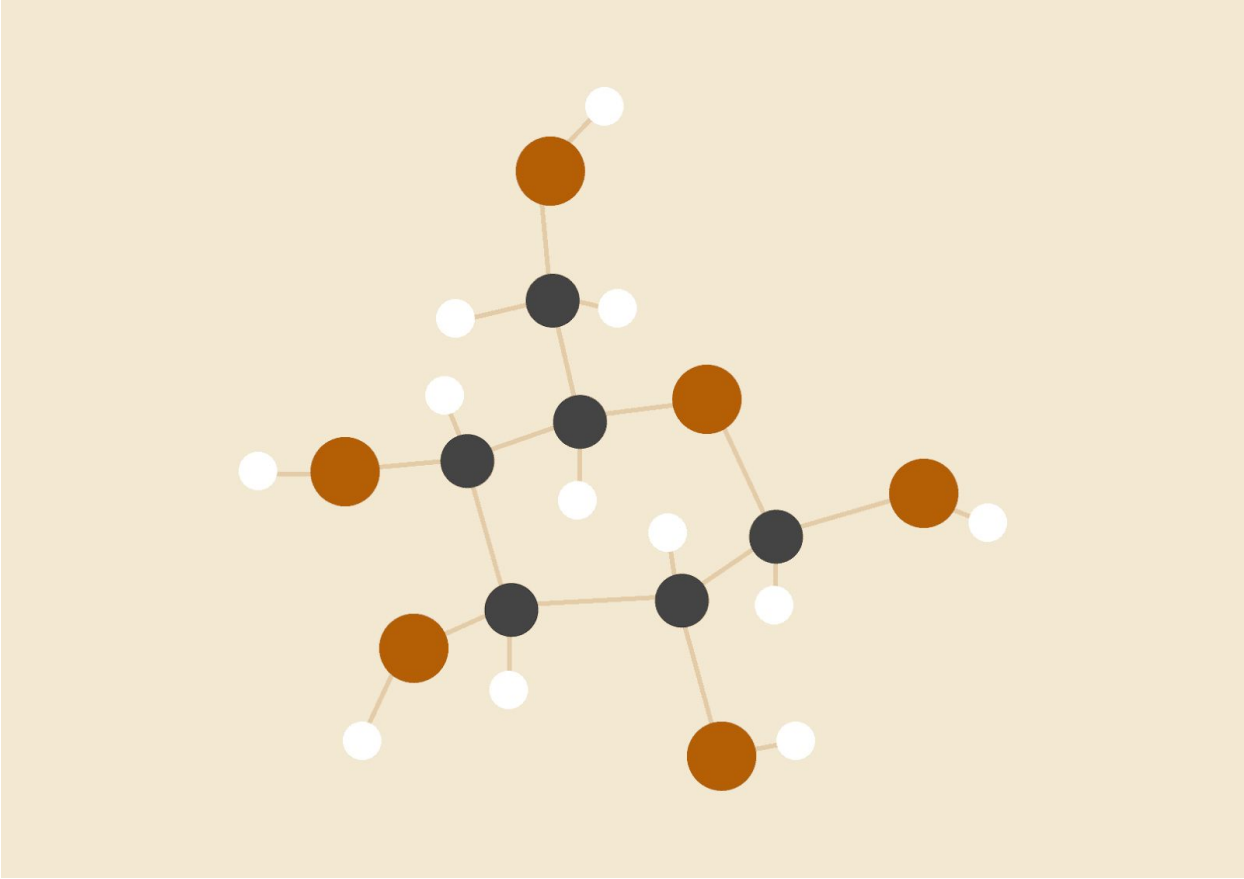
IL COMMESSO VIAGGIATORE

RELAZIONE

Allegro Luca 1211142, Hu Giovanni Jiayi 1206458, Mattiazzo Elena 1206695



Domanda 1

Istanza	HELD-KARP			Closest insertion			MST-approssimato		
	Soluzione	Tempo	Errore	Soluzione	Tempo (ms)	Errore	Soluzione	Tempo (ms)	Errore
burma14	3323	151ms	0%	3588	3	0.08%	4062	0.5	0.22%
ulysses22	7013	54s	0%	7816	1	0.11%	8132	0.1	0.16%
eil51	1221	2min	186%	490	4	0.15%	586	0.2	0.37%
kroD100	164904	2min	674%	25172	9	0.18%	27296	0.6	0.28%
gr229	190434	2min	414%	156823	39	0.16%	170205	2.1	0.26%
d493	113370	2min	223%	40907	337	0.16%	45262	3.3	0.29%
dsj1000	556080695	2min	2880%	22723010	2676	0.21%	25728577	14.5	0.37%

Domanda 2

Il comportamento dei tre algoritmi rispetto alle varie istanze è il seguente:

1. Held-Karp: essendo il tempo di esecuzione esponenziale O(n^2 * 2^n), aumentando anche di poco il numero di nodi, da 22 a 51, i tempi aumentano notevolmente e diventano presto improponibili per un normale computer. L'attuale implementazione riesce a risolvere in brevi tempi fino a 22 nodi grazie all'utilizzo di un set di bit per rappresentare i nodi visitati invece che altre strutture dati come una lista di nodi da visitare da cui rimuovere nodi man mano.
2. Closest insertion: approssima una soluzione in tempi accettabili O(n^2) sull'ordine dei secondi anche per n abbastanza grandi come 1000 nodi.
3. MST-approssimato: approssima in tempi nettamente migliori rispetto a Closest insertion in quanto seppur O(n^2) ha costanti migliori. In particolare l'implementazione non utilizza una heap, bensì un normale vettore di dimensione n che rappresenta le distanze dei punti rispetto all'ultimo visitato. Si è notato infatti che l'uso di una Heap rallenta i tempi di esecuzione. Difatti oltre all'aggiornamento causato dall'estrazione del minimo in tempo O(log(n)), vanno

aggiornate le distanze dei nodi nella Heap rispetto a quest'ultimo in tempo O(n * log(n)). L'uso di un normale vettore invece ha estrazione del minimo in O(n) ma l'aggiornamento delle distanze è O(n) invece che O(n * log(n)).

Per quanto riguarda invece gli errori di approssimazione, entrambi Closest Insertion e MST-Approssimato sono soluzioni 2-approssimato ma Closest Insertion è la migliore. Infatti MST-Approssimato ha il difetto di peggiorare la soluzione, rispetto a Closest Insertion, durante la visita dell'albero di copertura minimo nei momenti in cui passa da una foglia ad un altro ramo.

Sull'aspetto dell'efficienza invece l'algoritmo migliore è MST-Approssimato. Infatti sebbene entrambi gli algoritmi siano O(n^2), MST-Approssimato crea l'albero di copertura minimo calcolando n volte il nodo di distanza minima ma in seguito visita una sola volta l'albero. Closest insertion invece calcola n volta il vertice con distanza minima e per ciascun vertice deve inoltre trovare anche l'arco nel circuito parziale che minimizza la distanza triangolare.

Quindi le costanti di MST-approssimato sono migliori di quelle della Closest Insertion.