Tecniche di Preprocessing in C++ applicate a problemi lineari misto-interi (MIP)

Università degli Studi Di Milano

Marco Odore

20 ottobre 2016

Indice

1 Scopo del lavoro

Il lavoro propone una possibile implementazione in C++ di alcune delle tecniche di preprocessing applicate ai problemi di ottimizzazione lineare misto-interi (MIP) e binary (BIP). Per validare la correttezza del software è stato realizzato un generatore randomico di problemi MIP/BIP da sottomettere poi in AMPL.

1.1 Tecniche implementate

Sono state implementate diverse tecniche adatte a diversi contesti della programmazione lineare:

- Riduzione dei bound sulle variabili (Bounds Tightening)
- Ricerca di vincoli non soddisfacibili (Detecting Infeasibility)
- Eliminazione di vincoli ridondanti (Detecting Redundant Constraints)
- Fissaggio delle variabili (Variables Fixing)
- Riduzione dei coefficienti nei problemi BIP (Coefficients Reduction)

2 Bounds Tightening

La riduzione dei bound è una tecnica applicabile a tutti i tipi di variabili, e cioè a quelle di tipo continuo, intero e binario.

Questo metodo di preprocessing consiste nell'iterare sui vincoli del problema verificando la presenza di bound migliori per ogni variabile esaminata. Il procedimento continua finché, una volta iterati tutti i vincoli, ci sono stati degli aggiornamenti sui bound.

Procedimento:

- Per ogni vincolo si considerano separatamente le variabili con coefficienti positivi da quelle con coefficienti negativi:
$$\sum_{a_{ij}>0} a_{ij}x_j + \sum_{a_{ij}<0} a_{ij}x_j \leq b_i \quad \forall i$$
- Si isola una variabile k alla volta, cercando di ottimizzare i suoi bound:
$$a_{ik}x_k + \sum_{a_{ij}>0} a_{ij}x_j + \sum_{a_{ij}<0} a_{ij}x_j \leq b_i$$
- Si passa poi al calcolo del possibile nuovo bound, che nel caso di variabile con coefficiente positivo sarà il nuovo upperbound u'_k, mentre nel caso di variabile con coefficiente negativo sarà il nuovo lowerbound l'_k:

$$if \;\; a_{ij} > 0 :$$
$$u'_k = \frac{1}{a_{ik}} \left(b_i - \sum_{j \neq k, a_{ij}>0} a_{ij}x_j + \sum_{a_{ij}<0} a_{ij}x_j \right)$$
$$if \;\; a_{ij} < 0 :$$
$$l'_k = \frac{1}{a_{ik}} \left(b_i - \sum_{a_{ij}>0} a_{ij}x_j + \sum_{j \neq k, a_{ij}<0} a_{ij}x_j \right)$$

- Dopo i calcoli, i bound vengono effettivamente aggiornati, ponendo $u_k = u'_k$ o $l_k = l'_k$ se e solo se $u'_k < u_k$ o $l'_k > l_k$.

Nel caso le variabili siano intere (o binarie) viene semplicemente fatto il ceiling nel caso di nuovi lower bound ($\lceil l'_k \rceil$) e floor nel caso di nuovi upper bound ($\lfloor u'_k \rfloor$).

3 Detecting Infeasibility and Variables Fixing

Per verificare se un vincolo è non soddisfacibile va calcolato prima di tutto il suo *lower bound* L_i, che va poi confrontato con il termine noto b_i:

$$L_i = \sum_{a_{ij}>0} a_{ij} l_{x_j} + \sum_{a_{ij}<0} a_{ij} u_{x_j}$$

Se $L_i > b_i$, allora il vincolo risulta chiaramente insoddisfacibile e non esiste quindi una soluzione.
Se invece $L_i = b_i$, si possono fissare le variabili seguendo questo criterio:

$$\begin{array}{ll} \forall a_{ij} > 0 & \forall a_{ij} < 0 \\ x_j = l_{x_j} & x_j = u_{x_j} \end{array}$$

4 Detecting Redundant Constraints

Per individuare vincoli ridondanti viene calcolato l'*upper bound* U_i del vincolo che si sta controllando, che poi viene confrontato con il suo termine noto b_i:

$$U_i = \sum_{a_{ij}>0} a_{ij} u_{x_j} + \sum_{a_{ij}<0} a_{ij} l_{x_j}$$

Se $U_i \leq b_i$ allora il vincolo viene rispettato sempre, qualunque sia il valore assunto dalle variabili, e quindi può essere eliminato.

5 Coefficients Reduction

Questa tecnica può essere applicata solo a problemi di programmazione lineare binaria (BIP). Consiste nel ridurre la grandezza dei coefficienti dei vincoli eseguendo una serie di operazioni.

Procedimento:

- Si considera un vincolo alla volta:

$$\sum_{a_{ij}>0} a_{ij} x_j + \sum_{a_{ij}<0} a_{ij} x_j \leq b_i$$

- Si ottiene il valore M pari alla somma dei coefficienti positivi del vincolo i:

$$M = \sum_{a_{ij}>0} a_{ij}$$

- Viene poi creato un set S di coefficienti tali che il loro valore assoluto è maggiore di $M - b_i$:

$$S = \{a_{ij} : |a_{ij} > M - b_i\}$$

- Se il set S non è vuoto seleziona un coefficiente a_k al suo interno, e se $a_k > 0$:

$$a'_k = M - b_i$$
$$b'_i = M - a_k$$
$$a_k = a'_k$$
$$b_i = b'_i$$

- mentre se $a_k < 0$:

$$a'_k = b_i - M$$
$$a_k = a'_k$$

Il procedimento viene iterato sul singolo vincolo finché il set S ha almeno un elemento al suo interno.

6 Preprocessing in C++

Le tecniche citate sono state implementate in c++ sfruttando la programmazione a oggetti, cercando di realizzare un'astrazione dei tipi di variabili presenti all'interno dei problemi di programmazione MIP (continue, intere e binarie), in maniera tale da garantirne le proprietà intrinseche.