Cenni di Programmazione Lineare Intera

Domenico Salvagnin

2011-12-05

Un problema di programmazione lineare intera (MIP) consiste nella minimizzazione di una funzione lineare soggetta ad un numero finito di vincoli lineari, con in piú il vincolo che alcune variabili devono assumere valori interi. In generale si ha:

$$\begin{aligned} \min\ & cx \\ & a_i x \sim b_i \quad i = 1, \dots, m \\ & l_j \leq x_j \leq u_j \quad j = 1, \dots, n = N \\ & x_j \in \mathbb{Z} \quad \forall j \in J \subseteq N = \{1, \dots, n\} \end{aligned} \tag{1}$$

Se $J = N$, si parla di programmazione lineare intera pura, altrimenti di programmazione lineare intera mista. Ovviamente se $J = \emptyset$ si ha un semplice problema di programmazione lineare.

1 Algoritmo Branch & Bound

L'algoritmo branch-and-bound (B&B) è una tecnica di ottimizzazione generica che si basa su una enumerazione intelligente dell'insieme F delle soluzioni ammissibili di un problema di ottimizzazione combinatoria. L'algoritmo B&B ottimizza la ricerca sfruttando un principio molto semplice: non esplorare una regione dello spazio delle soluzioni se si può dimostrare che non contiene soluzioni migliori di quelle note.

L'algoritmo B&B usa una strategia *divide-and-conquer* per partizionare lo spazio di ricerca del problema originario in sottoproblemi, e poi li ottimizza separatamente. Ecco una breve descrizione del metodo.

Sia F l'insieme delle soluzioni ammissibili di un problema di minimizzazione, $c : F \to \mathbb{R}$ la funzione obiettivo da minimizzare e $\overline{x} \in F$ una soluzione ammissibile nota, determinata ad esempio euristicamente. Il costo $z = c(\overline{x})$ di tale soluzione ammissibile, che prende il nome di incumbent, è un upper bound sul costo della soluzione ottima. L'algoritmo B&B consta dei seguenti passi:

- Nella fase di bounding, si rilassa il problema, ammettendo soluzioni che non appartengono all'insieme F, ma ad un suo sovrainsieme $G \supseteq F$. La soluzione del rilassamente fornisce pertanto un lower bound sul valore della soluzione ottima. Se la soluzione di tale rilassamento appartiene a F o ha costo uguale a z, allora si ha finito: la nuova soluzione (se ammissibile) o quella nota è ottima.

- Altrimenti, si identifica una *separazione* F^* di F, vale a dire un insieme finito F^* di sottoinsiemi di F, tale che:

$$\bigcup_{F_i \in F^*} F_i = F$$

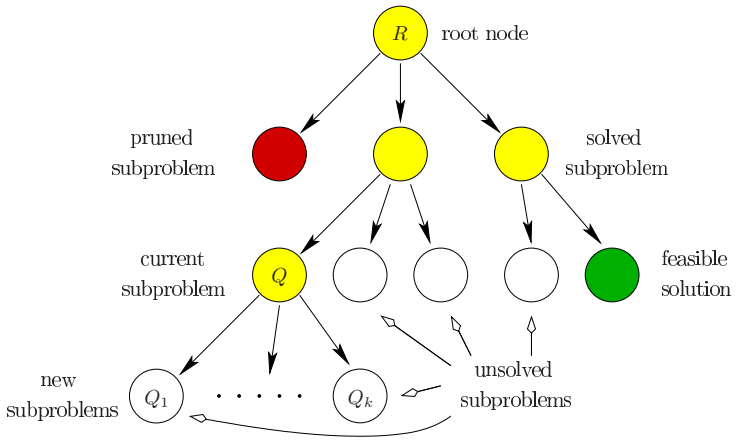

Figure 1: Ricerca ad albero di un algoritmo B&B.

Ogni sottoinsieme F_i è detto figlio di F. Tale fase, detta *branching*, è giustificata dal fatto che se F^* è una separazione, allora:

$$\min\{c(x) \mid x \in F\} = \min\{\min\{c(x) \mid x \in F_i\} \mid F_i \in F^*\}$$

F^* è spesso, anche se non necessariamente, una *partizione* di F. I figli di F vengono aggiunti alla coda $\mathcal{A}$ dei sottoproblemi da processare.

- Si seleziona un sottoproblema P_i dalla coda e se ne risolve un rilassamento. Ci sono quattro possibili casi:

  1. Se si trova una soluzione ammissibile migliore dell'incumbent $\overline{x}$, allora si sostituisce $\overline{x}$ con la nuova soluzione e si continua.
  2. Se il sottoproblema non ammette soluzione, allora si scarta (*pruning by infeasibility*).
  3. Altrimenti, si confronta il valore del rilassamento con l'upper bound globale z. Se è maggiore o uguale a z, allora è ancora possibile scartare il sottoproblema (*pruning by optimality*), in quanto non può portare ad una soluzione migliore di quella che si conosce già.
  4. Infine, se non è stato possibile scartare il problema, è necessario fare ulteriormente branching e aggiungere i figli del sottoproblema alla lista dei sottoproblemi da processare.

- Si continua in questo modo fino a che la lista dei sottoproblemi non diventa vuota. Quando ciò avviene, l'incumbent corrente è la soluzione ottima.

Lo pseudocodice per l'algortimo B&B è presentato in Figura 1, mentre una rappresentazione grafica della corrispondente ricerca ad albero è presentata nell'Algoritmo 1.

Vale la pena notare che:

- Il metodo è usabile anche senza un incumbent. In tale caso, l'algoritmo è ancora corretto, anche se in generale è molto meno efficace, in quanto è possibile scartare un sottoproblema solo per feasibility e non per optimality.

Algorithm 1: Schema generale B&B

S1: $\mathcal{A} \leftarrow \{P_0\}$, $z = \infty$
S2: Scegli un sottoproblema P_i da $\mathcal{A}$. Se la coda $\mathcal{A}$ è vuota, vai al passo S8
S3: Risolvi il rilassamento di P_i. Se la soluzione è ammissibile, vai al passo S6
S4: Se $g(P_i) \geq z$ vai al passo S7
S5: Scegli una separazione F_j^* per $F(P_j)$ e aggiungi i corrispondenti problemi ad $\mathcal{A}$
S6: Se $f(P_i) < z$, allora aggiorna $z = f(P_i)$ e l'incumbent $\overline{x}$
S7: $\mathcal{A} \leftarrow \mathcal{A} \setminus \{P_i\}$. Vai al passo S2
S8: Se $z = \infty$ il problema è impossibile, altrimenti la soluzione $\overline{x}$ è ottima

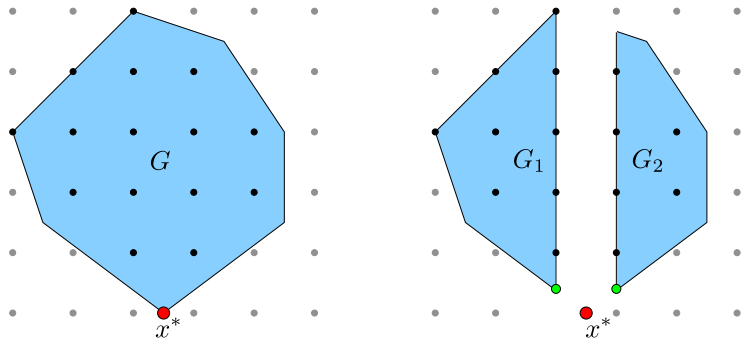

Figure 2: Branching sulla soluzione ottima del rilassamento lineare x^*.

- La scelta del rilassamento è una delle scelte piú importanti per un algoritmo B&B. Deve infatti essere scelto in modo da essere relativamente facile da risolvere, ma dare allo stesso tempo un lower bound ragionevole.

L'algoritmo B&B generico appena scritto si specializza in maniera immediata per la risoluzione di problemi MIP: dobbiamo semplicemente specificare come calcolare rilassamenti e separazioni. Per quanto riguarda il rilassamento, la scelta piú diffusa consiste nell'usare il rilassamento lineare dei sottoproblemi. Se la soluzione di tale rilassamento non è intera (si noti che il vincolo di interezza sulle variabili è l'unico rilassato), allora possiamo scegliere una x_j con un valore frazionario x_j^* in tale soluzione e considerare la seguente partizione:

$$x_j \leq \lfloor x_j^* \rfloor \vee x_j \geq \lceil x_j^* \rceil$$

Tale processo è illustrato in Figura 2.

2 Algoritmo Branch & Cut

Una versione migliorata dell'algoritmo B&B, sviluppata specificatamente per il caso MIP, è nota come *branch-and-cut* (B&C).

L'algoritmo B&C è essenzialmente un algoritmo ibrido B&B/piani di taglio, basato sulla seguente semplice idea: ad ogni nodo dell'albero decisionale, la formulazione associata al rilassamento del sot-

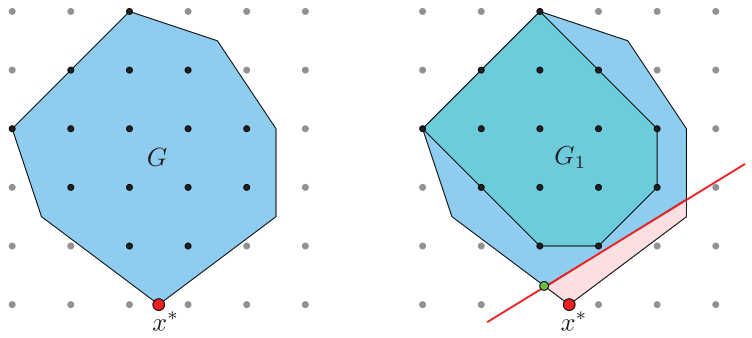

Figure 3: Rafforzamento di un rilassamento mediante piani di taglio.

toproblema corrente può essere rafforzata mediante la generazione di piani di taglio (vedi Figura 3). Tale rafforzamento è fatto per due motivi:

- per ottenere una soluzione intera del rilassamento lineare
- per ottenere un lower bound migliore e quindi più efficace per il pruning

Tale tecnica è spesso molto efficace e risolve molti dei difetti degli algoritmi B&B e a piani di taglio puri. Nonostante l'idea di base sia molto semplice, l'implementazione è peró tutt'altro che banale.

Infine, l'approccio B&C richiede l'esistenza di procedure efficienti per la risoluzione del seguente problema di *separazione*: data una soluzione frazionaria x^*, trovare una disugualianza valida $\alpha^T x \leq \alpha_0$, se esiste, violata da x^*, cioè tale che $\alpha^T x^* > \alpha_0$.