Pay Your Trip for Traffic Congestion: Dynamic Pricing in Traffic-Aware Road Networks

Lisi Chen,[1,2] **Shuo Shang,**[1]* **Bin Yao,**[3] **Jing Li**[2]
[1]UESTC, China
[2]Inception Institute of Artificial Intelligence, UAE
[3]Shanghai Jiao Tong University, China
{chenlisi.cs, jedi.shang}@gmail.com, yaobin@sjtu.edu.cn, jing.li@inceptioniai.org

Abstract

Pricing is essential in optimizing transportation resource allocation. Congestion pricing is widely used to reduce urban traffic congestion. We propose and investigate a novel *Dynamic Pricing Strategy* (DPS) to price travelers' trips in intelligent transportation platforms (e.g., DiDi, Lyft, Uber). The trips are charged according to their "congestion contributions" to global urban traffic systems. The dynamic pricing strategy retrieves a matching between n travelers' trips and the potential travel routes (each trip has k potential routes) to minimize the global traffic congestion. We believe that DPS holds the potential to benefit society and the environment, such as reducing traffic congestion and enabling smarter and greener transportation. The DPS problem is challenging due to its high computation complexity (there exist k^n matching possibilities). We develop an efficient and effective approximate matching algorithm based on *local search*, as well as pruning techniques to further enhance the matching efficiency. The accuracy and efficiency of the dynamic pricing strategy are verified by extensive experiments on real datasets.

Introduction

With the trend towards urbanization and the rapid development of urban transportation, traffic congestion becomes a huge problem for many major cities. For vehicles, traffic congestion means slower speeds, longer trip times, and increased vehicular queueing. For society and the environment, traffic congestion means more energy and money consumption, and increased greenhouse-gas emission and air pollution. How to reduce traffic congestion is the main problem faced by the governments in many countries.

Pricing plays an essential role in optimizing transportation resource allocation. Nowadays, congestion pricing is widely used to reduce traffic congestion. For example, in Singapore, congestion pricing is introduced for vehicles entering the center business area in peak hours (e.g., 18:00–19:30), while in London, Stockholm, and Milan, congestion pricing is charged for almost the whole day (e.g., 07:00–19:00) in the central area. However, these pricing strategies simply focus on "removing vehicles from hot areas", but have nothing to do with "balancing the global traffic load", which may lead to traffic congestion in other areas.

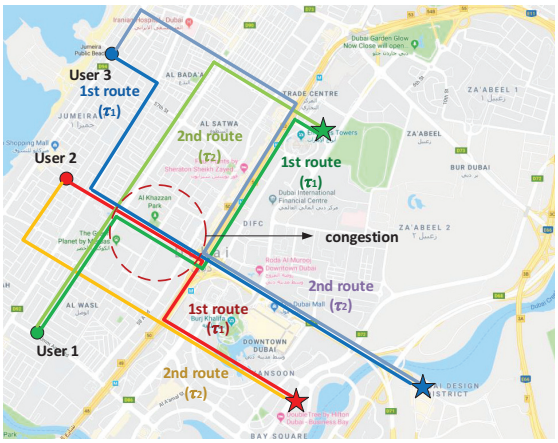


Figure 1: Trips and potential routes

In this light, we propose and investigate a novel *Dynamic Pricing Strategy* (DPS) in traffic-aware road networks. Assume that n travelers are planning their trips at the same time, and each trip may have k potential routes to connect the source and destination (e.g., the fastest route, the cheapest route, the greenest route, unobstructed routes). The trips are charged according to their "congestion contributions" to global urban traffic systems, and the DPS problem finds a matching between n travelers' trips and the potential travel routes to minimize the global traffic congestion. We believe that DPS holds the potential to benefit society and the environment, such as optimizing transportation resource allocation and enabling smarter and greener transportation.

An example is shown in Figure 1. Three users (u_1,u_2, and u_3) are planning their trips (n=3). The source and destination of a trip are denoted by circle and star, respectively. Each trip (user) has two potential routes (k=2), which are denoted by τ_1 and τ_2. So we have 2^3 possible route matchings in total. One of the matching is: $u_1 \rightarrow \tau_1$, $u_2 \rightarrow \tau_1$, and $u_3 \rightarrow \tau_1$. However, this matching may aggravate the traffic congestion in the road segment denoted by the red dot-

*Corresponding author.

ted circle in Figure 1. Another matching can be: $u_1 \rightarrow \tau_2$, $u_2 \rightarrow \tau_2$, and $u_3 \rightarrow \tau_1$, which may induce less traffic congestion in the circled road segment. We aim to find a matching between the three trips (users) and their potential routes to minimize the global traffic congestion.

The DPS problem is challenging due to its high computation complexity (there exist k^n matching possibilities). To address the challenge, we first propose a route pricing model to compute the congestion contribution made by each individual potential route. Next, based on the model we develop an efficient and effective approximate local-search based matching algorithm. In particular, given a collection of trips departing at the same time, we first generate an initial matching result. Then we explore the most effective route swaps regarding each trip. Our swapping-based approximate matching algorithm reduces the time complexity from $O(k^n)$ to $O(k \cdot n)$. To further improve the matching efficiency, we propose a pre-checking scheme to help prune unqualified swaps. Experiments on two real-life datasets show that our proposed algorithm is capable of achieving both high efficiency and high accuracy compared against the exact route matching algorithm.

Related Work

Existing studies related to the DPS problem can be classified into two categories: location-based route recommendations and traffic-aware route planning.

Location-based route recommendations The location-based route recommendations aim at finding a new route based on some user-defined criteria. Representative studies include mobile sequential recommendation that finds an optimal route with the minimum distance to a driver's next potential passenger (Ge et al. 2010; Ye et al. 2019; Ye, Xiao, and Deng 2018). Further, Ye et al. (2018) take a group of taxis from different locations as input. The problem is to find an optimal route for each taxi in order to reach an global optimization w.r.t. passenger deliveries. Another line of research aims to generate the shortest route by taking multiple costs into account (Dai et al. 2015; Xie et al. 2012; Shang et al. 2016a; Cao et al. 2012; Zeng et al. 2015). The problem of matching existing trajectories based on user requirements (Shang et al. 2014a; Chen et al. 2019) is also investigated. However, these studies only take individual travel distance or travel cost into consideration. They do not consider the effect of global traffic congestions contributed by a set of trips.

Traffic-aware route planning Traffic-aware route planning aims to derive an optimal route by avoiding potential traffic congestions. Existing studies on this topic can be classified into the following categories: (1) Shortest path search over traffic-aware road networks; (2) Pricing strategies in the context of spatial crowdsourcing; (3) Trajectory search and join. Specifically, Ding et al. (2008) and Hua et al. (2010) investigate the problem of finding the shortest path over traffic-aware road networks. Yang et al. (2014) aim to find stochastic skyline routes by considering multiple costs over a time-dependent road network with uncertainty. Shang et al. (2013; 2014b; 2015) study the problem of finding a route from a user-specified source location to a user-specified target location that has the minimum congestion probability on a given traffic-aware road network. Levin et al. (2014) study traffic-aware route search that takes a start location, a target location, and a set of terms as input. The goal is to find the fastest route from the start location to the target via PoIs that cover the query terms over a traffic-aware road network. Tong et al. (2018; 2017) Liu et al. (2018) develop pricing strategies in the context of spatial crowdsourcing. However, they focus on worker-task matching, which is different from our focus of minimizing global traffic congestions. Further, trajectory search aims to find similar trajectories based on a set of query locations (Shang et al. 2019; 2017a), and trajectory join finds similar trajectory pairs over a collection of trajectories (Shang et al. 2017b; 2018). The similarity functions may cover spatial domain (Chen et al. 2010), temporal domain (Shang et al. 2016b), and textual domain (Shang et al. 2012; Zheng et al. 2013). However, these studies focus on local optimization in query level. They do not regard the global traffic congestion as their optimization target.

Preliminaries

This section introduces the definitions of traffic-aware road network, trip, route, and traffic congestion level.

Traffic-Aware Road Networks

A traffic-aware road network is a connected graph $G = (V, E, W, c_{max})$, where V is a vertex set and $E \subseteq \{\{v_i, v_j\} | v_i, v_j \in V \wedge v_i \neq v_j\}$ is an edge set. A vertex $v_i \in V$ represents a road intersection or an end of a road, and an edge $e_k = \{v_i, v_j\} \in E$ represents a road segment that enables travel between vertices v_i and v_j. Function $W : E \mapsto \mathbb{R}$ assigns a real-valued weight $W(e)$ to an edge e that represents the corresponding road segment's length. Function $c_{max} : E \mapsto \mathbb{N}$ assigns a natural-valued *maximum capacity* $c_{max}(e)$ to an edge e that denotes the maximum number of routes (vehicles) traveling on the road segment. We use $\mathsf{n}(e, t)$ to denote the number of routes (vehicles) on road segment e at time t.

Trips and Routes

Next, we present the definitions of trip and route.

Definition 1: (*Trip*) A trip $tr = \{s, d, t_s\}$ consists of a source location s, destination location d, and the departure time t_s. □

We assume that each trip has k potential routes, offered by a map service, to connect the source and destination (e.g., fastest routes, shortest routes, greenest routes). The definition of route is presented as follows.

Definition 2: (*Route*) Given a traffic-aware road network $G = (V, E, W, c_{max})$, a route τ on G is defined by a tuple $\{\mathbf{p}, t_s\}$, where $\mathbf{p}$ is a finite sequence $\langle p_1, p_2, ..., p_n \rangle$ that consists of at least 2 vertices and t_s denotes the departure time of the route. Here, p_i and p_{i+1} ($i \in [1, n-1]$) are adjacent vertices in V. □

The price of route τ, denoted by $p(\tau)$, is calculated based on its congestion contribution to global urban traffic systems.

Level of Traffic Congestion

Following some popular online Map services (e.g., Google Maps[1]), we assign a traffic congestion level for edge e based on a pre-defined *traffic congestion level classification scheme* CL: $(\mathsf{n}(e,t),\Theta) \mapsto \mathbb{N}$. Here, Θ is a vector of thresholds $\langle \theta_0, \theta_1, \theta_2, ..., \theta_m \rangle$ where $\theta_0 = 0$, $\theta_i \in (0,1)$, m represents the total number of levels, and $\theta_i < \theta_j$ if $i \leq j$. CL generates a natural-valued traffic congestion level of e. For example, in Google Maps, m is set to 3 as it has three congestion levels (i.e., "green", "orange", and "red" levels). We assign $\mathsf{n}(e,t)$ to level i if $\theta_{i-1} \cdot c_{max}(e) \leq \mathsf{n}(e,t) < \theta_i \cdot c_{max}(e)$.

Problem Statement

This section defines the problem of our Dynamic Pricing Strategy (DPS). We aim to solve the following subproblems:

(1) *Individual traffic-aware route pricing*: Given a route τ, how to determine its congestion contribution to global urban traffic systems?

(2) *Global route matching*: Given a collection T of n trips departing at timestamp t and k potential routes for each of them, find a matching between n trips and their potential travel routes to minimize the global traffic congestion. The route matching problem is defined by Definition 3.

Definition 3: (*Global route matching problem*) Let $T = \{tr_1, tr_2, ..., tr_n\}$ be a collection of n trips departing at timestamp t (i.e., for any $tr_i \in T$, $tr_i.t_s = t$), $P_i = \{\tau_1, \tau_2, ..., \tau_k\}$ be the set of k potential routes of trip tr_i where for any $\tau_j \in P_i$, $\tau_j.t_s = t$. The GRM problem finds an optimal route τ_r from P_i for each $i \in [1,n]$ such that the *global traffic congestion factor* is minimized. □

Global Traffic Congestion Factor

The global traffic congestion factor measures the traffic congestion of a road network contributed by a collection of routes. It is computed by aggregating the increments of congestion levels regarding each road segment of a road network. Specifically, the increment of congestion level induced by route collection R regarding segment e (i.e., $\delta(R,e)$) is computed by Equation 1.

$$\delta(R,e) = \max_{t \in [t_{min}, t_{max}]} \{\mathsf{CL}([\mathsf{n}(e,t) + |R^{(e,t)}|], \Theta) - \mathsf{CL}(\mathsf{n}(e,t), \Theta)\} \tag{1}$$

where $R^{(e,t)}$ denotes a subset of routes in P that travel on segment e at time t, t_{min} denotes the earliest time that routes in R enter e, t_{max} denotes the latest time that routes in P leave e, CL represents the traffic congestion level function, and Θ denotes a vector of thresholds. Based on Equation 1, we can see that $\delta(P,e)$ computes the maximum increment of congestion level of e when taking the routes in R into account. Following existing study (Tong et al. 2018), we assume that $\mathsf{n}(e,t)$ and the real-time vehicle speed on each road segment are given.

Further, we observe that users are less likely to accept a route if it is charged at a higher price. To take such user behavior into consideration, we adopt a user acceptance model $\mathsf{U} : p(\tau) \mapsto prob$ and combine it with congestion increment level. Here, $prob$ denotes the probability that a user accepts the price charged by traveling on τ (Tong et al. 2018). Given P_i (cf. Definition 3), we use the model satisfying the following conditions *w.l.o.g*:

$$\sum_{\tau_j \in P_i} \mathsf{U}(p(\tau_j)) = 1$$

$$\forall (\tau_j, \tau_k \in P_i)\ (\mathsf{U}(p(\tau_j)) \times p(\tau_j) = \mathsf{U}(p(\tau_k)) \times p(\tau_k)).$$

We replace the component $|R^{(e,t)}|$ in Equation 1 by the sum of user acceptance probabilities of routes in $R^{(e,t)}$. Note that our proposal is independent to the user acceptance model. Based on Equation 1 and user acceptance model U, we compute the user-based congestion increment level induced by R regarding e (i.e., $\delta_u(R,e)$) by Equation 2:

$$\delta_u(R,e) = \max_{t \in [t_{min}, t_{max}]} \{\mathsf{CL}([\mathsf{n}(e,t) + \sum_{\tau \in R^{(e,t)}} \mathsf{U}(p(\tau))], \Theta) - \mathsf{CL}(\mathsf{n}(e,t), \Theta)\} \tag{2}$$

Next, we present the definition of global traffic congestion factor.

Definition 4: (*Global traffic congestion factor*) Global traffic congestion factor of R, denoted by $\mathsf{CF}(R)$, is computed by the sum of user-based congestion increment level regarding each road segment:

$$\mathsf{CF}(R) = \sum_{e \in E} \delta_u(R,e) \tag{3}$$

□

Individual Traffic-Aware Route Pricing

This section presents our pricing model for each individual route. Our objective is to charge a route according to its "congestion contribution" to global urban traffic systems. Based on Definition 4, Equations 1 and 2, we see that the global traffic congestion factor is directly related to the congestion increment level regarding each road segment. Thus, we consider the following two factors in determining the price of a route τ:

(1) *Congestion ratio*: The ratio of the vehicle count to capacity regarding each road segment traveled by τ;

(2) *Upgrade margin*: The gap between the current vehicle count and the minimum vehicle count of the next congestion level regarding each road segment traveled by τ (cf. Definition 5).

Definition 5: (*Upgrade margin*) Given a road segment e, the vehicle count of e at time t (i.e., $\mathsf{n}(e,t)$), and a traffic congestion level classification scheme $\mathsf{CL} : (\mathsf{n}(e,t), \Theta)$

[1]https://support.google.com/maps/answer/3092439

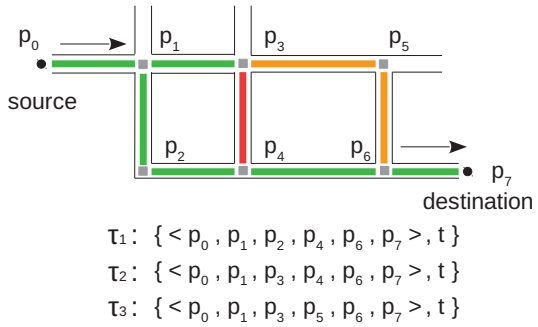


Figure 2: Traffic-aware road network

where $\Theta = \langle \theta_0, \theta_1, ..., \theta_m \rangle$, the upgrade margin, denoted by $\mathsf{UM}(e,t)$, is computed by:

$$\mathsf{UM}(e,t) = c_{max}(e) \times \theta_i - \mathsf{n}(e,t),$$

where $i < m$ and i denotes the current congestion level of $\mathsf{n}(e,t)$. □

The price model for route τ is presented by Equations 4 and 5. In particular, $\mathsf{pf}(\tau)$ denotes the price factor of τ.

$$\mathsf{pf}(\tau) = \sum_{e_i \in \tau} [\alpha \times \frac{\mathsf{n}(e_i,t)}{c_{max}(e_i)} + (1-\alpha) \times \frac{1}{\mathsf{UM}(e_i,t)}] \quad (4)$$

For each road segment e_i that traveled by τ, we linearly combines its congestion ratio (i.e., $\mathsf{n}(e_i,t)/c_{max}(e_i)$) and the inverse of upgrade margin (i.e., $1/\mathsf{UM}(e_i,t)$). Here, α is a parameter that balances the weight between congestion ratio and upgrade margin. The actual price of τ is set by Equation 5.

$$p(\tau) = b + \beta \times \mathsf{pf}(\tau), \quad (5)$$

where b is the base price, $\beta \times \mathsf{pf}(\tau)$ denotes the congestion surcharge, which is proportional to the price factor if τ, and β is a parameter indicating the weight of price factor.

Example 1: Figure 2 presents a small traffic-aware road network. We have a trip $tr = \{p_0, p_7, t\}$ departing at time t from p_0 to p_7. We use a uniform capacity $c_{max} = 150$ and a uniform traffic congestion level classification scheme $\Theta = \langle \theta_0, \theta_1, \theta_2 \rangle$, where $\theta_0 = 0$, $\theta_1 = 1/3$, and $\theta_2 = 2/3$, for all road segments. Note that in real-life scenario, road segments may have different capacity and congestion level classification schemes, which are determined based on their lengths, widths, speed limits, etc. The traffic conditions, i.e., vehicle count, congestion level, and upgrade margin (calculated based on Definition 5), of each road segment is presented in Table 1. We assume that trip tr has three route candidates: τ_1, τ_2, and τ_3 (cf. Figure 2), and we set α to be 0.5. Based on Equation 5 and the traffic conditions in Table 1, we compute the price factor for each route as follows: $\mathsf{pf}(\tau_1) = 0.61$; $\mathsf{pf}(\tau_2) = 0.87$; $\mathsf{pf}(\tau_3) = 0.74$. We can see that route τ_1 is the cheapest choice among the three route candidates if we consider trip tr exclusively. □

Global Route Matching (GRM)

In the previous section, we presented how to determine the price of each route candidate of a trip based on its congestion contribution to the road network. However, we may have a great number of trips that depart at different locations concurrently. We need to find a matching between n trips and their potential routes to minimize the global traffic congestion level.

Table 1: Traffic condition of each road segment

Segment (e)	$\mathsf{n}(e,t)$	Cong. Level	Up. margin
$\langle p_0, p_1 \rangle$	10	1 (green)	40
$\langle p_1, p_2 \rangle$	35	1 (green)	15
$\langle p_1, p_3 \rangle$	22	1 (green)	28
$\langle p_2, p_4 \rangle$	16	1 (green)	34
$\langle p_3, p_4 \rangle$	112	3 (red)	38
$\langle p_3, p_5 \rangle$	73	2 (orange)	27
$\langle p_4, p_6 \rangle$	45	1 (green)	5
$\langle p_5, p_6 \rangle$	68	2 (orange)	32
$\langle p_6, p_7 \rangle$	24	1 (green)	26

From Definition 3, we see that there exist k^n matching possibilities. It is computationally prohibitive to evaluate each matching. To solve the GRM problem efficiently, we develop a local search algorithm that explores effective potential route swaps. Specifically, the algorithm consists of two steps:

(1) *Initial matching*: Firstly, we generate an initial matching by selecting the route candidate with the minimum price factor for each trip;

(2) *Route swapping*: Next, for each trip we swap the selected route with other $k-1$ candidates if the swapping operation can reduce the value of global traffic congestion factor by at least a ratio of ϵ. Here, we propose a *route pre-check technique* that can filter out unqualified potential swapping pairs by evaluating their common sub-routes.

Initial Matching

Algorithm 1 presents the pseudo code of initial matching. We take a collection T of n trips, a uniform departure time t, a traffic-aware road network G, and k route candidates for each trip as input, and select an initial matching route among k candidates for each trip.

We first initialize $R[i]$ that stores the initial matching route for trip tr_i. Next, for each trip tr_i we evaluate its route candidate set and select the one with the minimum price factor (Lines 3–10). The reason of choosing price factor as our ranking criterion can be explained as follow: (1) the computation of price factor is very efficient (cf. Equation 4); (2) the price factor has strong positive correlation with our final objective function, the global traffic congestion factor. Before evaluating the route candidate set P_i for each trip tr_i we initialize τ_m and pf_{min} that denote the route with minimum price factor and it value of minimum price factor, respectively (Lines 4–5). For each route $\tau_j \in P_i$, we compute its price factor $\mathsf{pf}(\tau_j)$ based on Equation 4 and select the minimum one (Lines 7–9). After evaluating all candidate sets, we return R, an array of n initial matching routes, as the result.

Algorithm 1: InitialMatching

Data: Trip collection $T = \{tr_1, tr_2, ..., tr_n\}$, departure time t, set P_i of k route candidates for each trip tr_i, and traffic-aware road network G
Result: Initial matching route τ_i for each trip tr_i

```
1  for each tr_i in T do
2  |  Initialize R[i];
3  for each P_i (1 ≤ i ≤ n) do
4  |  Initialize τ_m;
5  |  pf_min ← +∞;
6  |  for each τ_j ∈ P_i do
7  |  |  if pf(τ_j) < pf_min then
8  |  |  |  pf_min ← pf(τ_j);
9  |  |  |  τ_m ← τ_j;
10 |  R[i] ← τ_m;
11 return R;
```

Route Swapping

After generating the initial result, we need to refine the matching by swapping a selected route with other route candidates of a trip if the swapping is considered to be "effective". Specifically, an effective swap should lead to reduction of the global traffic congestion factor by at least ϵ. The high-level idea of route swapping is as follow. For each trip tr_i, we compare the selected route $R[i]$ against other candidates in P_i and find the candidate τ_r that induce the highest reduction of global traffic congestion factor when swapping $R[i]$ with τ_r. However, if swapping $R[i]$ with τ_r cannot lead to reduction of the global traffic congestion factor by at least ϵ, we skip this swap.

Algorithm 2 presents the pseudo code of route swapping. For each trip tr_i we evaluate its route candidate set P_i and explore effective swaps (Lines 2–15). Specifically, we first initialize variables s, τ_m, and v_{max} that denote the current reduction of the global traffic congestion factor, the route that has the highest reduction value, and the corresponding value, respectively (Lines 2–3). Next, we calculate the reduction of the global traffic congestion factor regarding the swap between the route selected by initial matching (i.e., $R_{init}[i]$) and each τ_j in $P_i \setminus \{R_{init}[i]\}$ (Line 5), which is denoted by $\mathsf{CF}_{swap}(R_{init}[i], \tau_j)$. In particular, the computation of $\mathsf{CF}_{swap}(R_{init}[i], \tau_j)$ is detailed in the next subsection. If the reduction value is lower than $\epsilon \cdot \mathsf{CF}(R_{init})$, we skip this swap (Lines 6–7). Otherwise, we conduct further checking. Specifically, if the reduction value is larger than v_{max}, we update τ_m and v_{max} (Lines 9–11). After evaluation all potential swaps, we update $R[i]$ to τ_m if there exists an effective swap; otherwise, we assign the initial matching result $R_{init}[i]$ to $R[i]$ (Lines 12–15).

Computation of CF_{swap} with Pre-Checking

We present an efficient method to compute $\mathsf{CF}_{swap}(\tau_i, \tau_j)$ by our pre-checking technique, where τ_i denote the selected route in candidate set P and τ_j denote an unselected route in P. Let R be the initial matching route set returned by Algorithm 1. A straightforward way to compute $\mathsf{CF}_{swap}(\tau_i, \tau_j)$

Algorithm 2: RouteSwap

Data: Trip collection T, Set P_i of k route candidates for each trip tr_i, traffic-aware road network G, intial matching result R_{init}, and parameter ϵ
Result: Final matching result R

```
1  for each P_i (1 ≤ i ≤ |T|) do
2  |  Initialize s, τ_m;
3  |  v_max ← 0;
4  |  for each τ_j ∈ P_i \ {R_init[i]} do
5  |  |  s ← CF_swap(R_init[i], τ_j);
6  |  |  if s < ε · CF(R_init) then
7  |  |  |  continue;
8  |  |  else
9  |  |  |  if s > v_max then
10 |  |  |  |  τ_m ← τ_j;
11 |  |  |  |  v_max ← s;
12 |  if v_max > 0 then
13 |  |  R[i] ← τ_m;
14 |  else
15 |  |  R[i] ← R_init[i];
16 return R;
```

is directly based on Equation 3 (i.e., $\mathsf{CF}(R) - \mathsf{CF}(R')$ where $R' = R \setminus \{\tau_i\} \cup \{\tau_j\}$), which is very time consuming.

Nevertheless, we observe that it is unnecessary to evaluate all routes in R and R'. Instead, we only need to compare τ_i and τ_j and calculate the increments induced by τ_i and τ_j respectively. In particular, we first compute the global traffic congestion factor of the intersection of R and R' (i.e., $\mathsf{CF}(R_c)$ where $R_c = R \cap R'$). Next, we calculate the numbers of road segments whose congestion levels are upgraded when adding τ and τ_j into R_c, respectively, and return their difference as $\mathsf{CF}_{swap}(\tau_i, \tau_j)$. Note that the numbers are weighted by user acceptance probability (cf. Equation 2). Although this method is much more efficient than computing $\mathsf{CF}(R) - \mathsf{CF}(R')$ directly, we still need to evaluate each road segment covered by τ_i and τ_j. Here, we further alleviate the computational cost by pre-checking common road segments shared by both τ_i and τ_j. Let $t_x(\tau, e)$ be the time when τ enter e and $t_y(\tau, e)$ be the time when τ leave e. We present our pre-checking scheme in Theorem 1.

Theorem 1: Given routes $\tau_i = \{\langle p_1, p_2, ..., p_m\rangle, t\}$ and $\tau_j = \{\langle q_1, q_2, ..., q_n\rangle, t\}$, let $\tau_c = \{\langle p_1, p_2, ..., p_s\rangle, t\}$ ($s \leq m$ and $s \leq n$) be the common sub-route shared by τ_1 and τ_2. We have $\mathsf{CF}_{swap}(\tau_i', \tau_j') = \mathsf{CF}_{swap}(\tau_i, \tau_j)$ where $\tau_i' = \{\langle p_s, p_{s+1}, ..., p_m\rangle, t_y(\tau_c, \langle p_{s-1}, p_s\rangle)\}$ and $\tau_j' = \{\langle q_s, q_{s+1}, ..., q_n\rangle, t_y(\tau_c, \langle p_{s-1}, p_s\rangle)\}$.

Proof. Because τ_i and τ_j have the same departure time t and they share the same road segments from p_1 to p_s, they arrive point p_s at the same time. Thus, τ_i and τ_j have the same congestion increment level regarding the common sub-route τ_s, and we have $\mathsf{CF}_{swap}(\tau_i', \tau_j') = \mathsf{CF}_{swap}(\tau_i, \tau_j)$. □

Based on Theorem 1, we only need to compute $\mathsf{CF}_{swap}(\tau_i', \tau_j')$, which alleviates the computational cost by filtering out the common sub-route τ_s.

Table 2: Parameter Settings

	BT	NT
Number of trips n	50–2,500 / default 100	100–5,000 / default 200
Number of route candidates k	3–7 / default 3	3–7 / default 3
Number of congestion levels $\|\Theta\|$	2–5 / default 3	2–5 / default 3
Swap parameter ϵ	5–50 ($\times n^{-1}$) / default 10	5–50 ($\times n^{-1}$) / default 10
Route candidate selection parameter m	5–20 / default: 10	5–20 / default: 10

Experimental Study

We report on experiments with real road networks and trajectory datasets.

Experiment Settings

Datasets Two datasets are used in our experiments: Beijing Taxi Trajectories (BT) and New York Taxi Trips (NT). The two datasets are detailed as follow.

BT: The underlying space of BT is a Beijing road network. The network graphs are stored and indexed by adjacency lists. We use a real taxi trajectory data set collected by the T-drive project (Yuan et al. 2013). The real-time traffic condition of each road segment is generated based on the taxi trajectories. Specifically, for each road segment we find a snapshot timestamp, denoted by t_{vmax} when the segment has the highest number of vehicles. We regard $\mathsf{n}(e, t_{vmax}) \times 1.5$ as $c_{max}(e)$. The congestion level classification vector Θ is set as $\langle 0, 1/3, 2/3 \rangle$ by default. The timespan of the original trajectories from the T-drive project are quite long (i.e., lasting for days). To create trips with a realistic length and duration, we divide these trajectories into hour-long sub-trajectories.

NT: The underlying space of NT is a New York Road Network[2]. In NT, we use a real taxi data set from New York. Each item in the data set contains pick-up and drop-off locations of a taxi. The other settings of NT is the same as BT.

Generating trip collection We generate trip collection by randomly selecting n source-destination pairs in the trajectory dataset. Next, we randomly pick a trip in the trip collection and regard its departure time as the uniformed departure time of all trips in the collection. The cardinality of trip collection is specified in Table 2.

Generating route candidate set For each trip we offer k route candidates. In particular, for BT we select the original trajectory in the dataset as the first candidate. Next, we generate top-m ($m > k$) shortest routes from the source location to the destination. We randomly select $k - 1$ routes from the m shortest routes as the remaining candidates. As for NT, we directly generate top-m ($m > k$) shortest routes and randomly select k routes from the m shortest routes as the candidates. Here, m is a parameter and its effect is evaluated in experiments (Table 2, Figure 8).

[2]https://publish.illinois.edu/dbwork/open-data/

Compared Algorithms

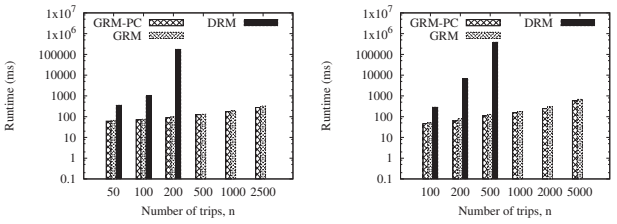
Direct Route Matching (DRM): The DRM is an exact search algorithm. It performs exhaustive search of k^n possible matching with a simple pruning method. In particular, we can safely prune routes that do not have any overlaps with other route candidates. The matching result generated by DRM has the lowest global traffic congestion factor.

Global Route Matching (GRM): The GRM algorithm consists of two steps: initial matching 1 and route swapping 2.

Global Route Matching with pre-check (GRM-PC): The GRM-PC is the GRM algorithm with pre-checking technique applied.

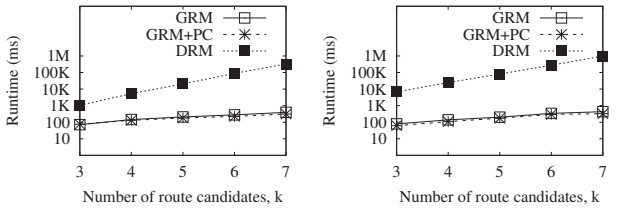
Evaluation settings All of the algorithms are run in memory. We report the cpu time for efficiency evaluation and report the ratio of global traffic congestion factor between GRM and DRM for effectiveness evaluation. Note that the matching results returned by GRM and GRM+PC are the same. Parameter settings are presented in Table 2.

Experimental Results



(a) BT (b) NT

Figure 3: Efficiency regarding the number of trips

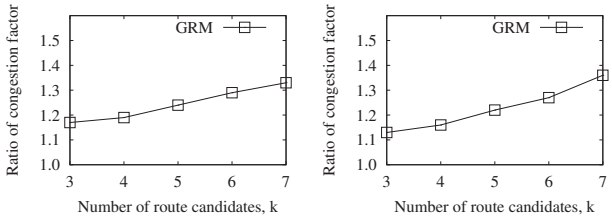


(a) BT (b) NT

Figure 4: Efficiency regarding # route candidates

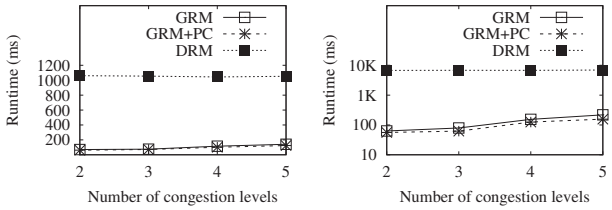
Effect of the number of trips This set of experiments (Figure 3) evaluates the runtime and efficacy performances as we vary the number of trips processed simultaneously. We see that the runtime of DRM increases significantly when we increase the number of trips. The reason is that in the worst case the DRM need to evaluate all possible matches (i.e., k^n). In contrast, the runtime of GRM and GRM+PC only

exhibits an linearly increasing trend when we increase the number of trips. Further, compared with GRM we observe that GRM+PC improves the runtime performance by a factor of 1.05–1.21, which underlines the effectiveness of our route pre-checking scheme.



(a) BT (b) NT

Figure 5: Effectiveness regarding # route candidates

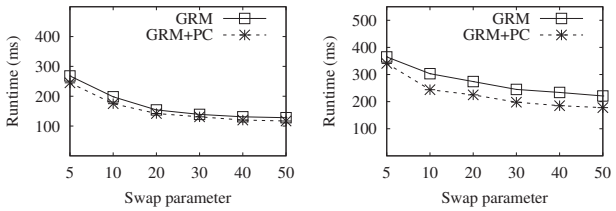


(a) BT (b) NT

Figure 6: Efficiency regarding the number of congestion levels

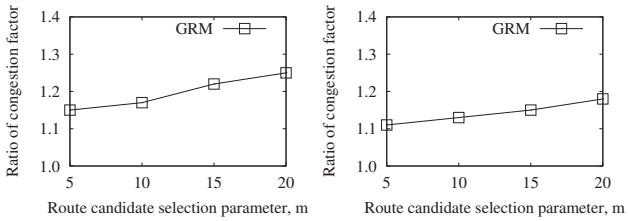
Effect of the number of route candidates in a trip Figure 4 presents the performance of the algorithms when varying k, the number of route candidates offered by each trip. A larger k leads to more matching possibilities. For DRM, it is obvious that the time cost increases exponentially as we increase k. In contrast, the GRM series are not sensitive to the value of k compared with DRM. The reason is that the total number of swap evaluations is just linearly related to k. Figure 5 shows the efficacy performance as we vary k. We can find that the ratio of congestion factor between GRM and DRM increases when we increase k. Nevertheless, even if we set k as 7 the ratios (1.33 and 1.36 in BT and NT, respectively) are still acceptable. Note that in real-life applications the value of k is generally small.

Effect of the number of congestion levels Figure 6 presents the performance when we vary $|\Theta|$, the number of congestion levels. Here, we let the thresholds be uniformly distributed. For example, we set Θ to be $\langle 0, 1/3, 2/3\rangle$ when $|\Theta| = 3$ and set Θ to be $\langle 0, 1/4, 1/2, 3/4\rangle$ when $|\Theta| = 4$. We observe that the performance of DRM is not affected by $|\Theta|$. This can be explained by the fact that the number of possible matching is independent of the number of congestion levels. However, the time cost of GRM series exhibits an increasing trend as we increase Θ. The reason is that a larger $|\Theta|$ will increase the number of effective swaps.



(a) BT (b) NT

Figure 7: Efficiency regarding the swap parameter



(a) BT (b) NT

Figure 8: Effectiveness regarding candidate selection parameter

Effect of swap parameter We proceed to evaluate the effect of varying the swap parameter (ϵ). The number of trips are set to 1,000 and 2,000 for BT and NT, respectively. From Figure 7, we find that when we increase ϵ, the runtime decreases for both GRM and GRM+PC. The reason is that increasing ϵ may decrease the number of swap operations. Additionally, we find that when the value of ϵ reaches $30/n$, the subsequent increase in runtime is modest.

Effect of candidate selection parameter Figure 8 shows the performance of accuracy when we vary the candidate selection parameter m. We find that the ratio of congestion factor increases as the value of m mounts up. The reason is that when we increase m, the similarity among route candidates of a trip will decrease, which may increase the discrepancy of matching results between DRM and GRM series.

Conclusion

We propose and study the DPS problem. To solve the problem, we first propose a route pricing model to compute the congestion contribution to global urban traffic systems made by a route. Next, based on the route pricing model we develop an efficient and effective approximate swap-based matching algorithm and its corresponding prechecking scheme to help prune unqualified swaps. Experiments on two real-life datasets show that the proposed swap-based algorithm is capable of achieving both high efficiency and high accuracy compared against the exact route matching algorithm.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (61932004, 61922054, 61872235, 61729202, 61832017, U1636210) and the National Key Research and Development Program of China (2018YFC1504504, 2016YFB0700502).

References

Cao, X.; Chen, L.; Cong, G.; and Xiao, X. 2012. Keyword-aware optimal route search. *PVLDB* 5(11):1136–1147.

Chen, Z.; Shen, H. T.; Zhou, X.; Zheng, Y.; and Xie, X. 2010. Searching trajectories by locations: an efficiency study. In *SIGMOD*, 255–266.

Chen, L.; Shang, S.; Jensen, C. S.; Yao, B.; Zhang, Z.; and Shao, L. 2019. Effective and efficient reuse of past travel behavior for route recommendation. In *KDD*, 488–498.

Dai, J.; Yang, B.; Guo, C.; and Ding, Z. 2015. Personalized route recommendation using big trajectory data. In *ICDE*, 543–554.

Ding, B.; Yu, J. X.; and Qin, L. 2008. Finding time-dependent shortest paths over large graphs. In *EDBT*, 205–216.

Ge, Y.; Xiong, H.; Tuzhilin, A.; Xiao, K.; Gruteser, M.; and Pazzani, M. J. 2010. An energy-efficient mobile recommender system. In *KDD*, 899–908.

Hua, M., and Pei, J. 2010. Probabilistic path queries in road networks: traffic uncertainty aware path selection. In *EDBT*, 347–358.

Levin, R., and Kanza, Y. 2014. TARS: traffic-aware route search. *GeoInformatica* 18(3):461–500.

Liu, A.; Wang, W.; Shang, S.; Li, Q.; and Zhang, X. 2018. Efficient task assignment in spatial crowdsourcing with worker and task privacy protection. *GeoInformatica* 22(2):335–362.

Shang, S.; Ding, R.; Yuan, B.; Xie, K.; Zheng, K.; and Kalnis, P. 2012. User oriented trajectory search for trip recommendation. In *EDBT*, 156–167.

Shang, S.; Lu, H.; Pedersen, T. B.; and Xie, X. 2013. Finding traffic-aware fastest paths in spatial networks. In *SSTD*, 128–145.

Shang, S.; Ding, R.; Zheng, K.; Jensen, C. S.; Kalnis, P.; and Zhou, X. 2014a. Personalized trajectory matching in spatial networks. *VLDB J.* 23(3):449–468.

Shang, S.; Guo, D.; Liu, J.; and Liu, K. 2014b. Human mobility prediction and unobstructed route planning in public transport networks. In *MDM*, 43–48.

Shang, S.; Liu, J.; Zheng, K.; Lu, H.; Pedersen, T. B.; and Wen, J. 2015. Planning unobstructed paths in traffic-aware spatial networks. *GeoInformatica* 19(4):723–746.

Shang, S.; Chen, L.; Wei, Z.; Guo, D.; and Wen, J. 2016a. Dynamic shortest path monitoring in spatial networks. *J. Comput. Sci. Technol.* 31(4):637–648.

Shang, S.; Chen, L.; Wei, Z.; Jensen, C. S.; Wen, J.; and Kalnis, P. 2016b. Collective travel planning in spatial networks. *IEEE Trans. Knowl. Data Eng.* 28(5):1132–1146.

Shang, S.; Chen, L.; Jensen, C. S.; Wen, J.; and Kalnis, P. 2017a. Searching trajectories by regions of interest. *IEEE Trans. Knowl. Data Eng.* 29(7):1549–1562.

Shang, S.; Chen, L.; Wei, Z.; Jensen, C. S.; Zheng, K.; and Kalnis, P. 2017b. Trajectory similarity join in spatial networks. *PVLDB* 10(11):1178–1189.

Shang, S.; Chen, L.; Wei, Z.; Jensen, C. S.; Zheng, K.; and Kalnis, P. 2018. Parallel trajectory similarity joins in spatial networks. *VLDB J.* 27(3):395–420.

Shang, S.; Chen, L.; Zheng, K.; Jensen, C. S.; Wei, Z.; and Kalnis, P. 2019. Parallel trajectory-to-location join. *IEEE Trans. Knowl. Data Eng.* 31(6):1194–1207.

Tong, Y.; Wang, L.; Zhou, Z.; Ding, B.; Chen, L.; Ye, J.; and Xu, K. 2017. Flexible online task assignment in real-time spatial data. *PVLDB* 10(11):1334–1345.

Tong, Y.; Wang, L.; Zhou, Z.; Chen, L.; Du, B.; and Ye, J. 2018. Dynamic pricing in spatial crowdsourcing: A matching-based approach. In *SIGMOD*, 773–788.

Xie, K.; Deng, K.; Shang, S.; Zhou, X.; and Zheng, K. 2012. Finding alternative shortest paths in spatial networks. *ACM Trans. Database Syst.* 37(4):29:1–29:31.

Yang, B.; Guo, C.; Jensen, C. S.; Kaul, M.; and Shang, S. 2014. Stochastic skyline route planning under time-varying uncertainty. In *ICDE*, 136–147.

Ye, Z.; Zhang, L.; Xiao, K.; Zhou, W.; Ge, Y.; and Deng, Y. 2018. Multi-user mobile sequential recommendation: An efficient parallel computing paradigm. In *KDD*, 2624–2633.

Ye, Z.; Xiao, K.; Ge, Y.; and Deng, Y. 2019. Applying simulated annealing and parallel computing to the mobile sequential recommendation. *IEEE Trans. Knowl. Data Eng.* 31(2):243–256.

Ye, Z.; Xiao, K.; and Deng, Y. 2018. A unified theory of the mobile sequential recommendation problem. In *ICDM*, 1380–1385.

Yuan, J.; Zheng, Y.; Xie, X.; and Sun, G. 2013. T-drive: Enhancing driving directions with taxi drivers' intelligence. *IEEE Trans. Knowl. Data Eng.* 25(1):220–232.

Zeng, Y.; Chen, X.; Cao, X.; Qin, S.; Cavazza, M.; and Xiang, Y. 2015. Optimal route search with the coverage of users' preferences. In *IJCAI*, 2118–2124.

Zheng, K.; Shang, S.; Yuan, N. J.; and Yang, Y. 2013. Towards efficient search for activity trajectories. In *ICDE*, 230–241.