Intermodal public transit routing using Linked Connections

Pieter Colpaert[1], Alejandro Llaves[2], Ruben Verborgh[1],
Oscar Corcho[2], Erik Mannens[1], and Rik Van de Walle[1]

[1] Ghent University – iMinds – Multimedia Lab
firstname.lastname@ugent.be
[2] Ontology Engineering Group – Universidad Politécnica de Madrid
allaves@fi.upm.es and ocorcho@fi.upm.es

Abstract. Ever since public transit agencies have found their way to the Web, they inform travelers using route planning software made available on their website. These travelers also need to be informed about other modes of transport, for which they have to consult other websites, or for which they have to ask the transit agency's server maintainer to implement new functionalities. In this demo, we introduce an affordable publishing method for transit data, called Linked Connections, that can be used for intermodal route planning, by allowing user agents to execute the route planning algorithm. We publish paged documents containing a stream of hops between transit stops sorted by departure time. Using these documents, clients are able to perform intermodal route planning in a reasonable time. Furthermore, such clients are fully in charge of the algorithm, and can now also route in different ways by integrating datasets of a user's choice. When visiting our demo, conference attendees will be able to calculate intermodal routes by querying the Web of data using their phone's browser, without expensive server infrastructure.

Keywords: Semantic Web, Linked Open Data, Linked Data Fragments, public transit, smart cities, route planning

1 Introduction

Public transit agencies inform travelers using route planning software made available on their website. However, for every public transit agency they intend to use for their travel, they would have to consult a different website, in which they need to look up connecting trips. Furthermore, the way travelers want their route planning advice to be calculated is diverse, going from calculating routes with the nicest pictures on social networking sites [3], to finding routes which are feasible with a certain disability.

Linked Data Fragments [5] is a conceptual framework that captures all Web APIs to RDF on an axis (Figure 1) to discuss the trade-offs between efforts done by the data publishers versus efforts by the data consumer. On the far-most right point of the axis, public transit agencies have chosen to allow HTTP clients to access a route planning service. In this case, the server is doing all the effort, and it will always, in some way, restrict the way user agents are able to plan routes. On the far-most left point of the

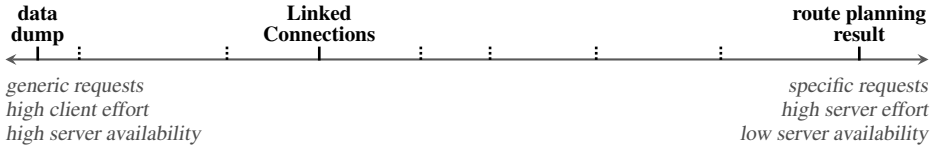

Fig. 1: The *Linked Data Fragments* axis illustrates that all HTTP interfaces offer data fragments, yet they differ in the specificity of the data they contain, and thus the effort needed to create them [5]. In this figure, the axis is applied to HTTP interfaces for the purpose of route planning.

Linked Data Fragments axis, data dumps of the transit schedules enable full flexibility. In this case, user agents execute the route planning algorithm on client-side, with the drawback that the effort that is required from user agents is high.

In this demo, we show that we can find a solution in the middle, which combines best of both worlds. We first discuss the related work, we then describe the point on the axis we have chosen and introduce the tools developed as part of the *Linked Connections* framework, and finally, we describe the set-up of the demo.

2 Related Work

In related work, a demo was given by choosing a different point on the Linked Data Fragments axis for the problem of solving SPARQL Queries. The solution introduced *Triple Pattern Fragments* (TPFs) to enable the client to perform reasonably fast joins [4]. This way, a low-end computer (Raspberry Pi) was able to host DBpedia, while offering the client the possibility to reliably solve *Basic Graph Patterns* (BGPs). In a similar way, this demo addresses route planning systems. Instead of focusing on high availability, our main goal is to let user agents stay in control of the route planning algorithm and to offer a way of combining different sources into intermodal route planning advice. In order to plan routes through a public transit system, recent research is slowly moving away from traditional shortest path algorithms such as Dijkstra or A* in favor of algorithms that work on top of the specific structure of transit schedules [1], or on top of a sorted list of connections [2]. These works, however, focus on creating algorithms optimized for one machine. This demo instead implements the basic *Connection Scan Algorithm* (CSA) [2] using *distributed* data through HTTP.

The *General Transit Feed Specification* (GTFS)[3] specifies the headers of 13 types of CSV files, describing the schedules using a set of rules. In recent years, GTFS gained a lot of popularity, thanks to its simplicity and its adoption in popular route planning systems such as Open Trip Planner, Navita.io, Google Maps or RRRR Rapid Real-time Routing. It is typically used as an exchange format: after downloading, existing tools convert the data to an in-memory, algorithm specific, structure. To update GTFS-files with real-time updates, GTFS-RT was created: a specification for a file that can be downloaded regularly, which includes a set of updates. In order to link the terms and identifiers of GTFS and its files with the Linked Open Data cloud, we have created

[3] *https://developers.google.com/transit/gtfs/*

Linked GTFS[4]. It helps transit agencies to provide context to the HTTP clients when executing the route planning by e.g., linking to the *gtfs:Route*[5] a vehicle follows, adding a *gtfs:headsign*, or indicating the type of the vehicle with a *gtfs:RouteType*.

3 Linked Connections

Linked Connections define a way to publish raw transit data, so that it can be used for intermodal route planning. Instead of solving each route planning query on the server-side Linked Connections enables route planning with a linear growing number of fragments with the number of connections.

A *connection*, in the context of this paper, is a hop from one transit stop to another. A connection is defined by an indication of the location and time of departure, and an indication of the location and time of arrival. Connections can be extracted from a set of rules, called a *transit schedule*, such as a GTFS dump. When all the connections of a certain transit system are available in one sorted (e.g., descending by departure-time) array, the basic CSA algorithm [2] can scan the list once and come up with the earliest arrival times for all stops. The results of the evaluation of extensions of this algorithm presented in their paper, seem promising.

A *Linked Connection* is a representation of a connection on the Web. It contains the data needed to define a connection, as well as useful links to other resources: e.g., links to GTFS terms, links to Linked Connections from different transit agencies, links to accessibility information, or links to nearby points of interests on geonames. In our implementation, we have published a sorted (by departure time) array of Linked Connections, as illustrated in Figure 2, using paged documents with *hydra:nextPage*[6] links. The array of sorted connections needed for CSA, now became a stream of connections which can be downloaded from the Web as the algorithm advances. For multimodal route planning, Linked Connections from different transit modes can be downloaded in parallel and merge-sorted just in time.

4 Demonstrator

The demonstrator can be viewed at *http://demo.linkedconnections.org/*[7]. When visiting the page, a map is served on which transit stops can be selected for various intermodal systems across the world. When both the start location and arrival destination are chosen, the algorithm will be executed. It will start downloading connections, until the earliest arrival time is found. In the process, the progress of visited transit stops will be visualised on the map.

---

[4] Linked GTFS is available at *http://vocab.gtfs.org/terms*. Mapping scripts to transform a GTFS file to Linked GTFS is available as open source at *https://github.com/OpenTransport/gtfs-csv2rdf*

[5] *gtfs:* is a prefix that can be expanded to *http://vocab.gtfs.org/terms#*

[6] *hydra:* is a prefix that can be expanded to *http://www.w3.org/ns/hydra/core#*

[7] The source code can be found at *https://github.com/linkedconnections*

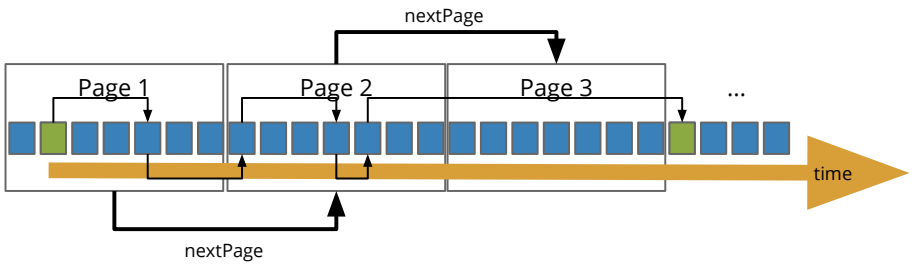

Fig. 2: This figure illustrates the demo's implementation. A sorted (by departure time) array of connections (squares) is published in pages or fragments. As the scanning algorithm advances, it decides to download more pages, discoverable through *hydra:nextPage* links. The algorithm returns a result when a path is found from a connection departing from a transit stop of choice (first green square), to a connection arriving at the destination stop of choice.

5 Conclusion

In this demo, we applied the Linked Data Fragments axis to the problem of intermodal route planning by introducing *Linked Connections*. We implemented a paged document containing a sorted list of connections between two transit stops, published on the Web. This way, we enable user agents to execute the route planning algorithm, and download connections from different agencies in parallel. Furthermore, this paged document has a finite amount of URLs per day. This results in better caching possibilities and thus lower server costs, compared to route planning APIs which execute the algorithm on the server-side.

The drawbacks are higher bandwidth consumption and slower results, as more data needs to be downloaded. Nevertheless, the HTTP clients can precache data, can show the results as the algorithm advances (streaming results) and/or calculate the routes on the application's server-side.

References

1. D. Delling, T. Pajor, and R. F. F. Werneck. Round-based public transit routing. In *Proceedings of the 14th Meeting on Algorithm Engineering and Experiments (ALENEX'12)*, 2012.
2. J. Dibbelt, T. Pajor, B. Strasser, and D. Wagner. Intriguingly Simple and Fast Transit Routing. In *Experimental Algorithms*, pages 43–54. Springer, 2013.
3. D. Quercia, R. Schifanella, and L. M. Aiello. The shortest path to happiness: Recommending beautiful, quiet, and happy routes in the city. In *Proceedings of the 25th ACM conference on Hypertext and social media*, pages 116–125. ACM, 2014.
4. R. Verborgh, O. Hartig, B. De Meester, G. Haesendonck, L. De Vocht, M. Vander Sande, R. Cyganiak, P. Colpaert, E. Mannens, and R. Van de Walle. Low-cost queryable linked data through triple pattern fragments. In *Proceedings of the 13th International Semantic Web Conference: Posters and Demos*, 2014.
5. R. Verborgh, O. Hartig, B. De Meester, G. Haesendonck, L. De Vocht, M. Vander Sande, R. Cyganiak, P. Colpaert, E. Mannens, and R. Van de Walle. Querying Datasets on the Web with High Availability. *Proceedings of the 13th International Semantic Web Conference*, 2014.