse direzioni ammesse:
- posso sfruttare parallelismo?
- esiste un criterio per scartare cammini parziali?

Esplorazione efficiente: **bound** + fathom (o prune)

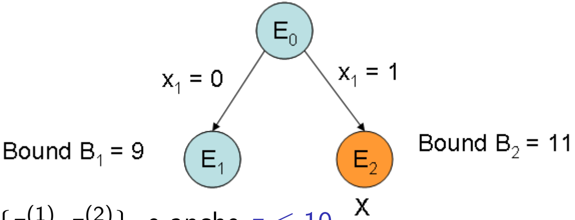
Esempio: $\min z = 11\,x_1 + 9\,x_2 + 10\,x_3 + y$

s.t. $x_1 + x_2 + x_3 \geq 1$

...molti altri vincoli su x_i e y (non formulabili)...

$x_i \in \{0,1\}, \quad y \geq 0$

È noto che $x_3 = 1$, $x_1 = x_2 = y = 0$ è ammissibile (con $z = 10$)



- $z = \min\{z^{(1)}, z^{(2)}\}$, e anche $z \leq 10$
- $z^{(1)} \geq 9$
- $z^{(2)} \geq 11 (\geq 10)$ $\Longrightarrow$ Possiamo **potare** E_2!!!

Branch-and-bound: idea base

- **Branch**: costruzione dell'albero delle soluzioni (enumerazione ricorsiva)
- **Soluzione ammissibile** (incumbent solution): valore possibile, ma non dimostrabilmente ottimo
- **Bound**: valutazione ottimistica della funzione obiettivo per le soluzioni associate ad un nodo (sottoalbero)
- **Fathom**: se il bound di un nodo non è migliore dell'incumbent, il relativo sottoalbero si può potare

Enumerazione implicita dello spazio delle soluzioni

Metodo del **Branch-and-Bound (B&B)**

Inizializzazione: Esegui una stima ottimistica B_0 della funzione obiettivo e poni $L = \{(P_0, B_0)\}$, $\bar{x} = \emptyset$, $\bar{z} = +\infty(\text{min})[-\infty(\text{max})]$

`Repeat`:

- *Criterio di Stop*: Se $L = \emptyset$, allora `stop`: $\bar{x}$ è la soluzione ottima.
- *Selezione nodo*: Seleziona ed estrai $(P_i, B_i) \in L$ per effettuare il branch
- *Branching*: Dividi P_i in t sotto-problemi $P_j, j = 1..t$ $(\cup_j P_j = P_i$
- `For each` sottoproblema $j = 1..t$ `:`
  - *Bounding*: Valuta una stima ottimistica B_j di P_j., ottenendo eventuali informazioni su soluzione "rilassata" x_j^R (e.g., parziale) di P_j, e/o su inammissibilità di P_j
  - *Fathoming*: `If` P_j non è ammissibile`: continue`
    `elseif` B_j non è migliore di $\bar{z}$ ammissibile`: continue`
    `elseif` x_j^R è ammissibile (e.g., completa)`:`
      `if` x_j^R anche migliore di $\bar{z}$`:`
        aggiorna $\bar{z} \leftarrow B_j$, $\bar{x} \leftarrow x_{ij}^R$
        elimina da L nodi k con L_k non migliore di $\bar{z}$
      `continue` (x_{ij}^R è ottima per P_j)
  - *Ricorsione*: `else` aggiungi (P_j, B_j) a L (B_j è più promettente di $\bar{z}$)

Esempio (dummy): scelta ottima di appalti

Una grossa azienda di costruzioni edili deve decidere la combinazione ottimale degli appalti da accettare per la costruzione degli edifici A, B e C.
I profitti attesi per i tre edifici sono di 3, 5 e 7 milioni di euro rispettivamente.
L'azienda dispone di 4 ruspe speciali e gli edifici richiedono risp. 3, 2 e 3 ruspe.
È possibile inoltre affittare fino a due altre ruspe speciali per la durata dei lavori, al costo di un milione di euro a ruspa.

Decisioni:

- accettare appalto i, $i \in \{A, B, C\}$. Possibili decisioni: sì/no.
- numero di ruspe da affittare. Possibili decisioni: 0, 1 o 2.

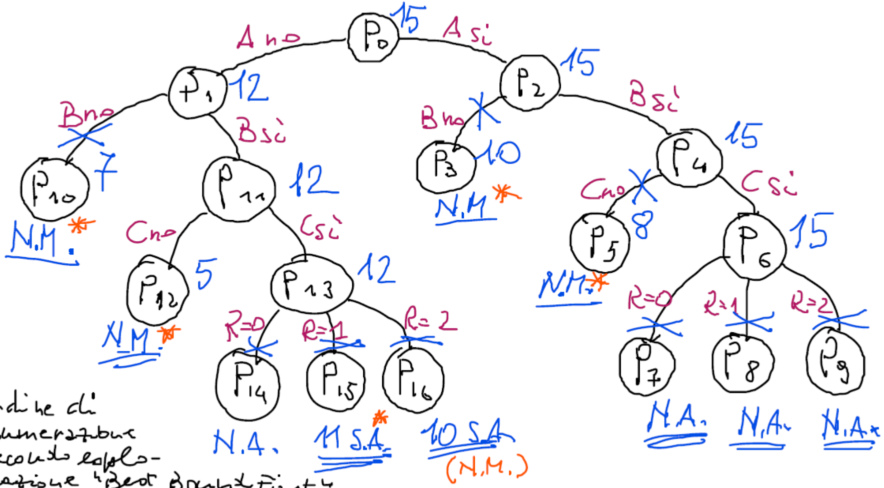
Possibili combinazioni: $2 \times 2 \times 2 \times 3 = 24$

Branch: scegliere una decisione (nell'ordine A-B-C-num.ruspe)
e creare un sottoproblema per ogni valore

Bound: somma profitti di tutti gli appalti possibili meno costo ruspe "fissate"
(valutazione imprecisa ma ottimistica e veloce,
senza ragionamenti su ruspe "necessarie")

Esempio: albero di branch-and-bound

A: 3 M$, 3 ruspe   B: 5 M$, 2 ruspe   C: 7 M$, 3 ruspe



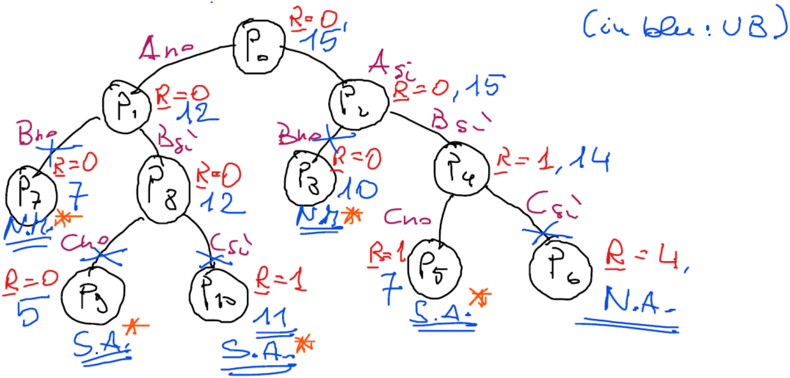
Ordine di numerazione secondo esplorazione "Best Bound First"

Sol. ottima in P_{15}!

*Esempio: regola alternativa per il calcolo dei bound

Bound: sommare i profitti di tutti gli appalti possibili *e valutare una stima per difetto R delle ruspe necessarie (sulla base degli appalti fissati)*

(A: 3 M$, 3 ruspe B: 5 M$, 2 ruspe C: 7 M$, 3 ruspe)



↳ la stima di R porta non solo a un U.B., ma anche a una soluzione ammissibile.

Progettazione

- **Regole di branching**: strategia per costituire sottoproblemi sempre più semplici (al limite una soluzione!)
  – $E_i : \cup_i E_i = E$ (must!) [e $E_i \cap E_j = \emptyset$ opzionale]
- **Bound**: *lower* bound (min, LB) o *upper* bound (max, UB).
  – Valutazione **ottimistica**...: $LB \leq f(E_i) \quad UB \geq f(E_i)$
  – ...ma non troppo! **efficienza** computazionale .vs. **qualità** bound*
- **Regole di fathoming**: evito di esplorare nodo se
  – **[N.M.]** Assenza di soluzione migliorante (B_i non migliora $f(\bar{x})$)
  – **[S.A.]** Valutazione ottimistica è anche di soluzione ammissibile
  – **[N.A.]** Sottoproblema non ammissibile ($E_i = \emptyset$)
- **Strategie di esplorazione**: *Depth First*, *Best Bound First*, *Mista*
- **Valutazione di soluzioni ammissibili**: opzionale!
  – sforzo computazionale .vs. possibilità di potare nodi
- **Criteri di arresto**: tutti i nodi *fathomed* (metodo esatto). Oppure...

B&B per PLI

Problema di PLI(M) (fissiamo le idee: max)

$$
\begin{aligned}
z_I = \max c^T x & \\
Ax \leq b & \\
x \geq 0 & \\
x_i \in \mathbb{Z}, & \qquad i \in I.
\end{aligned}
\tag{1}
$$

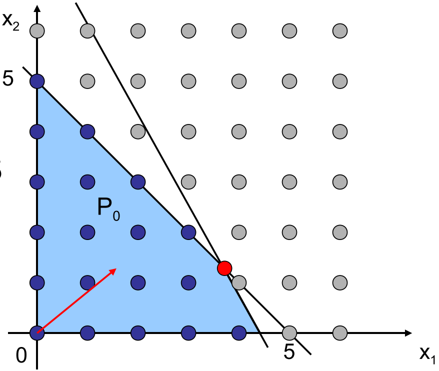
Rilassamento continuo

$$
\begin{aligned}
z_L = \max c^T x \\
Ax \leq b \\
x \geq 0
\end{aligned}
\tag{2}
$$

$z_L \geq z_I \qquad z_L$ è un UB!

Problema P_0

$$
(P_0) \quad \begin{array}{rrcl}
z_I^0 = \max & 5x_1 + \frac{17}{4}x_2 & & \\
& x_1 + x_2 & \leq & 5 \\
& 10x_1 + 6x_2 & \leq & 45 \\
& x_1, x_2 & \geq & 0 \\
& x_1, x_2 & \in & \mathbb{Z}
\end{array}
$$



Rilassamento lineare: $x_1 = 3.75$, $x_2 = 1.75$, con valore $z_L^0 = 24.06$

Branch da P_0 su variabile frazionaria $x_1 = 3.75$

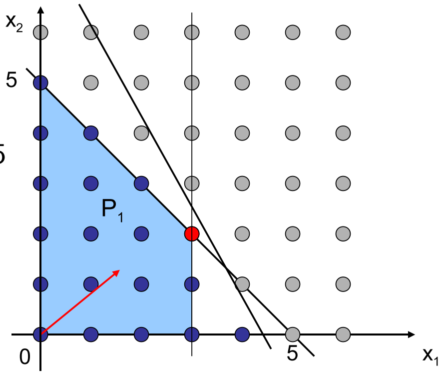
$$
\begin{array}{rrcl}
z_I^1 = \max & 5x_1 + \frac{17}{4}x_2 & & \\
& x_1 + x_2 & \leq & 5 \\
& 10x_1 + 6x_2 & \leq & 45 \\
& x_1 & \leq & 3 \\
& x_1, x_2 & \geq & 0 \\
& x_1, x_2 & \in & \mathbb{Z}
\end{array}
\qquad
\begin{array}{rrcl}
z_I^2 = \max & 5x_1 + \frac{17}{4}x_2 & & \\
& x_1 + x_2 & \leq & 5 \\
& 10x_1 + 6x_2 & \leq & 45 \\
& x_1 & \geq & 4 \\
& x_1, x_2 & \geq & 0 \\
& x_1, x_2 & \in & \mathbb{Z}
\end{array}
$$

(P_1) (P_2)

- Non perdo soluzioni intere: $E_1 \cup E_2 = E_0$, $z_I = \max\{z_I^1, z_I^2\}$
- z_L^0 esclusa!

Problema P_1

$$(P_1) \quad \begin{array}{rrcl} z_I^1 = \max & 5x_1 + \frac{17}{4}x_2 & & \\ & x_1 + x_2 & \leq & 5 \\ & 10x_1 + 6x_2 & \leq & 45 \\ & \color{red}{x_1} & \color{red}{\leq} & \color{red}{3} \\ & x_1, x_2 & \geq & 0 \\ & x_1, x_2 & \in & \mathbb{Z} \end{array}$$



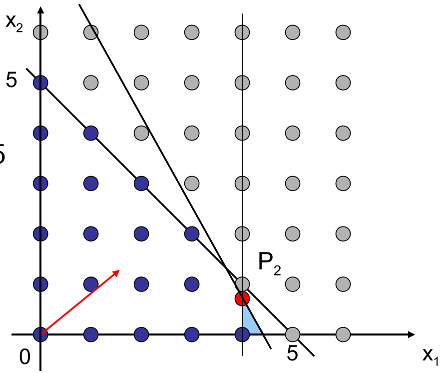
Rilassamento lineare: $x_1 = 3$, $x_2 = 2$, con valore $z_L^1 = 23.5$

Soluzione intera (rilassamento ammissibile): nodo potato per **S.A.**

aggiornamento incumbent, $\bar{z} = 23.5$

Problema P_2

$$(P_2) \quad \begin{array}{rrcl} z_I^1 = \max & 5x_1 + \frac{17}{4}x_2 & & \\ & x_1 + x_2 & \leq & 5 \\ & 10x_1 + 6x_2 & \leq & 45 \\ & x_1 & \geq & 4 \\ & x_1, x_2 & \geq & 0 \\ & x_1, x_2 & \in & \mathbb{Z} \end{array}$$



Rilassamento lineare: $x_1 = 4$, $x_2 = 0.83$, con valore $z_L^2 = 23.54$

Branch da P_2 su variabile frazionaria $x_2 = 0.83$

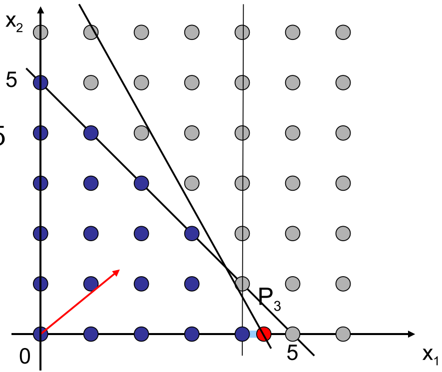
$$
\begin{array}{rrcl}
z_I^1 = \max & 5x_1 + \frac{17}{4}x_2 & & \\
& x_1 + x_2 & \leq & 5 \\
& 10x_1 + 6x_2 & \leq & 45 \\
& x_1 & \geq & 4 \\
& x_2 & \leq & 0 \\
& x_1, x_2 & \geq & 0 \\
& x_1, x_2 & \in & \mathbb{Z}
\end{array}
\qquad
\begin{array}{rrcl}
z_I^2 = \max & 5x_1 + \frac{17}{4}x_2 & & \\
& x_1 + x_2 & \leq & 5 \\
& 10x_1 + 6x_2 & \leq & 45 \\
& x_1 & \geq & 4 \\
& x_2 & \geq & 1 \\
& x_1, x_2 & \geq & 0 \\
& x_1, x_2 & \in & \mathbb{Z}
\end{array}
$$

(P_3) (P_4)

$$E_3 \cup E_4 = E_2$$

Problema P_3

$$
(P_2) \quad
\begin{array}{rrcl}
z_I^1 = \max & 5x_1 + \frac{17}{4}x_2 & & \\
 & x_1 + x_2 & \leq & 5 \\
 & 10x_1 + 6x_2 & \leq & 45 \\
 & {\color{red}x_1} & {\color{red}\geq} & {\color{red}4} \\
 & {\color{red}x_2} & {\color{red}\leq} & {\color{red}0} \\
 & x_1, x_2 & \geq & 0 \\
 & x_1, x_2 & \in & \mathbb{Z}
\end{array}
$$

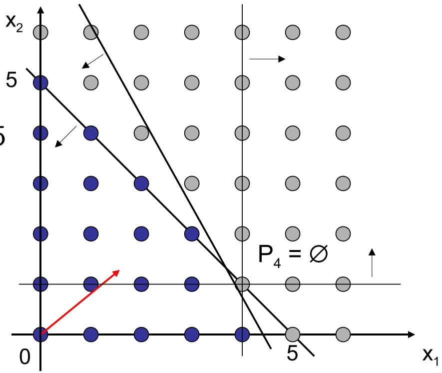


Rilassamento lineare: $x_1 = 4.5$, $x_2 = 0$, con valore $z_L^3 = 22.5$

$z_L^3 \leq \bar{z}$: nodo potato per **N.M.**

Problema P_4

$$(P_2) \quad \begin{array}{rrcl} z_I^1 = \max & 5x_1 + \frac{17}{4}x_2 & & \\ & x_1 + x_2 & \leq & 5 \\ & 10x_1 + 6x_2 & \leq & 45 \\ & {\color{red}x_1} & {\color{red}\geq} & {\color{red}4} \\ & {\color{red}x_2} & {\color{red}\geq} & {\color{red}1} \\ & x_1, x_2 & \geq & 0 \\ & x_1, x_2 & \in & \mathbb{Z} \end{array}$$

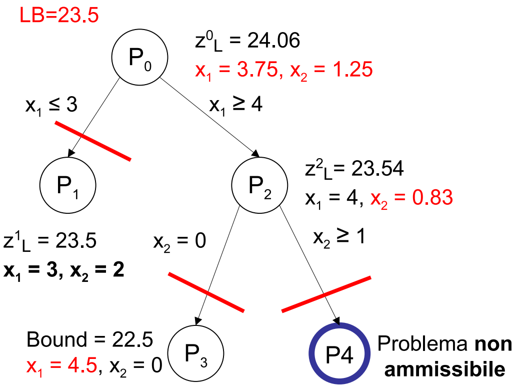


Rilassamento lineare: non ammissibile.

Anche (P_4) non è ammissibile: nodo potato per **N.A.**

Tutti i nodi fathomed: $\bar{x} = (2, 3)$ con $\bar{z} = 23.5$ ottima!

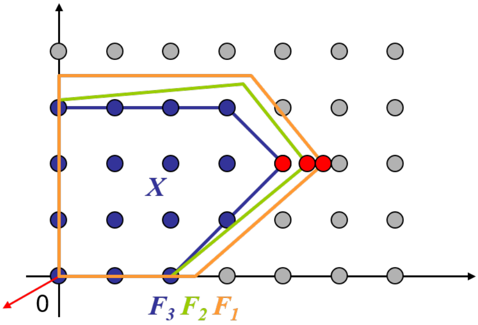
Albero di branch-and-bound



B&B per PLI: scelte progettuali

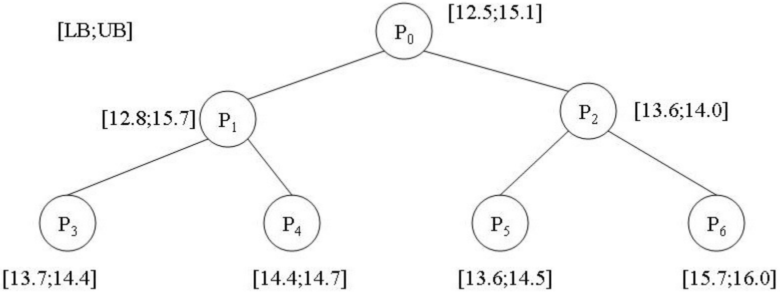
- **Bound** con rilassamento continuo
  - formulazioni alternative per bound più stringenti*
- **Branch** binario su una variabile frazionaria
  - come scelgo la variabile frazionaria? (e.g. “più” frazionaria, “più” intera, *diving* etc.)
  - possibile branching *t*-ario se pochi valori alternativi
- Fathoming standard
- Strategie di esplorazione: mista, *diving* etc.
- Soluzione ammissibile
  - euristiche (o meta-euristiche) ad-hoc prima del branch-and-bound
  - *rounding heuristic* sulla soluzione frazionaria ad ogni nodo (o sotto particolari condizioni)
  - etc.
- Arresto standard + max time, optimality gap etc.

*Esempio: bound e formulazioni alternative per PLIM



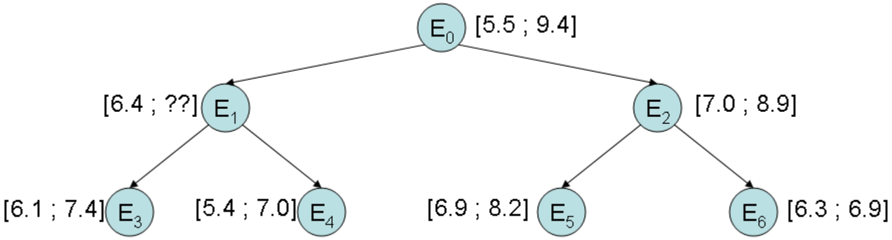
- F_2 è *migliore* di F_1: fornisce bound più stringenti (più vicini all'ottimo): UB più bassi (per problemi di max) o LB più alti (per problemi di min)
- F_3 è la *formulazione ideale*: permette di risolvere il problema al nodo radice (senza branching)

Esercizio



- min o max?
- nodi da chiudere?
- intervallo ottimo?
- best bound first?
- LB e UB per chiudere...

Esercizio



- min o max? valore '??'?
- intervallo ottimo?
- nodi da chiudere?
- best bound first?
- LB e UB per chiudere...