Università degli Studi di Padova

Dipartimento di Ingegneria dell'Informazione

Corso di Laurea Triennale in Ingegneria Informatica

Esperimenti di programmazione lineare intera per problemi di vertex cover

Relatore
Prof. Domenico Salvagnin

Laureando
Giacomo Camposampiero

Anno Accademico 2020/2021

Padova, 19 luglio 2021

Text.

Indice

1 Introduzione **1**

1.1 Programmazione lineare intera 2

1.1.1 Algoritmo Branch and Bound 2

1.1.2 Algoritmo Branch and Cut 3

2 Presentazione delle sperimentazioni **5**

3 Risultati sperimentali **7**

4 Conclusioni **9**

Bibliografia **11**

Capitolo 1

Introduzione

La ricerca operativa, dall'inglese *operational research*, è una disciplina scientifica relativamente giovane, nata con lo scopo di fornire strumenti matematici di supporto ad attività decisionali in cui occorre gestire e coordinare attività e risorse limitate. La ricerca operativa permette infatti di trovare, mediante la formalizzazione matematica di un problema, una soluzione ottima o ammissibile (quando possibile) al problema stesso. Costituisce di fatto un approccio scientifico alla risoluzione di problemi complessi, che ha trovato grande applicazione in moltissimi ambiti, non ultimo quello industriale.

Tra le diverse branche in cui si divide la ricerca operativa vi è anche l'ottimizzazione. Quest'ultima si occupa principalmente di problemi che comportano la minimizzazione o la massimizzazione di una funzione, detta funzione obbiettivo, sottoposta a un dato insieme di vincoli. Un problema di ottimizzazione può essere formulato come

$$\begin{aligned} &\min(or\ \max) f(x) \\ &S \\ &x \in D \end{aligned} \tag{1.1}$$

dove $f(x)$ è una funzione a valori reali nelle variabili x, D è il dominio di x e S un insieme finito di vincoli. In generale, x è una tupla $(x_1, ..., x_n)$ e D è un prodotto cartesiano $D_1 \times ... \times D_n$, e vale $x_j \in D_j$.

Un problema nella forma (1.1) risulta essere intrattabile, ovvero non esistono algoritmi efficienti (o non esiste proprio alcun algoritmo) per la sua risoluzione. Si rende quindi necessario considerare dei casi particolari di questa formulazione, i quali possiedono determinate proprietà che possono essere sfruttate nella definizione di algoritmi ad-hoc.

1.1 Programmazione lineare intera

Uno dei casi particolari della formulazione generale (1.1) a cui si è precedentemente fatto riferimento viene trattato dalla programmazione lineare intera. Un problema di programmazione lineare intera consiste nella minimizzazione (o massimizzazione) di una funzione lineare soggetta ad un numero finito di vincoli lineari, con in aggiunta il vincolo che alcune delle variabili del problema debbano assumere valori interi. In generale, il problema può quindi essere riformulato come

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

Se $J = N$ si parla di programmazione lineare intera pura, altrimenti di programmazione lineare intera mista (o MIP, dall'inglese *Mixed Integer Programming*).

La programmazione lineare intera restringe quindi notevolmente il tipo di vincoli a disposizione nel processo di formalizzazione matematica del problema, determinando una maggior difficoltà in fase di modellazione del problema. Tuttavia, questo non comporta eccessive restrizioni, almeno per la MIP, sui tipi di problemi che possono essere formulati secondo questo paradigma. Alcuni esempi di problemi risolvibili mediante MIP sono *knapsack*, problemi di *scheduling*, *facility location* e, naturalmente, *vertex covering* (di cui si discuterà più approfonditamente di seguito nella Sezione **TODO**).

Allo stesso tempo, l'introduzione dei vincoli di linearità ed interezza comporta notevoli vantaggi nella definizione ed implementazione di algoritmi risolutivi, di cui sono in seguito riportati alcuni esempi.

1.1.1 Algoritmo Branch and Bound

L'algoritmo branch-and-bound (B&B) è una tecnica di ottimizzazione generica basata sull'enumerazione dell'insieme delle soluzioni ammissibili di un problema di ottimizzazione combinatoria, introdotta nel 1960 da A. H. Land e A. G. Doig [1]. Questo algoritmo permette la gestione del problema dell'esplosione combinatoria mediante il taglio di intere porzioni dello spazio delle soluzioni, che può essere effettuato quando si riesce a dimostrare che queste ultime non contengono soluzioni migliori di quelle note, e una strategia *divide and conquer*, che permette di dividere lo spazio di ricerca in partizioni e risolvere ognuna di esse separatamente. Si riporta di seguito una breve descrizione dell'algoritmo.

Sia F l'insieme delle soluzioni ammissibili di un problema di minimizzazione (simmetrico nel caso della massimizzazione, a meno di un cambio di segno della funzione obiettivo), $c : F \to \mathbb{R}$ la funzione obiettivo e $\bar{x} \in F$ una soluzione ammissibile nota, generata mediante euristiche o mediante assegnazioni casuali. Sia $z = f(\bar{x})$ il costo di tale soluzione nota, anche detto *incumbent*, che rappresenta per sua natura un limite superiore al valore della soluzione ottima. L'algoritmo Branch and Bound si articola quindi in due diverse

1.1.2 Algoritmo Branch and Cut

Capitolo 2

Presentazione delle sperimentazioni

Capitolo 3

Risultati sperimentali

Capitolo 4

Conclusioni

Bibliografia

[1] A. H. Land e A. G. Doig. «An Automatic Method of Solving Discrete Programming Problems». In: *Econometrica* 28.3 (1960), pp. 497–520. ISSN: 00129682, 14680262. URL: http://www.jstor.org/stable/1910129.