Ottimizzazione Multi–Obiettivo

La *Programmazione Matematica* classica, lineare (PL) o intera (PLI), tratta problemi caratterizzati da una unica e ben definita funzione obiettivo. I problemi di ottimizzazione reali, invece, presentano tipicamente vari obiettivi

- Calcolo di percorsi:
  - Tempo di percorrenza totale
  - Costi totali (pedaggio, carburante, inquinamento,...)
  - Qualità delle strade (accesso ai veicoli pesanti,...)
  - Rischiosità (incidenti, esposizione al rischio per i residenti,...)
  - Rilevanza paesaggistica...
  - ...
- "Portfolio": scelta di *investimenti* (o *progetti*)
  - Profitto
  - Durata
  - Rischio (misurato secondo diversi parametri)
  - ...
- Scheduling:
  - Costo totale
  - Makespan (istante di fine operazioni)
  - Tardiness (ritardi nelle consegne)
  - ...

I metodi algoritmici per la PL e la PLI trovano una o più soluzioni che minimizzano (o massimizzano) una data funzione obiettivo

> *In generale, in un problema multi-obiettivo non esiste una soluzione che ottimizza tutti gli obiettivi!*

Per affrontare i problemi multi-obiettivo occorre dunque:

- definire l'insieme di soluzioni a cui siamo interessati
- sviluppare metodi "ad hoc" per determinare tali soluzioni

I metodi multi-obiettivo si basano generalmente sui metodi mono-obiettivo, o su loro varianti, eventualmente iterate più volte

Ovviamente, il fatto di dover fornire un *insieme* di soluzioni, potenzialmente esteso, complica notevolmente la difficoltà

Nel seguito ci concentriamo sui problemi di PLI multi-obiettivo, con obiettivi di massimo/minimo, e variabili intere/binarie; il generico problema avrà la seguente forma:

$$\begin{aligned} \max / \min\ (z_1, z_2, \ldots, z_p) &= (c_1^T x, c_2^T x, \ldots, c_p^T x) \\ Ax &= b \\ x &\text{ intero/binario} \end{aligned}$$

dove

- p è il numero di obiettivi
- $z_k = c_k^T x$ è la k-esima funzione obiettivo

Esempio

Un'azienda produce tre tipi di oggetti P_j, $j = 1, 2, 3$. Nella tabella sono riportati, per ogni unità di prodotto, il profitto unitario, le ore di lavoro necessarie, la quantità di materia prima utilizzata e il livello di inquinamento generato

	profitto unitario	ore lavoro	materia prima	inquinamento
P_1	10	4	3	10
P_2	9	3	2	6
P_3	8	2	2	3

L'azienda ha a disposizione mano d'opera per 1300 ore di lavoro e 1000 unità di materia prima

L'azienda ha due obiettivi:

- massimizzare il profitto
- minimizzare il livello di inquinamento

$p = 2$: problema *bi-obiettivo*, o più comunemente *bi-criterio*

Per formulare il problema introduciamo le varibili x_j, $j = 1, 2, 3$:
x_j rappresenta le unità di P_j prodotte

Modello matematico (trasformiamo la massimizzazione del profitto $c_1^T x$ in minimizzazione di $-c_1^T x$):

$$\begin{aligned} \min(z_1, z_2) &= (-10x_1 - 9x_2 - 8x_3, \\ &\qquad 10x_1 + 6x_2 + 3x_3) \\ 4x_1 + 3x_2 + 2x_3 &\leq 1300 \\ 3x_1 + 2x_2 + 2x_3 &\leq 1000 \\ x_1, x_2, x_3 &\geq 0, \text{intero} \end{aligned}$$

Soluzione ottenuta massimizzando singolarmente il profitto ($-z_1$):

$z_1 = -4300$

$z_2 = 2400$

$x_1 = 0,\ x_2 = 300,\ x_3 = 200$

Soluzione ottenuta minimizzando singolarmente l'inquinamento (z_2):

$z_1 = 0$

$z_2 = 0$

$x_1 = 0,\ x_2 = 0,\ x_3 = 0$

Come spesso accade nei problemi multi-obiettivo, le due soluzioni sono in netto contrasto tra loro!

Definizioni

Dominanza: date due soluzioni ammissibili $\bar{x}$ e $\bar{x}'$

- in un problema di minimo, $\bar{x}$ domina $\bar{x}'$ se:
  1. $c_i^T \bar{x} \leq c_i^T \bar{x}' \qquad i = 1, \ldots, p$
  2. $\exists k \in \{1, \ldots, p\} : c_k^T \bar{x} < c_k^T \bar{x}'$
- in un problema di massimo, $\bar{x}$ domina $\bar{x}'$ se:
  1. $c_i^T \bar{x} \geq c_i^T \bar{x}' \qquad i = 1, \ldots, p$
  2. $\exists k \in \{1, \ldots, p\} : c_k^T \bar{x} > c_k^T \bar{x}'$

Ottimo di Pareto una soluzione ammissibile $\bar{x}$ è un *ottimo di Pareto*, o *soluzione efficiente*, se non esiste nessuna soluzione ammissibile x che domina $\bar{x}$

Insieme Efficiente: è l'insieme delle soluzioni efficienti

> *A partire da una soluzione efficiente, si può migliorare un obiettivo solo peggiorandone un altro*

Spazio dei Criteri: a ogni soluzione x corrisponde un punto

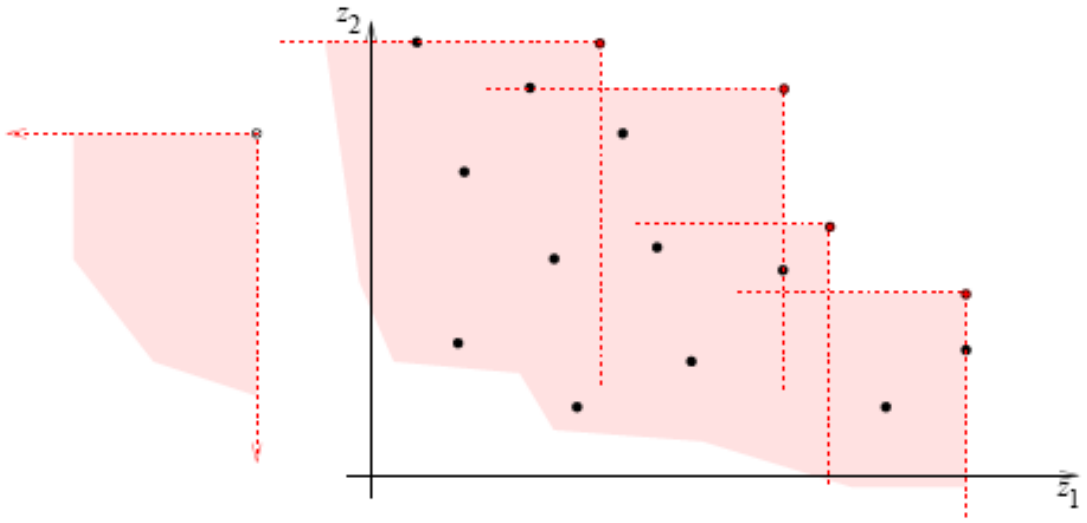
$$z(x) \quad = \quad (z_1, z_2, \ldots, z_p) \quad = \quad (c_1^T x, c_2^T x, \ldots, c_p^T x)$$

nello *spazio dei criteri* (di dimensione p); notare che lo stesso punto può corrispondere a diverse soluzioni

Punto Non-Dominato: un punto nello spazio dei criteri si dice *non-dominato* se corrisponde ad una soluzione efficiente

Frontiera di Pareto: è l'insieme dei punti non dominati

Esempio: caso bi-criterio (problema di massimo)



- 16 punti nello spazio dei criteri; ogni punto domina il quadrante a sinistra in basso
- la frontiera di Pareto è composta da 4 punti non-dominati
- l'insieme efficiente contiene *almeno* 4 soluzioni efficienti

Proprietà: nel caso bicriterio i punti della frontiera di Pareto sono ordinati equivalentemente

- *da sinistra a destra*, cioè per z_1 crescente
- *dal basso verso l'alto*, cioè per z_2 decrescente

Nessuna proprietà analoga vale se $p > 2$

Ricordiamo che l'*inviluppo (o involucro) convesso* di un insieme di punti è il minimo poliedro che contiene tali punti; nel caso bicriterio è un poligono convesso

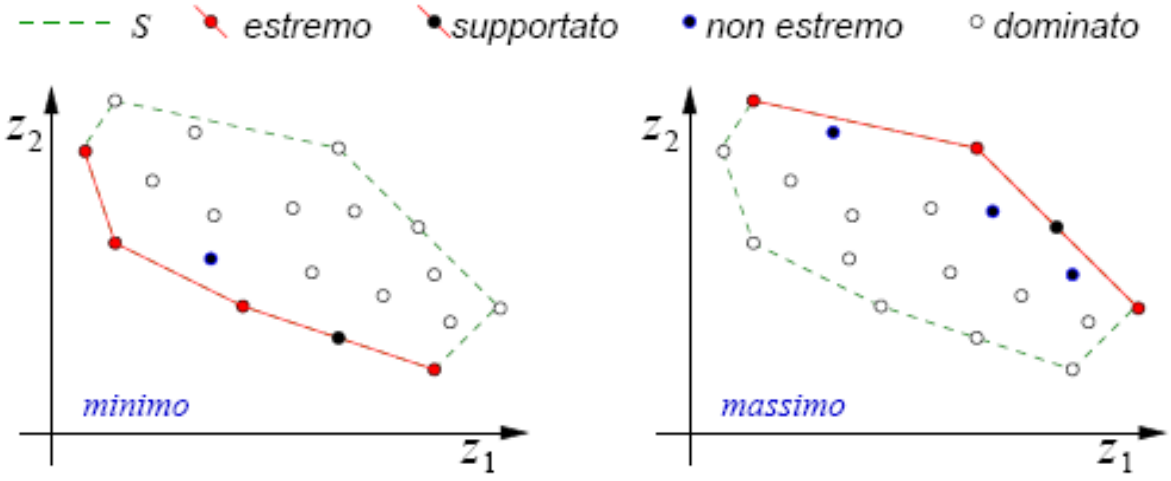
Denotiamo con S l'inviluppo convesso dei punti dello spazio dei criteri corrispondenti a soluzioni ammissibili

I punti della frontiera di Pareto, e analogamente le soluzioni efficienti, possono essere partizionati in:

- *supportati*: giacciono sulla superficie esterna di S
- *non supportati*: giacciono all'interno di S

Un punto supportato si dice *estremo* se è un vertice di S

Esempio: caso bicriterio



Osservazione: *è facile convincersi che se esiste una sola soluzione efficiente questa è estrema, altrimenti esistono almeno due soluzioni estreme*

Metodi Risolutivi

Tipicamente, lo scopo è individuare uno specifico ottimo di Pareto

Questo può tuttavia richiedere di generare tutta la frontiera (ammesso che ci si riesca) o un suo sottoinsieme

Si assume l'esistenza di un *decisore* in grado di selezionare o caratterizzare la soluzione considerata "migliore"

In base al ruolo svolto dal decisore nella strategia di soluzione del problema, i metodi risolutivi possono essere classificati in tre categorie

Metodi a posteriori, nei quali si genera tutto l'insieme efficiente, al cui interno il decisore sceglie la soluzione per lui migliore

Metodi a priori, nei quali il decisore specifica le sue preferenze prima che abbia inizio il processo risolutivo; in base alle informazioni avute dal decisore la ricerca si indirizza verso la soluzione "migliore", senza dover (necessariamente) generare tutti gli ottimi di Pareto

Metodi interattivi, nei quali il decisore specifica le sue preferenze mano a mano che il processo risolutivo procede, eventualmente scartando alcune soluzioni efficienti trovate, e guidando in tal modo il processo stesso verso la soluzione per lui più soddisfacente

Algoritmo ε–constrains

Si tratta di un metodo a posteriori.

Si seleziona una funzione obiettivo k e la si ottimizza, mentre le altre funzioni ($h = 1, \ldots, p$, $h \neq k$) vengono trasformate in vincoli, limitando opportunamente i loro valori

In dettaglio, consideriamo il problema *di minimo*:

$$\begin{aligned} \min\,(z_1, z_2, \ldots, z_p) &= (c_1^T x, c_2^T x, \ldots, c_p^T x) \\ Ax &= b \\ x &\geq 0, \text{ intero} \end{aligned}$$

Scelto $k \in \{1, \ldots, p\}$ definiamo per ogni altra funzioni obiettivo $h \neq k$ un *upper bound* ε_h. Il problema (mono-obiettivo) risultante è:

$$\begin{aligned} \min\ & c_k^T x \\ & c_h^T x \leq \varepsilon_h \qquad h = 1, \ldots, p,\ h \neq k \\ & Ax = b \\ & x \geq 0, \text{ intero} \end{aligned}$$

La soluzione ottima del problema mono-obiettivo *non appartiene necessariamente* alla frontiera di Pareto del problema multi-obiettivo; tuttavia, risolvendo iterativamente con valori ε_i opportunamente aggiornati, è possibile generare tutta la frontiera.

Come vedremo, il metodo risulta particolarmente semplice e intuitivo nel caso bicriterio; per $p > 2$ la scelta dei valori ε_h è più complicata

Nota: nei problemi *di massimo* si usano *lower bound* ($c_h^T x \geq \varepsilon_h$)

Algoritmo ε–constrained
Esempio: Knapsack Bi-Obiettivo

Consideriamo una applicazione in campo finanziario

Sia dato un insieme di n possibili investimenti j, ognuno richiedente un esborso in denaro w_j, e con un profitto p_j e un indice di sicurezza o affidabilità (inversamente proporzionale al rischio) s_j; sia dato inoltre un limite massimo di budget W per l'esborso iniziale

L'obiettivo è quello di selezionare l'insieme di investimenti il cui esborso totale non ecceda W e per cui siano massimi il profitto totale e la sicurezza totale.

Si consideri il caso in cui $n = 14$, $W = 100$ e i dati sono quelli nella tabella sottostante

j	1	2	3	4	5	6	7	8	9	10	11	12	13	14
w_j	16	18	21	9	7	23	14	38	16	41	13	26	31	22
p_j	23	13	53	2	2	8	33	43	13	13	10	11	31	30
s_j	34	44	14	22	11	28	9	7	27	9	15	6	16	10

Determiniamo la frontiera di Pareto con il seguente metodo ε-constrained:

1. Risolviamo un problema di knapsack massimizzando il profitto; siano P ed S il profitto e la sicurezza ottenuti
2. massimizziamo nuovamente il profitto, ma imponendo come vincolo che la sicurezza totale sia maggiore o uguale a $S + 1$
3. iteriamo aggiornando il vincolo sulla sicurezza con l'ultimo valore S ottenuto, fino a quando il modello risulta non ammissibile

Iterazione 1: risolviamo il problema di zaino

$$\begin{array}{lll} \max & p^T x & \\ & w^T x & \leq 100 \\ x_j \in \{0,1\} & & j = 1, \ldots, 14 \end{array}$$

ottenendo profitto massimo P= 159 e sicurezza S = 40, con gli investimenti: 3, 7, 8, 14

Iterazione 2: risolviamo il modello

$$\begin{array}{lll} \max & p^T x & \\ & w^T x & \leq 100 \\ & s^T x & \geq LB_S \\ x_j \in \{0,1\} & & j = 1, \ldots, 14 \end{array} \tag{1}$$

con $LB_S = S + 1 =$ **41**, dove S è il valore di sicurezza ottenuto all'iterazione precedente; in questo modo otteniamo i nuovi valori $P = 154$ e S = 105, con gli investimenti: 1, 3, 5, 7, 9, 14

Iterazione 3: risolviamo il modello (1) con $LB_S =$ **106**, ottenendo P= 154 e S = 122, con gli investimenti: 1, 2, 3, 5, 7, 14
Notare che P è rimasto invariato!

Iterazione 4: risolviamo il modello (1) con $LB_S =$ **123**, ottenendo P= 154 e S = 133, con gli investimenti: 1, 2, 3, 4, 7, 14
Notare che P è rimasto di nuovo invariato!

Iterazione 5: risolviamo il modello (1) con $LB_S =$ **134**, ottenendo P= 145 e S = 143, con gli investimenti: 1, 2, 3, 7, 9, 11

Iterazione 6: risolviamo il modello (1) con $LB_S =$ **144**, ottenendo P= 137 e S = 150, con gli investimenti: 1, 2, 3, 4, 7, 9

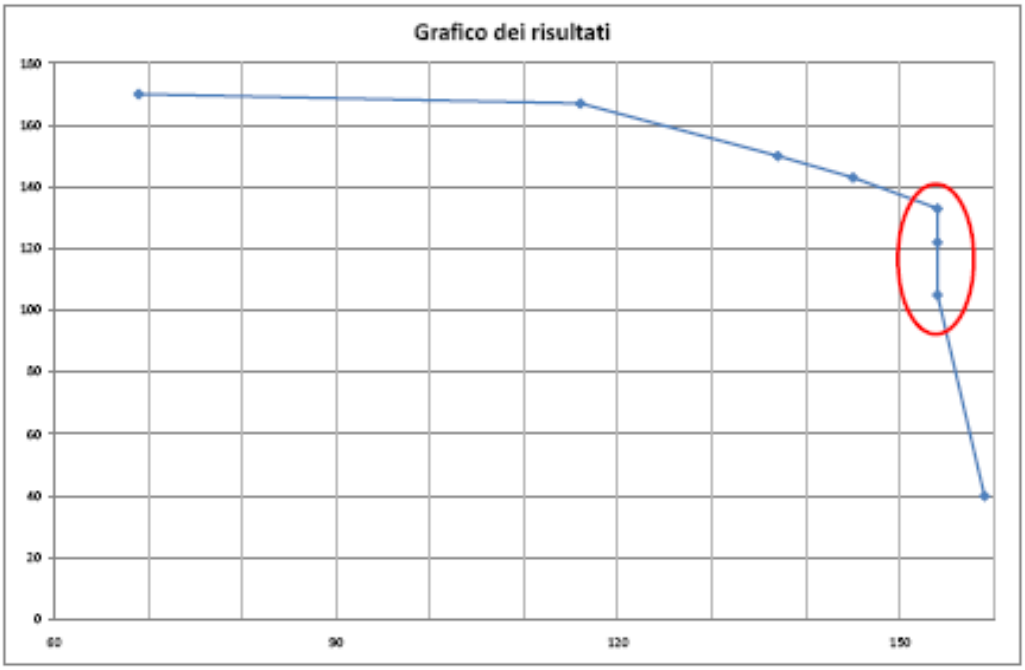
Iterazione 7: risolviamo il modello (1) con $LB_S = \mathbf{151}$, ottenendo P=116 e $S = 167$, con gli investimenti: 1, 2, 3, 4, 5, 9, 11

Iterazione 8: risolviamo il modello (1) con $LB_S = \mathbf{168}$, ottenendo P=69 e $S = 170$, con gli investimenti: 1, 2, 4, 6, 9, 11

Iterazione 9: risolviamo il modello (1) con $LB_S = \mathbf{171}$, e non otteniamo nessuna soluzione ammissibile: il metodo termina

Le coppie (P, S) generate dall'algoritmo sono: $(159, 40)$, $(154, 105)$, $(154, 122)$, $(154, 133)$, $(145, 143)$, $(137, 150)$, $(116, 167)$, $(69, 170)$

Il grafico dei punti corrispondenti a queste coppie è il seguente



Scalarizzazione ("metodo dei pesi")

Si tratta di un metodo a priori.

Il decisore assegna a ciascun obiettivo un "peso" (positivo) che ne riflette la rilevanza sulla scelta

Dunque ad ogni funzione obiettivo c_i si associa un moltiplicatore $\lambda_i > 0$, ottenendo la funzione obiettivo

$$z_\lambda(x) \;=\; \lambda^T(z_1, z_2, \ldots, z_p) \;=\; \sum_{i=1}^{p} \lambda_i c_i^T x$$

come somma pesata (con pesi λ) delle p funzioni obiettivo

Dato il problema:

$$\begin{aligned} \min\,(z_1, z_2, \ldots, z_p) &= (c_1^T x, c_2^T x, \ldots, c_p^T x) \\ Ax &= b \\ x &\geq 0, \text{intero} \end{aligned}$$

si ottiene il problema singolo obiettivo ("scalare")

$$\begin{aligned} \min\; z_\lambda(x) &= \textstyle\sum_{i=1}^{p} \lambda_i c_i^T x \\ Ax &= b \\ x &\geq 0, \text{intero} \end{aligned}$$

(e analogamente per un problema di massimo)

Teorema *(Geoffrion, 1968)* Ogni soluzione ottima del problema scalare è una soluzione efficiente *supportata* del problema multiobiettivo

Scalarizzazione: Esempio (Knapsack Bi-Obiettivo)

Consideriamo il problema di zaino bi-obiettivo definito in precedenza per l'applicazione in campo finanziario

Un decisore abbastanza propenso al rischio potrebbe assegnare peso $\lambda_P = 2$ all'obiettivo "profitto", e peso $\lambda_S = 1$ all'obiettivo "sicurezza"

Il problema singolo obiettivo risultante è il seguente:

$$\begin{array}{lrcll} \max & \lambda_P \cdot p^T x + \lambda_S \cdot s^T x & = & \sum_{j=1}^{14} (2p_j + s_j) x_j & \\ & w^T x & \leq & 100 & \\ & x_j & \in & \{0,1\} & j = 1, \ldots, 14 \end{array}$$

Il valore ottimo è 441 (ottenuto con gli investimenti 1, 2, 3, 4, 7, 14) e corrisponde alla soluzione efficiente con $P = 154$ e $S = 133$ trovata all'iterazione **4** del metodo ε-constrained

Un decisore decisamente avversio al rischio potrebbe invece assegnare pesi $\lambda_P = 1$ e $\lambda_S = 3$, ottenendo il problema:

$$\begin{array}{lrcll} \max & \lambda_P \cdot p^T x + \lambda_S \cdot s^T x & = & \sum_{j=1}^{14} (p_j + 3s_j) x_j & \\ & w^T x & \leq & 100 & \\ & x_j & \in & \{0,1\} & j = 1, \ldots, 14 \end{array}$$

Il valore ottimo è 617 (ottenuto con gli investimenti 1, 2, 3, 4, 5, 9, 11) e corrisponde alla soluzione efficiente con $P = 116$ e $S = 167$ trovata all'iterazione **7** del metodo ε-constrained