III: Small: Investigating Spatial Big Data for Next Generation Routing Services

December 17, 2012
Project Summary

Increasingly, location-aware datasets are of a size, variety, and update rate that exceed the capability of spatial computing technologies. This project addresses the emerging challenges posed by such datasets, which we call Spatial Big Data (SBD). SBD examples include trajectories of cell-phones and gps devices, temporally detailed (TD) road maps, vehicle engine measurements, etc. SBD has the potential to transform society. A recent McKinsey Global Institute report estimates that personal location data could save consumers hundreds of billions of dollars annually by 2020 by helping vehicles avoid congestion via next-generation routing services such as eco-routing. Eco-routing may leverage various forms of SBD to compare routes by fuel consumption or greenhouse gas (GHG) emissions rather than total distance or travel-time.

However, the envisaged SBD-based next-generation routing services pose several challenges for current techniques. First, SBD requires a change in frame of reference, moving from a global snapshot perspective to the perspective of an individual traveling through a transportation network. Second, SBD magnifies the impact of partial information and ambiguity of traditional routing queries specified by a start location and an end location. For example, traditional routing identifies a unique (or a small set of) route(s), given historical and current travel-times. In contrast, SBD may identify a much larger set of solutions, e.g., one route each for thousands of possible start-times in a week, significantly increasing computational costs. Third, SBD challenges the assumption that a single algorithm utilizing a specific dataset is appropriate for all situations. The tremendous diversity of SBD sources substantially increases the diversity of solution methods. For example, methods for determining fuel efficient routes leveraging engine measurement and gps track datasets may be quite different from algorithms used to identify minimal travel-time routes exploiting temporally detailed roadmaps. Newer algorithms may emerge as new SBD becomes available, creating the need for a flexible architecture to rapidly integrate new datasets and associated algorithms.

Intellectual Merit: This project is expected to result in III innovations in three areas. First, Lagrangian Xgraphs, a novel concept in computer science, will be explored at conceptual, logical and physical database levels to model travelers' frame of reference, a major departure from traditional binary relationship (e.g., adjacency) graphs. Second, to address increased computational cost from partial query specification, we will explore the concept of route-collections, and scalable algorithms for finding route-collections. For example, to identify a route-collection over all possible start-times of a given time-interval, we will investigate a critical time point approach which divides a given time-interval into a set of disjoint sub-intervals of stationary-rankings among alternative routes. The approach is not only novel but also very important for the field. We believe that critical time points may become a vital component of dynamic programming (DP) solutions, which will need reconsideration in the face of emerging temporally detailed SBD that violate DP assumptions about stationary ranking of alternate solutions. Third, to address the increasing diversity of SBD methods, we will investigate algorithm-ensembles, flexible architectures that allow rapid integration of new data sources and routing algorithms. The team has a track record of publications and innovation not only on the current generation of routing algorithms and digital roadmaps, but also on emerging TD roadmaps and spatio-temporal routing algorithms. The team also has access to required resources such as TD roadmaps, gps track data, and fuel consumption datasets.

Broader Impact: The proposed work, if successful, will serve U.S. goals for energy independence and sustainability by laying the ground work for eco-routing and other travel-related services that reduce fuel consumption and greenhouse gas emissions. By increasing the availability of SBD and related software prototypes, the project also enhances the research infrastructure for other researchers, few of whom have access to these datasets currently. Educational activities will include curriculum development and training of students in the emerging area of SBD and computational aspects of next generation routing services. The team includes female and minority graduate students and the PI has a track record of participation in summer institutes involving undergraduate students from historically black colleges and universities. Results will be submitted for publication in relevant peer-reviewed III conferences and journals.

Keywords: Spatial Big Data; Spatial Databases; Spatial Data Mining; Routing; Road-maps
1 Introduction

Routing and navigation services are a set of ideas and technologies that transform lives by understanding the physical world, knowing and communicating relations to places in that world, and navigating through those places. From Google Maps [1] to consumer Global Positioning System (gps) devices, society is benefiting immensely from routing services. Scientists use gps to track endangered species to better understand animal behavior, and farmers use gps for precision agriculture to increase crop yields while reducing costs. We’ve reached the point where a hiker in Yellowstone, a biker in Minneapolis, and a taxi driver in Manhattan know precisely where they are, their nearby points of interest, and how to reach their destinations.

Increasingly, however, the size, variety, and update rate of spatial datasets exceed the capacity of commonly used spatial computing and database technologies to learn, manage, and process the data with reasonable effort. We believe that this data, which we call Spatial Big Data (SBD), represents the next frontier in routing services. Examples of emerging SBD datasets include temporally detailed (TD) roadmaps that provide speeds every minute for every road-segment, gps track data from cell-phones, and engine measurements of fuel consumption, greenhouse gas (GHG) emissions, etc. Harnessing SBD has transformative potential. For example, a 2011 McKinsey Global Institute report estimates savings of “about $600 billion annually by 2020” in terms of fuel and time saved [2] by helping vehicles avoid congestion and reduce idling at red lights or left turns. Preliminary evidence for the transformative potential includes the experience of UPS, which saves millions of gallons of fuel by simply avoiding left turns (Figure 1(a)) and associated engine-idling when selecting routes [3]. Immense savings in fuel-cost and GHG emission are possible if other fleet owners and consumers avoided hot spots of idling, low fuel-efficiency, and congestion. Ideas advanced in this proposal are likely to facilitate ‘eco-routing’ to help identify routes which reduce fuel consumption and GHG emissions, as compared to traditional route services reducing distance traveled or travel-time. It has the potential to significantly reduce US consumption of petroleum, the dominant source of energy for transportation (Figure 1(b)). It may even reduce the gap between domestic petroleum consumption and production (Figure 1(c)), helping bring the nation closer to the goal of energy independence [4].

![Figure 1: Eco-routing supports sustainability and energy independence. (Best in color)](image)

However, SBD raises new challenges for the state of the art in spatial computing for routing services. First, it requires a change in frame of reference, from a global snapshot perspective to the perspective of the individual object traveling through a road network. Second, SBD increases the impact of the partial nature of traditional route query specification. It significantly increases computation cost due to the tremendous growth in the set of preference functions beyond travel-distance and travel-time to include fuel consumption, GHG emissions, travel-times for thousands of possible start-times, etc. Third, the growing diversity of SBD sources makes it less likely that single algorithms, working on specific spatial datasets, will be sufficient to discover answers appropriate for all situations.

The goal of this proposal is to investigate promising approaches to address SBD challenges towards developing next generation routing concepts. An ideal team to successfully carry out the proposed tasks needs expertise in routing algorithms, SBD relevant to next-generation routing services, spatial database management systems, and spatial data mining. Our team has the expertise required to carry out the proposed tasks. The P.I. was elected an IEEE Fellow for his contributions to spatial databases [8] and spatial data mining [9]. His research group has published extensively on routing algorithms [10–15] and roadmap storage methods [16, 17] underlying the current generation of routing services. His group is also pioneering exploration of SBD [18–22] such as TD roadmaps for next generation routing services through research on storage issues [23], spatio-temporal routing algorithms [24], and use-cases [25, 26].
2 Background

2.1 Traditional Routing Services

Traditional routing systems utilize digital road maps [27–30]. Figure 2(a) shows a physical road map and Figure 2(b) shows its digital, i.e., adjacency graph-based, representation. Road intersections are often modeled as vertices and the road segments connecting adjacent intersections are represented as edges in the graph. For example, the intersection of SE 5th Ave and SE University Ave is modeled as node N1. The segment SE 5th Ave between SE University Ave and SE 4th Street is represented by the edge N1-N4. The directions on the edges indicate the permitted traffic directions on the road segments. Digital roadmaps also include attributes for road-segments (e.g., center-lines, road-classification, speed-limit, historic speed, historic travel time, address-ranges, etc.) Figure 2(c) shows a tabular representation of nodes and edges in the digital road map. Additional attributes are shown in the edge table. For example, the entry for edge E1 (N1-N2) shows its speed and distance. Such datasets include roughly 100 million (10^8) edges for the roads in the U.S.A. [28]. Turn restrictions are difficult to model in either the node or edge table, as they represent relationships among edges. Commercial approaches [30] annotate nodes with turn information for procedural interpretation during graph traversals. Other limitations include lack of modeling of synchronized traffic lights and differences in delays across left turns and right turns.

Figure 2: Current representation of road maps as directed graphs with scalar travel time values.

Route determination services [31, 32], abbreviated as routing services, provide two basic types of service [33]. The first deals with determination of a best route given a start location, end location, optional waypoints, and a preference function. Here, the choice of preference function could be: fastest, shortest, easiest, pedestrian, public transportation, avoid locations/areas, avoid highways, avoid tollways, avoid U-turns, and avoid ferries. Route finding is often based on classic shortest path algorithms such as Dijkstra's [34], A* [35–40], hierarchical [10,11,41–61], materialization [11,59,62–81,81,81–96], Transit Nodes [97–101], Hub Labellings [102–106], Contraction Hierarchies [93, 107–111], GPU-based [112,113], and other algorithms for static graphs [114–127]. Shortest path finding is of greatest interest to tourists as well as drivers in unfamiliar areas. In contrast, commuters often know a set of alternative routes between their home and work. They use an alternate service to compare their favorite routes using real-time traffic information, e.g., scheduled maintenance and current congestion.

2.2 Spatial Big Data for Next Generation Routing Services

SBD are significantly more detailed than traditional digital roadmaps in terms of attributes and time-resolution. Examples of SBD include gps track data, temporally detailed (TD) road maps, spatio-temporal engine measurement data, etc. In this subsection we describe gps track data and TD roadmaps. Additional types of SBD are discussed in the “Facilities, Equipment, and Other Resources” document.

Gps Track Data: Gps trajectories are becoming available for a larger collection of vehicles due to rapid proliferation of cell-phones, in-vehicle navigation devices, and other gps data-logging devices [128] such as those distributed by insurance companies [129]. Such gps tracks allow indirect estimation of fuel efficiency and GHG emissions via estimation of vehicle-speed, idling, congestion, synchronized traffic lights, and turn delays. They also make it possible to offer personalized route suggestions to users to reduce fuel consumption and emissions. In recent years, consumer gps products [128,130] have been evaluating the potential of this
approach. In addition, GPS track mining [131–154] has also been used to create a subset of TD roadmaps connecting landmark locations and improve the quality of recommendation from web-based routing services leveraging traditional roadmaps. A key hurdle is the dataset size, which can reach $10^{13}$ items per year given constant minute-by-minute resolution measurements for all 100 million US vehicles.

**Temporally Detailed (TD) Roadmaps:** New datasets from companies such as NAVTEQ [28], who have provided us one of these datasets and a letter of commitment, use probe vehicles (e.g., GPS tracks) and highway sensors (e.g., loop detectors) to compile travel time information across road segments for all times of the day and week at fine temporal resolutions (seconds or minutes). While, traditional roadmaps (Figure 2(a)) have only one scalar value of speed for a given road segment (e.g., EID 1), TD roadmaps may potentially list speed/travel time for a road segment (e.g., EID 1) for thousands of time points (Figure 3(a)) in a typical week. This level of detail allows a commuter to compare alternate start-times in addition to alternate routes. It may even allow comparison of (start-time, route) combinations to select distinct preferred routes and distinct start-times. For example, route ranking may differ across rush hour and non-rush hour periods and in general across different start times. However, TD roadmaps are big and their size may exceed $10^{13}$ items per year for the 100 million road-segments in the US when associated with per-minute values for speed or travel-time. Thus, industry is using speed-profiles, a lossy compression based on the idea of a typical day of a week, as illustrated in Figure 3(b), where each (road-segment, day of the week) pair is associated with a time-series of speed values for each hour of the day.

![Figure 3: Temporally Detailed Roadmaps using Historical Speed Profiles. (Best viewed in color)](image)

In the near future, values for the travel time of a given edge and start time will be a distribution instead of scalar. We also envision richer temporal detail on many preference functions such as fuel cost, pot-holes [155], crime reports [156], and social media reports of events on road networks [157].

### 2.3 Preliminary Results

In our relevant preliminary work on SBDs, we developed logical [158,159] and physical storage models [23] suitable for TD roadmaps and extended classical shortest path algorithms such as Dijkstra’s and Bellman Ford’s algorithm for TD roadmaps [13,160].

Our logical model [158,159], called time aggregated graph, is based on associating a time series of attributes to nodes and edges of the graph representing the underlying topological roadmaps. Consider a series of snapshots of a sample TD roadmap as shown in Figure 4(a) for start times 1, 2, 3, and 4. An alternative representation, namely time-expanded graph (TEG) [161,162], is shown in Figure 4(c), which stitches all snapshots via edges representing not only start-node and end-node but also start-time and end-time. In our time aggregated graph representation shown in Figure 4(b), edge (A,C) would be associated with time series [1,1,2,2], representing the travel times at start-time $t = 1$, $t = 2$, $t = 3$, and $t = 4$, respectively. Time aggregated graph is a more concise representation relative to the snapshot and TEG model as it doesn’t replicate nodes and edges across timepoints. It also facilitates longitudinal reasonings and design of efficient algorithms. Due to the potential transformative impact, this work received the best paper award at the Sensor-KDD workshop, ACM-SIGKDD in 2007 [163,164].

We have explored routing algorithms [13,160] for time aggregated graphs via adaptation of both Dijkstra’s and Bellman Ford’s algorithms. The adapted Dijkstra’s algorithm, called SP-TAG, determined the shortest travel time path for a source-destination pair for a given start time. The adapted Bellman Ford’s algorithm, called BEst Start Time, recommended a route and a suitable start time such that total time spent in the...
network was minimized. Both SP-TAG and BEST showed superior performance \cite{13} when compared to existing counterparts based on time-expanded graphs \cite{165,166}.

2.4 New III Challenges

SBD raises significant new challenges for state of the art spatial computing. First, it requires a change in frame of reference from a snapshot perspective to the perspective of the individual traveling through a transportation network. For instance, new temporally detailed (TD) roadmaps provide historical travel-time (or speed) for every road-segment for every distinct minute of a week. A traveler moving along a chosen path in a TD roadmap would experience a different road-segment and its historical speed as well as traversal-time at different time-intervals, which may be distinct from the start-time. In addition, the traveler may experience synchronized traffic lights and different delays for left turns, right turns, and going straight, which are difficult to represent and compute with traditional road intersection (node table, edge table) models of roadmaps (Figure 2), and their temporal generalizations such as snapshots, TEG, and time aggregated graphs.

Second, the growing diversity of SBD significantly increases computational cost because it magnifies the impact of the partial nature and ambiguity of traditional routing query specification. Typically, a routing query is specified by a starting location and a destination. Traditional routing services identify a small set of routes based on limited route properties (e.g., travel-distance, travel-time (historical and current)) available in traditional digital roadmap datasets. In contrast, SBD face orders of magnitude richer information, more preference functions (e.g., fuel efficiency, GHG emission, safety, etc.) and correspondingly larger sets of choices. New questions thus arise: What is the computational structure of determining routes that minimize fuel consumption and GHG emissions? Does this problem satisfy the assumptions behind traditional shortest-path algorithms (e.g., stationary ranking of alternative routes assumed by a dynamic programming principle)? For example, temporally detailed roadmaps can potentially provide a distinct route for every possible start-time, even when we just consider travel-time. This raises an optimality challenge of correctly determining the fastest route corresponding to each start-time, since ranking of candidate routes might vary with time of day (rush hour vs non-rush hour). It also raises a representation challenge to summarize potentially large sets of routes in the result. In addition, there is a computational challenge of efficiently determining a large collection of routes (e.g., one for each start time and preference function) by identifying and reducing unnecessary computations.

Third, the tremendous diversity of SBD sources substantially increases the need for diverse solution methods. For example, methods for determining fuel efficient routes that leverage engine measurement and gps track datasets may be quite different from algorithms to identify minimal travel-time routes for a given start-time exploiting TD roadmaps. In addition, SBD differ in coverage, roadmap attributes and statistical details. For example, TD roadmaps cover an entire country, but only provide mean travel-time for a single road-segment for a given start-time in a week. In contrast, gps-track and engine-measurements cover only well-traveled routes and time-periods, but may provide better measurement of synchronized traffic lights and turn delays. New algorithms are likely to emerge as new SBD become available and as a result, a new, flexible architecture will be needed to rapidly integrate new datasets and associated algorithms.
3 Proposed Approach

The proposed CS research will investigate a set of novel computational ideas to support next generation routing services. We expect the system to be useful for tourists, commuters, researchers, etc. Tourists may wish to know the best routes to the airport for different start times during 7am and 9am. Commuters can use the system to determine which of their favorite commuter routes is most eco-friendly starting at 7am. Researchers may be interested in GPS tracks that are faster than the best paths in TD roadmaps to identify changes to roadmaps and enrich route recommendations.

The system we envision for next generation routing services is shown in Figure 5. Starting from the far left of the figure, we see different types of SBDs including TD roadmaps, GPS tracks, engine measurements, etc. The SBDs may be accessed via the proposed Lagrangian Xgraph data model (detailed in Section 3.1) addressing the challenge of the traveler’s perspective and experience.

The far right of Figure 5 shows different users querying the next generation routing system via a user interface (UI) which will be implemented in HTML5 for use in a variety of platforms including cellphones. The UI will send a request to the ensemble of route finders (detailed in Section 3.3), which will be architectured to address the challenge of the growing diversity of SBD sources. It will get route recommendations via querying SBDs (e.g., GPS tracks) or invoking Xgraph traversal algorithms. It will summarize recommendations to reduce redundancies based on alternative metrics such as travel time, fuel use, etc., and send those to the UI for presentation via maps, text, audio, etc. To addresses the partial and ambiguous nature of traditional routing query specification, the system will leverage the All-start-time Lagrangian Shortest Path (ALSP) module (detailed in Section 3.2), which will determine dominance zones of routes based on metrics of interest such as fuel economy, drive-time, etc.

3.1 Approach to Traveler’s Frame of Reference

Next generation navigation and routing services will need to be able to accommodate a traveler’s frame of reference where candidate routes are evaluated from the perspective of a person moving through the transportation network. Traditional graph models of roadmaps do not adequately model many aspects of the traveler’s experience such as turn delays and synchronized traffic lights. Let us consider computing challenges in modeling turns for illustration.

Figure 6(a), which is an expansion of intersection N5 from Figure 2(b), illustrates the limitations of traditional graphs in modeling turns [161, 167, 168] from the traveler’s perspective. Consider node N5a which illustrates four scenarios: left turn, right turn, straight ahead, and U-turn. The left turn is modeled by edge (N5a, N5b), the right turn is modeled by edge (N5a, N5d), straight ahead is modeled by edge (N5a, N5d), and the U-turn is modeled by edge (N5a, N5b). As can be seen, this model introduces several new nodes and edges per road intersection, thereby expanding the graph size by an order of magnitude. These turn edges may also have turn-delay attributes. Conceptualizing synchronized traffic lights further adds to the complexity of modeling the traveler’s perspective. This is because the capability of grouping straight ahead edges across road intersections with synchronized signals would be needed to support the fact that nearby traffic lights do not operate independently.

Modeling the traveler’s frame of reference requires innovation in three main areas. First, due to their potentially large and ever growing graph sizes, a concise representation is critical to reduce and possibly eliminate redundant information across different nodes and edges while still being able to model a traveler’s experience such as turn delays and traffic light synchronization. Second, new query language and logical data model concepts need to be investigated to represent and classify potentially new alternative semantics. A third challenge is the design of efficient and correct query processing strategies and algorithms.
**Proposed Approach:** To address these challenges, we propose a novel representational model called a *Lagrangian* X-graph, which is composed of a set of X-nodes and a set of X-edges representing trees over X-nodes. X-nodes and X-edges may have attributes with scalar or structured (e.g., time-series) values. Being a tree of X-nodes, X-edges are different from hyperedges, which represent subsets of nodes in a hypergraph, and directed hyperedges, which directly connect a set of sources to a set of destinations [169–174]. X-edges may be classified into sub-types to model different aspects of the traveler’s experience. For example, a binary-X-edge may model direct movement from an X-node to a neighboring X-node (also called a successor or a child) without going through any other X-node. An all-successors-X-edge represents a group of binary-X-edges from an X-node N to all children of N. A turn-X-edge may represent a movement relationship between two (or more) X-edges to model a left turn, a right turn, a U turn, or going straight ahead.

![Diagram](image)

**Figure 6: Modeling the traveler’s perspective (Best viewed in color).**

An example X-graph in Figure 6(b) has four X-nodes, namely, A, B, C, and D, representing road-intersections. It has seven X-edges including four binary-X-edges (A → B), (B → D), (A → C), (C → D) representing road-segments connecting adjacent road-intersections. It has an all-successors-X-edge (A → (B, C)) to represent the group of binary edges (A → B) and (A → C) from X-node A to its children X-nodes B and C. It also has two turn-X-edges (A → B → D) representing a right turn and (A → C → D) representing a left turn. The time-series associated with turn-X-edges represent turn delays. In this example, right turn has no wait. However, the left turn has a wait of 3 for start-time 1, 2 for start-time 2, 1 for start-time 3 and no wait for start-time 4. Turn-X-edges provide a concise representation of turn delays compared to models illustrated in Figure 6(a), since they do no split nodes of the original graph into multiple new nodes.

An X-route (modeling turn delays but not synchronized traffic lights) between two X-nodes n₀ and nₘ is a sequence of X-nodes (nᵢ), binary-X-edges (bᵢ₋₁,i) and turn-X-edges (tᵢ₋₁,i): n₀, b₀,₀, n₁, t₀,₁,₂, b₂,₂, ..., nₘ such that bᵢ₋₁,i is a binary-X-edge from node nᵢ₋₁ to X-node nᵢ and tᵢ₋₁,i is a turn-X-edge from X-edge bᵢ₋₁,i to X-edge bᵢ,i. The weight (e.g., travel time) of an X-route is the sum of the weights of its X-edges. In Figure 6(c), an example X-route is (A, (A → C), C, (A → C → D), (C → D), D) With this X-route, a traveler leaving road-intersection A at time 1 will reach road-intersection C at time 2, wait for a left-turn for 2 time units, and reach road-intersection D at time 5. Assuming travel-time is the X-edge attribute, this X-route has an attribute value of 4, which is the sum of travel-times for (A → C) and (C → D) and two units of travel-time for (A → C → D). For example, X-route (A, (A → B), B, (A → B → D), (B → D), D) has an attribute value of 3 due to the smaller wait for right turns. Note that the ranking of these two routes changes if we ignore left turn delay. This illustrates the value of modeling turn delay.

**Task T1: Investigate X-graph Conceptual Models of a broader set of Traveler’s Experiences:**

The goal of this task is to investigate the trade-offs between representational convenience (e.g., brevity) and computational scalability in defining X-graph concepts (e.g., X-edge subtypes and attributes) towards improving modeling of a larger set of traveler’s experiences (e.g., synchronized traffic lights). There are two major risks from the computational perspective. First, the number of possible X-edges (i.e., trees of X-nodes) is exponential (or worse) in the number of X-nodes (and binary-X-edges). Thus, unconstrained definition of X-edge sub-types for modeling traveler’s experiences can take a serious toll on computation and storage scalability. Second, unconstrained definition of X-edge attributes for modeling route cost (e.g., fuel use) can make it difficult to use common paradigms (e.g., divide and conquer, greedy, dynamic programming (DP)).

---

1In other fields, such a frame of reference is known as a Lagrangian frame of reference [162].
for designing scalable algorithms. For example, use of DP is facilitated if step costs are aggregated into route cost using a superior function [175], e.g., sum and maximum, where aggregate cost is never less than cost of any step and increase monotonically with step costs. In consultation with interdisciplinary colleagues at the Center for Transportation Studies (see letter of commitment), we plan to investigate alternative Xgraph conceptual models to understand the trade-offs between computational scalability and accuracy of representation of eco-routing relevant traveler experiences, such as synchronized traffic lights.

**Task T2: Investigate a Xgraph logical data model.** The goal of this task is to design a minimal set of datatypes and operations to concisely represent common SBD queries and efficient Xgraph traversal routing algorithms. This task is challenging due to the conflicting goals of concise representation and support for computationally efficient algorithms. Table 1 contrasts traditional snapshot graph operators with an initial set of operators for Lagrangian Xgraphs for four categories: node, node-pair relationship, edge-pair relationship, and route. The operators are specified using object-oriented notation of `<datatype> .operator_name`. For the node category, an example snapshot graph operator is `node.get(snapshot)`, which returns a node’s attribute values at a particular snapshot. In contrast, an example Xgraph operator is `Xnode.get-Xedges-with-turn-delay()`, which returns an Xnode’s associated turn-Xedges and their delay attributes. For the node-pair relationship category, an example snapshot graph operator is `edge.get(node1, node2, time)`, which returns an edge’s attribute values at a particular snapshot. On the other hand, an example Xgraph operator is `binary-Xedge.get(Xnode1, Xnode2)` which returns a binary-Xedge between Xnode1 and Xnode2 with its time-series valued attributes. Edge-pair relationships are not modeled by snapshot graphs but an example Xgraph operator would be `turn-Xedge.get(Xnode1, Xnode2)`, which returns a turn-Xedge between binary-Xedge1 and binary-Xedge2 with the associated turn-delay time series. For the routes category, an example snapshot graph operator is `route.eval()` which evaluates a route. By contrast, example Xgraph operators include `Xroute.eval()`, which evaluates an Xroute with turn X-edges and turn delay attributes, `Xroute.get-sub-route(from-Xnode, to-Xnode)`, which returns the subroute of the Xroute between the given Xnodes and Xroute1.glue(Xroute2), which connects Xroute1 to Xroute2 assuming that the last Xnode in Xroute1 is the same as the first Xnode in Xroute2.

<table>
<thead>
<tr>
<th>Category</th>
<th>Snapshot Operator</th>
<th>Lagrangian Xgraph Operator</th>
</tr>
</thead>
<tbody>
<tr>
<td>node</td>
<td>node.get(snapshot)</td>
<td>Xnode.get-Xedges-with-turn-delay()</td>
</tr>
<tr>
<td>node-pair relationship</td>
<td>edge.get(node1, node2, time)</td>
<td>binary-Xedge.get(Xnode1, Xnode2)</td>
</tr>
<tr>
<td>edge-pair relationship</td>
<td></td>
<td>turn-Xedge.get(binary-Xedge1, binary-Xedge2)</td>
</tr>
<tr>
<td>route</td>
<td>route.eval()</td>
<td>Xroute.eval()</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Xroute.get-sub-route(from-Xnode, to-Xnode)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Xroute1.glue(Xroute2)</td>
</tr>
</tbody>
</table>

The Lagrangian Xgraph datatypes and operations shown in Table 1 are preliminary and raise the following questions for investigation: Is this set of Xgraph data-types and operations closed, i.e., can operation results be represented within the set? Is this set expressive enough to represent spatio-temporal routing queries about a traveler’s experience relevant to eco-routing? Does it facilitate the design of computationally efficient algorithms? How should the set be refined to address the conflicting needs of expressive power and support for efficient algorithms?

**Task T3: Querying gps Tracks for Route Recommendations:** The task goal is to explore scalable query processing to make route recommendations from gps tracks without graph traversal algorithms. It is of value as gps tracks may reveal new route recommendations from gps tracks without graph traversal algorithms that are not available via a TD roadmap due to roadmap changes (e.g., traffic conditions) since the release of the TD roadmap. To test this value proposition, we will investigate the following route-from-gps query: Given a set of gps tracks and a TD roadmap, find new route recommendations for Lagrangian O-D pairs that are not discoverable by graph traversal on TD roadmaps. Processing this query is challenging since there are potentially a large number of origin-destination (O-D) pairs (i.e., \( \binom{n}{2} \)). This number approaches \( 10^{16} \) for the 100 million \( (10^8) \) road intersections in the US road network [28].

We propose to investigate scalable algorithms and indexes for processing the route-from-gps query. For
example, we will investigate an O-D join index data structure illustrated in Table 2 to speedup identification
of gps tracks passing via a given O-D pair. Given the geometric nature of gps tracks, we will first use map
matching [152] to identify the set of road intersections (Xnodes), road segments (binary-Xedges), and turns
(turn-Xedges) most likely traversed by a gps track. Its identifier will then be entered into the index entries
for all ordered Xnode pairs for which an Xroute can be recommended using the gps track at hand.

The O-D join index illustrated in Table 2 shows Lagrangian O-D
pairs and their corresponding routes. For example, if the Lagrangian
O-D pair “Home,Work” is of interest, then the index quickly allows us
to fetch routes 1, 2, and 3 versus searching for these routes between
Home and Work among all gps tracks on the fly. The existing spatio-
temporal literature [176,177] on indexes focuses primarily on geometry and time and does not consider
origin-destination pairs. One of the challenges in designing the O-D join index is the selection of data
structures to facilitate fast search on O-D pairs given an O-D pair or only one origin or one destination.
Ideally, one may use an O-D matrix with origins as rows and destinations as columns. However, this leads
to $10^{16}$ matrix entries given $10^8$ road intersections in the US roadmap. We will explore alternative ideas
such as sparse matrix data structures and partial indexes [178] (which indexes a subset of rows in the table)
to understand the tradeoff between storage overhead and access speedup issues in efficiently supporting
searching, inserting, and deleting index entries. After addressing the selection-based route-from-gps query
processing problem, we will investigate the join-based version. Direct gps tracks may not exist between
all O-D pairs, potentially requiring gps track joins via the Xroutes.glue() operation to determine route
recommendations. For example, for gps tracks $\langle A, B, C \rangle$ and $\langle D, B \rangle$, there is no direct path between $D$ and
$C$. However, joining the two (or more) tracks would reveal $\langle D, B, C \rangle$ as a new path. We will probe efficient
processing strategies for route recommendations using joins.

### 3.2 Approach to the Partial Nature of Traditional Routing Queries

Spatial Big Data magnifies the already partial and ambiguous nature of a traditional routing query. This
is because a typical routing query specified by a start location and a destination may result in multiple
answers. The apparent ambiguity in the shortest path query is clearly visible in the multiple candidate
routes returned for a single start and destination pair in common web-based routing services like [1,179,180].
For example, Figure 7(a) shows two candidate shortest travel time routes from the University of Minnesota
to MSP International Airport. Such ambiguity increases tremendously with the availability of SDB datasets,
resulting in increased computational costs due to the re-computation that may be necessary for thousands
of possible start times. We now consider this issue in the context of TD roadmaps via the all start-time
Lagrangian shortest paths (ALSP) problem.

Given a TD roadmap, a source, a destination, and a start-time interval, an ALSP problem determines a
collection of routes which includes the shortest path for every start time in the interval. The ALSP output
includes both the shortest paths and the corresponding set of time instants when the paths are optimal. For
example, for the problem of determining the shortest travel path time between the University of Minnesota
and the MSP International Airport during the interval from 7:00AM to 10:30AM. Figure 7(a) shows two
routes between the University and the Airport. The 35W route is preferred outside rush-hour, whereas the
route via Hiawatha Avenue is preferred during rush-hour (i.e., 7:00AM - 9:30AM) (see Figure 7(b)). Thus,
the ALSP route collection may be a set of two routes (one over 35W and one over Hiawatha Avenue) and
their corresponding time intervals.

Computing ALSP involves three computational challenges. First, the ranking of alternate paths between
any particular source and destination pair in the network is not stationary. In other words, the optimal
path between a source and destination for one start time may not be optimal for other start times. In
our previous example of shortest route between the university and airport, different routes were optimal
at different times. The principle of stationarity states that, if two reward sequences $R_1, R_2, R_3, \ldots$ and
$S_1, S_2, S_3, \ldots$ begin with the same reward, then the sequences should be preference-ordered the same as the
sequences $R_2, R_3, \ldots$ and $S_2, S_3, \ldots$ [181,182]. This means that in shortest route computation, if two journeys
$e_1, e_2, e_3, \ldots$ and $a_1, f_2, f_3, \ldots$ start at the same node, then the preference order of the journeys should not
change for another start time. This lack of stationarity in TD roadmaps eliminates the possibility of using
classic dynamic programming based techniques. Second, due to the potentially large and ever growing sizes
of SBD such as TD roadmaps, efficient computational methods need to developed which can scale to large
datasets. Lastly, many links in the network may violate the property of first-in-first-out (FIFO) behavior.

This is illustrated in Table 3, which shows the flight schedule for Delta airways [183] between Minneapolis and Austin, TX. Here, the travel time at 8:30 is 6 hrs 31mins, whereas waiting 40mins for the 9:10 flight would yield a quicker route 2hrs and 51 mins. This violation of first-in-first-out (FIFO) is called non-FIFO behavior. Surface transportation networks such as road networks also exhibit this behavior. For example, UPS [3,184] minimizes the number of left turns in their delivery routes during heavy traffic conditions. This leads to faster delivery and fuel savings.

To meet these challenges, we propose an approach based on two ideas: (a) critical time-point inspired divide and conquer, and (b) earliest arrival-time transformation. The first idea addresses the challenges of non-stationarity and redundant computation, while the second idea addresses the challenge of non-FIFO.

(a) Different routes between University and Airport [1]
(b) Preferred route varies over time.
(c) Total travel time of candidate paths between University and Airport

**Figure 7:** *Sample query for Spatial Big Data. (Best in color)*

**Critical time-point inspired Divide and Conquer Approach:** A naive approach for solving the ALSP problem would involve determining the shortest path for each start time in the interval using techniques [13, 40, 83, 185–189] developed for single-start time shortest paths. This leads to redundant re-computation of the shortest path across consecutive start times sharing a common solution. Some efficiencies can be gained using a time series generalization of a label-correcting algorithm [13]. However, this approach still entails a large number of redundant computations [24]. To reduce this redundancy, we propose the use of critical time points. For instance, consider again the problem of determining the shortest path between University of Minnesota and MSP international airport over a time interval of 7:30am through 10:30am. Figure 7(b) shows the preferred paths at some time instants during this interval, and Figure 7(c) shows the travel-times for the candidate paths for all start-times during the interval. The dotted lines across bars are drawn for ease of understanding only.

As can be seen, the Hiawatha route is faster for times in the interval [7:30am 8:30am], whereas 35W is faster for the interval [9:30am 10:30am]. This implies that the shortest path changed at some instant inside the interval [8:31 9:30]. For simplicity, we assume that the shortest path changed at 9:30am. We define this time instant as a *critical time point*.

Critical time points inspire a divide and conquer approach to handle network non-stationarity. In this approach we divide the time interval over which the network exhibits non-stationarity into smaller intervals which are guaranteed to show stationary behavior. Now, within these intervals, the shortest path can be computed using a single run of a dynamic programming (DP) based approach [13]. In our university-airport example, the critical time point was 9:30am. This created two discrete sub-intervals [7:30 9:29] and [9:30 10:30]. We can compute the ALSP using two runs of a DP based algorithm [13] on each sub-interval.

A key challenge in designing divide and conquer algorithms based on this idea is to minimize the amount of time needed to compute critical time points while ensuring correctness and completeness. Critical time points can be computed using one of two strategies, precomputing or lazy. In a precomputing based method, all the candidate solutions (routes) are enumerated and compared to determine the best route for each start time. This approach, however, would become a major bottleneck in the case of TD roadmaps, as there can be large number of candidate routes (one for each start time) between any source-destination pair. Therefore we propose using a lazy technique, which computes a superset of critical time points on-the-fly while exploring candidate routes. This technique is termed lazy because identification of first (\(n^{th}\)) critical time point is delayed until after determination of shortest path for the first (\(n^{th}\)) start time. For example, after computing
the shortest path for one start-time, the lazy technique will make a conservative estimate (while ensuring correctness) on the next critical time point at which the recomputation must start. In our university-airport example, this would mean that, while computing a shortest path at 7:30am we would estimate 9:30am as the next critical time point. The proposed lazy technique is described next using an ALSP solution adapting the Dijkstra’s algorithm.

Recall that Dijkstra’s algorithm initializes by associating labels (denoting the observed distance from a source) to each of the nodes in the graph. The source node is inserted into the priority queue (ordered on the distance label). In each iteration, a node with the least distance label is expanded and distance labels to its neighbors are either inserted into the priority queue or updated (if already present). This process continues until the destination node is expanded.

To adapt Dijkstra’s for our on-the-fly technique, we make two modifications. First, a data structure called, a Lagrangian distance array, is associated with each of the nodes (instead of a single label as in case of traditional Dijkstra’s). This array stores the current observed distance of the node from the source for each start time. Second, every node is associated with another data structure called a Lagrangian predecessor array. This data structure stores the current best predecessor found so far.

<table>
<thead>
<tr>
<th>Start Time</th>
<th>Route</th>
<th>Travel Time</th>
<th>Arrival Time</th>
<th>Earliest Arrival Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>8:30</td>
<td>via Detroit</td>
<td>6 hrs 31 mins</td>
<td>15:01</td>
<td>12:01</td>
</tr>
<tr>
<td>9:10</td>
<td>direct flight</td>
<td>2 hrs 51 mins</td>
<td>12:01</td>
<td>12:01</td>
</tr>
<tr>
<td>11:00</td>
<td>via Memphis</td>
<td>4 hrs 30 mins</td>
<td>15:38</td>
<td>15:38</td>
</tr>
<tr>
<td>11:30</td>
<td>via Atlanta</td>
<td>6 hrs 28 mins</td>
<td>17:58</td>
<td>17:21</td>
</tr>
<tr>
<td>14:30</td>
<td>direct flight</td>
<td>2 hrs 51 mins</td>
<td>17:21</td>
<td>17:21</td>
</tr>
</tbody>
</table>

The priority queue in our adapted Dijkstra orders the Lagrangian distance arrays based on the values at the first start-time (or current start time as algorithm proceeds). The rest of the searching and pruning proceeds as in Dijkstra’s except that now we would update the Lagrangian distance and Lagrangian predecessor arrays for all start times instead of a single scalar value. This continues until the destination node is expanded. At this stage the algorithm has found the shortest path to the destination for the first start-time (or current start time). In addition, our distance and predecessor arrays could potentially contain shortest paths for other start-times as well. However, it is important to note these are based on decisions made for first start-time (or current start time) as the priority queue was ordered on it. Thus, each entry in the Lagrangian predecessor array is now scanned starting at the first start-time (or current start time) to determine the next time instant when there is a change in predecessor. If the Lagrangian predecessor array of a node X is [A A A B B B], then, it shows that A was predecessor for times $t = 1, 2, 3$, whereas B was predecessor for times $t = 4, 5, 6$. If the current start time is $t = 1$, then the previous procedure would give $t = 4$ as the result. This time instant represents a possible critical time point. This procedure is repeated for each of the closed nodes and the earliest of the possible critical time points is set as the next potential critical time point, and is used as the current start time for the next iteration of the above steps.

**Task P1:** Generalize critical time points (CTP) to other shortest path algorithms: Generalization of CTP based divide and conquer approaches for adaptation of other shortest path algorithms (e.g. A*, Bidirectional, materialization) is important for two reasons. First, it will allow the evaluation of generalizability of the idea. Second, it may provide faster algorithms for ALSP. This is because many false potential CTP (not affecting the best route for a given node pair but leading to unnecessary computations) are generated [24] by Dijkstra’s algorithm as it explores a much larger set of candidates than is necessary. Thus, adapting critical time point-based divide and conquer to focused search strategies (e.g., A*, bi-directional, etc.) may reduce unnecessary computations. Since A* and bi-directional searches have shown remarkable performance [36–38, 107, 109, 110] in pruning the search spaces relative to Dijkstra’s, we plan to explore the benefits of A* and bi-directional search strategies to prune the search space for the critical time point based approach.

We would leverage an A* strategy for a critical time-point based divide and conquer approach by developing heuristics that satisfy the properties of consistency (or monotonicity) and admissibility. The admissibility property requires that a heuristic always underestimate the distance to the destination, while the consistency property requires that the heuristic follow triangle inequality. These properties are necessary to ensure the
correctness of the algorithm. We plan to investigate A*’s ability to reduce false potential critical time points and redundant computations.

We also plan to probe bi-directional search strategies for critical time point approach. A bi-directional search typically has two search frontiers, a forward and a backward search. Here, the forward search starts from the source, whereas the backward search starts from the destination on a reverse graph. In a reverse graph all edges in the original graph would be reversed. Designing bi-directional search for ALSP problem on TD roadmap is non-trivial, due to the need to know the arrival time at destination [40,83,189]. Therefore, we propose creating a virtual destination node and adding links of zero cost to all the temporal copies of all the other destination nodes. Now, the backward search would start from this virtual destination node to address the lack of knowledge about arrival time at the destination. After designing a backward search strategy for ALSP, we will investigate its ability to reduce false potential critical time points and redundant computations.

**Task P2: Generalize critical time points to a broader set of problems:** The goal of this task is to investigate generalizability of critical time points (CTP) based the divide and conquer approach to a larger set of computational problems. We will start with evaluating the CTP idea for a few specific problems. For illustration, consider the All-pair All start-time Lagrangian Shortest path Problem (AALSP). Given a TD roadmap and a start-time interval the AALSP problem determines a route collection which includes the shortest path, for every start time in the interval, for each possible (start, destination) pair. Given the TD roadmap of Figure 4(c), start-time interval [1 2], the output will include (B→D, [1 2]), (A→C, [1 2]), (A→B, [1 2]), (C→D, [1 2]), (A→C→D, [1 2]). Note that the shortest paths didn’t change across the start-times 1 and 2 raising the possibility of using CTP based divide and conquer to save unnecessary computation cost across these start times.

CTP may adapt traditional All-pair Shortest Path (ASP) algorithms (e.g., Floyd Warshall’s [34, 35], Johnson’s [34, 35]) to reduce redundant work across start-times 1 and 2. However, this task is much harder than that of adapting Dijkstra’s algorithm for ALSP, since we do not know the start-times for the sub-paths being combined in the recurrence step. This issue invalidates even a naive approach of invoking ASP algorithm for all distinct start-times. A valid naive AALSP algorithm may determine sub-paths utilizing the first \((K - 1)\) nodes in an adapted Floyd Warshall’s (FW) algorithm for all start-times before examining sub-paths utilizing first \(K\) nodes. This naive algorithm has redundant computation across start-times, where shortest paths do not change for some node-pairs, and raises two questions. First, how may redundant computations be reduced using node-pair sensitive CTP based divide and conquer adaptation of FW algorithm? We will probe this question via alternative approaches such as time-series compression (e.g., run-length encoding) and decomposition (e.g., start with a subset of start times ensuring solution for first start time). Second, will savings be enough for adapted FW algorithm to outperform alternative AALSP solutions based on invoking an ALSP algorithm for each node-pair (or each source node in case of an adapted Dijkstra’s algorithm for shortest path trees)? After understanding alternative ways of leveraging CTP for AALSP, we would investigate its applicability for other dynamic programming use-cases for temporal graphs.

**Address non-FIFO challenge:** Related work in the area of route collections with minimization of travel time as the preference function [190–192] are limited due to FIFO assumptions. Other related work in the area of route collections include skyline based techniques [193] which assume independence among dimensions and do not incorporate a traveler’s frame of reference. We now describe our approach for addressing the non-FIFO challenge for travel-time metric for route quality. This is a two step process [160, 187]. First, the travel time information is converted into arrival time information. Second, the arrival time information is converted into earliest arrival time information. The second step captures the possibility of arriving at an end node earlier by waiting (non-FIFO behavior). For example, consider again Table 3 which shows the flight schedule between Minneapolis and Austin, TX. Here, the result of the first step, shown in fourth column, was obtained by adding the start time with the travel time (flight time). Now, we would scan the arrival time information (in the fourth column) to capture any benefits associated with waiting. For example, we can observe that a quicker path for start time 8:30am can be obtained by waiting for 40mins. The last column of the table shows the result of the second step.

**Task P3: Generalize to a broader set of preference metrics:** The goal of this task is to investigate ways to address non-FIFO and non-stationarity challenges for a broader set of non-travel-time preference
metrics such as fuel usage, GHG emission, etc. We will investigate the idea of *Patience-payoff*, which represents the gains obtained by postponing start time. For example, a driver has a patience-payoff of a gallon of fuel in postponing her start time from 7:45am to 8:15am if her route needs 2 gallons of fuel during rush hour (e.g., 7am - 8am) and 1 gallon outside rush-hour. Generalization of earliest-arrival-transform and critical-time-point based divide and conquer to patience-payoff is challenging due to mismatch between units for waiting time (e.g. minutes) and preference metrics (e.g. gallons, tons, etc) such as fuel efficiency and GHG emissions. One approach to address this challenge is to introduce functions mapping preference-metric domains and wait time-interval to a common domain (e.g., carbon footprint). However, unrestricted definition of such functions can make it difficult to use common algorithm design paradigms, e.g., dynamic programming (DP). For example, DP is facilitated if the step (e.g., Xedge traversal) details do not matter beyond the cost attribute of the step. In other words, the functions mapping preference-metric domains and wait time-intervals should not be sensitive to the details of the steps (e.g., acceleration during Xedge traversal). In consultation with interdisciplinary colleagues at the Center for Transportation Studies, we plan to investigate alternative function definitions to understand the trade-offs between the computational scalability and accuracy of the patience-payoff representation.

### 3.3 Approach to the Growing Diversity of Spatial Big Datasets

SBD challenges the assumption that a single algorithm, applied on a specific dataset, will be appropriate for all situations. The tremendous diversity of SBD sources substantially increases the need for the diversity of solution methods. For example, leveraging engine measurement and GPS track datasets to find fuel efficient routes will likely require algorithms quite different from those that exploit TD roadmaps to find Fastest routes for commuters. Newer algorithms may emerge as new SBD become available, making it necessary to develop flexible architectures to rapidly integrate new datasets and associated algorithms.

**Task D1: Investigate new architecture for the Route Determination Component.** With the rise in complexity of software systems, lower level algorithms and data structures are no longer the only major design problems. The overall organization of the subcomponents making these large systems – the software architecture – presents a brand new category of challenges [194]. Significant research [195] has moved software architecture design into a grounded and challenging discipline.

The goal of this task is to investigate flexible but efficient architecture to deal with the growing diversity of SBD. This task is challenging due to following reasons. First, we cannot anticipate all possible future SBDs which may be relevant to route recommendation. Yet it is desirable to reduce the lag between availability of new SBDs and its use, particularly when it improves recommendation quality. Second, the goals of flexibility and computational efficiency are often conflicting and require trade-offs. Lastly, the architecture should scale up and down to platforms ranging from cell-phones to cloud computing.

Figure 5 illustrates the proposed *algorithm ensemble architecture* for next generation routing services. SBD sources feed into the algorithm portfolio where a collection of algorithms solve the proposed query using different approaches. A core challenge in choosing an ensemble of route finding methods is to reduce redundancy. If the route collection from a route finder method are always dominated by the route collection from another using less computational resources, then the former method doesn’t add any value to the ensemble. We propose to examine ways to identify and remove dominated methods from ensembles.

We propose to evaluate our straw-man architecture for flexibility and scalability. The initial system prototype will only include two SBDs (e.g., TD roadmaps and GPS tracks) and associated route finders. We will next add a new SBD, namely engine measurements (and associated route finders). This will test the architecture for the design goals of allowing for new algorithms and datasets to be added rapidly while remaining scalable for SBD. The architecture will be refined based on the experience of adding new SBDs. We will also compare our architecture with other common generic architectures (e.g., layered systems, repositories, event-based systems, etc. [194]) as well as domain-specific architectures (e.g., OpenLS [31,33]). In addition, we intend to utilize Oracle and ESRI’s expertise in designing architecture, as we have existing relations with both companies and both have provided letters of commitment.

The system components will include modules to merge results from different algorithms for removal of duplicates and grouping of overlapping routes. This is a non-trivial research task as detailed in Task D2. The initial portfolio will include single-SBD as well as multi-SBD methods. Single-SBD-based methods will in-
include traditional routing algorithms (A*, Dijkstra’s, bi-directional) and proposed TD-roadmap based ALSP algorithms (Section 3.2) to extract route collections, as well as gps-track querying methods (Section 3.1). Design of multi-SBD method is a computationally challenging and will be investigated in Task D3.

Task D2: Probe route summarization, merging, and grouping. The goal is to investigate methods to summarize a large set of routes from an ensemble of route finders into a smaller set of routes, balancing diversity and brevity. This task is challenging due to conflicting requirements of diversity and brevity.

![Example Input and Output of K-Median for Route-Collection](image)

We propose a preliminary approach for grouping routes by formulating the K-Median for Route-Collection (KMRC) problem as follows: Given a spatial roadmap graph, a desired number of summary routes, $k$, and a collection of routes, $R$, find a subset of $k$ routes in $R$ that minimizes route similarity. For illustration, route similarity may be defined as the number of nodes and edges that two routes $r_1$ and $r_2$ have in common. If $r_1$ and $r_2$ have no nodes or edges in common, then they would have a route similarity of zero; whereas if they had all edges in common and they were of equal length, they would have a route similarity of $|r_1|$, or the length of $r_1$ or $r_2$. Figures 8(a) and 8(b) illustrate an input and output example of KMRC respectively. The input consists of thirteen nodes, fifteen edges (with edge weights of 1 for simplicity), $k = 2$, indicating that two routes are desired, and the collection of routes (shown in the table in Figure 8(a)). The output contains two routes from the given collection of routes that minimize route similarity, $\langle S, 1, 2, 3, 4, D \rangle$ and $\langle S, 8, 9, 11, D \rangle$. Additionally, merging and grouping route candidate answers may be done for all start times.

We propose to investigate novel algorithmic refinements such as seeding each subsequent time instant after the first time instant with the results of previous time instants to improve computational savings without reducing result quality. The idea is to minimize iterations by providing the algorithm with an answer that is close to the final answer and avoid redundant calculations for each time instant as would be done in a naïve algorithm. After maturing the KMRC approach to summarizing routes using illustrative similarity measure, we plan to investigate its generalization to other similarity and diversity measures based on SBD sources (e.g., TD roadmap, gps tracks), preference metrics (e.g., travel time, fuel used), path chosen, road type used (e.g., highway, side-streets), etc.

Task D3: Explore Multi-SBD Algorithms for Route Finding: The goal of this task is to investigate the potential of using multiple SBDs simultaneously in a single algorithm to provide richer route recommendations and perhaps lower computational costs compared to single-SBD algorithms. This task is challenging as SBDs may differ in concepts, positional accuracy, coordinate systems, recency, update rate, etc. An example of a quality issue arises from using aggressive driving gps trajectories that violate traffic laws (e.g., speed limits, running red lights, private roads). Similarly, positional accuracy may arise due to issues like gps drifts and weak signals.

In order to address these challenges we propose to explore the idea of “Lagrangian Xgraph registration”. Given an Xgraph and a gps track, Lagrangian Xgraph registration identifies the Xroute using the Xnode sequence which best approximates the gps track. This process yields the following two advantages. First, the total journey time obtained by summing the travel times of road segments in TD roadmaps, corresponding to Xedges (of the identified Xroute) may now serve as an upper bound on the fastest path via traversal of TD roadmap. This upper bound may be used in the Bi-directional search described in Task P1 to prune candidate partial paths resulting in lower computational costs. It may even be possible to use the above gps-track derived Xroute to reduce the cost of finding a feasible path before pruning phase.

Second, failure of “Lagrangian Xgraph registration” may indicate off-road trajectories or recent updates in the road networks not captured by TD roadmaps. For example, a new road or bridge might have been
constructed after the release of the current TD roadmaps. If a large number of gps-tracks fail to register in a geographic area, one may recommend manual review to decide about possible updates to the TD roadmap to provide richer route recommendations. However, this task is challenging due to the potential of false alarms generated from off-road gps tracks. We will investigate automatic methods to recommend manual inspection of geographic concentration of Lagrangian Xgraph registration failures to balance costs of false alarms and benefits of improved route recommendation. These methods will be refined iteratively via intermediate evaluation using SBDs (e.g., TD roadmaps, gps tracks) at hand.

4 Curriculum Development Activities

The current computer science curriculum has yet to incorporate the topics of spatial big data (SBD) and next generation routing services (e.g., eco-routing). Given the growing enthusiasm of industry and government for such services, we believe that a broad range of undergraduate and graduate students should be exposed to these topics. Thus, we propose to leverage the research activities described in previous sections to develop a new curriculum on spatial big data and next generation routing services for graduate and undergraduate courses.

We plan to dedicate two weeks to SBD in CSci 8715, a semester long course on Spatial Database Research. We will use our proposed research to develop a text-book chapter for the Spatial Database textbook [8] along with other instructional material such as slides, lecture notes, research projects, API's for handling SBD, exercises and laboratories. We plan to introduce SBD topics and research results in a couple of undergraduate courses such as algorithms and databases. Introductory courses on algorithms discuss classical dynamic programming (DP) based shortest path algorithms, such as Dijkstra’s [34, 35]. We plan to develop a guest lecture to discuss the DP assumption (e.g., stationary ranking), provide examples (e.g., TD roadmap and other SBDs) where DP assumptions do not hold, and introduce core ideas (e.g., critical time point) to develop next generation routing algorithms as well as other problems.

5 Broader Impact

Benefits to Society: The proposed framework, if successful, will lead to reductions in overall greenhouse emissions and related pollutants by increasing the availability of eco-routing to industry, government and the general public. Transportation sources are one of the primary contributors to greenhouse gas emissions. A recent study by the U.S. Environmental Protection Agency [196] reported that in 2008, transportation sources contributed to approximately 27% of total U.S. greenhouse gas emissions. Additionally, transportation was reported to be the fastest-growing source of U.S. greenhouse gas emissions, accounting for 47% of the net increase in total U.S. emissions since 1990. Also, it was found to be largest end-use source of CO₂ (most prevalent greenhouse gas). This creates urgent need for research and development of new technologies (e.g., eco-routing) which can mitigate the problem. This problem could be addressed by developing novel technologies which would assist fuel efficient route planning. This project could potentially lead to lower greenhouse gas emissions, given that petroleum based fuels form a major component in transportation.

Education and Workforce development: A significant outcome of this research project will be the training and development of graduate level researchers. PhD students supported through this project will participate in sustainability minor as well as leadership development program described in Facilities section under Institute on Environment. The research will also provide projects for the Undergraduate Research Opportunity Program (UROP) at the University of Minnesota and undergraduate honors theses. Results from SBD, next generation routing, and eco-routing research will be used to enhance the syllabi and teaching materials for courses in computer science as described in Section 4. We also plan to work closely with MathCEP at the University of Minnesota for outreach to K-12. MathCEP sponsors programs for K-12, undergraduate, and graduate students as described in Facilities section.

Broadening Participation of Underrepresented Groups: Two Ph.D. students out of six current ones come from groups underrepresented in Computer Science and are likely to participate in this project. Our proposed education plans include participation from historically black colleges and universities via a summer internship program. Each summer, we currently provide a 2-month long research experience for four to six undergraduate students from underrepresented groups as a part of the Expedition: Understanding Climate Change: A Data-Driven Approach project (see letter of commitment from Prof Kumar). In our proposed research on SBD for next generation routing services, we plan to engage students in software
development projects related to this proposal, along with presentations and demonstrations. In addition, the PI has supervised many female and minority graduate students (e.g., 4 Ph.D. and many M.S. students).

**Dissemination:** The research results will be submitted for publication in relevant forums including, IEEE Transactions on Knowledge and Data Engineering, the ACM SIGSPATIAL conference, the International Symposium on Spatial and Temporal Databases, and the International Conference on Geographic Information Science. We will also develop and maintain a web-site to distribute research results.

**Enhance Research Infrastructure:** Research undertaken in this proposal will enhance the current research infrastructure with hardware, software and dataset contributions. Car monitoring hardware used to record and analyze driving habits and behavior, along with fuel efficiency and consumption, will be purchased and made available to future research projects. In addition, open source software will be made available with the algorithms and techniques discovered. Anticipated functionality would be finding routes to enhance eco-routing and reducing fuel consumption. Lastly, we will work hard to obtain release permission for gps track datasets so they can be shared with other researchers interested in SBD and routing services.

## 6 Evaluation and Management Plan

**Evaluation Plan:** The Proposed ideas will be evaluated for computational scalability and route quality (e.g., travel time, fuel use, GHG emission) via prototyping within the open-source database management system PostgreSQL [197] with spatial and routing extensions of PostGIS [198], pgRouting [199]. A Lagrangian Xgraph logical data model, including new time routing operations and algorithms, will be prototyped by extending SQL with user defined data-types. Sources of SBD for in-laboratory experiments evaluating algorithm design will include TD roadmaps from Navteq, engine measurements from ORNL, gps-tracks from BI Inc., airline schedules (see letters of commitments and Facilities section), and synthetic datasets from network based traffic generators [200,201]. Airline flight schedule data will be used to evaluate the proposed EAT-based ideas to address non-FIFO behavior (task P3). Engine measurement SBD will be used to test the flexibility of architecture (task D1).

In addition, we will use test-vehicles with on-board equipment to monitor gps-track and engine measurements (e.g., fuel use, GHG emissions) to compare the proposed ideas with traditional routing services (e.g., Google Maps). We will use a smart-phone based client using HTML5 and javascript to not only access PostgreSQL based realization of the proposed next generation routing services from vehicles, but also collect new gps-tracks. We will also analyze the theoretical properties of the proposed algorithms (e.g., correctness, optimality) and data models (e.g., algebraic closure of proposed data-types and operations). Evaluation results will be used to improve approaches via gap analysis and refinements.

**Management Plan:** The PI will provide the general leadership for the project, as well as expertise in the computer science research areas described in Section 3. In addition, he will manage contact with transportation experts from Center for Transportation studies at UMN who will be helping to validate the research. He will also lead curriculum development activities. He will track the progress of the project and ensure the timely and accurate reporting of the project’s status to the NSF. The schedule of research tasks to be divided between two PhD students is given in Table 4.

### Table 4: Schedule of project tasks by years

<table>
<thead>
<tr>
<th>Year 1</th>
<th>Year 2</th>
<th>Year 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Student 1</td>
<td>Task T1, T2</td>
<td>Task T3, D1</td>
</tr>
<tr>
<td>Student 2</td>
<td>Task P1</td>
<td>Task P2, P3</td>
</tr>
</tbody>
</table>

## 7 Results from Prior Support

The PI’s work has been supported by multiple NSF grants [202–211] resulting in 20 Ph.D students and many publications [212–220]. His most recent grant relating to spatio-temporal network databases was “NSF: III-CXT: Spatio-temporal Graph Databases for Transportation Science” [202]. This yielded scalable algorithms for emergency evacuation planning and resulted in 2 Ph.D. dissertations, journal articles [14,221,222] and numerous conference papers [12,13,23,24,158,223]. In another relevant grant, “Databases for Spatial Graph Management” [208], the objective was to develop, evaluate and implement a set of network storage and access methods and network analysis algorithms. This project resulted in one Ph.D. thesis and several papers in journals [17] and conferences [11,16].
References


