Complementi ed Esercizi di Informatica Teorica II

Vincenzo Bonifaci

25 giugno 2008

9 Problemi di ottimizzazione: Set Cover

Il problema della COPERTURA DI INSIEME (in inglese *Set Cover*) è il seguente: dati un insieme finito U, una collezione di suoi sottoinsiemi $\mathcal{S} = \{S_1, S_2, \ldots, S_\ell\}$ e una funzione di costo $c : \mathcal{S} \to \mathbb{Q}_+$, trovare una sottocollezione $\mathcal{S}' \subseteq \mathcal{S}$ tale $\bigcup_{S \in \mathcal{S}'} C = U$; l'obiettivo è minimizzare il costo della sottocollezione scelta.

Possiamo formulare il problema attraverso il seguente programma lineare 0/1. Utilizziamo una variabile x_S per ogni insieme $S \in \mathcal{S}$, che è pari a 1 se e solo se l'insieme S è selezionato nella soluzione. Il vincolo è che per ogni elemento $e \in U$ esista almeno un insieme contenente e tra quelli selezionati.

$$\begin{aligned} \min \quad & \sum_{S \in \mathcal{S}} c(S) x_S \\ & \sum_{S : e \in S} x_S \geq 1, \quad \forall\, e \in U \\ & x_S \in \{0,1\}, \qquad \forall\, S \in \mathcal{S}. \end{aligned} \tag{1}$$

In seguito vedremo come usare il rilassamento frazionario di questa formulazione per ottenere un algoritmo approssimato per Set Cover.

9.1 Un algoritmo goloso

In questa sezione analizziamo il seguente algoritmo per Set Cover: selezioniamo iterativamente l'insieme dal migliore rapporto $\frac{\text{elementi coperti}}{\text{costo}}$ e rimuoviamo gli elementi coperti, continuando così finché tutti gli elementi non sono stati coperti. Diamo una descrizione più precisa. Sia C l'insieme degli elementi già coperti all'inizio di una iterazione. Durante questa iterazione, definiamo il *rapporto costo-efficacia* di un insieme S come il costo medio al quale copre nuovi elementi, ovvero $c(S)/|S \setminus C|$. Definiamo il *prezzo* di un elemento come il rapporto costo-efficacia dell'insieme dal quale è coperto per la prima volta. In altre parole, quando un insieme S viene selezionato, possiamo pensare che il suo costo sia distribuito equamente tra i nuovi elementi coperti da S, e che la corrispondente quota vada a formare il loro prezzo. Si veda il riquadro Algoritmo 1 per una descrizione in pseudocodice dell'algoritmo.

Algoritmo 1 L'algoritmo goloso per Set Cover

```
C := ∅
while C ≠ U do
    Trova l'insieme S con rapporto costo-efficacia minimo.
    Sia α = c(S)/|S \ C| il rapporto costo-efficacia di S.
    Includi S nella soluzione e poni prezzo(e) = α per ogni e ∈ S \ C.
    C := C ∪ S.
end while
Restituisci gli insiemi selezionati.
```

Per analizzare l'algoritmo, numeriamo gli elementi di U nell'ordine in cui sono coperti dall'algoritmo (a parità di ordine, ovvero se alcuni elementi sono stati coperti nella stessa iterazione, stabiliamo un ordine arbitrario tra di essi). Sia $e_1, \ldots, e_n$ un tale ordinamento.

Lemma 9.1 *Per ogni* $k \in \{1, \ldots, n\}$*,* $\text{prezzo}(e_k) \le \text{opt}/(n-k+1)$.

Dimostrazione. Ad ogni iterazione, gli insiemi della soluzione ottima $\mathcal{S}^*$ sono in grado di coprire gli elementi residui ad un costo totale al più pari ad opt. Il costo medio al quale vengono così coperti gli elementi è al più $\text{opt}/|U \setminus C|$. Ora, tra questi insiemi, deve essercene almeno uno con rapporto costo-efficacia pari al più a $\frac{\text{opt}}{|U \setminus C|}$: infatti, se tutti gli insiemi della soluzione ottima necessari a coprire $U \setminus C$ avessero rapporto maggiore, considerando il loro costo complessivo avremmo

$$\begin{aligned}\sum_{S \in \mathcal{S}^*} c(S) \ge \sum_{S \in \mathcal{S}^*} \frac{c(S)}{|S \setminus C|} |S \setminus C| &> \sum_{S \in \mathcal{S}^*} \frac{\text{opt}}{|U \setminus C|} |S \setminus C| \\ &= \text{opt} \sum_{S \in \mathcal{S}^*} \frac{|S \setminus C|}{|U \setminus C|} \ge \text{opt},\end{aligned}$$

contraddicendo il fatto che il costo complessivo è al più opt (l'ultima disuguaglianza segue dal fatto che $(U \setminus C) \subseteq \cup_{S \in \mathcal{S}^*}(S \setminus C)$).

Ora, durante l'iterazione in cui l'elemento e_k viene coperto, $U \setminus C$ contiene almeno $n-(k-1)$ elementi. Poiché e_k è coperto dall'insieme con rapporto costo-efficacia minimo, abbiamo

$$\text{prezzo}(e_k) \le \frac{\text{opt}}{|U \setminus C|} \le \frac{\text{opt}}{n-k+1}.$$

□

Teorema 9.2 *L'Algoritmo 1 è un algoritmo H_n-approssimato per* COPERTURA DI INSIEME*, dove* $H_n = 1 + \frac{1}{2} + \ldots + \frac{1}{n}$.

Dimostrazione. Poiché il costo di ogni insieme selezionato è distribuito equamente tra i nuovi elementi coperti, il costo totale della soluzione è pari a

$\sum_{k=1}^{n} \text{prezzo}(e_k)$. Dal Lemma precedente, questa quantità è al più $(1 + \frac{1}{2} + \ldots + \frac{1}{n}) \cdot \text{opt}$. $\square$

Lemma 9.3 *Per ogni $n \geq 1$, $\ln n \leq H_n \leq 1 + \ln n$.*

Dimostrazione. Si consideri la funzione reale $x \mapsto 1/x$. Per definizione di integrale,

$$\int_1^n \frac{1}{x} \leq \sum_{i=1}^{n} \frac{1}{i} = 1 + \sum_{i=2}^{n} \frac{1}{i} \leq 1 + \int_1^n \frac{1}{x}\,dx.$$

Calcolando gli integrali, otteniamo

$$[\ln x]_1^n \leq H_n \leq 1 + [\ln x]_1^n$$

$$\ln n \leq H_n \leq 1 + \ln n.$$

$\square$

Corollario 9.4 *L'Algoritmo 1 è un algoritmo $O(\log n)$-approssimato per* COPERTURA DI INSIEME.

9.2 Un algoritmo di arrotondamento

In questa sezione mostriamo un algoritmo di arrotondamento (*rounding*) per il problema del Set Cover. Sia f la frequenza dell'elemento che più frequentemente appare nei sottoinsiemi dati, ovvero $f = \max_{e \in U} |\{i : e \in S_i\}|$. L'algoritmo ci permetterà di ottenere un fattore di approssimazione pari ad f. Notiamo che su alcune istanze f può essere più piccolo di H_n, mentre su altre può capitare l'opposto.

L'algoritmo è basato sul rilassamento frazionario del programma lineare intero (1). Questo rilassamento si ottiene rimuovendo i vincoli di interezza sulle variabili, ottenendo il seguente programma lineare:

$$\begin{aligned} \min \quad & \sum_{S \in \mathcal{S}} c(S) x_S \\ & \sum_{S : e \in S} x_S \geq 1, \quad \forall\, e \in U \\ & 0 \leq x_S \leq 1, \qquad \forall\, S \in \mathcal{S}. \end{aligned} \tag{2}$$

Dato che la programmazione lineare ammette algoritmi polinomiali, possiamo ottenere in tempo polinomiale una soluzione (frazionaria) x che soddisfa tutti i vincoli del programma (2) e tale che $\sum_S c(S) x_S = \text{opt}_{\text{LP}}$, dove opt_{LP} è il valore ottimo del programma (2). Notare che $\text{opt}_{\text{LP}} \leq \text{opt}$, poiché l'insieme delle soluzioni del programma (1) è contenuto nell'insieme delle soluzioni del programma (2). Naturalmente non abbiamo ancora risolto il problema originario, perché ora abbiamo una soluzione frazionaria anziché intera. Per questo motivo useremo una procedura di arrotondamento che converte la soluzione frazionaria in una soluzione intera senza aumentarne eccessivamente il costo.

La procedura di arrotondamento è la seguente. Definiamo per ogni $S \in \mathcal{S}$

$$\hat{x}_S = \begin{cases} 1, & \text{se } x_S \geq 1/f \\ 0, & \text{se } x_S < 1/f. \end{cases}$$

L'algoritmo restituisce tutti gli insiemi S per cui $\hat{x}_S = 1$. Lo pseudocodice è dato nel riquadro 2.

Algoritmo 2 L'algoritmo di arrotondamento per Set Cover

Trova una soluzione ottima del programma lineare (2).
Restituisci gli insiemi S per i quali $x_S \geq 1/f$.

Teorema 9.5 *L'Algoritmo 2 è un algoritmo f-approssimato per* COPERTURA DI INSIEME.

Dimostrazione. Per iniziare, spieghiamo perché la soluzione $\hat{x}$ rappresenta una soluzione ammissibile del Set Cover. Consideriamo un elemento arbitrario $e \in U$. Poiché e compare in al più f insiemi, e poiché $\sum_{S:S \ni e} x_S \geq 1$, per almeno un insieme contenente e dobbiamo avere $x_S \geq 1/f$. Ma allora $\hat{x}_S = 1$ per costruzione, e quindi l'elemento e è coperto da qualche insieme nella soluzione restituita dall'algoritmo.

Per quanto riguarda il costo, notiamo semplicemente che per come è effettuato l'arrotondamento abbiamo $\hat{x}_S \leq f \cdot x_S$ per ogni S. Concludiamo che il costo della soluzione restituita dall'algoritmo è

$$\sum_{S \in \mathcal{S}} c(S)\hat{x}_S \leq \sum_S c(S)(f x_S) \leq f \cdot \sum_S c(S) x_S \leq f \cdot \mathrm{opt}_{\mathrm{LP}} \leq f \cdot \mathrm{opt}.$$

□

Notare che il problema del Vertex Cover può essere visto come un caso particolare di Set Cover in cui $f = 2$, dato che ogni elemento (arco) può essere contenuto in al più due insiemi (archi incidenti a ciascun vertice). Come corollario otteniamo quindi un algoritmo 2-approssimato per Vertex Cover.