Programmazione Lineare Intera: Introduzione

Alessandro Hill

Basato sul materiale di

Daniele Vigo (D.E.I.)

rev. 1.1(AH) – 2024

P identifica il problema nel suo complesso e zP è il
valore minimo della funzione obiettivo per il problema P

Programmazione Lineare Intera

Funzione Obiettivo (minimizzare un costo)

$$(P) \quad z_P = \min c^T x$$
$$A\,x \geq d$$
$$x \geq 0, \text{ intere}$$

Rappresenta un insieme di vincoli:
A è una matrice che contiene i coefficienti dei vincoli.
x è il vettore delle variabili decisionali.
d è un vettore di costanti.

La differenza chiave tra la PLI e la programmazione lineare classica è la restrizione che impone che tutte le variabili decisioni x devono assumere valori interi.

Il resto del problema (funzione obiettivo e vincoli) rimane lineare

- vincoli di interezza: non lineari
  - x intera $\Leftrightarrow \sin \pi x = 0$
  - x binaria $\Leftrightarrow x\,(x-1) = x^2 - x = 0$
- PLI $\overset{\text{simile}}{\approx}$ NLP
- in realtà la non linearità del problema è "concentrata" nella prescrizione di interezza

Anche se la funzione obiettivo e i vincoli del problema sono lineari, la condizione che le variabili debbano essere intere introduce una non linearità implicita nel problema.

Se x è una variabile binaria (cioè può assumere solo valori 0 o 1)

Poiché i vincoli di interezza e binarietà sono intrinsecamente non lineari, la PLI si avvicina in qualche modo a un problema di ottimizzazione non lineare (NLP).

Il concetto di rilassamento continuo consiste nel rimuovere i vincoli di interezza dal problema di Programmazione Lineare Intera (PLI), trasformando così il problema in un normale problema di Programmazione Lineare (PL). In altre parole, si permette che le variabili x possano assumere qualsiasi valore reale (continui), anziché essere vincolate a valori interi.

Rilassamento continuo di PLI

- rimuovendo il vincolo di interezza:

C(P) rappresenta il problema rilassato, cioè il problema in cui le variabili non sono più costrette a essere intere, che è più semplice da risolvere rispetto al problema originale P

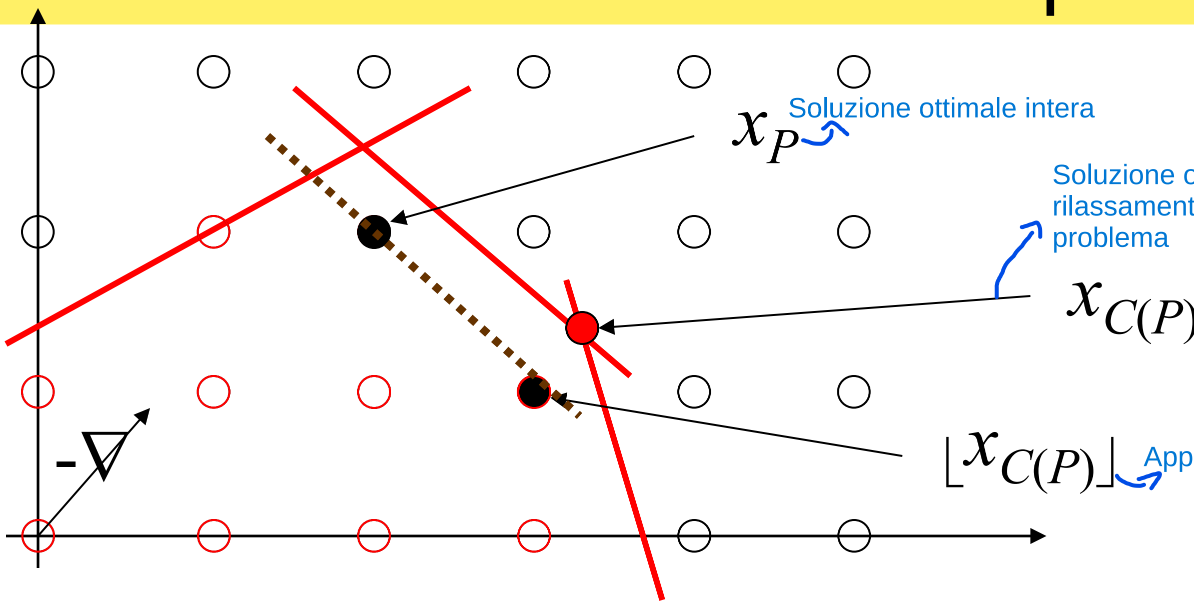
rilassamento continuo $C(P)$ "associato" a P

$$\boxed{z_{C(P)} \leq z_P}$$

zP è il valore ottimale del problema originale (con le variabili intere).
zC(P) è il valore ottimale del problema rilassato (con variabili continue).

Dim.: si cerca il minimo in un insieme più ampio

Poiché il rilassamento continuo consente soluzioni in un insieme più ampio (valori reali anziché interi), il valore ottimale zC(P) del problema rilassato sarà minore o uguale rispetto al valore ottimale zP del problema originale. Questo perché, nel rilassamento continuo, stiamo cercando il minimo in un insieme più grande (insieme delle soluzioni continue) rispetto a quello più ristretto delle soluzioni intere.



Rilassamento continuo (2)

Th.: *se la soluzione del rilassamento continuo è* ***ammissibile*** *per P* ($\overset{\text{cioè}}{=}$ *è intera*),

allora è ***ottima*** *per P*

Se la soluzione del rilassamento continuo rispetta anche i vincoli di interezza del problema originale P, essa è ammissibile per P

Dim.:

1) $z_{C(P)} \leq z_P$

il valore ottimo del rilassamento continuo zC(P) è minore o uguale al valore ottimo del problema intero zP

2) $x_{C(P)}$ è ammissibile per P

la soluzione del rilassamento continuo xC(P) è intera e quindi ammissibile per P

$\Rightarrow \quad z_{C(P)} = \underline{c^T x_{C(P)} \geq z_P} \quad \Rightarrow \quad z_{C(P)} = z_P$

Poiché xC(P) è intera e ammissibile anche per il problema intero, il valore cT xC(P) è una soluzione valida per zP (ma potrebbe non essere ottimale, al momento, quindi >=).

Problemi ed algoritmi

- Algoritmo esatto: determina la soluzione ottima
  - Se il problema è "difficile" il tempo di calcolo necessario ad un algoritmo esatto cresce molto rapidamente (= esponenzialmente) con la dimensione del problema
  - Si risolvono in modo esatto problemi "piccoli"
  - Molti problemi reali sono "difficili" e "grandi"
- Algoritmo euristico o approssimato:

  determina in tempo ragionevole una soluzione ammissibile di "buona" qualità
  - Si risolvono problemi "grandi"
  - In alcuni casi è possibile dare garanzie sulla qualità della soluzione ottenuta (Es. al più il 1.5 volte la soluzione ottima)

Gli algoritmi esatti sono utili per trovare la soluzione ottima, ma sono poco pratici per problemi molto complessi o grandi a causa del tempo di calcolo.
Gli algoritmi euristici o approssimati sono utilizzati per risolvere problemi grandi in tempi più rapidi, ottenendo soluzioni che, pur non essendo ottimali, sono di buona qualità e risolvibili entro un tempo ragionevole.

Algoritmo (euristico) per PLI

begin

Si risolve il problema continuo C(P) tramite il metodo del simplesso, che permette di trovare la soluzione ottima del problema rilassato, senza considerare i vincoli di interezza.

determina con simplesso la soluzione x di $C(P)$

if $C(P)$ impossibile then STOP (P impossibile)

Se C(P) non ha soluzioni, allora anche il problema originale P non ha soluzioni.

else

if $C(P)$ illimitato then STOP (P illimitato, salvo casi particolari)

Se il rilassamento continuo ha una soluzione illimitata, allora è molto probabile che anche il problema originale P sia illimitato

else

if x intero then STOP (x sol. ottima di P)

else

"arrotonda" ogni x_j frazionaria all'intero più vicino

Se la soluzione x ottenuta non è intera, l'algoritmo "arrotonda" ogni xj frazionaria all'intero più vicino. Questo arrotondamento è una tecnica euristica, che cerca di trovare una soluzione ammissibile per il problema intero originale P.

end

Che soluzioni produce questo algoritmo ?

CASO 1: soluzioni utili

- Problemi per cui i valori delle variabili della soluzione ottima sono molto elevati
- Es. pezzi da produrre (elevata quantità)

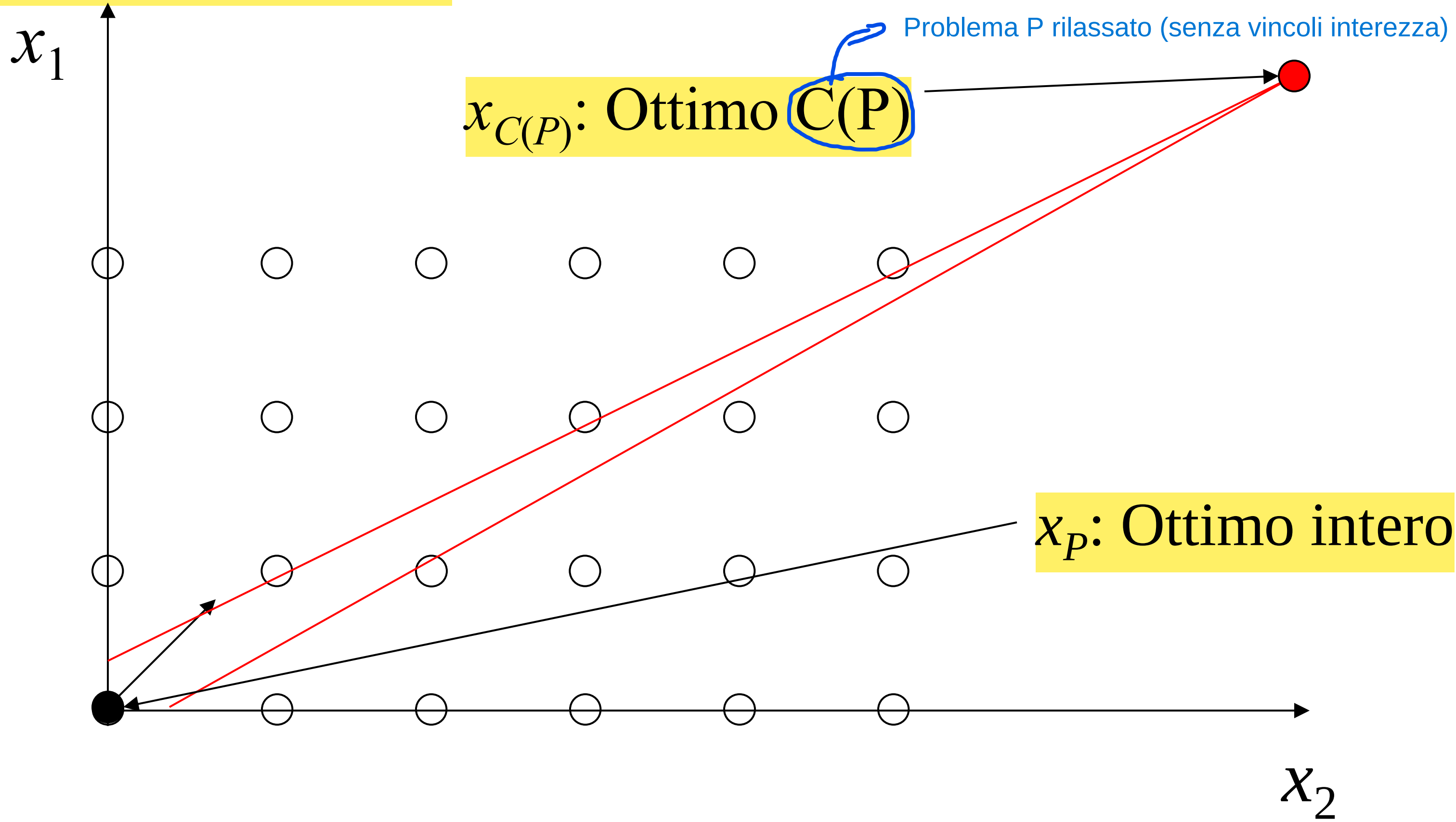
	$C(P)$	$C(P)$ arrotondato
$x_1 =$	2449.51	2450
$x_2 =$	14301.1	14301
$x_3 =$	7800.92	7801
$\max x_1+x_2+x_3$	24551.53	24552
$3x_1+x_2 \leq 21650$	21649.63	21651

CASO 2: soluzioni inutili

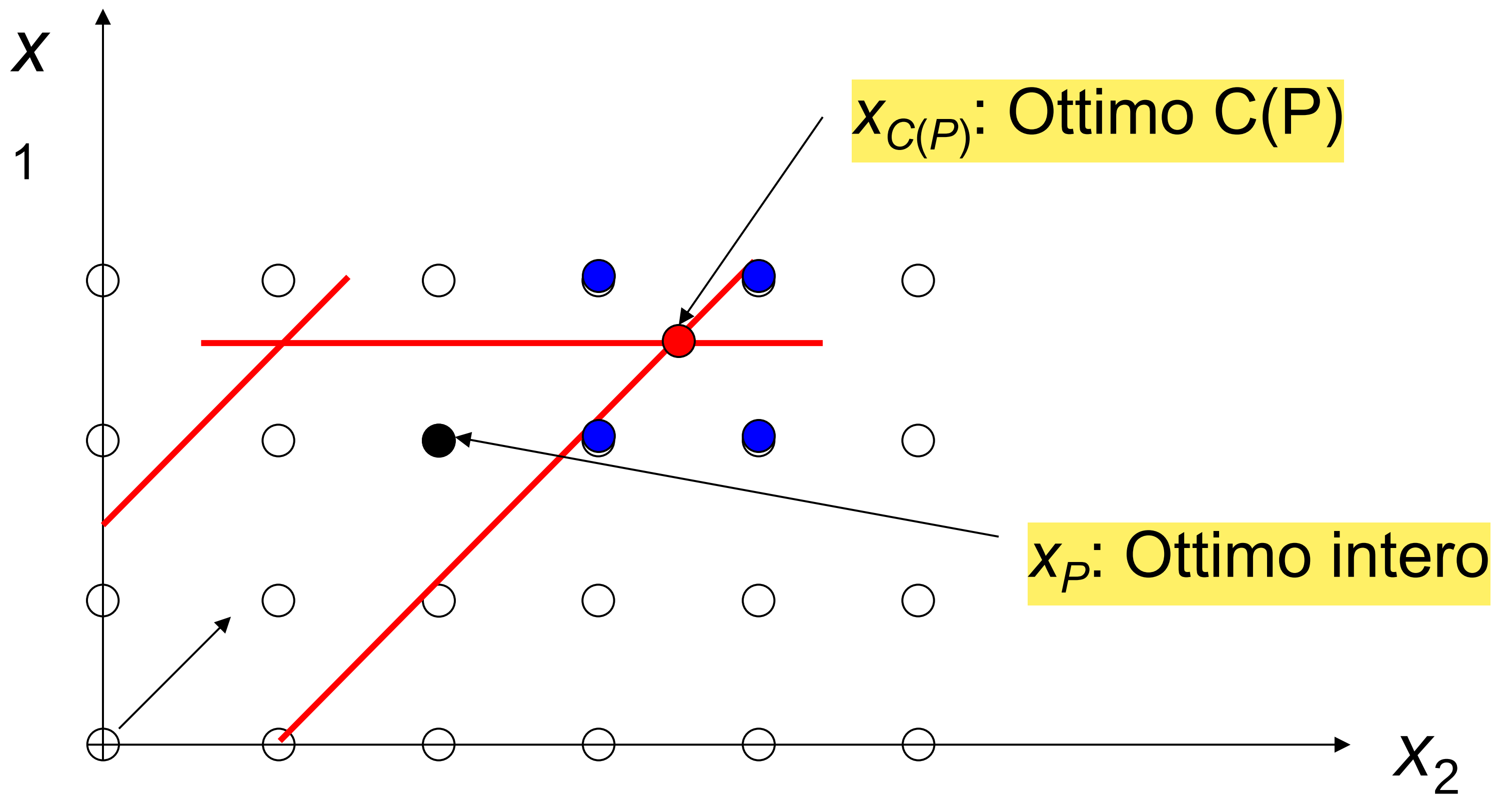
- Problemi in cui i valori delle variabili decisionali all'ottimo sono molto piccoli:
  - Numero di edifici da realizzare
  - Numero di veicoli da assegnare ad un servizio
  - Opportunità di una scelta
  - uso o meno di un tratto di strada in un percorso (sì/no)
  - ………
- La parte frazionaria non è trascurabile e l'arrotondamento può produrre facilmente soluzioni non ammissibili

CASO 2: soluzioni inutili (2)

- Soluzione intera e continua possono essere molto “lontane”



CASO 3: soluzioni non ammissibili



- Nessuno dei quattro punti interi attorno a $x_{C(P)}$ è ammissibile per P

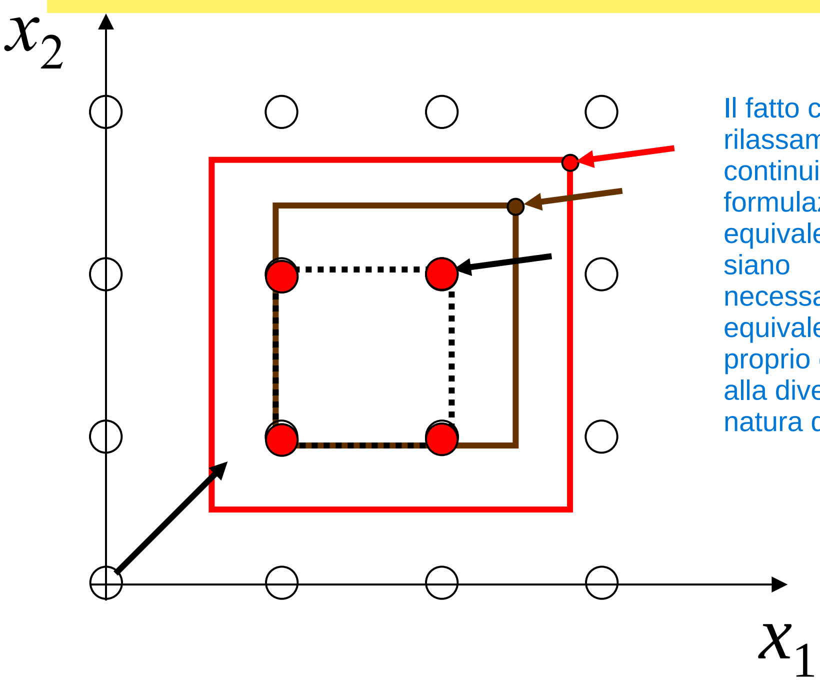
Formulazioni equivalenti

Questa è una formulazione generale del problema.

- dato $z_P = min \{c^T x : x \in X\}$ esistono molte formulazioni equivalenti:

Questa è una formulazione specifica del problema, dove i vincoli sono espressi in forma di disuguaglianze lineari.

$$z_P = min \{c^T x : Ax \geq d, x \geq 0, x \text{ intero}\}$$

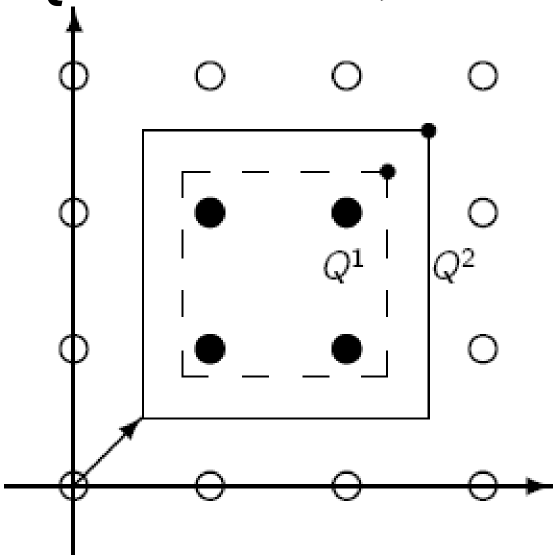


Il fatto che i rilassamenti continui di formulazioni equivalenti non siano necessariamente equivalenti è proprio dovuto alla diversa natura dei vincoli.

- i corrispondenti rilassamenti continui non sono però equivalenti !

Confronto di formulazioni

- Esistono formulazioni migliori di altre ?
- Una formulazione $Q^1 = \{A^1x = d^1, x \geq 0\}$ valida per P è migliore di una formulazione $Q^2 = \{A^2x = d^2, x \geq 0\}$ se $Q^1 \subset Q^2$



Se Q^1 e Q^2 sono due formulazioni di un problema di min con $Q^1 \subset Q^2$, allora $z_{C(Q1)} \geq z_{C(Q2)}$

La "qualità" di una formulazione in PLI dipende da quanto è stretta rispetto al problema originale. Se una formulazione è più stretta (Q1 sottoinsieme di Q2), la soluzione ottimale del rilassamento zC(Q1) sarà più vicina al valore ottimale intero zI.

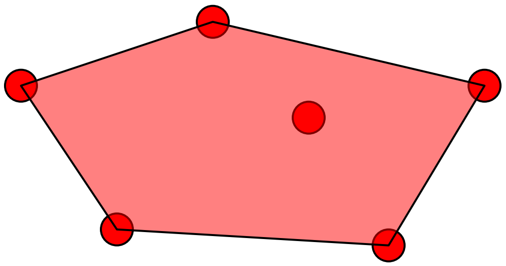
Trovare una formulazione ideale significa cercare la rappresentazione più efficiente e precisa del problema, tale che le sue soluzioni siano facili da trovare e interpretare.

Formulazione "ideale" di PLI

- Esiste una formulazione "ideale" di PLI ?

Def.: *Dato un insieme $S \subseteq R^n$ si dice* convex hull (*guscio convesso*) *di S il più piccolo insieme convesso* conv(S) *che contiene S*

Esempio: Se prendi un insieme di punti in un piano, come diversi vertici di un poligono, il guscio convesso è il poligono più piccolo che può essere disegnato attorno a quei punti. Ogni punto del poligono sarà o un vertice o una combinazione convessa dei vertici.



Formulazione Ideale di PLI: Quando riesci a esprimere un problema PLI attraverso il suo guscio convesso, hai una formulazione ideale. Questo semplifica il problema perché puoi trattarlo come un problema di programmazione lineare su un politopo che ha solo vertici interi, eliminando la necessità di applicare metodi di arrotondamento o di branch and bound.

- Se X è un insieme di punti interi, conv(X) è un politopo $\tilde{P}$ i cui vertici sono tutti punti *interi*

Algoritmi generali per PLI

- Metodi esatti tradizionali (anni 60-oggi):
  - Metodo dei piani di taglio (cutting planes)
  - Branch-and-Bound
  - Programmazione Dinamica
- …
- Metodi esatti più avanzati (anni 90-oggi):
  - Branch-and-Bound + Cutting planes = Branch-and-Cut
  - Branch-and-Price/Column generation