Business Analytics - Vehicle Routing Problem

Matteo Montrucchio
s302683@studenti.polito.it

Susanna Olivero
s304124@studenti.polito.it

Daniele Poggio
s305734@studenti.polito.it

3 agosto 2022

1 Introduzione

Abbiamo svolto un problema VRP simmetrico con capacità monodimensionale, veicoli identici e in numero dato.

Sono stati implementati i seguenti script MatLab i quali generano due soluzioni, una semplice costruttiva e una iterativa che utilizza come punto di parteza la soluzione trovata in precedenza.

- `main`
- `SimpleHeuristicSolution`
- `GeneralizedAssignment`
- `InsertionBasedMethod`
- `LocalSearchTWOPT`
- `TabuSearch`
- `nmEX`

2 Metodi Implementati

Nel `main` vengono caricati i dati presi da una libreria online e viene costruisce la matrice delle distanze euclidee; a seguire vengono eseguiti i diversi metodi per generare le 2 soluzioni e, infine, vengono stampati i risultati con i rispettivi plot.

La prima funzione che viene richiamata è `SimpleHeuristicSolution`, questa è una semplice euristica costruttiva del tipo *cluster-first, route-second*. All'interno viene prima richiamata la funzione `GeneralizedAssignment` la quale inizialmente assegna i seed in

modo che siano i più lontani fra di loro e dal centro e del nodo centrale; successivamente assegna i nodi alla singola route risolvendo il problema di minimo

$$\begin{aligned}
&\min \sum_{i=1}^{n}\sum_{k=1}^{m} c_{ik}y_{ik} \\
&\text{s.t.} \sum_{k=1}^{m} y_{ik} = 1, i = 1, ..., n \\
&\sum_{i=1}^{n} p_{ik}y_{ik} < R_k, k = 1, ..., m \\
&y_{ik} \in \{0, 1\}.
\end{aligned}$$

dove le variabili decisionale y_{ik} valgono 1 se il nodo i è inserito nel percorso k, 0 altrimenti. c_{ik} è un'approssimaziona del costo di assegnare il nodo i al percorso k, questa approssimazione è fatta utilizzando il criterio delle extra miglia rispetto al deposito 0 e al seed σ_k, ovvero

$$c_{ik} = d_{0,i} + d_{i,\sigma_k} - d_{0,\sigma_k}.$$

Lo step successivo è l'esecuzione dell'`InsertionBasedMethod`, un'euristica costruttiva che dato un insieme di nodi crea un percorso passante per tutti i punti inserendone uno per volta, il nodo da inserire viene scelte sempre secondo il criterio delle extra miglia. Questa funzione viene chiamata separatamente per ogni percorso, si ottiene così una soluzione ammissibile per il VRP.

Prima di usare questa soluzione come punto di partenza per il metodo iterativo, ottimizziamo la soluzione trovata eseguiendo `LocalSearchTWOPT`, un algoritmo che ottimizza separatamente ogni singola route esplorando dei vicini generati dalla regola *2-Opt*. Viene ora eseguita la funzione `TabuSearch`, essa è un'euristica iterativa i cui parametri sono elencati di seguito:

- N_{max}: numero massimo di iterazioni senza nessun miglioramento
- α_{max}: numero massimo di vicini generati nell'intorno
- β_{max}: numero massimo di scambi 2-Opt
- M: numero massimo di vertici scambiati
- $(\theta_{min}, \theta_{max})$: bound sulla durata del tabu

Gli steps fondamentali del metodo sono i seguenti:

1. Generare un vicino
   - 1.1. Scegliere casualmente due percorsi R1 e R2
   - 1.2. Scegliere casulamente n e m vertici (compresi tra 0 e M) rispettivamente da R1 e R2
   - 1.3. Se questo inserimento porta all'inammissibilità si ripete il passaggio precendente
   - 1.4. Inserire i vertici scelti utilizzando la procedura di inserimento: script *"nmEX"*

1.5. Applicare la procedura 2-Opt a R1 e R2 indipendentemente per β_{max} volte

1.6. Calcola il valore della funzione obiettivo della situazione finale

2. Ripetere i passaggi precedenti per α_{max} volte

3. Implementare la mossa con il miglior valore della funzione obiettivo, dopo di che gli scambi tra i 2 percorsi scelti diventano tabù.

Se tutte le mosse nell'intero vicinato diventano tabù, l'algoritmo sceglie la mossa tabù più vecchia e procede di conseguenza.
Nella fase di miglioramento della soluzione ogni $3 \cdot numVerteces$ l'algoritmo verifica se è stata trovata una nuova soluzione migliore. In caso contrario, torna all'ultima soluzione migliore.

3 Risultati computazionali

I punti di partenza utilizzati nelle simulazioni sono 75 e 100. Per ciasco dei 2 casi sono stati analizzati 4 diverse distribuzioni iniziali chiamate semplicemente a, b, c, d. I metodi sono quelli descritti in precedenza: SimpleHeuristic, 2-Opt e Tabù search con due diverse inizializzazioni del parametro N_{max}, ovvero $12 \cdot numVerteces$ $(12n)$e $6 \cdot numVerteces$ $(6n)$.

La seguente tabella mostra i risultati per i vari casi e il valore ottimo nell'ultima colonna.

	A	B	C	D	E	F	G	H	I	J
1			simpleHeuristic		2-Opt	Tabu search (12n)		Tabu search (6n)		
2		numRoutes	minDist	time (s)	minDist	minDist	time (s)	minDist	time (s)	minimum
3	75a	10	2152,8	2,2	2139,5	1728,8	467,7	1913,2	127,1	1618,4
4	75b	9	2000,8	4,1	1985,5	1450,7	584,5	1492,7	122,5	1344,6
5	75c	9	1830,4	1,6	1802,1	1395,3	329,5	1331,3	148,7	1291
6	75d	9	1858,6	2	1846,7	1494,2	420,5	1645	58,8	1365,4
7	100a	11	2421,7	5,7	2406,4	2212,1	1674,3	2355,6	258,7	2041,3
8	100b	11	2345,6	3,8	2327,3	2210,4	959,7	2274,2	400,6	1939,9
9	100c	11	2381,1	3	2366	1580,6	2474,4	1659,9	760,8	1406,2
10	100d	11	2230,7	8,3	2224,5	1770,2	1313,6	1937,5	340	1580,5

I dati evidenziati in verde sono il valore ottimo mentre i dati evidenziati in arancio rappresentano la migliore delle soluzioni fra quelle ottenute dall'euristiche implementate.

Per quanto riguarda l'euristica costruttiva possiamo, in generale, osservare che se esiste una soluzione ammissibile, allora questa viene sicuramente trovata. Tuttavia, nei casi presi in esame si può vedere che la soluzione trovata è ben lontana dalla soluzione ottima.

Soffermandoci invece sulla soluzione ottimizzata con $2-Opt$ notiamo che essa produce dei risultati migliorando ogni singolo percorso, ma per quanto riguarda il problema nella sua interezza non vi è un grosso miglioramento.

Notiamo, infine, che la Tabù search $(12n)$ è quella che si avvicina di più al minimo assoluto con un un'unica eccezione nel caso 75c in cui il risultato migliore è dato dalla

Tabù search ($6n$). Come si può vedere sono stati calcolati anche i tempi di esecuzione. Essi sono molto bassi per quanto riguarda il calcolo dell'euristica semplice che però porta a risultati di bassa qualità. La tabù serch ha dei tempi di esecuzione decisamente più alti, ma ciò si traduce in soluzioni di più alta qualità. Osservando le 2 diverse inizializzazioni della tabù search si nota un *trade-off* tra tempo di esecuzione e qualità della soluzione trovata (entrambi maggiori nel caso ($12n$)).

Di seguito sono riportate le figure ottenute tramite MatLab per un caso con 75 punti (75a) e uno con 100 punti (100c). Le figure sono tre per ogni caso e rappresentano in ordine l'euristica semplice, la tabù search ($6n$) e, infine, la tabù search ($12n$). Notiamo che è possibile vedere il miglioramento della soluzione anche graficamente.

Figura 1: 75a

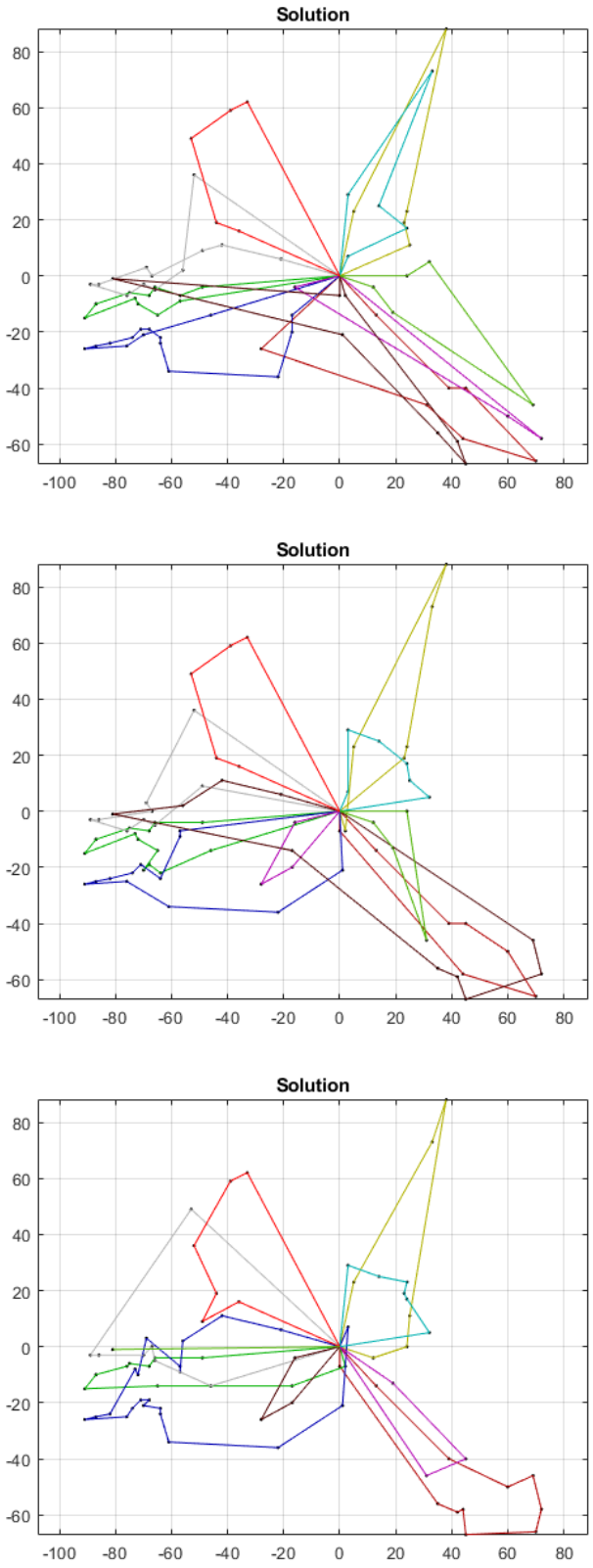
Solution
80
60
40
20
0
-20
-40
-60
-100
-80
-60
-40
-20
0
20
40
60
80
Solution
80
60
40
20
0
-20
-40
-60
-100
-80
-60
-40
-20
0
20
40
60
80
Solution
80
60
40
20
0
-20
-40
-60
-100
-80
-60
-40
-20
0
20
40
60
80

Figura 2: 100c