Adaptive Collective Routing Using Gaussian Process Dynamic Congestion Models

Siyuan Liu, Yisong Yue, Ramayya Krishnan
Carnegie Mellon University
{siyuan,yisongyue,rk2x}@cmu.edu

ABSTRACT

We consider the problem of adaptively routing a fleet of cooperative vehicles within a road network in the presence of uncertain and dynamic congestion conditions. To tackle this problem, we first propose a Gaussian Process Dynamic Congestion Model that can effectively characterize both the dynamics and the uncertainty of congestion conditions. Our model is efficient and thus facilitates real-time adaptive routing in the face of uncertainty. Using this congestion model, we develop an efficient algorithm for non-myopic adaptive routing to minimize the *collective travel time* of all vehicles in the system. A key property of our approach is the ability to efficiently reason about the long-term value of exploration, which enables collectively balancing the exploration/exploitation trade-off for entire fleets of vehicles. We validate our approach based on traffic data from two large Asian cities. We show that our congestion model is effective in modeling dynamic congestion conditions. We also show that our routing algorithm generates significantly faster routes compared to standard baselines, and achieves *near-optimal performance* compared to an omniscient routing algorithm. We also present the results from a preliminary field study, which showcases the efficacy of our approach.

Categories and Subject Descriptors: H.2.8 Database applications: Data mining I.2.6 Artificial Intelligence: Learning - parameter learning

General Terms: Algorithms; Experimentation.

Keywords: Collective routing; Gaussian Process; Dynamic congestion model.

1. INTRODUCTION

We consider the problem of collectively routing a fleet of cooperative vehicles, with the goal of minimizing the total travel time. Such problem settings naturally arise in contexts such as delivery services and cooperative fleets of automated self-driving vehicles.

Intelligently routing vehicles in urban environments is a challenging problem due to uncertainty in the traffic conditions of the road network. Consider the example depicted in Figure 1, where seven cooperative vehicles wish to travel from point A (left side) to point B (right side) using either the top road or the bottom road. Suppose from historical measurements that we know the bottom road is faster in expectation. However, there is a reasonable chance that the top road is currently faster. A good *collective* strategy in this setting would be to send the first two vehicles down separate roads in order to observe current traffic conditions on both roads (i.e., vehicles can also be viewed as sensors). Afterwards, the remaining vehicles can be routed using much more reliable traffic information.[1]

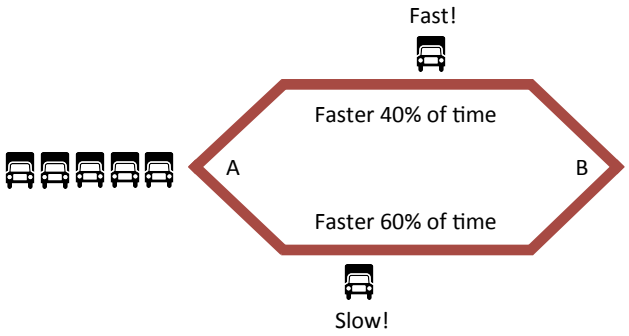

Figure 1: Example collective routing problem with uncertainty. Seven vehicles wish to travel from A (left) to B (right). The bottom road is faster in expectation, but there is a chance that the top road is currently faster. A good collective strategy is to send the first two vehicles down separate roads to observe which road is currently the faster one.

Two technical challenges arise from this example. First, we require a (probabilistic) model that reliably captures the distribution, or uncertainty, of traffic conditions (e.g., the model must be able to predict that the top road in Figure 1 is faster 40% of the time). Furthermore, such a model must be able to (efficiently) predict reliable *posterior*, or updated, distributions given real-time observations.

Second, we require routing algorithms that can balance the exploration/exploitation trade-off underlying all prob-

[1]Note that our approach does not require vehicles to wait while the first two vehicles are exploring, but rather automatically balances the collective exploration/exploitation trade-off for the entire fleet (see Section 5).

KDD'13, August 11–14, 2013, Chicago, Illinois, USA.

lems pertaining decision-making under uncertainty – this issue is typically not explicitly considered in the routing literature. In Figure 1, the first two vehicles are engaged in "exploratory" routing so that later vehicles can be routed more optimally. Exploration is inherently non-myopic since it often reduces the utility (i.e., increases the travel time) of the exploring vehicle. As such, effective routing algorithms must be careful to not over-explore, which requires reasoning about the long-term impact of exploration. Our setting is further complicated due to collectively routing a fleet of cooperative vehicles, rather than a single vehicle in isolation.

In this paper, we make the following contributions:

- We propose a Gaussian Process Dynamic Congestion Model (GPDCM), which can model both the uncertainty and the dynamics of traffic conditions.
- We propose an adaptive collective routing algorithm, called Planning using Canonical Routes (PCR), that can effectively reason about the long-term value of exploration for an entire fleet of cooperative vehicles. The PCR algorithm can efficiently construct long-term plans that are optimized over the full action space of all possible routes. The PCR algorithm accomplishes this task by calculating the long-term value of exploration only over a small set of "canonical routes" – some of which are the true optimal routes (see Section 5.1).
- We validate our approach using real traffic data collected from two large Asian cities. We show that our traffic model effectively captures the dynamics of real-time traffic conditions.
- We also show that our PCR algorithm dramatically out-performs conventional baselines, and achieves near-optimal performance compared to an omniscient routing algorithm with perfect knowledge of traffic conditions. We also present results from a preliminary field study showcasing the efficacy of our approach.

2. RELATED WORK

Traffic data analysis: Traffic modeling is a very diverse research area, which is primarily due to there being a large variety of measurement types (e.g., traffic cameras, GPS traces) as well as modeling goals. Our work is most closely related to the area of congestion estimation.

Congestion or traffic estimation has been studied using a variety of mathemetical tools, ranging from flow patterns [20], to per-individual statistical models [18], to Markov chain forecasting [29]. The two main types of measurement data are GPS or low-bandwidth cellular updates which are associated with individual vehicles [32, 7], and static traffic cameras which are associated with known location [2].

In contrast to most prior work, our goal is to derive a holistic generative probabilistic model of dynamic traffic conditions. Since our goal is to incorporate such a traffic model into a real-time routing algorithm, we require our model to not only accurately capture the distribution of future traffic scenarios (given real-time observations), but also be sufficiently fast to evaluate. For measurement, we use GPS traces collected from thousands of taxis from two large Asian cities (see Section 6).

Other work on traffic modeling have studied phenomenon such as traffic accidents or other outlier activitiy [3, 22] and evaluating the overall health of a road network design [34].

Adaptive Routing: Although there have been some work on personalized or adaptive routing based on historical and real-time traffic conditions, prior work did not model the benefits of real-time exploration, and thus must resort to myopic routing [33, 8].

The routing problem we study is an instance of the general problem of decision-making under uncertainty. Such problems are typically cast as partially observable Markov decision processes (POMDPs) [14], where the world (i.e., road congestion conditions) is assumed to behave according to Markovian dynamics, and actions (i.e., routing) reveal partial observations regarding the state of the world (i.e., local congestion measurements).

We cast our problem as a POMDP with a continuous state space modeled using a Gaussian Process (GP). In contrast to previous work on continuous POMDPs (e.g., [10]), we are focused on solving problems with large structured action spaces (i.e., all possible routings of the cooperative vehicles).

There are three common types of approaches for solving discrete POMDPs with large action spaces. The first is to simplify the problem using canonical or macro actions [11]. After identifying the best plan consisting of macro actions, the planner can optionally refine at finer granularities. The second type is to always plan at finer granularities and use pruning approaches to avoid enumerating the full exponentially large decision tree (e.g., Monte-Carlo tree search [28]).

The third, more generic, type of approach is to sample from a generative distribution of the POMDP and then find a good plan that optimizes for this empirical distribution [12, 4, 15, 25].[2] The most general approaches are called stochastic planning with recourse [12, 4, 26], which is a very general framework[3] where the goal is to compute a good initial plan while also allowing adaptive behavior, or recourse, during operation. Stochastic planning with recourse approaches are often focused on planning for relatively short time horizons due to efficiency reasons. Conversely, approaches such as PEGASUS [25] plan for longer time horizons but typically optimize over restricted classes of policies.

Our routing algorithm can be viewed as a stochastic planning approach (over a continuous state space) that utilizes canonical actions. In contrast to previous work, our approach optimizes both for the *long-term plan* and over the *full action space of all possible routes*. To control the complexity of long-term planning, we rely on the observation that typically only a few routes are optimal for any vehicle – we call these the canonical routes (see Section 5.1). Our approach efficiently trades off between exploration and exploitation by choosing routes that optimize the combination of expected travel time (exploitation) and uncertainty reduction of the canonical routes (exploration to determine which canonical route is the best) – this bears affinity to algorithms for Gaussian Process bandit optimization [30, 17].

From a combinatorial routing perspective, our work is focused on a relatively simple setting where each vehicle has a pre-defined destination. More complicated settings typically assume that each vehicle is "exchangeable" and can be routed to any of the target destinations (cf. [9]). This leads to a challenging combinatorial optimization problem

[2]It can be shown that a finite number of samples is sufficient for any planning problem (cf. [15, 25]).

[3]It can be shown that all MDP planning problems can be instantiated as a (large) stochastic planning with recourse problem (cf. Chapter 4 of [13]).

even without considering uncertainty. We instead focus on the complementary problem of how best to collectively route each vehicle to their pre-assigned destination while accounting for uncertain traffic conditions.

Another line of related work is motivated from the network packet routing problem [1]. A key difference is that [1] is focused on the repeated games problem (i.e., gradually learning about network conditions as more packets are routed) for a single routing problem at a time, whereas we seek to solve a single-shot multi-agent routing problem.

3. GAUSSIAN PROCESS PRELIMINARIES

Let $\mathcal{R}$ denote our road nework, and $\mathcal{Z}$ denote the space of contexts (e.g., containing information about time of day or day of week). We wish to model the travel speed of road segments $r \in \mathcal{R}$ under varying contexts $z \in \mathcal{Z}$. We do so via a function $f : \mathcal{R} \times \mathcal{Z} \to \Re_+$, that outputs the travel speed for a given (r, z) pair.

We assume that f is sampled probabilistically from a Gaussian process prior distribution $f \sim P(f)$ [27]. A Gaussian process prior is fully specified by its mean function

$$\mu(r, z) = \mathbf{E}[f(r, z)]$$

and its covariance, or kernel, function

$$\begin{aligned} k((r, z), (r', z')) &= \mathbf{E}[(f(r, z) - \mu(r, z))(f(r', z') - \mu(r', z'))] \\ &= \texttt{cov}((r, z), (r', z')). \end{aligned}$$

A major computational benefit of Gaussian processes is that posterior inference can be computed in closed form. Suppose we have collected observations $Y = [y_1, \ldots, y_T]^\top$ at $X = [(r_1, z_1), \ldots, (r_T, z_T)]$, where each $y_i \sim N(f(r_i, z_i), \sigma^2)$ is corrupted by i.i.d. Gaussian noise. We also define the deviation of Y from its prior mean as

$$\delta Y = [y_1 - \mu(r_1, z_1), \ldots, y_T - \mu(r_T, z_T)]^\top.$$

Then we can write the posterior distribution given X and Y also as a Guassian process distribution with mean

$$\mu_{Y,X}(r, z) = \mu(r, z) + \hat{k}_{Y,X}(r, z)^\top (\hat{K}_{Y,X} + \sigma^2 I)^{-1} (\delta Y)^\top \quad (1)$$

and covariance $k_{Y,X}((r, z), (r', z')) =$

$$k((r, z), (r', z')) - \hat{k}_{Y,X}(r, z)^\top (\hat{K}_{Y,X} + \sigma^2 I)^{-1} \hat{k}_{Y,X}(r', z'), \quad (2)$$

where

$$\hat{k}_{Y,X}(r, z) = [k((r_1, z_1), (r, z)), \ldots, k((r_T, z_T), (r, z))]^\top \in \Re^T$$

is a column vector of the kernel values between (r,z) and every observed location in X, and

$$\hat{K}_{Y,X} = [k((r_i, z_i), (r_j, z_j))]_{i,j \in [1,\ldots,T]} \in \Re^{T \times T}$$

is the kernel Gram matrix of all observed locations in X. Note that the posterior variance of $f(r, z)$ is $k_{Y,X}((r, z), (r, z))$.

Essentially, (1) states that the posterior mean $\mu(r, z|Y, X)$ of $f(r, z)$ deviates from its prior mean $\mu(r, z)$ according to how much (and in what direction) the past observations deviated from their prior means. If some $(r', z') \in X$ had high positive covariance with (r, z), then $\mu(r, z|Y, X)$ would shift along the deviation of the associated observation y' from its mean $y' - \mu(r', z')$. Conversely, if some $(r', z') \in X$ that had little covariance with (r, z), then the associated observation y' would have little impact on $\mu(r, z|Y, X)$.

Similarly, (2) states that the posterior covariance between (r, z) and (r', z') decreases as we gather more observation that is related to (i.e., has high prior covariance with) (r, z) and (r', z'). Thus, the variance of our posterior estimate, $k_{Y,X}((r, z), (r, z))$, of $f(r, z)$ decreases faster when our past observations have higher prior covariance with (r, z).

4. GAUSSIAN PROCESS DYNAMIC CONGESTION MODELS

For a given road segment $r \in \mathcal{R}$ and context $z \in \mathcal{Z}$, we wish to model the background distribution of the traveling speed $y_{r,z}$. Furthermore, we wish to model the appropriate temporal and spatial dependences such that, after observing recent traveling speeds, we can make more confident inferences of the traveling speeds of nearby roads in the near future. We now present the Gaussian Process Dynamic Congestion Model (GPDCM), which captures such properties within the Gaussian Process framework. We assume for simplicity that the observation noise variance σ^2 is known.

We essentially assume that road traffic conditions behave according to Gaussian Process dynamics. In other words, given the current observations, the posterior GP distribution should accurately represent the distribution of future traffic conditions. It remains to define a suitable choice of μ and k. We do so by estimating μ and k using historical traffic data S, which we describe in the following.

4.1 Estimating μ

Estimating μ requires assuming regularities about how μ behaves relative to z. For example, suppose we assume that temporal regularities can be characterized by time-of-day, which we denote as $\tau(z)$. Then we can estimate $\mu(r, z)$ as

$$\mu(r, z) = \texttt{mean}\left\{y_{r,z'} : (y_{r,z'} \in S) \wedge \left(\tau(z') = \tau(z)\right)\right\}, \quad (3)$$

where $y_{r,z'}$ is a historically observed traveling speed on road r and time-of-day $\tau(z') = \tau(z)$. This allows computing the mean traveling speed on road r at any future time, but restricts us to modeling all days as being sampled from the same (prior) distribution.[4]

4.1.1 Temporal Smoothing

Since it is unlikely that historical observations contain many data points with exactly matching time-of-day contexts, we use standard temporal smoothing techniques to estimate a smoothed version of (3) [21].

We first partition time into intervals (which correspond to each time step in the routing problem definition). Let $\tau(t)$ now denote the time-of-day interval corresponding to time step t (e.g., 5:10pm - 5:20pm). For road segment r and time step t, we define $Y_S^{(t)}(r)$ as the set of all the reported speeds matching $\tau(t)$ at r in the historical data S, i.e.,

$$Y_S^{(t)}(r) = \{y | ((r, z), y) \in S \wedge \tau(z) = \tau(t)\}.$$

We can now use $Y_S^{(t)}$ to construct an empirical distribution of travel times on road r at time step t.

Following [21], we employ temporal smoothing of the empirical *cumulative distribution functions* (CDF). We can write this smoothed CDF at road segment r as

$$\mathrm{H}_S^{(t)}(y|r) = \beta \cdot \mathrm{H}_S^{(t-\gamma)}(y|r) + (1 - \beta)(1 - \mathrm{CDF}(y|Y_S^{(t)}),$$

[4]We can alternatively use a feature representation of the context $\phi(z)$ and a parametric model to estimate $\mu(r, z)$.

where $\mathrm{CDF}(y|Y_S^{(t)})$ denotes the empirical CDF w.r.t. $Y_S^{(t)}$, and β and γ control for the degree of smoothing. We refer to [21] for more details. Afterwards, we can compute the empirical mean (3) using $H_S^z(y|r)$ via $\mu(r,z) = \mathbf{E}_{y\sim H}[y]$.

4.2 Estimating k

For simplicity, we assume the covariance kernel can be decomposed by road and context, i.e.,

$$k((r,z),(r',z')) = k_1(r,r')k_2(z,z'). \tag{4}$$

This decomposition $k = k_1k_2$ assumes that all road segments share the same temporal covariance k_2; while somewhat idealistic, this offers a compact and efficient representation.

4.2.1 Estimating Road Covariance k_1

Estimating $k_1(r,r')$ is relatively straightforward, as it is essentially the covariance of the distribution of travel speeds between two road segments. We can estimate k_1 from historical data S as

$$k_1(r,r') = \texttt{mean}\left\{(y_{r,z} - \mu(r,z))(y_{r',z} - \mu(r',z)) : y_{r,z}, y_{r',z} \in S\right\}. \tag{5}$$

Note that, for (5), the two historical observations $y_{r,z}$ and $y_{r',z}$ share the exact same context (e.g., were observed at the exact same time). As such, we also employed temporal smoothing (see Section 4.1.1) to estimate k_1 more reliably.

Intuitively, if two roads r and r' have high covariance according to S, then $k(r,r')$ will be high. As such, observations of r' provide more information about the conditions on r than another road segment with low covariance with r'.

Note that since the roads in our system form a finite set, we can precompute the entire road-road kernel matrix.

4.2.2 Estimating Temporal Covariance k_2

Estimating $k_2(z,z')$ is slightly more complicated, as it requires us to assume additional regularities about z. For example, we can assume that $k_2(z,z')$ only depends on the difference in time $|z-z'|$, which leads to

$$k_2(z,z') = \texttt{mean}\left\{(y_{r,z_1} - \mu(r,z_1))(y_{r,z_2} - \mu(r,z_2)) : (y_{r,z_1}, y_{r,z_2} \in S) \wedge \left(|z_1 - z_2| = |z - z'|\right)\right\}. \tag{6}$$

Here we compute the empirical covariance of traveling speeds at time distance $|z-z'|$ averaged over every road. As when estimating k_1, we also employ temporal smoothing to estimate k_2 more reliably.

Intuitively, the covariance should decrease as the time difference increases. From the perspective of each y_{r,z_1}, the distribution of the corresponding y_{r,z_2} should look more like the background distribution as $|z_1 - z_2|$ increases.

4.2.3 Other Considerations

Although (5) and (6) correspond to (modulo regularity assumptions) the "correct" covariance estimates, their non-parametric form can be expensive to evaluate. Naively, (5) and (6) requires reprocessing S for each kernel evaluation. However, one can specify k_1 and k_2 using other forms as well. For example, one could use a RBF kernel: $k_1(r,r') = \exp\left\{-\|r-r'\|^2/\gamma_1\right\}$, where $\|r-r'\|$ denotes the distance between r and r', and γ_1 is a tunable parameter used to fit the historical data (i.e., to be similar to (5)). One can also define an analogous kernel model for k_2. This definition has an advantage of being more compact and thus more efficient than (5) and (6). We defer a more extensive study on the choice of kernels to future work.

4.3 Posterior Inference

The main purpose of using the GPDCM is to be able to make tractable probabilistic inferences of future road conditions given real-time observations. After estimating μ and k from historical data, the posterior distribution is fully specified via (1) and (2).

Note that the $K_{Y,X}$ matrix in (2) grows quadratically in size w.r.t. the number of observations. This can make inference very costly as we collect more observations. Intuitively, we should expect that very old observations do not impact our model much when trying to predict future conditions (since the time difference is large, the k_2 component from very old observations should be close to 0). As such, we should be able to discard old observations from our model without compromising model accuracy.

5. ADAPTIVE COLLECTIVE ROUTING

We consider the setting where N cooperative vehicles must travel from source locations $A = \{a_1, \ldots, a_N\}$ to destinations $B = \{b_1, \ldots, b_N\}$. We desire an adaptive routing policy that can minimize the total travel time of all vehicles.

Let `Planner` denote the routing algorithm being employed. For simplicity, we discretize time into intervals (e.g., every minute). At time step t the following occurs:

- Receive observations $Y_t = \{y_1, \ldots, y_{t-1}\}$
- Receive state of vehicles $L_t = \{\ell_1, \ldots, \ell_N\}$
- Compute routing $\Psi_t \leftarrow \texttt{Planner}(L_t, Y_t)$
- Execute routing Ψ_t until next time step
- $t \leftarrow t+1$

In our setting, each observation $y_t = \{y_{t,1}, \ldots, y_{t,N}\}$ corresponds to the observed speed of the roads traveled by each vehicle for that time step t. In this section, we assume that congestion conditions behave according to a GPDCM.[5]

We develop our algorithm, called Planning using Canonical Routes (PCR), in three steps. First, we observe that typically only a few routes are ever optimal for each vehicle – we call these the *canonical routes*. Second, we show how to utilize canonical routes to efficiently model the long-term value of exploration. Finally, we show how to compute collective routing plans that balance the exploration/exploitation tradeoff for an entire fleet of cooperative vehicles. The PCR algorithm is described in Algorithm 1.

5.1 Canonical Routes

At first glance, computing (near-)optimal routing plans may seem hopelessly intractable, since the full decision space for even a single vehicle can be exponentially large. However, it can often be the case that only a few routes are ever optimal for a given vehicle's routing problem. For instance, given a GPDCM of congestion conditions, one can sample probable future traffic scenarios. Each such sample gives *complete information* of one probable future scenario (cf. [15, 25]), which allows us to compute the optimal route under that scenario. We call such optimal routes the *canonical*

[5]Since we discretize time into intervals, we use the average travel speed of each road segment for routing.

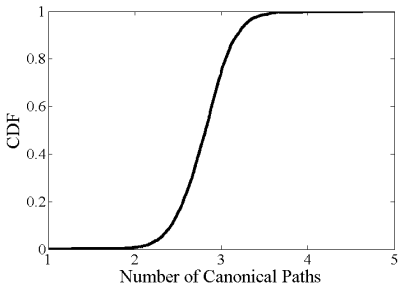

Figure 2: Distribution of the number of canonical routes for each individual vehicle routing problem (see Section 5.1). On average, there are 3.2 canonical routes per vehicle.

routes. Depending on the specific conditions, the optimal canonical route could be much faster than simply routing according to expected travel time.

Figure 2 shows the distribution of the number of canonical routes per vehicle.[6] We see that the number of canonical routes is often quite small – about 3.2 on average.

If a routing algorithm *knew exactly* which of the sampled scenarios reflects future traffic conditions, then that corresponding canonical route would be the optimal action to take. But due to uncertainty regarding which scenario actually reflects future conditions, then it may be beneficial to perform some "exploratory routing" in order to more confidently identify which canonical route is actually best.

5.2 Value of Exploration

The standard approach for quantifying the value of exploration is to measure the expected utility gain of future routing decisions [23]. Given the exponentially many possible future routing decisions, computing this value of exploration exactly is intractable using naive approaches.

We use two approximations in order to efficiently quantify the long-term value of exploration. First, we focus exploration only on the canonical routes, rather than all possible routes. This approximation is motivated by the assumption that each vehicle should be routed (as closely as possible) according to the optimal canonical route. Therefore, the goal of exploration should be to determine which canonical routes are actually optimal. In addition, focusing on canonical routes is a form of long-term exploration, since canonical routes are complete paths to the vehicles' destinations.

We next approximate the value of exploration using variance reduction.[7] For any canonical route $\vec{r}^*$, the variance reduction of $\vec{r}^*$ from running some other route $\vec{r}$ is

$$\delta k(\vec{r}, \vec{r}^* | X, Y) = k(\vec{r}^*, \vec{r}^* | X, Y) - k(\vec{r}^*, \vec{r}^* | X, Y, \vec{r}), \qquad (7)$$

where $k(\vec{r}^*, \vec{r}^* | X, Y)$ denotes the variance reduction of $\vec{r}^*$ under the posterior distribution induced by observations X and Y, and $k(\vec{r}^*, \vec{r}^* | X, Y, \vec{r})$ denotes the variance of $\vec{r}^*$ under posterior distribution induced by observations X, Y and $\vec{r}$.

Estimating (7) exactly is expensive, since the variance of each road segment in $\vec{r}^*$ depends on the exact arrival times at that segment (which depends on previous segments). We instead approximate $k(\vec{r}^*, \vec{r}^* | X, Y)$ using the expected travel times for each segment in $\vec{r}^*$. We first write $\vec{r}^*$ as

$$\vec{r}^* = \langle (r_1^*, \tau_1^*), \ldots, (r_{|\vec{r}^*|}^*, \tau_{|\vec{r}^*|}^*) \rangle,$$

where τ_i denotes the expected arrival time at road segment r_i^*. We can then approximate $k(\vec{r}^*, \vec{r}^* | X, Y)$ as

$$k(\vec{r}^*, \vec{r}^* | X, Y) \approx \sum_{i=1}^{|\vec{r}^*|} k((r_i^*, \tau_i^*), (r_i^*, \tau_i^*) | X, Y, \vec{r}_{[1:i-1]}^*),$$

where $\vec{r}_{[1:i-1]}^*$ denotes the first $i-1$ road segments of $\vec{r}^*$. We can similarly approximate $k(\vec{r}^*, \vec{r}^* | X, Y, \vec{r})$ as

$$\sum_{i=1}^{|\vec{r}^*|} k((r_i^*, \tau_i^*), (r_i^*, \tau_i^*) | X, Y, \vec{r}_{[1:i-1]}^*, \vec{r}_{\prec(r_i^*, \tau_i^*)}),$$

where $\vec{r}_{\prec(r_i^*, \tau_i^*)}$ denotes the segments of $\vec{r}$ with expected arrival before τ_i^*.

In summary, we quantify the long-term value of exploration of any route by how much that route reduces the variance, or uncertainty, of the travel times of the canonical routes. If we were routing a vehicle for purely exploratory purposes, then we would choose a route $\vec{r}$ that maximizes (7) summed over every canonical route $\vec{r}^*$ in our set.

Algorithm 1 Planning using Canonical Routes (PCR)

1: **input**: L, Y, α
2: Define posterior mean $\forall (r, z):\ \mu(r, z | Y)$
3: Define posterior cov. $\forall (r, z), (r', z'):\ k((r, z), (r', z') | Y)$
4: Sample K traffic scenarios $\{\hat{f}_k\}_{k=1}^K$ from posterior using $\mu(r, z | Y)$ and $k((r, z), (r', z') | Y)$.
5: For each vehicle n, compute all optimal routes $\{\vec{r}_{n,i}^*\}_{i=1}^K$ from the K traffic scenarios.
6: Initialize $W \leftarrow \{1, \ldots, N\}$
7: Initialize $\Psi \leftarrow \emptyset$
8: **for** $j = 1, \ldots, N$ **do**
9: For each vehicle $n \in W$, compute route $\vec{r}_n$ that minimizes $C(\vec{r}_n) = \texttt{travel-time}(\vec{r}_n) - (\alpha/K) \sum_{n',i} \delta k(\vec{r}_n, \vec{r}_{n',i}^* | Y, \Psi)$ *//see* (7)
10: $\hat{n} \leftarrow \text{argmin}_n\, C(\vec{r}_n)$
11: $W \leftarrow W \setminus \{\hat{n}\}$
12: $\Psi \leftarrow \Psi \cup \{\vec{r}_{\hat{n}}\}$
13: **end for**
14: **Return** Ψ

5.3 Balancing Exploration vs Exploitation

Our algorithm, called Planning using Canonical Routes (PCR), is described in Algorithm 1. At each time step t, the algorithm first computes a set of canonical routes by sampling from the posterior GPDCM distribution (Lines 4-5). The algorithm then computes the lowest cost (or highest utility) route $\vec{r}_n$ for each vehicle n (Line 9).[8] Cost here is measured as a weighted combination of expected travel time of $\vec{r}_n$ (under the posterior distribution) and negative average variance reduction (averaged over all canonical routes for all vehicles), with the parameter α controlling for the trade-off.

Intuitively, a route with a high travel time might still have the lowest cost if it greatly reduces the uncertainty of many canonical routes for many vehicles. Likewise, a route that does not offer much in terms of variance reduction may still have the lowest cost if it has (by far) the shortest travel time.

[6] We computed Figure 2 by sampling from a GPDCM estimated using traffic real data (see Section 6).

[7] Variance reduction exactly quantifies the value of information in some settings (cf. [30, 17]).

[8] This can be done using the standard Dijkstra's shortest path algorithm using our cost function.

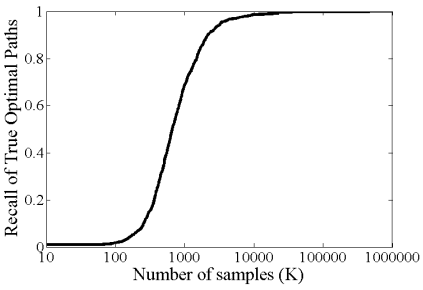

Figure 3: Recall (or coverage) of the optimal paths with respect to the number of samples K.

After computing the lowest cost route for each vehicle, the algorithm greedily commits to the route with the overall lowest cost (Lines 10-12). This process is repeated until all vehicles have been committed to a route.

5.3.1 Connection to Submodular Optimization

The reason why we need to recompute the lowest cost routes each iteration (Lines 8-13) is because the variance reduction of canonical routes change after the algorithm has commited to a route. For instance, if the two best routes from the current iteration both provide uncertainty reduction to the same canonical routes, then after the algorithm commits to one of those routes, the value of exploration for the other route has decreased dramatically (i.e., it no longer provides the same variance reduction). This notion of diminishing returns is often modeled using submodularity [24].

Algorithm 1 is implicitly optimizing for a utility function (the negative of the cost function) that we empirically observe is close to monotone submodular in our application. As such, our routing problem is qualitatively similar to a submodular optimization problem under matroid constraints, and we should expect that the greedy approach employed by Algorithm 1 achieves near-optimal solutions (for this utility function) [6, 31].

One nice property of submodular function optimization is that one can employ a lazy variant of the greedy approach to significantly speed up computation [19]. It is straightfoward to incorporate this lazy variant into Algorithm 1.

5.3.2 Other Design Decisions

Variance reduction is often an over-estimate of the true uncertainty reduction. As such, it is often beneficial to choose α to be small. In our experiments, we use $\alpha = 0.1$.[9]

Another design decision is the number of samples K (Line 4 in Algorithm 1). We would like to choose K sufficiently large so that all true canonical paths (those that are actually optimal) are covered. Figure 3 shows the recall (or coverage) of true canonical paths collectively for 50 vehicles. In our experiments, we choose $K = 5000$.[10]

6. EMPIRICAL EVALUATION

We conduct two types of empirical evaluation. First, we verify whether our Gaussian Process Dynamic Congestion Model can accurately predict the distribution of future traffic scenarios given real-time observations. Second, we verify how our PCR algorithm compares versus conventional routing baselines and also versus omniscient routing with perfect knowledge of congestion conditions.[11]

We obtained two datasets consisting of GPS traces from a large number vehicles:

- **City 1**: We obtained one year (from January 2008 to December 2008) of GPS data from approximately 15,000 taxis in a large Asian city.
- **City 2**: We obtained one year (from January 2006 to December 2006) of GPS data from approximately 5,600 taxis in a second large Asian city.

6.1 Gaussian Process Validation

We first validate the effectiveness of our Gaussian Process Dynamic Congestion Model (GPDCM), and verify that it can accurately predict the distribution of future traffic scenarios given real-time observations.

6.1.1 Kolmogorov-Smirnov Test

Since our goal is to evaluate the how well our GPDCM predicts *a distribution* of the future, we require a method a measure of how well a predicted distribution matches the true or empirical distribution observed in nature (i.e., the observed distribution of travel speeds). For our evaluations we use the Kolmogorov-Smirnov test [16], which has a long tradition in the statistics community.[12]

The Kolmogorov-Smirnov score quantifies the distance between any two distributions. In our case, we wish to quantify the distance between the empirical distribution of travel speeds and the predicted distribution of our GPDCM.

Let F_n denote the empirical cumulative distribution function for n i.i.d. random variables X_i:

$$F_x(x) = \frac{1}{n}\sum_{i=1}^{n} I_{X_i \leq x} \tag{8}$$

where $I_{X_i \leq x}$ is an indicator function (i.e., 1 if $X_i \leq x$, and 0 otherwise). The Kolmogorov-Smirnov score for a given cumulative distribution function $F(x)$ and empirical distribution $F_n(x)$ is

$$Z_n = \sup_x \left| F_n(x) - F(x) \right|, \tag{9}$$

which corresponds to the supremum over all absolute distances between the two distributions. The lower the KS score, the better the distributional fit. Two identical distributions will have a KS score of 0, and typically a KS score of less than 0.1 is indicative of a strong distributional fit.

6.1.2 Results

For each city, we fit a GPDCM (μ and k) using six months of data (from February to June). We then evaluated the predictive accuracy of our GPDCM (via the KS score) on held-out data. For each day, we provide the GPDCM with two hours of observations from 8am-10am. We then measure the KS score of the GPDCM posterior compared to future

[9] One can also tune α using a validation set.

[10] We note that since sampling is efficient (due to the GPDCM allowing sampling in closed form) and routing is efficient (due to it being a modified Dijkstra's algorithm), choosing $K = 5000$ still allows Algorithm 1 to be run efficiently.

[11] All of our experiments were conducted using a workstation with four Intel Core Quad CPUs (Q9550 2.83 GHz) and 32 GB of main memory.

[12] Other options include the Berk-Jones test, the score test and their integrated versions [16, 5].

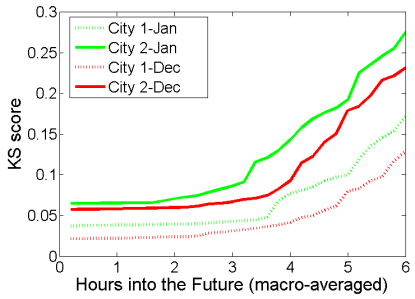

Figure 4: Showing the macro-averaged KS score (computing KS score for each day separately) on the two datasets for months of January and December.

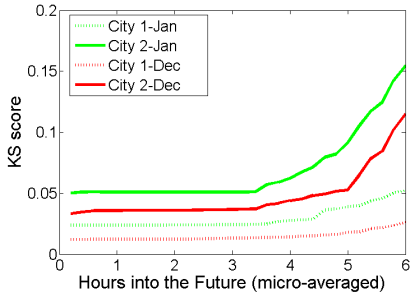

Figure 5: Showing the micro-averaged KS score (by aggregating all days into a single day) for the two datasets for the months of January and December.

empirical distributions partitioned into one hour windows (e.g., [0-1], [1-2], etc., hours into the future).

Figure 4 shows the KS score averaged over multiple days (i.e., macro-averaged) for both datasets for the months of January and December (which are the furthest apart months in both datasets). We observe that the KS score is quite low for the first three hours, indicating that our GPDCM predicts accurate near-term traffic distributions. We also observe that the KS score degrades as the GPDCM predicts traffic distributions further into future, and eventually decays to the "background" prior KS score.

The performance difference for different cities and months in Figure 4 can be explained as follows. Since we have more data from City 1, we should expect a better model fit. Likewise, our datasets also contain more data from December, which explains why our model makes more accurate (from a distributional point of view) predictions for the month of December. As such, one can interpret this difference in performance between December and January as being primarily attributed to data sparsity in the test set.

Figure 5 shows the KS score computed over all days (i.e., micro-averaged) for both datasets for the months of January and December. Because we construct an empirical distribution by aggregating over all days (e.g., all [10am-11am] time windows for each day in January), we should expect the KS score here to be lower than the results in Figure 4 due to less data sparsity. As we can see, the KS score is indeed uniformly lower for both datasets and both months.

These results suggest that the GPDCM is an effective model for predicting the distribution of future traffic conditions given real-time observations. In the following, we evaluate how our PCR algorithm can utilize a GPDCM to effectively plan for non-myopic routing under uncertainty.

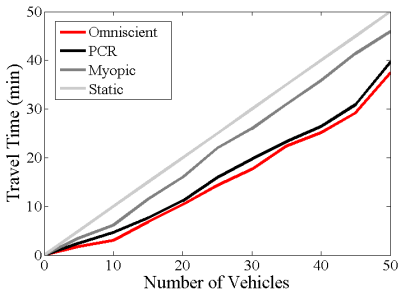

Figure 6: Travel time versus number vehicles to be routed. The PCR algorithm performs nearly as well as the omniscient lower bound.

6.2 Routing Validation

We now validate the effectiveness of our Planning using Canonical Paths (PCR) algorithm for adaptive collective routing under uncertainty. We compare our approach against both conventional baselines (described below) as well an omniscient lower bound:

- **Static Routing**: Each vehicle is routed (via Dijkstra's algorithm) according to the GPDCM prior mean. The routing plan is not adaptively updated as vehicles collect real-time observations.
- **Myopic Routing**: Each vehicle is routed (via Dijkstra's algorithm) according to the GPDCM posterior mean. The routing plan changes adaptively in response to real-time observations, but the value of exploration is not considered (hence myopic).[13]
- **Omniscient Routing**: Each vehicle is routed (via Dijkstra's algorithm) according to **perfect information** about future traffic conditions. Omniscient routing represents a lower bound on the performance of any routing algorithm in this setting.[14]

All routing algorithms utilize the same GPDCM for modeling travel times.

We run all routing algorithms via simulation using congestion conditions mined from the same day in our historical data [21],[15] and assign each vehicle a source and destination pair (A, B).

6.2.1 Results

Figure 6 compares the total travel time with respect to the number of vehicles to be routed. We observe that the PCR algorithm significantly outperforms both Static and Myopic routing, and nearly matches the performance of Omniscient routing. Figure 7 shows a comparison of total travel time with respect to the routing distance. We again observe that the PCR algorithm is near-optimal relative to Omniscient routing. These results suggest that the PCR algorithm is effective at balancing the exploration/exploitation trade-off in order to quickly identify the best possible routing paths for the majority of the vehicles in the system.

[13]Myopic routing is essentially how conventional adaptive routing algorithms behave (cf. [33]).

[14]Omniscient routing is only computable during simulation.

[15]We compute for each road the "crowdedness" metric in [21]. The crowdedness of a road is essentially the complement value of the CDF on that road's speed distribution (i.e., crowdedness of 0.9 means the road is at its 10th percentile traveling speed).

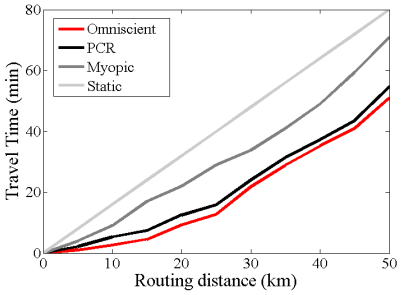

Figure 7: Travel time versus routing distance. The PCR algorithm performs nearly as well as the omniscient lower bound.

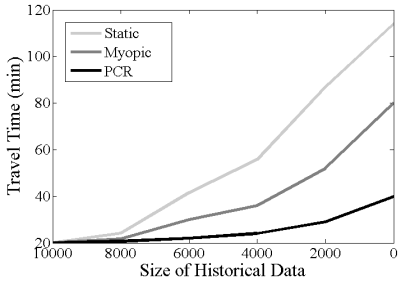

Figure 8: Travel time (min) versus size of historical data used to estimated μ and k.

6.2.2 *Efficiency Study*

Although the PCR algorithm described in Algorithm 1 is already efficient due to relying on efficient subroutines such as Gaussian Process posterior inference and the Dijkstra's shortest path algorithm, one can further trade off routing performance for computational or storage efficiency (the two notions of efficiency are often inter-related).

Our use of non-parametric mean μ and kernel k functions leads to more computationally expensive posterior inference as the training data S grows. Figure 8 shows how the total travel time degrades (for 20 vehicles) as the amount of historical data S (used to estimate μ and k) decreases. We see that the PCR algorithm is robust to using less reliable GPDCMs, which can lead to significant computational and storage savings for relatively small sacrifices in performance.

Another way to trade off performance versus efficiency is by varying the number of posterior samples K. Figure 9 shows how the total travel time degrades as fewer samples are used. We observe that the PCR algorithm can tolerate a small degree of undersampling of traffic scenarios.

6.2.3 *Preliminary Field Study*

We conducted a preliminary field study using twelve vehicles in City 1. Each routing trial comprises two groups of six cooperative vehicles, with each vehicle being assigned identical source and destination locations. The first group employs a version of Myopic routing, where the drivers are instructed to route according to their best knowledge and can communicate with each other. The second group employs a restricted version of the PCR algorithm, where three drivers are routed to explore three canonical routes, and the other three are routed according to updated knowledge.

Table 1 shows the results from five trials. We observe

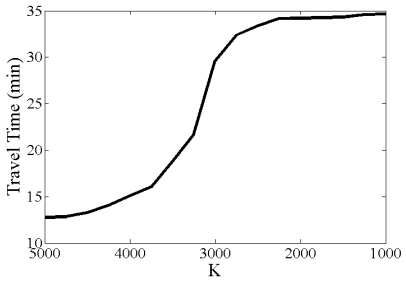

Figure 9: Travel time versus K.

Table 1: Travel Times (minutes) from Field Study

Trial	Myopic	PCR	Improvement
1	28.0	22.3	25.7%
2	36.2	27.6	31.2%
3	12.7	11.2	13.4%
4	51.0	25.7	42.9%
5	32.5	24.9	30.6%

that PCR routing consistently outperforms Myopic routing, which lends evidence that the PCR approach can be more broadly deployed to improve routing performance for larger fleets of cooperative vehicles. The improvement in Trial 3 is small, which is due to low congestion during that time.

7. DISCUSSION

Our approach is related to other approaches that first sample from a POMDP distribution and then optimize for the resulting empirical distribution [15, 25]. One important theoretical question is what number of samples K guarantees good performance (at least within one iteration).

Another theoretical question is whether one can prove guarantees for the entire execution of the PCR algorithm. Such analysis may rely on using variance reduction to bound the performance gap between omniscient routing, which is is related to analysis for Gaussian Process bandit optimization [30, 17]. Two important differences between [30, 17] and our setting are that (1) variance reduction decays over time (i.e., past observations have low value) and (2) we must optimize over a structured action space (i.e., all possible routings), which significantly complicates the analysis.

From a combinatorial routing perspective, our work is focused on a relatively simple setting where each vehicle has a pre-defined destination. It would be interesting to investigate whether our PCR algorithm can be extended to accommodate more complicated settings, such as having each vehicle visiting a set of destinations, or assuming "exchangeability" so that each vehicle can be routed to any of the destinations (cf. [9]), or assuming uncertainty in when and where destination goals arise (such as in a taxi service).

From a game-theoretic perspective, it would be interesting to consider cases where vehicles were not assumed to cooperative, but rather could be incentived to cooperate. One possibility is to properly price the value of exploration.

8. CONCLUSION

We proposed a Gaussian Process Dynamic Congestion Model to capture both the dynamics and the uncertainty of congestion conditions, and the Planning using Canoni-

cal Routes algorithm to balance the exploration/exploitation trade-off for entire fleets of vehicles. We conducted extensive evaluations using GPS traces collected from two large Asian cities. Our results show that our GPDCM an effectively predict the distribution of future congestion conditions, and that our PCR algorithm can achieve near-optimal performance relative to omniscient routing. We also conducted a preliminary field study, where we found our approach to signficantly outperform conventional myopic routing.

Acknowledgements. This research was supported by the T-SET University Transportation Center sponsored by US DoT Grant No. DTRT12-G-UTC11 and the Singapore National Research Foundation under its International Research Centre @ Singapore Funding Initiative and administered by the IDM Programme Office. Yisong Yue was also supported in part by ONR (PECASE) N000141010672 and ONR Young Investigator Program N00014-08-1-0752. The authors also thank Emma Brunskill, Geoff Gordon, Sue Ann Hong, Lavanya Marla, and Lionel Ni for valuable discussions and support regarding this work.

9. REFERENCES

[1] B. Awerbuch and R. Kleinberg. Adaptive routing with end-to-end feedback: Distributed learning and geometric approaches. In *Proc. of STOC*, 2004.

[2] J. Bacon, A. I. Bejan, A. R. Beresford, D. Evans, R. J. Gibbens, and K. Moody. Dependable and historic computing. chapter Using real-time road traffic data to evaluate congestion. 2011.

[3] C. Benjamin. Detecting the onset of congestion rapidly with existing traffic detectors. *Transportation Research*, 2003.

[4] J. Birge and F. Louveaux. *Introduction to stochastic programming*. Springer Verlag, 1997.

[5] M. C. Bryson. Heavy-tailed distributions: Properties and tests. *Technometrics*, 16(1), 1974.

[6] G. Calinescu, C. Chekuri, M. Pál, and J. Vondrák. Maximizing a submodular set function subject to a matroid constraint. *Integer programming and combinatorial optimization*, pages 182–196, 2007.

[7] P. S. Castro, D. Zhang, and S. Li. Urban traffic modelling and prediction using large scale taxi gps traces. In *Proc. of Pervasive*, 2012.

[8] C. Chekuri, N. Korula, and M. PÃąl. Improved algorithms for orienteering and related problems. In *Proc. of SODA*, 2008.

[9] J. Cordeau, M. Gendreau, A. Hertz, G. Laporte, and J. Sormany. New heuristics for the vehicle routing problem. *Logistics systems: design and optimization*, pages 279–297, 2005.

[10] P. Dallaire, C. Besse, S. Ross, and B. Chaib-draa. Bayesian reinforcement learning in continuous pomdps with gaussian processes. In *Proc. of IROS*, 2009.

[11] R. He, E. Brunskill, and N. Roy. Efficient planning under uncertainty with macro-actions. *Journal of Machine Learning Research*, 40:523–570, 2011.

[12] J. Higle and S. Sen. Stochastic decomposition: An algorithm for two-stage linear programs with recourse. *Mathematics of operations research*, 16(3):650–669, 1991.

[13] S. A. Hong. *Distributed Market-Based Algorithms for Multi-Agent Planning with Shared Resources*. PhD thesis, Carnegie Mellon University, 2013.

[14] L. Kaelbling, M. Littman, and A. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101:99–134, 1995.

[15] M. Kearns, Y. Mansour, and A. Ng. Approximate planning in large pomdps via reusable trajectories. In *Proc. of NIPS*, 1999.

[16] A. Koning and L. Peng. Goodness-of-fit tests for a heavy tailed distribution. Econometric institute report, Erasmus University Rotterdam, Econometric Institute, Nov 2005.

[17] A. Krause and C. S. Ong. Contextual gaussian process bandit optimization. In *Proc. of NIPS*, 2011.

[18] H.-P. Kriegel, M. Renz, M. Schubert, and A. Zuefle. Statistical density prediction in traffic networks. In *Proc. of IEEE ICDM*, 2007.

[19] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance. Cost-effective outbreak detection in networks. In *Proc. of KDD*, 2007.

[20] X. Li, J. Han, J.-G. Lee, and H. Gonzalez. Traffic density-based discovery of hot routes in road networks. In *Proc. of SSTD*, 2007.

[21] S. Liu, Y. Liu, L. Ni, J. Fan, and M. Li. Towards mobility-based clustering. In *Proc. of KDD*, 2010.

[22] W. Liu, Y. Zheng, S. Chawla, J. Yuan, and X. Xing. Discovering spatio-temporal causal interactions in traffic data streams. In *Proc. of KDD*, 2011.

[23] J. Mockus, V. Tiesis, and A. Zilinskas. The application of bayesian methods for seeking the extremum. *Towards Global Optimization*, 2:117–129, 1978.

[24] G. Nemhauser, L. Wolsey, and M. Fisher. An analysis of the approximations for maximizing submodular set functions. *Mathematical Programming*, 14:265–294, 1978.

[25] A. Y. Ng and M. Jordan. Pegasus: A policy search method for large mdps and pomdps. In *Proc. of UAI*, 2000.

[26] G. Polychronopoulos and J. Ttsitsiklis. Stochastic shortest path problems with recourse. *NETWORKS*, 27:133–143, 1996.

[27] C. Rasmussen and C. Williams. *Gaussian processes for machine learning*, volume 1. MIT press Cambridge, MA, 2006.

[28] D. Silver and J. Veness. Monte-carlo planning in large pomdps. In *Proc. of NIPS*, 2010.

[29] K. Sirvio and J. Hollmén. Spatio-temporal road condition forecasting with markov chains and artificial neural networks. In *Proc. of HAIS*, 2008.

[30] N. Srinivas, A. Krause, S. Kakade, and M. Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *Proc. of ICML*, 2010.

[31] M. Streeter, D. Golovin, and A. Krause. Online learning of assignments. In *Proc. of NIPS*, 2009.

[32] J. Yoon, B. Noble, and M. Liu. Surface street traffic estimation. In *Proc. of MobiSys*, 2007.

[33] J. Yuan, Y. Zheng, X. Xie, and G. Sun. Driving with knowledge from the physical world. In *Proc. of KDD*, 2011.

[34] Y. Zheng, Y. Liu, J. Yuan, and X. Xie. Urban computing with taxicabs. In *Proc. of UbiComp*, 2011.