Laboratorio di Algoritmi

Progetto "Guida autonoma" (luglio 2021)

Nota: La scadenza del progetto è fissata per lunedì 12 luglio **compreso**.

Nota: Si consiglia di controllare gli eventuali aggiornamenti di questo documento, che potrebbero riportare le risposte a dubbi degli studenti e la correzione di eventuali errori. Si consiglia anche di consultare l'altro documento che riporta avvertenze utili per lo svolgimento del progetto.

Il problema Si fa un gran parlare, ultimamente, dei veicoli a guida autonoma. Il problema che vogliamo affrontare riguarda la gestione di una flotta di taxi elettrici a guida autonoma. I clienti chiamano la centrale di servizio e descrivono lo spostamento a cui sono interessati, il sistema valuta se esiste un veicolo adatto a servire la richiesta, lo assegna e ne definisce il percorso in modo che raggiunga il cliente al momento desiderato o, se questo non è possibile, al più presto. Il sistema gestisce anche il meccanismo di ricarica e valuta il guadagno realizzato con il servizio.

I veicoli si muovono in una città rappresentata da un elenco di strade, per semplicità tutte a doppio senso. Nel sistema di gestione, ogni strada è descritta da una coppia di indici numerici, che corrispondono agli incroci o alle piazze poste sugli estremi della via, e da un numero intero positivo, che ne indica il tempo di percorrenza misurato in secondi. Queste informazioni sono sufficienti a determinare la topologia della rete stradale e a calcolare i tempi necessari a muoversi da un punto a un altro della rete lungo un percorso dato, nonché a calcolare i percorsi più opportuni.

Il sistema conosce il numero dei veicoli disponibili e la loro autonomia, espressa dal numero totale massimo di secondi durante i quali possono muoversi prima di scaricare completamente la batteria. Conosce anche la durata in secondi della ricarica, che può avvenire solo nella sede della ditta di gestione. Questa si trova nel punto di indice 1 della rete stradale.

Il sistema ha l'elenco delle chiamate in ordine di ricezione crescente (in caso di orari coincidenti, si rispetta comunque l'ordine dato). Ogni chiamata è caratterizzata da:

- l'ora in cui viene ricevuta;
- il cognome del cliente;
- gli indici dei punti di origine e destinazione del tragitto richiesto;
- l'ora minima prima della quale il cliente non può partire dall'origine;
- l'ora massima entro la quale il cliente deve arrivare alla destinazione.

Per cominciare, sono richieste alcune elaborazioni documentali: bisogna generare

- l'elenco dei clienti in ordine alfabetico;
- l'elenco della chiamate in ordine di durata decrescente della corsa.

Per *durata* di una corsa si intende il tempo minimo necessario a un veicolo a muoversi sulla rete stradale dal punto di origine al punto di destinazione.

Il sistema deve poi definire le posizioni iniziali dei veicoli sulla rete stradale. Supponiamo per semplicità che le posizioni lecite siano tutti e soli gli estremi delle strade (incroci e piazze). Per coprire efficacemente la città, il sistema cerca l'insieme

di posizioni reciprocamente più distanti tra loro, vale a dire tali che la somma delle relative distanze sia massima. Siccome questo problema è computazionalmente complesso, viene risolto con un'euristica *greedy*, descritta nel seguito.

La parte principale del progetto consiste nel simulare l'andamento del servizio. Al principio della giornata ogni veicolo è nella posizione assegnatagli, ha la batteria completamente carica ed è libero. Le chiamate arrivano una dopo l'altra, e ogni chiamata provoca la ricerca di un veicolo che sia in grado di servirla, cioè rispetti opportune condizioni elencate più avanti. Se nessun veicolo è in grado di servire la chiamata, questa viene rifiutata. Altrimenti, uno dei veicoli viene scelto (in base al criterio discusso nel seguito) e parte immediatamente per raggiungere il punto di origine dove il cliente lo aspetta in un momento compreso fra l'ora minima di partenza e l'ora massima di arrivo. Se il veicolo arriva al punto di origine prima dell'ora minima, rimane fermo ad aspettare il cliente (si suppone che possa farlo senza intralciare il traffico). Poi porta il cliente a destinazione, arrivando prima dell'ora massima di arrivo (è una delle condizioni per accettare la chiamata). Il cliente abbandona il veicolo e il sistema verifica la carica della batteria. Se questa è inferiore al 20% dell'autonomia massima, il veicolo torna alla sede per ricaricarsi; altrimenti, rimane fermo in attesa della chiamata successiva[1]. Siccome in sede è disponibile una sola colonnina di ricarica, i veicoli che ne hanno bisogno aspettano in coda il proprio turno nell'ordine con cui sono rientrati. Ovviamente, un veicolo che sta raggiungendo o servendo un utente, o sta rientrando in sede, o attende in coda la ricarica o si sta ricaricando, non è libero per rispondere a chiamate. Torna libero immediatamente alla fine della ricarica. Per evitare spostamenti inutili, un veicolo ricaricato non torna alla posizione iniziale, ma rimane in sede ad aspettare la chiamata successiva assegnatagli.

Un veicolo è in grado di servire una chiamata se:

1. è libero, cioè non sta servendo altri clienti, tornando in sede, aspettando in coda la ricarica o ricaricandosi;

2. è in grado di raggiungere l'origine della chiamata, caricare l'utente e raggiungere la destinazione entro l'ora massima di arrivo indicata;

3. ha carica sufficiente a raggiungere l'origine della chiamata, da lì la destinazione e dalla destinazione la sede per ricaricarsi (anche se non torna in sede a ricaricarsi tutte le volte, deve poter essere in grado di farlo).

Il prezzo pagato dal cliente è proporzionale (per semplicità, uguale) alla durata della corsa in secondi. A questo prezzo si aggiunge un premio fisso di puntualità nel caso in cui il cliente parta esattamente all'ora minima di partenza indicata, senza dover attendere. Qualunque ritardo in partenza cancella il premio, mentre i ritardi in arrivo non sono permessi, dato che l'intera corsa viene cancellata senza nemmeno cominciare.

La simulazione si baserà sul concetto di *evento discreto*, che viene descritto in dettaglio nella sezione seguente. In sintesi, si tratta di gestire da un lato lo stato del sistema, dall'altro un insieme di eventi, ciascuno caratterizzato dall'istante di occorrenza. All'inizio, il sistema è in un dato stato e un insieme di eventi futuri sono già noti. Quindi, il meccanismo della simulazione a eventi discreti procede iterativamente in questo modo:

[1] È possibile che tutte le successive chiamate richiedano spostamenti per i quali la carica residua è insufficiente. In tal caso, questa regola euristica porterà il veicolo a stare fermo e rifiutare clienti che potrebbe servire dopo una ricarica. Le euristiche per definizione non garantiscono soluzioni ottime.

1. considera il primo evento dell'insieme (quello cronologicamente minimo) e in base a regole che dipendono dallo stato corrente del sistema, modifica lo stato stesso (per esempio, se arriva una chiamata, il sistema le assegna un veicolo, ne definisce la traiettoria e marca il veicolo come impegnato);

2. sempre in base all'evento corrente, "innesca" altri eventi discreti (cioè li genera e li aggiunge all'insieme) in istanti futuri determinati;

3. infine, rimuove l'evento elaborato dall'insieme e torna al punto 1.

La simulazione termina quando l'insieme degli eventi è vuoto.

Al termine della simulazione, si richiede un rapporto dettagliato degli eventi, e il valore di alcuni indicatori di prestazione riassuntivi:

- il numero di chiamate rifiutate,
- il numero di ricariche effettuate,
- il tempo totale di movimento dei veicoli (in secondi),
- il ricavo totale del servizio.

Per concludere, siccome la scelta del veicolo a cui assegnare ogni chiamata e del momento in cui ricaricare i veicoli è del tutto euristica, è utile cercare di valutare quanto potenzialmente il servizio stesso sia redditizio. Calcolare l'ottimo è computazionalmente complesso[2]. Ci limitiamo quindi a calcolare una stima per eccesso del guadagno realizzabile, che si basa sul tempo complessivo messo a disposizione dai veicoli e sul tempo minimo richiesto per servire ciascuna chiamata. Il modo per ottenere questa stima è descritto nel seguito.

Il progetto Il programma da realizzare riceve dalla linea di comando i nomi di tre file di testo, che forniscono i dati. Nell'ordine:

1. il primo file rappresenta la rete stradale,

2. il secondo file rappresenta le caratteristiche dei veicoli,

3. il terzo file rappresenta le chiamate degli utenti.

Il file di testo che rappresenta la rete stradale riporta nella prima riga il numero di punti (incroci e piazze) nei quali possono stazionare i veicoli ed essere caricati e scaricati i clienti e il numero di strade. Per esempio,

```
10 20
```

indica che la rete ha 10 punti e 20 strade. Segue l'elenco delle strade, una per riga, rappresentate dai due indici dei punti estremi e dal tempo di percorrenza in secondi. Per esempio:

```
3 10 89
2 8 132
...
```

indica che i punti 3 e 10 sono collegati da una strada che si percorre in 89 secondi, i punti 2 e 8 da una strada che si percorre in 132 secondi, e così via. Le strade sono

[2]Senza contare che bisognerebbe anche distinguere il caso in cui le chiamate vengono rivelate una alla volta, come nella simulazione che si sta conducendo in questo caso, da quello in cui le chiamate sono tutte note a priori e questa conoscenza viene sfruttata per servirle meglio.

tutte a doppio senso di marcia e con tempi di percorrenza identici nei due versi. Ignoriamo per semplicità semafori, tempi di svolta, ecc. . .

Il file di testo che descrive i veicoli consiste in una sola riga. Nell'ordine, sono riportati il numero dei veicoli, la durata in secondi dell'intervallo di servizio, l'autonomia della batteria in secondi di movimento e la durata in secondi della ricarica. Per esempio:

```
3 3600 1000 200
```

indica che ci sono 3 veicoli, che l'orizzonte temporale del servizio è di 3 600 secondi, l'autonomia dei veicoli di 1, 000 secondi in movimento, e infine la durata della ricarica di 200 secondi.

Il file di testo che conserva le chiamate riporta nella prima riga il loro numero. Ogni successiva riga descrive una chiamata, riportando l'ora (espressa in secondi a partire dal principio del servizio), il cognome del cliente (per semplicità, sempre una singola parola di 25 caratteri al massimo), gli indici dei punti di origine e destinazione del tragitto richiesto, l'ora minima e massima entro le quali il cliente deve effettuare il proprio viaggio (espresse in secondi a partire dal principio del servizio) e il premio guadagnato se il cliente viene caricato esattamente al momento desiderato. Per esempio:

```
200
1107 Palmieri 2 7 1765 2994 1018
1335 Hijjoui 4 5 1674 3388 2545
...
```

indica che ci sono 200 chiamate, la prima delle quali avviene dopo 1 107 secondi dal principio del servizio (e considerata istantanea per semplicità), richiede per il sig. Palmieri uno spostamento dal punto 2 al punto 7 che cominci non prima di 1 765 secondi e termini non oltre 2 994 secondi dal principio del servizio; il premio di puntualità è pari a 1 018, da aggiungere al prezzo della corsa (pari alla durata del percorso dal punto 2 al punto 7) solo se il viaggio inizia effettivamente al secondo 1 765.

Il primo punto del progetto è disporre i clienti in ordine alfabetico. Per semplicità, assumeremo che tutti i clienti facciano una sola chiamata e abbiano cognomi diversi. Quindi il numero dei clienti è uguale a quello delle chiamate. Bisognerà stamparli a video in ordine alfabetico, uno per riga, preceduti da una riga con la parola chiave `Clienti:`. Per esempio:

```
Clienti:
Bartolini
Dolciotti
Fiore
...
```

Quindi, bisogna ordinare i viaggi richiesti, compresi quelli che non sarà possibile servire, per durata decrescente del tragitto da origine a destinazione. A pari durata, si ordineranno per ora di chiamata crescente. Si stamperà a video una riga con la parola chiave `Viaggi:`, seguita dalle chiamate, una per riga, per le quali si riporterà l'ora di chiamata, il cognome del cliente e la durata in secondi, seguita dall'elenco degli indici dei punti attraversati lungo il percorso (dal punto di origine a quello di destinazione). Per esempio:

```
Viaggi:
661 Fiore 699
1107 Hijjoui 675
```

```
302 Dolciotti 587
...
```

Prima di eseguire la simulazione, occorre determinare le posizioni iniziali dei veicoli sulla rete. Lo faremo tenendo il primo veicolo in sede (cioè nel punto 1 della rete), ponendo il secondo nel punto più lontano possibile dal primo (cioè nel punto che richiede il massimo tempo di viaggio per essere raggiunto dal primo), il terzo nel punto più lontano dai primi due (cioè con somma massima dei tempi di viaggio dai punti precedenti), e così via sino all'ultimo veicolo. In caso di pari distanza totale, si sceglie il punto di indice minimo. Ogni veicolo deve stare in un punto diverso. Terminata questa fase, si stamperanno le posizioni iniziali dei veicoli su una sola riga, precedute dalla parola chiave `Posizioni:` e separate da spazi. Per esempio:

```
Posizioni:  1 3 2
```

Per la simulazione, scandiamo il tempo in secondi a partire dal principio del servizio. I veicoli sono inizialmente tutti liberi, hanno la batteria completamente carica e stanno nelle posizioni calcolate subito sopra. Consideriamo i seguenti possibili tipi di eventi discreti:

1. una chiamata;
2. l'arrivo a destinazione di un cliente;
3. il rientro in sede di un veicolo che deve ricaricare la batteria;
4. la fine della ricarica di un veicolo.

Per semplicità, carichiamo subito tutti gli eventi chiamata nell'insieme, anche se elaboreremo ogni chiamata come se non conoscessimo le successive, dato che stiamo simulando la gestione di un servizio in tempo reale.

Un evento di tipo chiamata modifica lo stato del sistema cercando un veicolo adatto a servire la chiamata. Se lo trova, il veicolo diventa impegnato, parte subito verso il punto di origine, dove si ferma ad aspettare il cliente (se questi non è ancora arrivato), lo carica (istantaneamente), e prosegue verso il punto di destinazione. I due percorsi sono ovviamente i più brevi possibili sulla rete. Per semplicità, i veicoli disponibili per servire una chiamata sono solo quelli liberi al momento della chiamata. Fra questi veicoli si sceglie:

- quello che arriva prima al punto di destinazione;
- in caso di parità, si sceglie quello che arriva più tardi al punto di origine (per ridurre il rischio di tenere il veicolo in attesa per niente);
- in caso di ulteriore parità, si sceglie quello di indice minimo.

Se nessun veicolo è libero, si rifiuta la chiamata[3]. Un evento di tipo chiamata innesca anche un evento successivo: se la chiamata viene servita, infatti, è determinata l'ora in cui il cliente arriva a destinazione: si tratta di valutare la durata dei due percorsi dalla posizione corrente del veicolo al punto di origine della corsa e da questo al punto di destinazione, nonché l'eventuale attesa nel punto di origine. Il nuovo evento di arrivo alla destinazione va aggiunto all'insieme.

Un evento di arrivo a destinazione modifica lo stato del sistema rendendo libero il veicolo che trasportava il cliente. Per risparmiare energia, il veicolo non torna alla posizione iniziale, ma rimane fermo nella nuova posizione. Se però la carica

[3] A rigore, questo spreca la possibilità che un veicolo stia servendo un cliente o si stia ricaricando, ma torni libero abbastanza presto da poter servire il cliente.

della batteria è strettamente inferiore al 20% dell'autonomia, il veicolo non torna libero, ma va immediatamente in sede a ricaricarsi seguendo il percorso più breve. L'evento innesca quindi un evento di rientro in sede, all'ora determinata dalla durata del percorso.

Un evento di rientro in sede mette il veicolo in coda per la ricarica. La permanenza in coda dipende dal numero di veicoli in coda e dalla durata della ricarica stessa, ma anche dal tempo residuo di ricarica del veicolo che attualmente è in testa alla coda stessa. In base a questi fattori si può prevedere l'ora alla quale il veicolo appena arrivato in sede terminerà la ricarica stessa, e quindi innescare l'evento di fine della ricarica.

Un evento di fine della ricarica rende nuovamente libero il veicolo ricaricato, che rimane in sede finché non gli viene assegnata una nuova chiamata.

È ovviamente possibile che si verifichino eventi simultanei. In questo caso, per convenzione, assumeremo il seguente ordine, a parità di orario:

1. fine della ricarica di un veicolo,
2. arrivo a destinazione di un cliente,
3. rientro in sede di un veicolo,
4. chiamata di un cliente.

Questo serve a rendere disponibili per le chiamate i veicoli che si liberano da una ricarica o da un servizio contemporaneamente alla chiamata stessa[4]. Se due eventi sono simultanei e dello stesso tipo, si considera per primo quello relativo al veicolo di indice minimo. Per gli eventi simultanei di chiamata, che non hanno un indice di veicolo, si segue l'ordine originale delle chiamate.

La simulazione produce un elenco di eventi ordinati come sopra indicato. Si stamperà a video una prima riga con la parola chiave `Eventi:`, seguita dall'elenco degli eventi, riportando di ciascuno in una riga separata l'ora, il tipo di evento (`CHIAMATA`, `FINE_SERVIZIO`, `RIENTRO_SEDE`, `FINE_RICARICA`), l'indice del veicolo coinvolto (0 per gli eventi di chiamata, che non coinvolgono veicoli) e il cognome del cliente coinvolto (nessuno per gli eventi di rientro in sede e fine ricarica). Per esempio:

```
Eventi:
302 CHIAMATA 0 Dolciotti
432 CHIAMATA 0 Rainesi
661 CHIAMATA 0 Fiore
...
```

L'elenco degli eventi termina con alcune informazioni riassuntive, cioè il numero di chiamate rifiutate, il numero di ricariche effettuate, il tempo totale di movimento dei veicoli e il guadagno totale del servizio. Ciascuna informazione occupa una riga ed è preceduta da un'opportuna parola chiave, come nel seguente esempio:

```
Rifiuti:
Ricariche:
Tempo totale:
Guadagno:  8339
```

Per concludere, si farà una stima per eccesso del guadagno ottenibile dalle chiamate date. Questa stima si basa sul fatto di ignorare alcuni vincoli fondamentali

[4] E anche a garantire che, se la destinazione di un cliente è la sede e poi il veicolo si ferma a ricaricare, il cliente scenda dal veicolo prima che questo si metta in coda alla colonnina

del problema, come gli intervalli temporali richiesti dai clienti, e i tempi di viaggio necessari a spostare ogni veicolo fino ai punti di origine delle chiamate, nonché dai punti di destinazione alla sede in caso di ricarica, i tempi di attesa in coda per la ricarica e i tempi di attesa del cliente quando il veicolo arriva in anticipo. A questo punto, si può vedere il servizio come il tentativo di riempire il più possibile uno "zaino virtuale":

- la capacità corrisponde al tempo totale disponibile, pari al numero dei veicoli moltiplicato per l'ampiezza dell'orizzonte temporale del servizio[5];
- gli oggetti corrispondono alle singole chiamate, dove:
  - il volume corrisponde al solo tempo di spostamento da origine a destinazione;
  - il valore corrisponde al guadagno fornito dalla corsa più il premio di puntualità, che si ipotizza di poter sempre ottenere.

La soluzione del problema di zaino complessivo fornisce un valore che è certamente non inferiore al guadagno ottimo realizzabile. Probabilmente, la stima è molto superiore all'ottimo, ma quanto meno può dare un'idea del suo ordine di grandezza (oltre che dei metodi tipicamente utilizzati per affrontare problemi di Ottimizzazione Combinatoria). Questa stima deve essere stampata a video preceduta dalla parola chiave `UB:` . Per esempio:

```
UB: 18794
```

Chiarimenti

[5] A rigore, ogni veicolo è uno zaino a sé, ma per evitare di complicare il problema ignoriamo anche la distinzione tra i veicoli, come se si potessero prestare tempo a vicenda.