Complementi ed Esercizi di Informatica Teorica II

Vincenzo Bonifaci

23 giugno 2008

6 Problemi di ottimizzazione: Commesso Viaggiatore

Il problema del COMMESSO VIAGGIATORE (in inglese *Traveling Salesman Problem* o TSP) è il seguente: dato un insieme di città e una funzione distanza tra coppie di città, determinare un percorso chiuso che visiti ogni città *esattamente* una volta (ad eccezione di quella di partenza che è anche quella di arrivo) minimizzando la distanza totale percorsa. Formalmente, sia $V = \{1, \ldots, n\}$ l'insieme di città e sia $d : V^2 \to \mathbb{Z}_+$ la funzione distanza. Una soluzione è una permutazione $\pi : V \to V$ e il suo costo è dato da

$$\sum_{i=1}^{n-1} d(\pi(i), \pi(i+1)) + d(\pi(n), \pi(1)).$$

Un caso particolarmente interessante per le applicazioni (TSP *metrico* o Δ-TSP) è quello in cui la funzione distanza è simmetrica e soddisfa la disuguaglianza triangolare:

$$\begin{aligned} d(x,y) &= d(y,x) && \forall x, y \in V \\ d(x,z) &\leq d(x,y) + d(y,z) && \forall x, y, z \in V. \end{aligned}$$

Esercizio 6.1. Dimostrare che il TSP con la disuguaglianza triangolare è equivalente al seguente problema: dato un insieme di città e una funzione distanza, determinare un percorso chiuso che visiti ogni città *almeno* una volta, minimizzando la distanza totale percorsa. Per equivalente qui si intende che è possibile trasformare ogni istanza del primo problema in un'istanza del secondo in maniera tale che ogni soluzione di costo C della seconda istanza può essere convertita in una soluzione di costo al più C della prima istanza; e viceversa scambiando i ruoli dei due problemi.

Suggerimento: in un verso, mostrare che l'omissione dal percorso di un nodo già visitato non incrementa il costo totale. Nell'altro verso, usare la funzione distanza che associa a due nodi x e y la lunghezza di un cammino minimo da x a y.

6.1 L'algoritmo dell'albero ricoprente minimo

Ricordiamo che per il TSP metrico esiste un algoritmo 2-approssimato basato sul calcolo di un albero ricoprente minimo nel grafo completo su n nodi in cui gli archi sono pesati secondo la funzione distanza. Dopo aver calcolato un albero ricoprente a peso minimo, l'algoritmo considera il multigrafo ottenuto prendendo due copie di ciascun arco, ne calcola un cammino chiuso Euleriano (ovvero un cammino chiuso che attraversa ogni *arco* esattamente una volta) e infine converte questo cammino Euleriano in un ciclo che tocca ogni nodo esattamente una volta, semplicemente omettendo le occorrenze successive alla prima di ciascun nodo (ovvero "scorciando" il cammino).

Mostriamo qui che esiste una famiglia di istanze che mostra che l'analisi dell'algoritmo è stretta, ovvero non migliorabile. Infatti su queste istanze l'algoritmo trova soluzioni che hanno un fattore di approssimazione che tende a 2 quando il numero di nodi cresce.

Si consideri l'istanza raffigurata in Figura 1(a). Si tratta di una istanza non solo metrica, ma Euclidea, per cui i nodi sono punti del piano Euclideo e le distanze sono pari alle distanze Euclidee. Un albero ricoprente a peso minimo per questa istanza è rappresentato in Figura 1(b) (il fatto che sia questo l'albero a peso minimo può essere verificato applicando per esempio l'algoritmo di Kruskal all'istanza in Figura 1(a)). Un possibile ciclo restituito dall'algoritmo a partire dall'albero trovato è rappresentato in Figura 1(c); si noti che per $\epsilon \to 0$, la lunghezza del ciclo tende a $4L$. D'altra parte la soluzione rappresentata in Figura 1(d) ha peso (per $\epsilon \to 0$) pari a $2L + 2$. Il rapporto di approssimazione risultante è quindi $4L/(2L+2)$. Poiché L può essere reso arbitrariamente grande (semplicemente aumentando il numero di nodi), il rapporto di approssimazione può essere reso arbitrariamente vicino a 2.

6.2 L'algoritmo di Christofides

Si ricordi l'algoritmo di Christofides per il TSP metrico:

1. Costruisci un albero ricoprente T a peso minimo nel grafo completo su n nodi in cui gli archi sono pesati secondo la funzione distanza;
2. Costruisci un *matching* M^* a peso minimo sui nodi di grado dispari di T;
3. Trova un cammino chiuso Euleriano per il multigrafo che è l'unione di T ed M^*, e convertilo in un ciclo che tocca ogni nodo esattamente una volta, con la tecnica delle "scorciatoie".

È noto che l'algoritmo di Christofides è un algoritmo 1.5-approssimato per il TSP metrico. Mostriamo qui che esiste una famiglia di istanze che dimostra che l'analisi dell'algoritmo è stretta.

Si consideri l'istanza Euclidea raffigurata in Figura 2. In questa istanza il numero di nodi è 11. È facile verificare che esiste una soluzione di costo pari a 11 (che deve essere ottima perché la distanza tra qualunque due nodi è almeno 1, e ci sono 11 nodi). L'algoritmo di Christofides trova come albero ricoprente

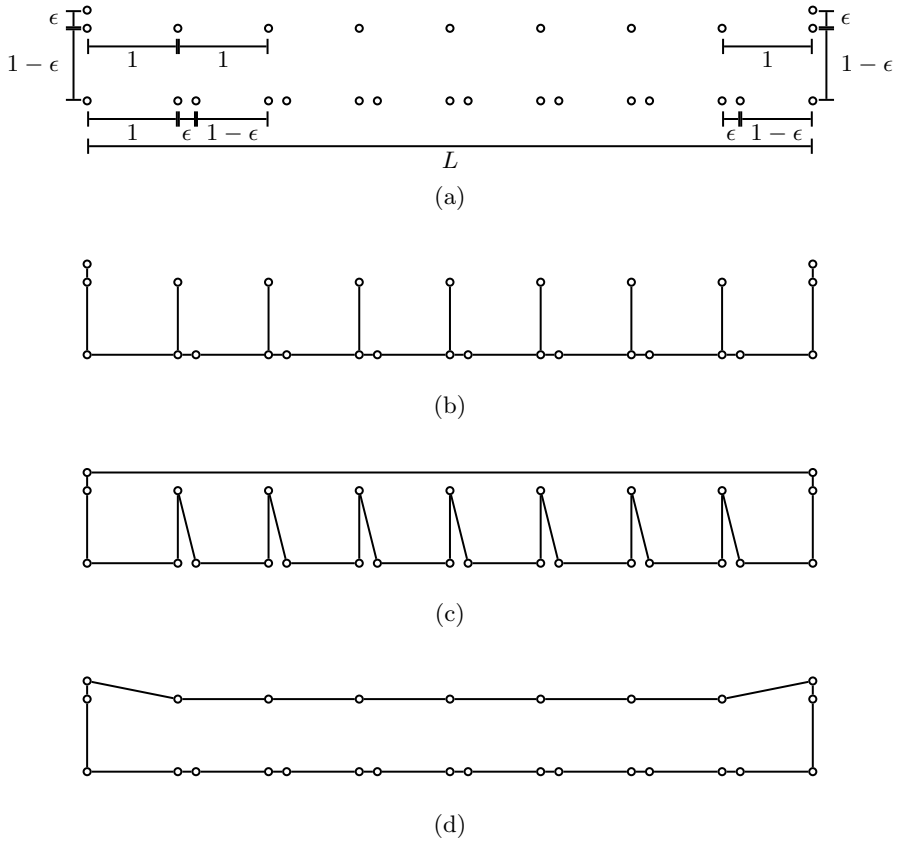

Figura 1: Controesempio per l'algoritmo dell'albero ricoprente minimo. (a) Istanza di TSP metrico (Euclideo). (b) Albero ricoprente minimo. Lunghezza $\simeq 2L+1$. (c) Soluzione restituita dall'algoritmo. Lunghezza $\simeq 4L$. (d) Soluzione ottima. Lunghezza $\simeq 2L + 2$.

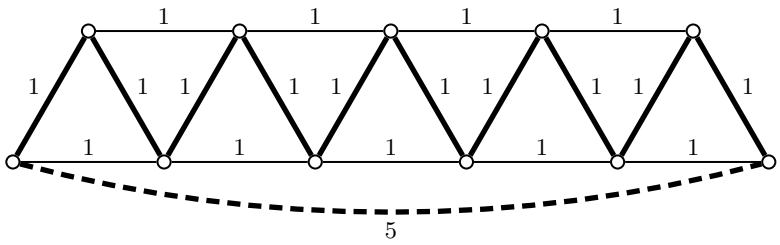

Figura 2: Controesempio per l'algoritmo di Christofides

l'insieme di archi segnato con le linee in grassetto, di peso 10. Poi vi aggiunge il matching dato dal singolo arco di peso 5 segnato con una linea tratteggiata. Infine la terza fase dell'algoritmo semplicemente non fa nulla, perché il cammino così ottenuto non contiene nodi ripetuti. Il costo totale è 15, per cui il rapporto è 15/11. Generalizzando l'esempio otteniamo istanze di n nodi con rapporto pari a

$$\frac{n-1+\frac{1}{2}(n-1)}{n},$$

che per $n \to \infty$ tende a 1.5.

6.3 Ricerca locale: 2-opt

Si consideri l'algoritmo di ricerca locale 2-opt per COMMESSO VIAGGIATORE con funzione distanza simmetrica. In questo algoritmo, il vicinato di una soluzione è dato dall'insieme dei cicli che si possono ottenere togliendo e aggiungendo due archi alla soluzione corrente. Seguendo l'approccio tipico della ricerca locale, l'algoritmo parte da una soluzione costruita in maniera arbitraria (ad esempio generata casualmente o tramite un'euristica) e sostituisce di volta in volta la soluzione corrente con quella di costo minore nel suo vicinato, fino a fermarsi in un minimo locale.

La Figura 3 raffigura una instanza per la quale 2-opt termina con una soluzione arbitrariamente lontana dall'ottimo. Nella figura, gli archi sottili hanno costo $1/\alpha n$, l'arco dal nodo 1 al nodo $n = 8$ ha costo pari a 1 e tutti gli archi non disegnati hanno costo molto grande, ad esempio $2n$. La soluzione ottima è data dal ciclo $s^* = (1\,2\,7\,8\,3\,4\,5\,6)$ ed ha costo $1/\alpha$. La soluzione $s = (1\,2\,3\,4\,5\,6\,7\,8)$, invece, ha costo $1 + \frac{n-1}{\alpha n}$. Inoltre s è anche un minimo locale per 2-opt (come invitiamo il lettore a verificare): rimuovendo e aggiungendo due archi si ottengono solo soluzioni di costo superiore al costo di s (si noti come per ottenere la soluzione s^* è necessario rimpiazzare *tre* archi di s). Ne concludiamo che se 2-opt usa come soluzione iniziale s, terminerà immediatamente restituendo s e il rapporto di approssimazione sarà

$$\frac{1+(n-1)/\alpha n}{1/\alpha} = \frac{\alpha n + n - 1}{n} > \alpha.$$

Dato che α può essere fissato in maniera arbitraria, concludiamo che 2-opt non ha un rapporto di approssimazione limitato. Notiamo che la stessa costruzione è valida anche quando il numero di vertici n è maggiore di 8: è sufficiente inserire i nodi in più tra il nodo 4 e il nodo 5. Infine, questa costruzione è un caso particolare di una famiglia di esempi che mostra che anche l'euristica di ricerca locale k-opt, in cui si tolgono e aggiungono k archi alla volta, per k costante, puà dare in generale soluzioni di costo arbitrariamente lontano dall'ottimo.

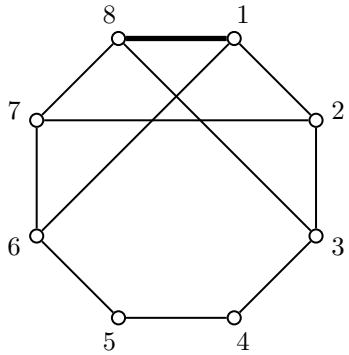

Figura 3: Controesempio per 2-opt