# Verifying Geostatistical Travel Time Properties on Routing Networks

Renaud De Landtsheer and Christophe Ponsard
*CETIC Research Centre, Gosselies, Belgium*

Keywords: Vehicle Routing, Geographic Information System, Speed Profile, Turn Restrictions, Contraction Hierarchies, Emergency Response.

Abstract: Nowadays, many systems are increasingly relying on interconnected, geolocated, and mobile devices. In order to cope with this, geographical information system (GIS) have evolved to precisely capture not only the spatial characteristics of real world transportation networks but also temporal dimension, including the variability of travel duration related to traffic jams. This paper explores the verification of a number of interesting spatiotemporal properties identified from a set of real world cases and expressed on enriched GIS data structures. It details our progress on developing efficient algorithmic modules to verifying such properties. We have applied our algorithms on the medical emergency infrastructures deployed in Belgium.

## 1 INTRODUCTION

The increasing deployment of interconnected and geolocated mobile devices has triggered the design of a new generation of Geographic Information Systems (GIS) able to provide more flexible and dynamic answer to several real-world routing problems such as the routing of vehicle fleets or the dispatching of emergency services. Such systems share the common needs of having to cope with the synchronization of several geographically distributed resources and specific requirements both of spatial and temporal nature.

For example for emergency medical care systems, it is important to make sure all locations can be reached within 10 minutes, 90 % of the cases. The travel time can be affected by a number of external factors that must be taken into account, such as known structural traffic jam, or crisis-specific scenarios such as the unavailability of some road in the context of a flooding. A complete analysis of such factors are reported by (Lin and Zito, 2005).

Geographic Information Systems (GIS) have evolved a lot in terms of level of details, accuracy but also ability to capture behavioural information both of spatial (turn restrictions, vehicle restriction w.r.t. height, width, weight,...) and temporal nature (e.g. statistical speed profiles as illustrated in Figure 1). These improvements require more complex algorithms to enable the design of innovative applications.

This paper presents our current achievements on the verification of spatiotemporal properties on such feature-rich maps. Our goal is to benchmark a number of specific data structures and algorithms able to efficiently verify those properties, especially temporal ones like worst case trip duration and computing accurate isochrones around one or multiple geographical points. This work was mostly driven by concrete industrial needs originating from a GIS company. It is mainly addressing a design-time verification perspective of geographically distributed infrastructure, with the system being designed by experts possibly with the guidance of optimization tools. Efficiency of such verification is important because we could not only consider them at design phase, but also deploy them in the system runtime in the future.

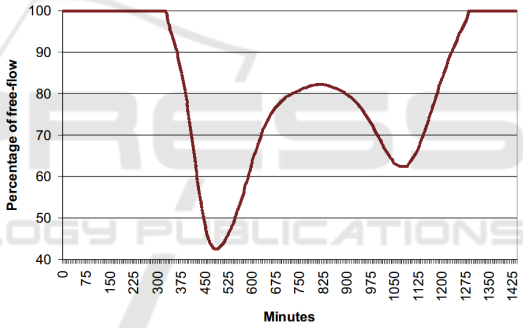

Figure 1: Typical speed profile.

The addressed challenge is not in the discovery of brand new algorithms but rather in the innovative and efficient combinations and adaptations of existing algorithms able to cope with the targeted verification.



Landtsheer, R. and Ponsard, C.
Verifying Geostatistical Travel Time Properties on Routing Networks.
DOI: 10.5220/0005794403960401
In *Proceedings of 5th the International Conference on Operations Research and Enterprise Systems (ICORES 2016)*, pages 396-401
ISBN: 978-989-758-171-7

More precisely, we are addressing the following **spatial restrictions**:

- one-way roads
- turn restrictions at crossing-points
- vehicle specific attributes like physical limit on height, width, weight
- vehicle specific regulation like no trucks in city centre, local traffic roads only allowed at the start or end of a trip

and the following **temporal restrictions**:

- speed limit on road segments
- speed profiles: speed can be lower than the free flow speed for various reasons such as traffic jam, traffic lights, ...
- temporary closed road during known periods, anticipated events or what-if scenarios like floods

This paper is structured as follows. Section 2 presents key algorithmic modules that we identified and structured to answer our needs. Section 3 describes our implementation, based on an existing GIS system. Section 4 reports our benchmarking of these algorithms to compute a number of typical verification queries. Section 5 applies and discusses them on a real emergency medical system. Section 6 presents some related work. Finally Section 7 draws some conclusions and identifies further work.

# 2 BACKGROUND: KEY ALGORITHMIC MODULES

This section presents the main algorithmic modules adapted to cope with the spatial and temporal restrictions mentioned above. We will consider as starting point that a road network is a graph where edges are road and nodes are intersections. Edges are decorated by some scalar information enabling to evaluate the travel time (e.g. distance + maximum speed). Edges are directed because the timing properties can differ considering the direction (or even be absent in case of one-way).

We also identify a collection of interesting algorithm to support our goal. Such algorithms are variants of Dijkstras shortest path algorithm (Dijkstra, 1959). Its principle is to perform a prioritized breadth-first exploration of the graph that grows the minimal travel cost tree. It can be used both for point to point search and for one-to-many exploration while other more efficient variants such as the A* (Hart, 1968) are focusing on directed search, and are restricted to point-to-point searches. A more complete survey recent of recent advances in algorithms for route planning in transportation networks is also available in (Bast et al., 2014).

## 2.1 Contraction Hierarchies (CH)

The search performance of shortest path algorithms can benefit from large speedups by using multi-level graph abstractions provided by *contraction hierarchies* (CH) (Geisberger et al., 2008). The multi-level graph is computed once for all in a pre-processing phase. Nodes are ordered by importance (or priority) using some heuristics. Shortcuts are introduced from the least to the most important and the contracted node is removed. This result in a new contracted graph with less nodes and where connection are faster routes. The process is iteratively repeated until the whole graph is contracted and yields a number of layer that can be interpreted as network of ”secondary”, ”national”, or ”high-speed” (virtual) roads. This phase can be very expensive both in space and time and also produces a graph with significantly more edges.

Queries require special variants of Dijkstras algorithm which proved extremely efficient: on large graphs, speed-up of several order of magnitudes (30-100 faster) when compared to the non-CH query.

It is also possible to compute efficient contraction hierarchies on time-dependent graph (Batz et al., 2009). It is also possible deal with graphs with turn costs (and thus the more specific case of turn restrictions) (Geisberger and Vetter, 2011).

## 2.2 Travel Time Functions (TTF) and Related Operations

Road edges in recent maps can now be annotated with *cost function* rather that scalar to capture speed profiles. Indeed, the speed on a road segment typically varies with the time of the day, depending on the level of congestion of the road segment. Such information is gathered from statistical data over long periods and classified over specific types of days (e.g. week days, Saturdays, Sundays/public holidays). *Speed profiles* are typically expressed as percentage of the nominal road free flow speed as show in Figure 1. They can exhibit a variety of curves typically with speed reductions during high traffic time resulting from recurring structural congestions in network bottlenecks such as town entrances.

At the level of the algorithms, the cost associated to a segment is represented as a *Travel Time Function* (TTF): for any possible start time on the segment, it

specifies the estimated duration of the travel on the segment. The TTF is the product of the speed profile by the free flow speed. Based on this, forward and backward path search can be implemented to compute time specific paths respectively for given start time and arrival time. However, in a verification perspective, a more interesting operation is to compute the best end-to-end travel time function and reason independently of a given time. For Dijkstras algorithm, two specific operations need to be efficiently supported by the TTF algebra: (1) *TTF composition* across two adjacent segments (taking into account the time taken to cross the first one) and (2) *TTF minimisation* when updating a node. These operations are illustrated in Figure 2.

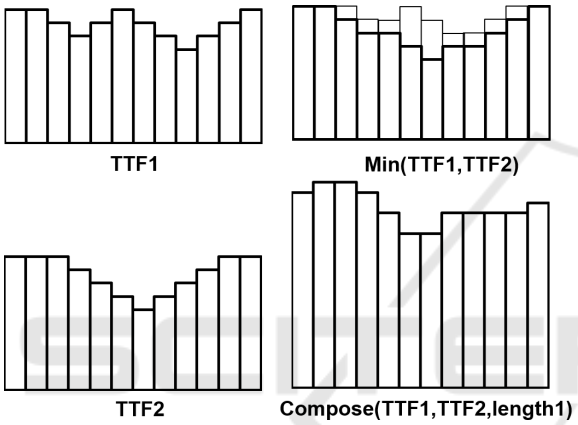


Figure 2: Travel time functions and related operators.

As the time complexity of profile search is high, the specific case of computing minimal and maximal time is addressed by more efficient algorithm computing min-max approximation of the resulting TTF. We also used this simplified algorithm to prune the search when implementing a precise TTF computation.

## 2.3 Dealing with Turn Restrictions (TR)

Turn restrictions can be represented either as an attribute attached to each crossroad of the map, or implicitly, by expanding each crossroad into several nodes, representing the entrance points to it, so that each allowed turn in the crossroad would be represented by a specific directed edge. The representation of turn restriction in the GIS has a great impact on the efficiency of the algorithms as well as the size of the maps. Our choice to model them as attribute was first dictated by size considerations as unfolding turn restrictions on the graph would result in an explosion of the number of nodes and would result in maps of unmanageable size. A similar conclusion was also reported by (Byrd et al., 2015) who took this approach at some point. The drawback is that all the algorithm related to contraction hierarchies need to be modified to cope with this additional attribute. Additionally, turns need to be checked for restriction on incoming and out-going edges, to that the whole reasoning become edge-based rather than node-based, increasing the complexity by $O(d^4)$ in the worst case where $d$ is the mean node degree. The degree of nodes tend to increase dramatically during the contraction process. However, as turn restrictions only affects a small portion of nodes, efficient solutions can be devised to cope with them.

# 3 IMPLEMENTATION

For the implementation, we started from an existing Open-Source routing engine called GraphHopper (Karich et al., 2015). It is among the few routing engines supporting contraction hierarchies with OSRM (OSRM Community, 2015) and Open Trip Planner (Byrd et al., 2015). However, none are currently supporting the discussed extensions, despite some ongoing work.

GraphHopper is implemented in Java and provides an efficient indexed storage representation to quickly retrieve edges and nodes of the graph and their related attributes such as edge geometries. These storages were extended to capture specific temporal and spatial edge attributes to cope with TTF and TR, respectively. These storages can either be mapped in memory for better performance or on kept disk for larger maps.

We represent TTFs through histograms with a constant number of slots. The composition is therefore approximate and can result in some computation sensibility, i.e. different algorithms (typically with and without CH) can produce different paths, however of near equivalent quality. This could be solved by using more precise representations such as piecewise linear TTF but at the price of some performance loss, and increase of storage size. Some algorithm extensions are able to cope efficiently with such extensions (Batz et al., 2010). In order to efficiently support min-max algorithms, min and max values are pre-computed, and explicitly stored.

Four main algorithms were implemented, covering the main GIS goals:

- **forward:** fastest path query for a given start time
- **backward:** fastest path query for a given arrival time
- **profile:** end-to-end travel time function from start point to destination point

- **flood:** map coverage give a number of starting points and travel duration

Those main algorithms rely on a number of software modules described in the previous section and that are efficiently shared across them forming a coherent global library, also easing future extensions. Only the three first functions are supporting contractions hierarchies.

# 4 BENCHMARKING

We benchmarked our algorithms both in CPU time and memory consumption on a sample of 400 routes (20 x 20 matrix between starting and arrival points) located inside a Belgian map. This map includes about 500.000 nodes, 640.000 edges with TTF, and 15.000 TR. For flooding algorithms we only used the 40 start points. A CoreI7 laptop was used using in-RAM storage with 1GB allocated JVM RAM. The reported CPU times are average time on the whole sample and are measured as user time. The RAM is the maximal memory consumption on the whole test run with a disk mapped storage for the map graph.

Table 1: Global Performances in CPU time and memory.

| Algorithm (TTF+TR) | Without CH | With CH |
|---|---|---|
| Forward Search | 420 ms / 732 MB | 7 ms / 694 MB |
| Backward Search | 485 ms / 707 MB | 8 ms / 682 MB |
| Single Profile Computation | 1285 ms / 794 MB | 31 ms / 738 MB |
| Single Min-Max Computation | 572 ms / 315 MB | N/A |
| Profile flooding (80% map) | 28s / 1.7 GB | N/A |
| Min-Max flooding (80% map) | 13s / 1.5 GB | N/A |

Table 1 shows the performance summary for typical usage of our algorithms. We could achieve speedup factors of 60 w.r.t the non-CH version with query times under 10ms. The speedup is also important for profile computation (40x) with computation time about 30ms. This enables the production of the large profile matrix required for optimizing vehicle routing in a few minutes. As expected, the computation of Min-Max are faster than the computation of full profile (about twice). The computation of flooding coverage is also very time-consuming. It grows quickly with the number edges in the graph (the contraction of this algorithm was not considered). Looking at the memory consumption, it is well kept under control because all CH version require less RAM although the graph are bigger in size (at least 2 times more important: about 167MB for the non-contracted map of Belgium and 355 for the contracted version). When using TTF, backward algorithms are slower than forward ones because TTFs are organized by entering time on the segment, so that querying them by time leaving the segment requires $O(\log(n))$ queries.

Table 2: Impact of introducing TTF and TR support).

| Algorithm | GH | TTF | TTF+TR |
|---|---|---|---|
| Map contraction | 2 min | 7 min | 26 min |
| Forward Search (NOCH) | 40 ms | 155 ms | 420 ms |
| Forward Search (CH) | 2 ms | 4 ms | 7 ms |
| Backward Search (NOCH) | 40 ms | 171 ms | 485 ms |
| Backward Search (CH) | 485 ms | 7 ms | 8 ms |
| Profile Computation (NOCH) | N/A | 339 ms | 1285 ms |
| Profile Computation (CH) | N/A | 21 ms | 31 ms |

As expected, the run time degraded significantly when introducing TTF and then TR, compared to the performance of the original GraphHopper algorithms as shown in Table 2. However, the degradation is lower in the CH version and is globally kept under control. Finally, contraction time is also increasing but within control. The main difficulty in the design was to avoid the explosion of the degree of the node (and shortcuts) resulting in a large size, compression time (possibly exhausting the machine) and poor query speed-ups. The current result might however prevent the contraction of very large (continental size) maps at the finest level of detail.

# 5 CASE STUDY: BELGIAN EMERGENCY RESPONSE SYSTEM

Our case study is the emergency medical care in Belgium, it is an ambulance dispatching system similar to the London Ambulance Service which has been extensively used in the literature (Finkelstein and Dowell, 1996)(Letier and van Lamsweerde, 2004). Critical time and space elements in an intervention scenario are summarized in Figure 3 (Moore, 2002).

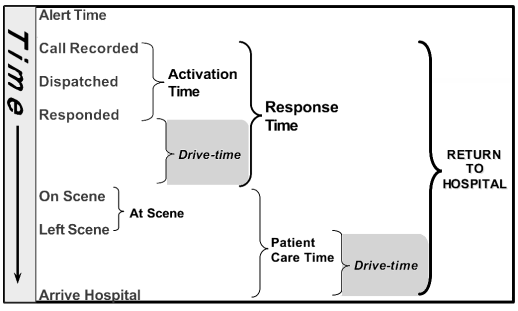


Figure 3: Critical time / space elements for intervention.

Two drive-time are present: from the allocated ambulance dispatch point to the intervention scene and, if patient conditions requires it, from the scene

to the most appropriate hospital. The belgian law put specific requirements requiring that *"most of the population should be reachable by road within 10 minutes at the maximal authorised speed, while the population unreachable within 15 minutes should be as small as possible"* (Federal Public Service for Health, 2007). We will focus here on this legal requirements.

The Belgian Emergency Care is composed of 370 departures points of different types which are distributed across the country. Let's call this set SMUR : `List[Location]`. Let us also define some GIS predicates such as within$(l,a)$ which is true when a location $l$ is within a given geographical are $a$. We can express the first part of the above property as follows:

$$\Box\forall l : \text{Location} \cdot \text{timeToReach}(l, \text{SMUR}) \leq 10\text{minutes}$$

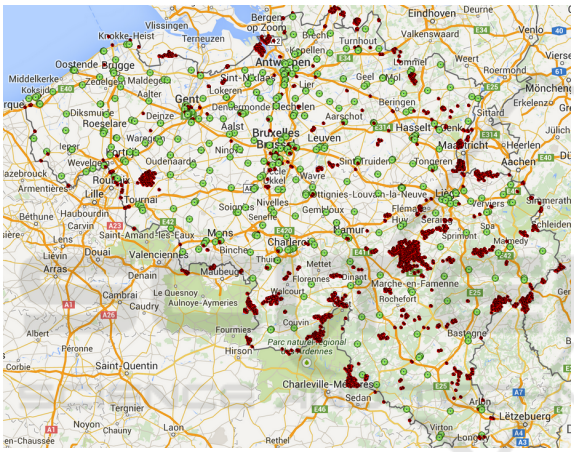


Figure 4: Map of emergency medical care coverage in Belgium within 10 minutes.

The timeToReach is implemented using the Profile on Map flooding algorithm with simultaneous search from all the SMUR locations. The search is also bounded with the maximum time allowed by law. The results is shows in Figure 4 in counter-example form, i.e. locations not fulfilling the property are displayed on the map as dark dots while the SMUR locations are displayed in lighter pinned dots. The coverage for 10 minutes is about 87 % of the nodes while 99 % are covered within 15 minutes. A closer analysis of the map shows that SMUR location are indeed at the centre of covered zones. Less covered areas match less populated areas and are actually covered by helicopter interventions. Uncovered zones around the national border are also addressed by agreements with emergency services of neighbouring countries, with some extra dispatch transfer time to be taken into account for these cases.

# 6 RELATED WORK AND DISCUSSION

Distance oracles are succinct data structures encoding shortest path information among a carefully selected subset of pairs of vertices in a graph. The encoding is done in such a way that the oracle can efficiently answer shortest path queries for arbitrary origin-destination pairs. The design of such oracles for time dependent graphs has been described in (Kontogiannis and Zaroliagis, 2014). In our work we rely on contraction hierarchies (Geisberger et al., 2008), which are considered as heuristics implementing such oracles. Our works extends the heuristics to efficiently cope with space restrictions by using a position-based graph, and time dependencies through a smart prioritization of the contraction as only a small part of the network is decorated with time-dependent information.

Dealing with large networks requires a lot of time to contract the data, and the achieved query times are sometimes not fast enough to compute large matrixes required in applications like vehicle routing. A solution reported in (Kieritz et al., 2010) is to use distributed memory parallelization. On a medium size network, 64 processes will cut down the preprocessing duration by a factor 28 and the queries duration by a factor 25. Globally 6 times more memory is required but per processor there is a reduction by a factor of 10.5. In our work we did not consider such techniques to accelerate the contraction: we rather improved the contraction and query procedures.

An extensive benchmarking has been conducted in the context of transportation networks (Bast et al., 2014). Such systems also have a strong time-dependent dimension because they are heavily constrained by schedules. The benchmarks where conducted on many algorithms that were implemented in C++. Our implementation is in Java. Although the domain, implementation and test environment are different, the results confirm the good performance of contraction hierarchies on large graphs.

A legitimate question is how representative are travel times estimates. A study conducted for travel times to hospital revealed that estimates were rather close approximations to reported times (Haynes et al., 2006), i.e. 50% of travel time estimates were within five minutes of the time reported by respondents. Compare to our target response time of 10 minutes, 5 minutes can seem huge. However the reported times are not for emergencies and include parking times, which can vary a lot. It also does not include statistical travel times while we do. We did not yet validate against a database of measured response time

although our analysis shows coherent results.

# 7 CONCLUSION AND PERSPECTIVES

This paper presented our on-going work in analysing complex GIS-based systems involving complex spatiotemporal constraints. We showed how map data structures can capture constraints such as travel time functions and turn restrictions. We then focused on building a toolbox of efficient algorithms. Coping with the complexity induced by the extra graph constraints required non-trivial work. So far, the available modules were quite easy to deploy. They can be enriched to deal with more restrictions such as truck attributes which would enable richer verifications. A language can also be designed to express constraints in purely declarative form as illustrated the case study and compile it using our emerging toolbox. In a larger context, such requirements could be combined with other temporal requirements and a mixed verification could be conducted at the system level, e.g. to show that some global intervention time is guaranteed in all possible scenarios. Finally, we can also incorporate stochastic TTF, to be able to more easily manage requirements with a more complex and accurate probabilistic formulation.

# ACKNOWLEDGEMENT

This work was financially supported by the Walloon Region by the REDIRNET project (nr 607768). We thanks Market-IP for sharing their fully featured maps and Raphael Michel for sharing his expertise on the Belgian emergency medical care.

# REFERENCES

Bast, H. et al. (2014). Route Planning in Transportation Networks. Technical report, Microsoft Research.

Batz, G. V., Delling, D., Sanders, P., and Vetter, C. (2009). Time-dependent contraction hierarchies. In *in Proc. 11th Workshop on Algorithmic Engineering and Experiments (ALENEX)*, pages 97–105. SIAM.

Batz, G. V., Geisberger, R., Neubauer, S., and Sanders, P. (2010). Time-dependent contraction hierarchies and approximation. In *Experimental Algorithms*, pages 166–177. Springer.

Byrd, A. et al. (2015). OpenTripPlanner. http://www.opentripplanner.org.

Dijkstra, E. W. (1959). *A note on two problems in connexion with graphs*, volume 1. Springer.

Federal Public Service for Health (2007). Vade-mecum for Emergency Medical Aid (Law of July 8, 1964) for ambulance services and ambulance rescuers (in French).

Finkelstein, A. and Dowell, J. (1996). A comedy of errors: The london ambulance service case study. In *Proc. of the 8th Int. Workshop on Software Specification and Design*, IWSSD '96, pages 2–, Washington, DC, USA. IEEE Computer Society.

Geisberger, R., Sanders, P., Schultes, D., and Delling, D. (2008). Contraction hierarchies: Faster and simpler hierarchical routing in road networks. In McGeoch, C. C., editor, *WEA*, volume 5038 of *Lecture Notes in Computer Science*, pages 319–333. Springer.

Geisberger, R. and Vetter, C. (2011). Efficient routing in road networks with turn costs. In *Experimental Algorithms - 10th International Symposium, SEA 2011, Kolimpari, Chania, Crete, Greece, May 5-7, 2011. Proceedings*, pages 100–111.

Hart, P. E. (1968). A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100107.

Haynes, R., Jonesaa, A. P., Sauerzapf, V., and Zhao, H. (2006). Validation of travel times to hospital estimated by GIS. *International Journal of Health Geographics*, 5(40).

Karich, P. et al. (2015). GraphHopper Routing Library. https://graphhopper.com.

Kieritz, T., Luxen, D., Sanders, P., and Vetter, C. (2010). Distributed time-dependent contraction hierarchies. In *Experimental Algorithms, 9th Int. Symposium, SEA 2010, Ischia Island, Naples, Italy, May 20-22*, pages 83–93.

Kontogiannis, S. C. and Zaroliagis, C. D. (2014). Distance Oracles for Time-Dependent Networks. In *Automata, Languages, and Programming - 41st Int. Colloquium, ICALP 2014, Copenhagen, Denmark, July 8-11*, pages 713–725.

Letier, E. and van Lamsweerde, A. (2004). Reasoning about partial goal satisfaction for requirements and design engineering. *SIGSOFT Softw. Eng. Notes*, 29(6).

Lin, H. E. and Zito, R. (2005). A review of travel-time prediction in transport and logistics. In *In Proceedings of the Eastern Asia Society for transportation studies*, volume 5, pages 1433–1448.

Moore, D. (2002). GIS in Emergency Planning, a spatial analysis of ambulance services.

OSRM Community (2015). Open source routing machine. http://project-osrm.org.