A Demonstration Of PreGo: A System For Dynamic Multi-Preference Routing

Mohamed Ali1 Aqeel Rustum1 Dev Oliver2 David Hazel1 Abdeltawab M. Hendawi1,2 Ankur Teredesai1

1Center for Data Science, Institute of Technology, University of Washington Tacoma, WA, USA
2Department of Computer Science and Engineering, University of Minnesota, MN, USA

{mhali, binrusas, dhazel, hendawi, ankurt}@uw.edu
{hendawi}@cs.umn.edu
doliver@esri.com

ABSTRACT
This demo presents the PreGo framework. PreGo can best be described as a flexible and dynamic routing system. It is dynamic in the sense that it does not depend on static road network graph. Rather, we first introduce a new structure termed: Attribute Time Aggregated Graph (ATAG), constructed from volunteered geographic information attributes (VGIs) such as GPS traces, crime, accident reports, open access maps and so on. Each edge in the ATAG augmented road network graph has multiple attributes with multiple values per attribute. Moreover, ATAG provides the ability to store, retrieve and inference of values depending on query time of the day. Within the context of the PreGo system, a Time-Parameterized Multi-Preference Shortest Path (TP_SP) algorithm is introduced to process multi-preference routing queries through a single traversal of the ATAG structure. TP_SP algorithm returns the best fit path(s) based on users’ specified start time. In case the start time is not given by the user, PreGo suggests the best start time that guarantees a route that optimally fits the user’s preferences. To further reduce the response latency, we introduce the bidirectional TP_SP algorithm that processes the query from the two ends at the same time. PreGo is also flexible in the sense that it allows its user to decide his own routing preferences and even vary their weights. During the demo, users will enjoy interacting with the PreGo system through its nicely designed interface to (1) Find the optimal route w.r.t user’s specified preferences for a given start time and (2) Obtain the best start time for commuting along with its corresponding route that optimally satisfies the chosen preferences. (3) Examine the correctness, efficiency and scalability by running batches of burning tests.

1. INTRODUCTION

People have become highly dependent on routing services in their daily commuting especially with the availability of these services on a variety of platforms, e.g., web-mapping, mobile applications, in-car GPS [3]. The shortest route is the typical answer returned by such routing services. However, this answer does not always comply with commuters’ preferences when distance and time are not the only attributes being considered. For example, a truck driver shipping a heavy load is more interested in taking a route that is mostly comprised of flat roads for safety purposes while tourists driving at night are more interested in passing by roads that have a higher number of open shops. More features and preferences can be addressed by considering additional factors beside the distance and the time such as the number of traffic lights, road conditions and risks, and the number of interest points which in turn can lead to discovering the optimal route w.r.t multiple user’s preferences.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
Copyright 20XX ACM X-XXXXX-XX-X/XX/XX ...$10.00.
work graph characterized by the ATAG structure in which the edges are represented by multiple attributes, each of which is time parameterized. This means each attribute will be having a cost value for each time slot of the day.

(2) The graph construction and maintenance through which the underlying road network that is augmented by ATAG structure is initialized, built and updated from the input VGI data.

(3) The query processor component that is responsible for answering user routing queries. Two kinds of routing queries are supported within the PreGo framework; (a) finding the optimal path(s) for multiple attributes for a given start time, and (b) for users more interested in optimized travel time, versus a specific start or arrival time, PreGo can recommend a time to start their trip such that their preferences are best fulfilled.

To process those queries while preserving the efficiency and the scalability, the PreGo framework is equipped with the Time Parameterized Multi-Preference Shortest Path (TP_SP) algorithm that is proposed to efficiently extract the best-fit route from the ATAG structure. Following the Dijkstra’s approach [13], via a single traverse of the ATAG structure, the TP_SP algorithm smartly discovers the optimal path(s) from a source to a destination w.r.t. a combination of subset/all attributes by applying the prune and wait approach. Prune stops the expansion of graph traverse for all branches except that branch that is promising to carry an optimal value for at least one attribute. Hence, the prune strategy assures the system efficiency by saving the computation resources from being drained in unrequited graph expansion. The wait enforces the TP_SP algorithm to wait and not to declare the in-hand path that reached the destination node is the optimal path until all other branches are pruned. That means we are sure that the total cost at the destination node represents the cost of the dominant path and it will not be beaten by any other branch. Thus, the wait strategy guarantees the optimality of the returned route.

In order to further decrease the response time, we introduce the bidirectional version of our TP_SP algorithm that processes the query from the two ends at the same time.

During the demo, attendees will be able to interact with the PreGo system through its web-based graphical user interface to accomplish a number of different scenarios as follows. (1) Submitting routing queries to search for an optimal route for a specified source-destination pair of locations (2) Selected set of preferred attributes to be considered in the resulting route and freely set the weight for each preference. (3) Choose to either decide a start time, or let the system suggest the best start time for commuting so the optimal route is guaranteed. (4) Contributing to the system by reporting events associated with their location and time, and sharing their GPS trajectories. (5) Exploring the system internals through animated examples showing the behavior of the shortest path algorithms implemented in the system. (6) Examining the system behavior under stress tests to show how it behaves when being overwhelmed with a huge number of queries, and large combination of attributes. (7) Finally, to compare the computation costs for executing the TP_SP versus the bidirectional version.

2. OVERVIEW OF PREGO

The architecture of the PreGo system is given in Figure 1. PreGo has three main types of data sources for the leveraged Volunteered Geographic Information (VGI), namely, public GPS traces, open access maps, and risk reports, e.i., crime and car accidents reports. The system also has three main components, namely, the attributes time aggregated graph (ATAG) data structure, the graph construction and maintenance module, (Section 3), and the query processing module, (Section 4).

As a brief description for the system functionality, PreGo can be triggered from the query side or from the data source side. For the former, a commuter sends a routing query, (e.g., start and and locations, preferences, weights, start time), then the system dispatches the query processing module to find the optimal path(s) that satisfies the received parameters. For the later, when new data comes from a source, the PreGo calls the graph construction and maintenance module to refresh the edges cost of the ATAG structure, and hence, be reflected in the next queries answer. In the rest of this section, we elaborate more about the main data sources and ATAG structure.

2.1 VGI SOURCES

The proposed framework relies on data extracted from three main sources.

(1) Public GPS Traces. Our main source of volunteered data is the GPS tracks given by the crowd. We consider this a source as a vital one as we are able to extract the real travel time at each edge in the in-hand road network. OSM allows the download of volunteered GPS traces filtered by areas of interest. We use a tool provided on the OSM wiki called JOSM [12] which allows us to view OSM data and collect the GPS traces for a specified area.

(2) Open Access Maps. To obtain the base for the ATAG structure, we rely on the available free accessible map resources such as shape files [6], and the TAREEG web-service [2]. Through those resources we are able to extract the road network graph as set of nodes, edges and compute the basic weights, e.g., distance. In addition, they give us the ability to extract some indicators about the near by services and points of interests around each edge, e.g., lakes, parks, commercial buildings, schools etc.

(3) Risk Reports. The final data source contains data about car accidents and crime linked to their locations and times during the day, e.g., NHTSA [1]. The number of recorded crimes and accidents events around an edge gives an indicator of how risky this edge is.

2.2 ATAG DATA STRUCTURE

We introduced the ATAG structure to save all the data in a way that allows us to compute the shortest path from a given source location to a destination. Intrinsically, ATAG supports multi-preference routing functions, e.g., less fuel consumption, less car accidents, less pollution, or paths with more services, at different start times. Unlike the existing time aggregated graphs [5] in which each edge can have multiple weights for one attribute such as travel time cost, each edge in the ATAG data structure can have more than one attribute, e.g., travel time, distance, and risk. For each edge, we store multiple weights in different time slots. Figure 2 illustrates the idea of storing multiple attributes for each single edge in the ATAG data structure. As can be seen in the figure, each edge has three different attributes (i.e., travel time, distance, and risk) and each attribute can have either the one value like the distance attribute or different values like the travel time attribute which its values varies according to the time instance of the day. For exam-
example, the edge between \((n_1, n_3)\) has the same weight of distance unit for all times of the day, and different values for travel time and risk attributes.

3. GRAPH CONSTRUCTION AND MAINTENANCE

The graph construction and maintenance component takes care of processing the VGI data collected from the considered sources in addition to the data shared by the users. This component has two major roles. First, it is responsible to build the underlying road network graph \(ATAG\) via extracting map data, set of nodes, edges, and edges' attributes. Second, it initializes and refreshes the edges’ weights for each included attribute, e.g., travel time or risk level. Edges’ weight extraction approaches can be classified into two types: simple and complex. Simple weights extraction is basically a straightforward interpretation of the input data into an attribute’s weight for an edge. For example, translating the number of car accidents and crimes into a risk weight is computed by normalizing that number over the distance of the edge. To extract the travel time, we initially map-match each input GPS track into its corresponding consecutive edges based on the in-hand underlying road network map [10]. Then, we apply a sequence of validation on each GPS track, e.g., maximum speed limit violation. Only valid tracks are leveraged for obtaining the travel time weights. After that, each computed weight is saved at its corresponding time slot index of the day.

4. QUERY PROCESSING

The query processing module is in charge for processing users’ multi-preference routing queries. There are two cases, the query processor is able to handle for a given query expressed as source-destination pair of nodes, and a set of user’s preferences and weights. Case1: if the user provides the trip start time, the query processing module directs the search over the \(ATAG\) for the optimal travel time, in this case, we find out the best start time at which the optimal path(s) on the map as given in Figure 3. Throughout the rest of this section, we give the main idea of the basic TP_SP algorithm and explain an illustrative example. The pseudo code and detailed description of the TP_SP algorithm and the other two algorithms will be provided in the paper version of this demo.

Main Idea. In a single traverse, the TP_SP algorithm can find all optimal paths at a given start time w.r.t. to all/subset of the attributes stored in the ATAG structure. This is achieved based on two concepts, (1) The prune concept stops the expansion of all in-hand branches, obtained after expanding the start node in the ATAG, unless for those branches that are likely to contribute to the optimal paths result. We call the one of these branches a non-dominated sub-route. This internally means there is at least one attribute’s cost in this route that dominates the costs of the same attribute(s) in all other ongoing branches as well as the cost of the obtained route, if any, at the destination node. (2) The wait concept tells the TP_SP algorithm to wait and not to consider the first route to reach the destination node as the optimal until all the ongoing branches are pruned and no further expansion. By doing this, we guarantee that the paths collected at the destination node represents the optimal set of paths.

5. DEMO SCENARIOS

This section illustrates the users’ scenarios of using the proposed PreGo system. Through its intuitive user interface 3, audience will be able to interact with the system and examine its features and capabilities.

Through the following scenarios we briefly describe how the attendees of the conference will be able to interact with the system from different perspectives.

This demonstration relies on real implementation of the PreGo system components. A number of programming languages and frameworks have been leveraged to build various parts of the system. For example, the basic TP_SP algorithm and all of its variations along with the ATAG data structure are implemented in Java. The map-matching technique employed by the graph construction and maintenance module is implemented in C [10]. The user interface is built using a combination of Java scripts, HTML, and CSS, and leverages the Google maps APIs and the D3.js visualization packages. The demo is also based on real data for road network graphs obtained from the OpenStreetMap [12] with the assistance of the Tareeg framework [2]. To generate queries workloads, implemented a query generator in Python. These heterogeneous components talk to each other through sockets.

At the time of the demonstration at the conference venue, attendees can try the following scenarios.

5.1 Scenario 1: Find Multi-Preference Optimal Path

As a fundamental scenario of using the PreGo system, users can submit multi-preference routing queries to obtain the optimal route(s) from a source location to a destination given a set of preferred attributes and a specific start time as well. From the left expandable panel of 3 under "Directions", the user first need to provide the source, which is initially will be set to user current location as a default, and the destination locations either in a text address format or in latitude/longitude format. Then, specify his preferred starting time for commuting by choosing the "At" that is initially set to the current local time as a default. After that, specify his preferences through selecting the attributes he would like them to be considered in his routing plan. The attributes are presented with checkboxes through which he is able to specify more than one attribute. Finally, click the "Get Path" button and as a result the system will efficiently handle his request and after that will show the optimal path(s) on the map as given in Figure 3.
5.2 Scenario 2: Find Best Start Time

For the same routing query posted in the previous scenario, the user can choose to get recommendation from PreGo about the best time to begin traveling. This service is beneficial for users with flexible time and not constrained by a specific start time. To direct the system to achieve that request, the user simply checks the "Looking for best start time" radio button, Figure 5.2, and then clicks the "Get Path" button. The best-start \(TP_{SP}\) algorithm will be fired to compute the best start time for best fulfillment of the user’s preferences combination. In the best case, one starting time along with its relevant optimal route will be recommended. Nevertheless, it is possible to be recommended with more than a time, each of which associated with a route, when there is no single optimal path according to the user specifications can be achieved at any time of the day.

5.3 Scenario 3: System Insight

In this scenario, attendees who want to learn more about how the PreGo system works can visually inspect the internal behavior of the system when answering a routing query through a simple animated demonstration. In general, audience can see how the \(TP_{SP}\) algorithm is executed on a road network graph augmented by the \(ATAG\) structure. Along with that, users can see how the dominance relationship among a set of candidate nodes is computed for each graph expansion at each iteration. In addition, audience will view the animated demonstration of the bidirectional \(TP_{SP}\) behavior. Basically, we show how the forward and backward threads can collaboratively work to find an optimal path in faster pace of processing. These services can be reached from the "More" panel 3.

5.4 Scenario 5: Volunteering / Reporting

In this scenario, users will be able to contribute in promoting the system’s real time data by sharing their GPS tracks which in turn is going to be utilized through adapting the cost values and weights into the \(ATAG\). In addition, users can report events associated with their location and time of happening, second and third left panel in Figure 3. Example of such events includes risky events, e.g., crimes or accidents, and services report, e.g. liked restaurant, as well as disturbances, e.g., road construction.

6. REFERENCES