Appunti su algoritmi approssimati per il Problema del Commesso Viaggiatore (TSP)

A. Agnetis*

1 Algoritmi approssimati ed euristici

Il Problema del Commesso Viaggiatore (Traveling Salesman Problem, TSP) è uno dei problemi combinatori più importanti e più studiati. Applicazioni del TSP si possono trovare nei campi più disparati, dalla produzione manifatturiera all'instradamento di veicoli, al progetto di circuiti integrati... Inoltre il TSP stesso nasce come sottoproblema di problemi ancora più complessi, in cui ad esempio occorre pianificare i tragitti che contemporaneamente diversi veicoli dovranno compiere.

Per il TSP sono state studiate a fondo formulazioni di programmazione intera e approcci poliedrali (*branch and cut*), che hanno dato risultati estremamente significativi, tanto da consentire, grazie anche ai progressi dei calcolatori e a un'accurata ingegnerizzazione del software, la soluzione esatta di istanze di TSP con diverse migliaia di nodi. Tuttavia, non sempre si hanno il tempo e i mezzi di calcolo necessari per utilizzare strumenti metodologici e algoritmici cosi' sofisticati. In queste note vogliamo affrontare il problema in un'ottica diversa, ossia quella di rinunciare alla ricerca della soluzione ottima in cambio di un considerevole risparmio computazionale. Gli algoritmi che vedremo qui non assicurano quindi di fornire la soluzione ottima al problema, ma solo una soluzione approssimata. Tra questi algoritmi vanno distinte due classi: gli algoritmi che forniscono una garanzia a priori sulla distanza dall'ottimo della soluzione fornita (algoritmi *ρ-approssimati*), e quelli che non danno alcuna garanzia a priori, ma che comunque, da una verifica a posteriori (di tipo statistico), si rivelano comunque efficaci. A quest'ultima categoria vanno ascritte le cosiddette meta-euristiche, ossia schemi euristici che possono essere adattati a molti diversi problemi di ottimizzazione, e che, va detto, hanno datto ottimi risultati pratici negli ultimi anni (tabu search, simulated annealing, algoritmi genetici...). Di questi si parla in un'altra sezione. Qui vogliamo occuparci degli algoritmi approssimati.

*Dipartimento di Ingegneria dell'Informazione - Università di Siena

Si consideri un algoritmo per un problema di ottimizzazione, e sia z_A il valore della funzione obiettivo della soluzione fornita dall'algoritmo, mentre z^* è il valore della soluzione ottima. Se l'algoritmo non assicura di trovare la soluzione ottima, ma garantisce di trovare una soluzione che non è superiore a ρ volte il valore dell'ottimo ($\rho > 1$), ossia $z_A \leq \rho z^*$, allora l'algoritmo si dice ρ-approssimato. Chiaramente, siamo interessati a trovare algoritmi in cui il valore di ρ sia il più possibile vicino a 1, e aventi però complessità non proibitiva.

Nel seguito, vedremo due algoritmi molto efficienti in grado di fornire soluzioni ρ-approssimate per un caso particolare del TSP. Ricordiamo la formulazione del TSP (in forma di riconoscimento). Nel seguito, $\hat{K}_p(d)$ indica il grafo completo orientato con p nodi, con pesi d_{ij} sugli archi.

TSP

- *Dati:* L'insieme dei pesi d_{ij} degli archi del grafo $\hat{K}_n(d)$; un intero h.
- *Domanda:* Esiste un ciclo hamiltoniano su $\hat{K}_n(d)$ di costo $\leq h$?

Un caso particolare di TSP si ha quando la matrice C dei pesi è simmetrica, ossia $d_{ij} = d_{ji}$ per ogni coppia di nodi i, j, e soddisfa la *disuguaglianza triangolare*, ossia, presi comunque tre nodi i, j, k si ha che

$$d_{ij} + d_{jk} \geq d_{ik}. \tag{1}$$

In questo caso, si parla di Δ-TSP. Un caso rilevante di questo tipo si ha quando i pesi rappresentano distanze euclidee tra punti nel piano (si parla allora di *TSP euclideo*). Infatti, una caratteristica della funzione distanza è proprio quella di soddisfare le (1). Ci chiediamo a questo punto se aver considerato un caso particolare di TSP renda il problema più facile o no.

Teorema 1 *Δ-TSP è NP-completo.*

Dim. Anzitutto, Δ-TSP $\in NP$ (al pari di TSP). Mostriamo ora che CICLO HAMILTONIANO $\rightarrow$ Δ-TSP. Data dunque un'istanza di CICLO HAMILTONIANO, ossia un grafo $G(N, A)$ non orientato, costruiamo l'istanza di Δ-TSP nel seguente modo. Sia $n = |N|$, e consideriamo il grafo $\hat{K}_n(d)$, ove il peso dell'arco (i, j) è cosi' definito: se $(i, j) \in A$, allora $d_{ij} = 1$, altrimenti $d_{ij} = 2$. Infine, poniamo $h = |N|$. Si noti che i pesi sono definiti in modo tale che valgono tanto la simmetria (in quanto per ogni coppia i, j, i pesi d_{ij} e d_{ji} sono ambedue pari a 1 o ambedue pari a 2) quanto la disuguaglianza triangolare (in quanto nella (1) il termine di sinistra vale almeno 2 e quello di destra al più 2).E'

immediato verificare che esiste un ciclo hamiltoniano in G se e solo se esiste, in $\hat{K}_n(d)$, un ciclo hamiltoniano di peso non superiore a n. Infatti, un tale ciclo, essendo formato da n archi, sarà composto tutto da archi di costo 1, corrispondenti cioè ad archi (non orientati) di G, che formano dunque un ciclo hamiltoniano su G. Viceversa, se tale ciclo su G non esiste, non potremo trovare su $\hat{K}_n(d)$ un ciclo hamiltoniano di costo non superiore a n. □

In un'istanza di TSP generica, per 'proibire' l'uso di un determinato arco nella soluzione ottima, si può pensare di porre il peso di quell'arco pari a un valore arbitrariamente elevato. Si osservi che in un'istanza di Δ-TSP questo non si può avere, ossia tutti gli archi del grafo completo sono percorribili.

2 Un algoritmo 2-approssimato per il Δ-TSP

Un primo algoritmo approssimato per il Δ-TSP è il seguente. Si consideri il grafo completo pesato $\hat{K}_n(d)$. Indichiamo con C^* il ciclo hamiltoniano di peso minimo e con T l'albero ricoprente di peso minimo, mentre z^* e $z(T)$ indicheranno i rispettivi pesi. Il ciclo C^*, essendo hamiltoniano, è costituito da n archi. Togliendo un qualsiasi arco da C^* si ottiene dunque un albero ricoprente, di peso senz'altro non maggiore, dal momento che i pesi sono tutti non negativi. Dunque,

$$z(T) \leq z^* \tag{2}$$

Ora, dato l'albero T, *raddoppiamo* tutti i suoi archi, ottenendo cosi' un multigrafo $\bar{T}$. Tale multigrafo sarà euleriano, in quanto tutti i nodi hanno grado pari. Il peso di un ciclo euleriano su tale multigrafo è pari alla somma dei pesi di tutti gli archi, ossia $2z(T)$. Possiamo allora definire un ciclo hamiltoniano nel seguente modo. Percorrendo un ciclo euleriano sul grafo $\bar{T}$, annotiamo la successione dei nodi via via visitati, ad esempio:

$$1 — 2 — 3 — 2 — 4 — 5 — 4 — 6 — 4 — 2 — 1$$

Chiaramente, alcuni nodi (invero, tutti tranne le foglie) saranno visitati più di una volta. Ora, scorrendo nuovamente la lista dall'inizio, cancelliamo tutte le ripetizioni di ciascun nodo:

$$1 — 2 — 3 — — 4 — 5 — — 6 — — —$$

Quello che rimane è un ciclo hamiltoniano C_A. Qual è il peso di tale ciclo hamiltoniano? Vediamo di darne un upper bound. Il ciclo C_A segue molti archi del ciclo euleriano, ma ne "bypassa" alcuni. Ad esempio, C_A passa dal nodo 3 al nodo 4, mentre il ciclo euleriano ripassava dal nodo 2. Ricapitolando:

Algoritmo ALBERO;

1. Data la matrice dei pesi D, calcola l'albero ricoprente di peso minimo T;
2. Raddoppia tutti gli archi di T, ottenendo il multigrafo euleriano $\bar{T}$;
3. Dato un ciclo euleriano su $\bar{T}$, si calcoli un ciclo hamiltoniano C_A cancellando tutti gli archi ripetuti.

Grazie alla disuguaglianza triangolare, allora, possiamo concludere che la lunghezza complessiva del ciclo hamiltoniano cosi' ottenuto, $z(C_A)$, non è superiore a quella del ciclo euleriano, ossia:

$$z(C_A) \leq 2z(T) \tag{3}$$

e dunque, ricordando che $z(T) \leq z^*$, si ha:

$$\frac{z(C_A)}{z^*} \leq 2 \tag{4}$$

in definitiva, risulta dimostrato il seguente risultato:

Teorema 2 *Il ciclo hamiltoniano prodotto dall'algoritmo ALBERO è 2-approssimato.* □

Venendo alla complessità dell'algoritmo, osserviamo che il passo più oneroso è il calcolo dell'albero ricoprente di costo minimo, che può ad esempio essere svolto in $O(n^2)$ con l'algoritmo di Prim-Dijkstra.

Certamente un errore massimo del 100% non è poco. Tuttavia, va detto che *nel caso medio* le prestazioni di questo algoritmo sono molto superiori a quelle del caso peggiore.

3 Un algoritmo 3/2-approssimato per il Δ-TSP

Vediamo ora un algoritmo un po' più complesso, che consente di avere un'approssimazione migliore. Utilizzando le stesse notazioni di cui sopra, anche ora consideriamo l'albero ricoprente di peso minimo T. In tale albero, come su ogni grafo, avremo un numero pari di nodi di grado dispari. Sia Q tale insieme di nodi, e sia q la sua cardinalità. Possiamo allora considerare il sottografo completo $K_q(d)$, in cui compaiono solo i nodi di Q, e in cui gli archi sono pesati, sempre con i pesi $d(i, j)$. Su questo grafo cerchiamo il matching perfetto di peso minimo. Tale problema è risolubile in tempo polinomiale. Chiamiamo M tale matching ottimo, e $z(M)$ il suo peso. Se noi aggiungiamo all'albero ricoprente T gli archi di M, otteniamo un grafo G_E euleriano, in quanto il grado di tutti i nodi che avevano grado dispari in T aumenta di 1. Anche qui, come prima, percorriamo un ciclo euleriano ed eliminiamo poi i nodi già visitati, ottenendo un ciclo hamiltoniano C_A.

Algoritmo di Christofides (1980);

1. Data la matrice dei pesi D, calcola l'albero ricoprente di peso minimo T;
2. Calcola il matching perfetto di peso minimo sul sottografo di $K_n(d)$ indotto da tutti i nodi che hanno grado dispari in T; sia M tale matching
3. Aggiungi gli archi di M a T, ottenendo il grafo euleriano G_E;
4. Dato un ciclo euleriano su G_E, si calcoli un ciclo hamiltoniano C_A cancellando tutti gli archi ripetuti.

Vogliamo mostrare che il peso $z(C_A)$ del ciclo prodotto dall'algoritmo MATCHING non si discosta oltre un certo limite dall'ottimo z^*. Anzitutto, il peso di un ciclo euleriano su G_E è pari a $z(T)+z(M)$, e dunque, grazie alla disuguaglianza triangolare, dal momento che C_A è stato ottenuto bypassando alcuni nodi del suddetto ciclo euleriano, si ha

$$z(C_A) \leq z(T) + z(M) \tag{5}$$

Consideriamo ora il ciclo hamiltoniano ottimo C^*. Questo ciclo attraversa tutti i nodi del grafo, e in particolare dunque quelli dell'insieme Q. Se noi allora cancelliamo da C^* tutti i nodi che non sono in Q, e supponiamo invece di collegare questi direttamente fra loro, otteniamo un ciclo C_Q che ovviamente, sempre per la disuguaglianza triangolare, ha lunghezza complessiva $z(C_Q)$ tale che

$$z(C_Q) \leq z^* \tag{6}$$

Il ciclo C_Q ha esattamente q archi. Ricordando che q è un numero pari, possiamo dividere gli archi del ciclo C_Q in due insiemi alternati: gli archi pari (il secondo, il quarto, ..., il q-esimo) e gli archi dispari (il primo, il terzo,..., il $((q-1)$-esimo). Ciascuno di questi due insiemi costituisce un matching perfetto tra i nodi di Q, ed ha quindi peso complessivo non inferiore a $z(M)$. In definitiva dunque

$$z(C_Q) \geq 2z(M) \tag{7}$$

e quindi, dalle (6) e (7):

$$z^* \geq 2z(M) \tag{8}$$

Dalle (2), (7) e (8) si ha quindi

$$z(C_A) \leq z(T) + z(M) \leq z^* + (1/2)z^* \tag{9}$$

e dunque giungiamo alla conclusione:

Teorema 3 *Il ciclo hamiltoniano prodotto dall'algoritmo di Christofides è 3/2-approssimato.* □

Anche qui valgono considerazioni analoghe a quanto visto prima, a proposito dell'efficacia pratica di tale algoritmo (che è notevolmente superiore a quella dell'algoritmo ad albero). Osserviamo che la complessità dell'algoritmo è dominata dal calcolo del matching perfetto di peso minimo ($O(n^3)$ senza considerare algoritmi ancora più sofisticati). Questi risultati sono ancor più significativi se si tiene conto del fatto che è possibile dimostrare che, per il TSP generale, *non* può esistere alcun algoritmo ρ-approssimato, per nessun valore di ρ, neanche arbitrariamente alto (a meno che P=NP). Dunque, in questo senso, il Δ-TSP è un problema più trattabile.