Peer Reviewed, Open Access, Free Online Journal
Published monthly : ISSN: 2308-832X
Issue 10(19); October 2014

Original Article: SOLUZIONE DEL PROBLEMA CON IL PERCORSO MODIFICATO DUE FASI ALGORITMO PROVIDER RESTRIZIONI TEMPO E IL NUMERO DI PERSONE

Citation
Egorova O.E., Zakirova U.V., Osechkina T.A. Soluzione del problema con il percorso modificato due fasi algoritmo provider Restrizioni tempo e il numero di persone. *Italian Science Review.* 2014; 10(19). PP. 20-25.
Available at URL: http://www.ias-journal.org/archive/2014/october/Egorova.pdf

Authors
O.E. Egorova, Perm National Research Polytechnic University, Russia.
U.V.Zakirova, Perm National Research Polytechnic University, Russia.
T.A. Osechkina, Perm National Research Polytechnic University, Russia.

Submitted: September 20, 2014; Accepted: September 27, 2014; Published: October 2, 2014

Problema di routing è stato evidenziato in una classe separata dopo problema del commesso viaggiatore è diventato particolarmente popolare. Al momento l'organizzazione del processo di fornitura delle merci dal fornitore al consumatore impegnato in scienza chiamata logistica. Pertanto, per tutte le aziende che in un modo o nell'altro è una gestione logistica, percorsi di ottimizzazione gioca un ruolo chiave. Uno dei più importanti e interessanti della classe selezionata è il compito della raccolta.

Scelto dagli autori per studiare le condizioni supplementari del problema (in tempo e numero di persone), sono già state esaminate in diversi studi.

Russell Bent e Pascal Van Hentenryck nel suo studio considerare vincoli di tempo e il numero di veicoli. In questo lavoro, le conclusioni che l'algoritmo in due fasi è più efficiente per risolvere tali problemi. [1] Uno studio condotto da Christos D. Tarantilis, George Ioannou, Chris T. Kiranoudis e Gregory P. Prastacos, suggerisce che usando il loro metodo di variazione metaeuristica proposto, si può efficacemente pianificare il traffico rotte. Inoltre, l'algoritmo basato su Open Vehicle Routing problema, è adattata alle condizioni reali. [2]

In questo lavoro, Feiyue Li, Bruce oro, Edward Wasil analizzato studi sul tema «Apri veicolo Problema Routing». Gli autori concludono che l'interesse per OVRP negli ultimi anni è cresciuta notevolmente e ha sviluppato una vasta gamma di nuovi algoritmi che risolvono questo problema. [3]

Raccolta problema appartiene alla classe dei problemi NP-difficili. Il più delle volte, questi problemi possono essere risolti con metodi euristici, perché permettono di trovare rapidamente una soluzione. Ma la soluzione così ottenuta non è sufficientemente accurato. Gli stessi metodi esatti sono ricerca come esaustiva di tutte le possibili combinazioni, quindi non sono efficaci. [4]

I modi più ottimali per trovare soluzioni ai metodi riconosciuti metaeuristica. Il più comunemente usato nella pratica sono: algoritmi genetici, il metodo di colonia di formiche, annealing simulato [5].

Gli algoritmi genetici non garantiscono che la soluzione è ottimale. Il vantaggio

principale di questi algoritmi è che è versatile e può essere utilizzato per differenti tipi di attività. I vantaggi del metodo di ricottura sono restrizioni sul tipo di funzione da ridurre al minimo, la possibilità di cercare il minimo globale e l'efficienza nel risolvere i problemi delle diverse classi. Tra gli svantaggi sono: la bassa velocità dell'algoritmo e la complessità dell'algoritmo di adattamento ai dati in questione [6].

Questi metodi sono alla base degli algoritmi a due stadi. Con loro, vi è una realizzazione della seconda fase -. Miglioramento della soluzione approssimata, la ricerca viene eseguita nella prima fase [7]

Il seguente è un classico problema di routing con restrizioni aggiuntive. Come già accennato, le limitazioni di aspetti quali il tempo e numero di persone coinvolte nella routine di servizio dell'oggetto. Un modello matematico del problema, tenendo conto di questa affermazione è la seguente: (1), (2), (3), (4), (5).

La funzione obiettivo (1) riduce al minimo la struttura di manutenzione qui t_{ij}^{l} - il momento in cui il veicolo passa sul percorso dall'oggetto i all'oggetto; j ; l - Un indice che è responsabile per bypassare rotte; m - Il numero di rotte. Limitazione (2) significa che ogni oggetto è servito una volta, ed appartiene ad un lobo. Limitazione (3) - la capacità massima del veicolo, dove il q - capacità di carico del veicolo; Q^{+} - Il peso del punto, che deve essere immerso in macchina; Q^{-} - Il peso che si desidera scaricare. (4) - è un limite di tempo per la chiusura: l_j - la chiusura dell'oggetto a_j - i punti di tempo di servizio e_o^l - check out l ° Brigata del deposito. Vincolo (5) - la restrizione che implica che il numero di persone che servono per essere non inferiore a quella richiesta.

In questo studio, un algoritmo comprende due fasi, vale a dire la prima fase - la costruzione di una soluzione approssimata. Per comodità l'algoritmo è uno schema a blocchi della sua attuazione (vedi fig. 1). Descrive le procedure, la cui realizzazione vi aiuterà a trovare i percorsi approssimativi, tenendo conto di tutti i vincoli.

Consideriamo più in dettaglio lo schema a blocchi della stessa. Ordinamento procedura divide l'insieme dei punti in 2 gruppi: gli oggetti per i quali si richiede il servizio, per esempio, 3 persone, e gli oggetti per il servizio che richiede 2 persone. Io variabile viene assegnato il valore 3 Ciò significa che saranno eseguite tutte le ulteriori procedure per una serie di punti che servono 3 persone.

Procedura DividePoint specifica il numero minimo di gare che sono necessarie per servire tutti i punti.

MaxDistantPoint senso è che, in questa unità viene eseguito per trovare il punto più lontano in cui viene suddiviso in gruppi di oggetti (petali). Numero di punti remoti corrisponde al numero di giri.

ConstructPetals impegnati in formazione diretta di questi gruppi. Cosa è incluso nella scheda sulla base della distanza minima da una postazione remota al punto-richiedente e indietro.

Procedura TransfertPoint permette di bilanciare i petali in peso. Se la scheda è sovraccarico, il trasferimento viene effettuato in termini di lobo sovraccarico vicinanze. In questo caso anche sostenuto il principio di minima distanza da una postazione remota al punto, che viene trasmesso.

Dopo tutte le operazioni su una serie di punti che servono 3 persone, un controllo i = 3 Questo contatore indica che il programma ha soddisfatto tutti per le "triplette" e ora si deve andare al "due" - una serie di punti, che sono serviti da due persone.

Significato procedura AddMassThree è che in questa fase è necessario integrare i petali esistenti "triple", "due a due", vale a dire rendere l'aggiunta di petali "triple" (se possibile) i punti, che sono gestiti da due persone ed è adatto per il peso, per evitare

di sovraccaricare il petalo. Va notato che i punti vengono aggiunti anche sulla base della distanza più breve da una postazione remota, per considerazione.

Una volta che la procedura è stata eseguita AddMassThree, il contatore i viene impostato a 2, e non vi è una funzione di rollback DividePoint. Questo suggerisce che tutto il procedimento sopra descritto, tranne AddMassThree essere eseguita per una serie di "due a due". Dal momento che i <> 3, comincia a eseguire i CreateRoutes procedura, che formavano percorsi per ogni petalo. La base di questa funzione è un algoritmo che è una modifica del metodo greedy, ossia vengono aggiunti in sequenza i punti di percorso situate ad una distanza minima l'uno dall'altro. Qui formano una matrice in cui i punti sono disposti in ordine crescente di chiusura degli oggetti utilizzando la formula: $PrioritetClose[i] := l[i] - a[i] - TimeInWay - t[j^0, i]$ dove j^0 - quest'ultimo incluso nel percorso. Dopo la formazione della matrice prende il primo elemento $j*$ di esso, ed è calcolata rispetto alla quantità $ProblemTime = PrioritetClose[i] - t[j^0, i] - a[j^*] + t[j^*, i]$. Se $ProblemTime$ tutti i punti assume un valore maggiore di zero, allora significa che la squadra riesce a tutti i punti di arrivo per quanto riguarda l'oggetto $j*$. Avanti, formata una serie di punti che sono più vicini al punto $j*$. E 'in questa procedura è l'idea di "algoritmo greedy", vale a dire unità dei vigili nel punto prossimale in cui riesce. Una volta selezionato un oggetto $k*$ dalla matrice, viene provata la condizione per l'inclusione nel percorso di questo punto dalla formula: $PrioritetClose[j^*] = TimeRoute[i] + t[j^0, i]$ dove $TimeRoute[i] = t[j^0, i] + t[i, j^*] + a[i]$. Se questo valore è maggiore di zero, quindi attendere l'apertura dell'oggetto non è necessario ed è possibile rilasciare l'oggetto, prima di andare al primo closing. Una volta che l'oggetto è inclusa nel percorso, è necessario modificare l'ora dei vigili sulla strada.

Per testare l'algoritmo è stato portato a 81: Custodia e 80 punti per i quali è necessario creare percorsi. Ogni oggetto è servita da 2 o 3 persone, controllare brigate dal deposito per tutti i lobi allo stesso modo, i veicoli hanno la stessa capacità di. Ogni oggetto è caratterizzato da un peso diverso del carico, il tempo di apertura, gestione e chiusura.

È stata ricevuta il 6 petali. Per visualizzare i risultati presentati due percorsi petali. La Figura 2 mostra il percorso del veicolo 1 lobo.

La figura 3 è indicato dal movimento coerente del veicolo sul petalo 2.

Le distanze tra i punti nei petali del minimo. Percorsi posati in modo ottimale, ma scegliere il percorso non sono raccolti in modo coerente. Ciò è dovuto al fatto che l'apertura dei punti dato differente. Ad esempio, nel 2 a 4 petalo prima di tutti gli oggetti che sono più vicini al punto 3. E, dal momento che il veicolo ha il tempo di chiamare al punto 4, si attiva automaticamente la rotta, e prima di 6.

La soluzione risultante ha un errore che è differente dalle soluzioni teoriche, che è 26% sulla rotta ottimale senza restrizioni. In connessione con l'introduzione di limiti di tempo, c'è un deterioramento di percorsi ottimali, quindi l'errore risultante può essere considerato accettabile.

Tra gli svantaggi notevoli dell'algoritmo consiste nel prevedere che esso è implementato per punti concentrati in una piccola area.

Plus è la capacità di gestire grandi quantità di dati, nonché relativamente piccolo errore.

Secondo lo studio, le seguenti conclusioni: la soluzione può essere considerato realistico in considerazione del fatto che i costi relativi a risorse di trasporto possono essere considerati del tutto adeguato; nella costruzione di percorsi tenendo conto di tutti i vincoli; Ottenere lo stesso margine di errore, causato dal fatto

che l'algoritmo non contiene metodi miglioramento.

References:

1. Bent R. 2003. A Two-Stage Hybrid Algorithm for Pickup and Delivery Vehicle Routing Problems with Time Windows. pp. 123-137.
2. Tarantilis D.C. A threshold accepting approach to the open vehicle routing problem.
3. Li F. 2007. The open vehicle routing problem: Algorithms, large-scale test problems, and computational results. pp. 2918 – 2930.
4. Laporte G. 1998. Classical Heuristics for the Vehicle Routing Problem .
5. Eyrikh S.N. 2012. Review of methods for solving transport routing. p. 22 – 29.
6. Gendreau M. 1994. Metaheuristics for the vehicle routing problem.
7. Sam R. 1999. Thangiah. A Hybrid Genetic Algorithm, Simulated Annealing and Tabu Search Heuristic for Vehicle Routing Problems with Time Windows.

Fig.1 Schema a blocchi dell'algoritmo

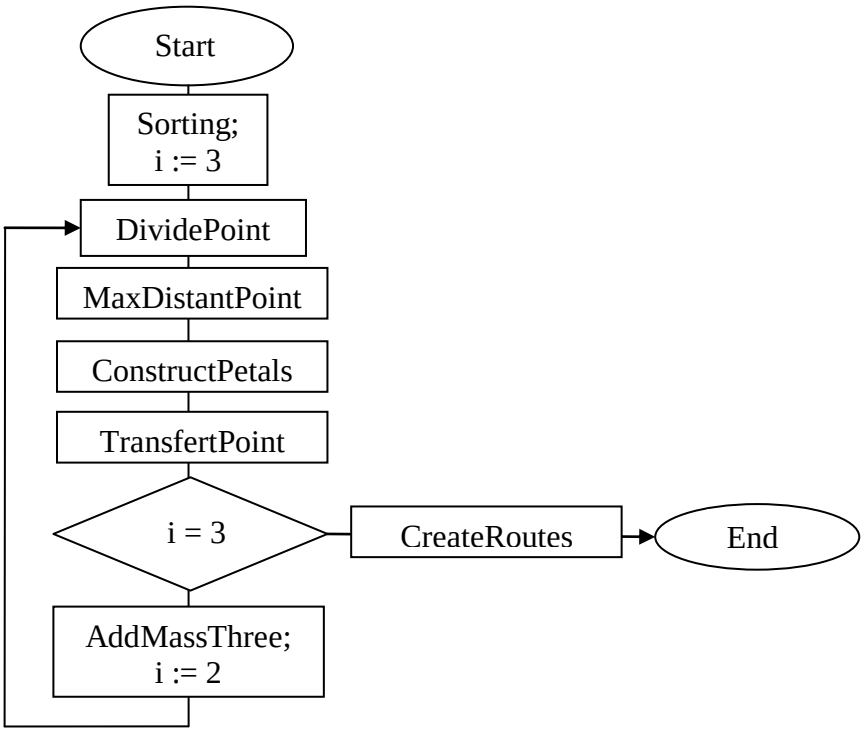


Fig. 2 Route 1 scheda

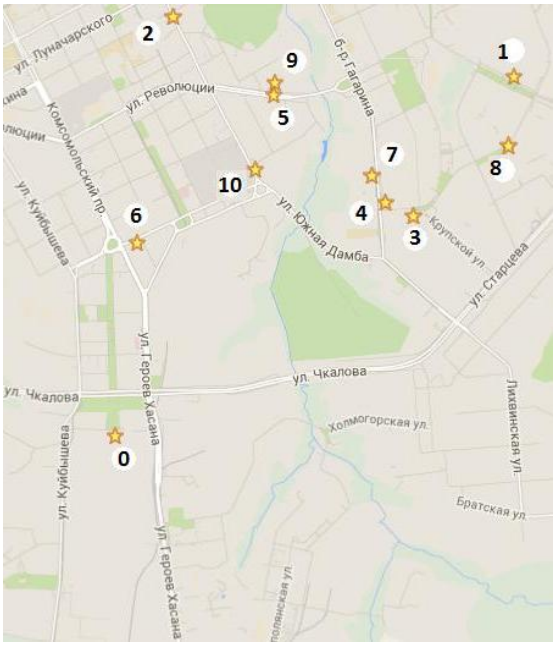
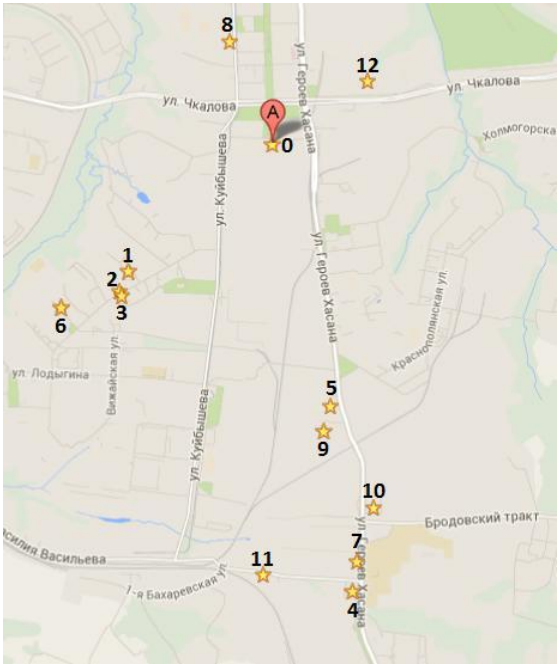


Fig. 3 Route 2 petali



$$F = \sum_{l=1}^{m} \sum_{\substack{i=1 \\ j=1 \\ i \neq j}}^{k} t_{ij}^{l} \to \min \qquad (1)$$

$$\sum_{l=1}^{m} \sum_{i,j=1}^{n} x_{ij}^{l} = 1, \ \sum_{i} x_{ip}^{l} - \sum_{j} x_{pj}^{l} = 0, \quad l = \overline{1,m}, \ x_{ij}^{l} \in \ 0;1 \qquad (2)$$

$$\sum_{l=1}^{m} Q_{l}^{+} \ x_{ij} \ + \sum_{l=1}^{m} Q_{l}^{-} \ x_{ij} \ \leq q \qquad (3)$$

$$l_{j} - a_{j} - e_{o}^{l} - \sum_{i=0}^{j-1} \sum_{p=1}^{j} \ t_{ip}^{l} \geq 0, \qquad l = \overline{1,m} \qquad (4)$$

$$r \ x_{ij} \ \leq r_{ij} \qquad (5)$$