Algoritmi e strutture dati per piattaforme efficienti di ride sharing

— Compendio di tesi magistrale —

Francesco Tosoni

Settembre 2020

Compendio breve

Studi recenti [1, 2] hanno dimostrato come il *ride sharing* (talvolta denotato in italiano come "covetturaggio") offra un enorme potenziale in termini di riduzione del numero di veicoli necessari per servire una data richiesta di mobilità, come ad esempio le corse su taxi.

Ad ogni modo, sebbene negli ultimi anni siano stati proposti diversi servizi di *car pooling*, e sebbene molti di essi siano ora largamente utilizzati in tutto il mondo (si pensi a Uber, Didi, Lyft), le tecniche di *ride sharing* sono tuttora soggette a serie limitazioni di scalabilità, ed in particolar modo quando si prende in esame il problema di combinare in tempo reale richieste di corse multiple in un una singola corsa condivisa da più utenti all'interno di una grande area urbana. Per queste ragioni, lo studio sistematico e la progettazione di algoritmi veloci e scalabili per la gestione su richiesta di grandi flotte di veicoli all'interno di grandi aree geografiche riveste in questo ambito un ruolo chiave di innovazione.

Ogni piattaforma di mobilità su richiesta deve quindi necessariamente approcciare il seguente problema: dato un numero di richieste di taxi e data una flotta di veicoli disponibili, definire una procedura per la stima efficiente dei tempi di attraversamento tra origini e destinazioni delle corse da un lato, e posizioni dei veicoli dall'altro. Le stime sui tempi costituiscono, infatti, un passaggio fondamentale per la selezione di un opportuno sottoinsieme di richieste di taxi, e per la successiva costruzione e per l'instradamento dei viaggi combinati.

Va di per sé che gli approcci forza bruta, ossia caratterizzati da un confronto esaustivo tra tutte le possibili combinazioni (quadratiche) di richieste di corse e di veicoli non scalano a sufficienza per essere implementati su una piattaforma in tempo reale.

Questa tesi presenta un nuovo approccio che, basato su un filtro di località, e combinato con un algoritmo di ride sharing scalabile facente uso di *shareability network*, guadagna un fattore di *speed up* significativo rispetto agli approcci noti, assicurando al contempo un impatto minimale sulla qualità della soluzione di

ride sharing calcolata. La nuova proposta è corroborata da un vasto insieme di esperimenti, per i quali si è utilizzato un dataset che consta di un mese di richieste di corse ($\sim 10^6$) eseguite in due distinte aree urbane: Manhattan (NYC), e Singapore.

La ricerca condotta in questa tesi si inquadra in un progetto scientifico sostenuto dal PROGRAMMA MIT-UNIPI 2019 e che ha visto la collaborazione tra il gruppo di ricerca *A³Lab* (`acube.di.unipi.it`) dell'Università di Pisa, diretto dal Prof. Ferragina, e il *MIT Senseable City Lab* (`senseable.mit.edu`), diretto dal Prof. Ratti. Durante il mio periodo di tesi ho potuto dunque collaborare con ricercatori esperti di algoritmi scalabili per problemi di mobilità condivisa (afferenti al laboratorio MIT) e ricercatori esperti di algoritmi su grafi e compressione dati (afferenti al laboratorio *A³Lab*). Il risultato finale di questa tesi è oggetto di un articolo scientifico (*in allegato* alla documentazione presentata) che è stato sottomesso, ed è in fase di revisione, alla rivista IEEE Transactions on Intelligent Transportation systems (IF = 6.319).

Parole chiave— Sistemi di trasporto intelligente, smart cities, economie condivise, ride sharing, instradamento di veicoli, mobilità urbana su richiesta, filtro di località, strutture dati geometriche

Introduzione

Il ride sharing (anche noto come *carpooling*, oppure "covetturaggio" in italiano) rappresenta una proposta di lungo corso per ridurre il traffico di veicoli, soddisfacendo allo stesso tempo il bisogno di mobilità di una popolazione di passeggeri che si spostano all'interno di una determinata area urbana. Ad oggi sono disponibili molteplici servizi di ride sharing (Uber, Didi, Lyft, solo per citarne alcuni), e l'impatto delle tecniche di ride sharing promette di diventare ancora più centrale in un prossimo futuro con l'introduzione dei veicoli a guida autonoma.

Lavori recenti hanno messo in evidenza che in aree urbane ad alta densità demografica – come ad esempio Manhattan (NYC) – sia possibile in linea di principio combinare la maggior parte delle richieste di corse punto-punto (ad esempio, corse di taxi), e guadagnare pertanto una significativa riduzione tanto nel numero di veicoli necessari a servire tali corse, quanto nel numero totale di chilometri percorsi dalla flotta di taxi; cfr. [1, 2, 3].

Sfortunatamente, l'applicazione dei sopra-menzionati approcci a scenari in tempo reale pone tuttora sfide di scalabilità significative. Tipicamente, un servizio di ride sharing su richiesta necessita della raccolta di tre tipi di dati in ingresso, al fine di ottimizzare la combinazione di richieste di taxi in una singola corsa condivisa da poter successivamente assegnare ad un veicolo.

La FIGURA 1 rappresenta il tipico processo in tempo reale di raccolta e di matching ottimale di richieste di corse.

Un tipo di input consta di richieste di corse, le quali si raccolgono tipicamente da utenti tramite un'applicazione (utente) per smartphone. Le richieste di corse definiscono – tra gli altri parametri – il luogo di partenza della corsa,

nonché la destinazione prescelta, che noi indicheremo nel seguito come *endpoint* della corsa. Un secondo tipo di input è la posizione e lo status dei veicoli facenti parte della flotta; questi dati vengono generalmente acquisiti tramite una seconda applicazione per smartphone, a disposizione degli autisti. Un terzo tipo di input, che si rende anch'esso necessario per l'ottimizzazione del servizio di mobilità, è costituito da una stima accurata dei tempi di attraversamento tra (un sottoinsieme di) endpoint e (un sottoinsieme di) veicoli della flotta. Questa stima sui tempi risulta la chiave del processo di ottimizzazione, permettendo di stimare con accuratezza l'efficienza di diverse combinazioni di richieste di corse in viaggi condivisi, oltre alla loro assegnazione ai veicoli disponibili.

Minore è la dimensione di tale sottoinsieme di *corse candidate per un match*, minore è il tempo di calcolo per la soluzione del problema di ottimizzazione; tuttavia la selezione di un numero eccessivamente esiguo di candidati *potrebbe degradare* la soluzione finale di ride sharing calcolata dall'algoritmo di ottimizzazione, sia in termini di chilometri totali percorsi, sia di tempi medi di attesa a cui sono sottoposti i passeggeri, sia in termini di numero di veicoli usati.

Finora la ricerca di soluzioni scalabili per la mobilità condivisa su richiesta si è focalizzata principalmente sull'efficacia del problema di ottimizzazione in sé, assumendo di avere già a disposizione in input: le richieste di corse, le posizioni dei veicoli, le stime sui tempi di attraversamento, e soprattutto il sottoinsieme di corse candidate per un match. Questo è il caso, ad esempio, degli approcci introdotti in [1, 2, 3]. Tuttavia, mentre l'acquisizione delle richieste di corse e delle posizioni dei veicoli è relativamente semplice da implementare, questo non è il caso per la selezione di un sottoinsieme opportuno di corse candidate per un match. Chiaramente un approccio forza bruta che calcoli i tempi di attraversamento per tutte le possibili combinazioni di endpoint di corse e di posizioni di veicoli non è proponibile, sia in termini di tempi di calcolo (che esploderebbero quadraticamente nel numero di richieste di corse $\mathcal{T}$), sia in termini di occupazione in memoria, sia di costo. Difatti, ogni confronto tra richieste di corse risulta in alcuni calcoli (talvolta dispendiosi) di cammini

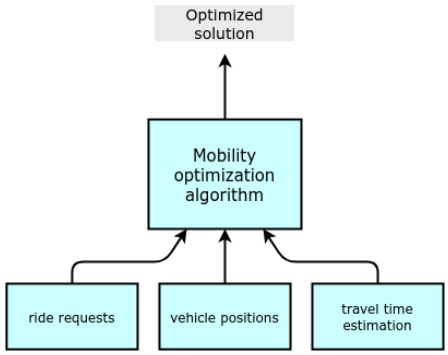
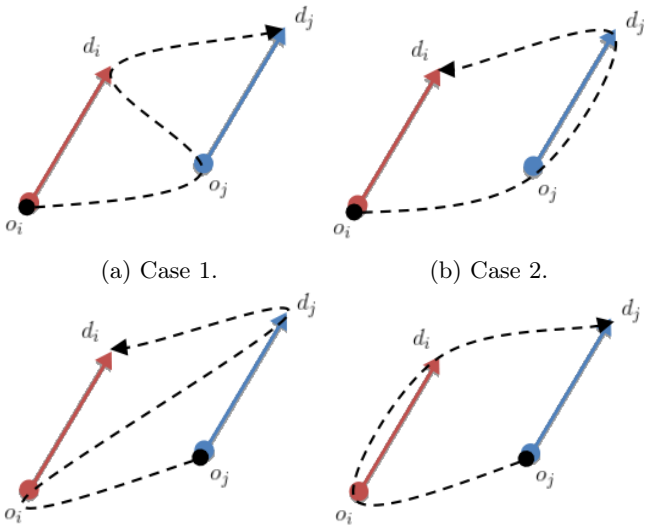


Figura 1: Il processo di ottimizzazione della mobilità su richiesta necessita di tre tipi di dati in input.



(a) Case 1. (b) Case 2.

(c) Case 3. (d) Case 4.

Figura 2: Modi validi di combinare una coppia di corse nel problema del ride sharing. Le linee rossa e blu rappresentano due diverse richieste di corse, che denotiamo rispettivamente con T_i e T_j. Le linee nere tratteggiate rappresentano i 4 diversi percorsi che il veicolo può seguire per combinare T_i e T_j. Nelle figure, o_i (risp. o_j) rappresenta l'origine della corsa T_i (risp. T_j), mentre d_i (risp. d_j) ne rappresenta la destinazione.

minimi [4, 5, 6, 7], che peraltro potrebbero portare a valutare distanze anche tra coppie di corse i cui endpoint appaiono molti lontani gli uni dagli altri, e che quindi con ottime probabilità non risulteranno in un match valido.

La FIGURA 2 mostra i quattro possibili pattern che possono essere usati per combinare due diverse richieste di taxi nel problema del ride sharing.

Qualora le stime sui tempi di attraversamento siano ottenute tramite servizi commerciali come Google Traffic, occorre tenere in considerazione un addebito di costi proporzionale al numero di query sottoposte, le quali potrebbero ammontare a $\Theta(|\mathcal{T}|^2)$ nel caso in cui si adottino approcci banali. Si delinea pertanto la necessità di progettare soluzioni intelligenti al fine di filtrare efficientemente e poi calcolare i tempi di attraversamento solo per un sottoinsieme appropriato di corse candidate per un match, rinunciando ad esaminare tutte le coppie di corse possibili, pur prestando attenzione a non perdere quelle coppie che contribuiscono al calcolo della soluzione di ride sharing ottimale.

Per abbracciare questa sfida, introduciamo in questa tesi un nuovo approccio di filtraggio basato su considerazioni sugli spazi e sulle geometrie delle reti stradali. L'approccio si propone di ridurre sostanzialmente il numero di stime sui tempi di attraversamento; di conseguenza si ridurrà anche il numero di corse candidate per un match, necessarie per il processo successivo di ottimizzazione.

Sebbene l'approccio di filtraggio proposto risulti sufficientemente generale da essere applicato ad una varietà di altri problemi di ottimizzazione nell'ambito della mobilità su richiesta, in questa tesi dimostriamo la sua applicazione al problema dell'approssimazione efficiente dell'insieme ottimale di corse valide per una condivisione nel contesto di una applicazione di ride sharing.

Il nostro contributo

Presentiamo un nuovo algoritmo e le strutture dati correlate che risolvono efficientemente il problema del ride sharing.

Il nostro algoritmo migliora quello proposto in [1], tramite un'attenta orchestrazione di condizioni su geometrie e tempi che permette di ottenere un forte abbattimento del numero σ di corse candidate che devono essere controllate per un dato match.

Sfruttiamo misure empiriche statistiche rilevate sul grafo della città per introdurre una tecnica di filtraggio in grado di definire un'area di località in cui con maggiore probabilità sia possibile individuare candidati per una determinata richiesta di mobilità.

La FIGURA 3 dimostra la forte correlazione che insiste tra distanze tra endpoint nello spazio (espresse come distanze euclidee sulle coordinate GPS), e tempi di attraversamento in: (a) Manhattan (NYC), e (b) Singapore.

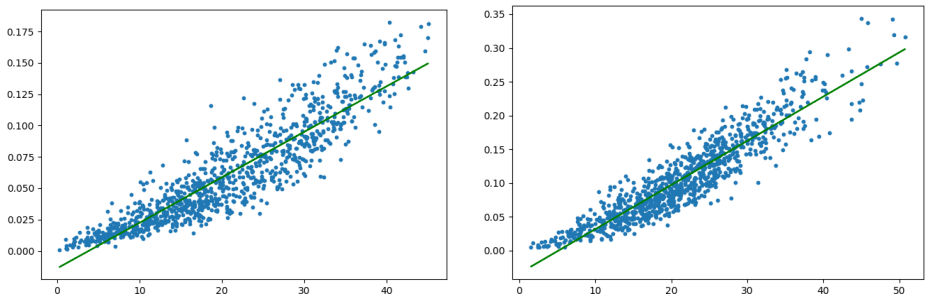
Il filtro di località, che costituisce un tassello chiave nella nostra soluzione, può essere adattato per trattare diverse reti stradali e condizioni di traffico. Il filtro è inoltre progettato per *auto-adattarsi* quando si passa a considerare richieste di taxi effettuate in aree eterogenee della città (ad esempio: il centro e le periferie), in cui è ricorrente sperimentare rapide variazioni nelle velocità medie di percorrenza.

Di seguito, denoteremo l'approccio originario [1] come algoritmo *legacy*, il quale incorre nelle due seguenti limitazioni principali:

- Manca di una correlazione tra spazi e tempi. Si procede quindi alla verifica della validità di un match tra due corse anche se le due richieste di taxi si riferiscono a regioni della città molto lontane le une dalle altre. Per questa ragione il numero di coppie (T_i, T_j) di corse da controllare cresce *quadraticamente* nel numero di richieste effettuate, vale a dire $\Theta(|\mathcal{T}|^2)$;
- Fa uso massiccio del costoso algoritmo dei cammini minimi di Dijkstra, al fine di valutare le distanze tra ogni coppia di nodi (origini e destinazioni) nel grafo della città.

Il nostro nuovo algoritmo apporta invece il seguente triplice contributo:

- Riduce il numero di coppie (T_i, T_j) che devono essere esaminate applicando un insieme di *condizioni di prossimità geometrica* tra le corse T_i e T_j.



(a) Manhattan (NYC) (b) Singapore

Figura 3: I grafici (a) e (b) riportano i valori delle distanze euclidee (asse y) in funzione dei valori dei tempi di attraversamento (asse x). Più in dettaglio, ciascuno dei 10^3 punti blu rappresenta un qualche cammino minimo tra due incroci stradali estratti in maniera random da (a) Manhattan, e (b) Singapore. I tempi di attraversamento sono misurati in minuti. I valori delle distanze sono calcolati come distanze euclidee usando le coordinate del *global positioning system* (GPS), e costituiscono pertanto dei numeri puri. Si osserva immediatamente che i valori di tempi e distanze evidenziano una forte correlazione, che permette di approssimare i valori delle distanze (asse y) a partire dai valori dei tempi (asse x), ad esempio tramite un'approccio basato sulla regressione lineare. Difatti, la linea verde nei grafici rappresenta una approssimazione lineare di $d = d(\tau)$ come $a \cdot \tau + b$, in cui d denota le distanze, e τ denota i tempi di attraversamento; a e b rappresentano rispettivamente il coefficiente angolare e la quota. Utilizzando questa tecnica di regressione lineare abbiamo ottenuto un *coefficiente di determinazione* $R^2 = 0.83$ per Manhattan ($a = 3.63 \cdot 10^{-3}$, $b = -1.39 \cdot 10^{-2}$), e $R^2 = 0.85$ per Singapore ($a = 6.55 \cdot 10^{-3}$, $b = -3.40 \cdot 10^{-2}$).

- Riduce il tempo necessario per l'individuazione delle coppie (T_i, T_j) che appaiono *geometricamente vicine*, sfruttando alcune strutture dati geometriche di comprovata efficienza per le ricerche di oggetti su spazi bidimensionali.
- Si avvale di alcuni algoritmi allo stato dell'arte per il calcolo delle distanze, sfruttando in particolare la struttura planare del grafo della città e le annesse (e talvolta nascoste) *highways* (autostrade, vie principali). Si registra di conseguenza un notevole miglioramento nei calcoli dei cammini minimi.

Come risultato, il nostro filtro di località è in grado di costruire efficientemente sia in tempo che in spazio una versione *sparsa* della *shareability network* introdotta in [1], permettendo di determinare in modo ottimale la combinazione tra richieste di corse, e di calcolare quindi una soluzione ottima di ride sharing in tempo reale. Abbiamo testato il nostro approccio basato sul filtro di località, in particolare confrontando l'ottimalità della soluzione di ride sharing individuata per mezzo della rete sparsa calcolata tramite il nostro filtro, con la soluzione

ottima ottenuta considerando una rete *completa* in cui sono valutati i tempi di attraversamento tra tutte le possibili coppie di corse, cfr. [1].

L'analisi asintotica, proposta nella tesi per descrivere le prestazioni del nuovo approccio, è successivamente corroborata da una batteria di esperimenti condotti su due dataset di richieste di corse di taxi effettuate nel 2011 e facenti riferimento a due aree urbane di diverse estensioni: il piccolo distretto di Manhattan nella città di New York (59.1 km^2), e la metropoli di Singapore (721.5 km^2). Abbiamo inoltre analizzato l'efficienza della nostra soluzione su diversi slot temporali (ad esempio, ore di punta e ore notturne). Durante l'indagine sperimentale è emerso come il nostro approccio sia in grado di indurre un fattore di speed up fino a 5X (o persino maggiore) rispetto a [1], e specialmente durante le cosiddette "ore di punta" (*rush time*), vale a dire quando i sistemi di ride sharing devono farsi carico di un numero maggiore di richieste da parte degli utenti. Siamo inoltre in grado di combinare migliaia di richieste nell'arco di pochi secondi, cosicché il possibile ritardo indotto dai tempi di calcolo del nostro algoritmo può essere considerato di fatto trascurabile; pertanto le prestazioni del nostro approccio risultano conformi con le specifiche di tempo di una piattaforma di ride sharing.

In sintesi, i risultati riportati in questa tesi mostrano chiaramente come la nostra soluzione sia robusta sotto diversi scenari. La soluzione risulta sufficientemente flessibile per bilanciare *recall* (ossia, numero di candidati validi individuati) da un lato, e speed up temporale dall'altro; risulta inoltre possibile estendere l'approccio di filtro di località anche al problema dell'assegnazione dei veicoli alle corse, fornendo quindi suggerimenti per lo sviluppo di algoritmi e strutture dati simili anche in questo contesto.

Riferimenti bibliografici

[1] P. Santi, G. Resta, M. Szell, S. Sobolevsky, S. H. Strogatz, and C. Ratti, "Quantifying the benefits of vehicle pooling with shareability networks," *Proceedings of the National Academy of Sciences*, vol. 111, no. 37, pp. 13 290–13 294, 2014.

[2] M. Vazifeh, P. Santi, G. Resta, S. Strogatz, and C. Ratti, "Addressing the minimum fleet problem in on-demand urban mobility," *Nature*, vol. 557, 05 2018.

[3] J. Alonso-Mora, S. Samaranayake, A. Wallar, E. Frazzoli, and D. Rus, "On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment," *Proceedings of the National Academy of Sciences*, vol. 114, no. 3, pp. 462–467, 2017.

[4] M. Potamias, F. Bonchi, C. Castillo, and A. Gionis, "Fast shortest path distance estimation in large networks," in *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, ser. CIKM '09. New York, NY, USA: Association for Computing Machinery, 2009, p. 867–876.

[5] H. Bast, D. Delling, A. Goldberg, M. Müller-Hannemann, T. Pajor, P. Sanders, D. Wagner, and R. Werneck, *Route Planning in Transportation Networks*, 11 2016, vol. 9220, pp. 19–80.

[6] I. Abraham, A. Fiat, A. V. Goldberg, and R. F. Werneck, "Highway dimension, shortest paths, and provably efficient algorithms," in *Proceedings of the Twenty-first Annual ACM-SIAM Symposium on Discrete Algorithms*, ser. SODA '10. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2010, pp. 782–793.

[7] R. Geisberger, P. Sanders, D. Schultes, and C. Vetter, "Exact routing in large road networks using contraction hierarchies," *Transportation Science*, vol. 46, no. 3, p. 388–404, Aug. 2012.