MASTER'S THESIS

Time-dependent routing with contraction hierarchies

Author

J.H. Hesseling

August 2023

Supervisors University of Twente

A. Trivella (1st supervisor)

E. Lalla (2nd supervisor)

Supervisor Simacan

B. Toersche

UNIVERSITY OF TWENTE.

Drienerlolaan 5

7522 NB Enschede

The Netherlands

simacan

Valutaboulevard 16

3825 BT Amersfoort

The Netherlands

Management summary

Introduction

This research on time-dependent routing is conducted for Simacan, a company that helps its customers by providing a platform that gives insights into the delivery performance within supply chains. In this research, the term routing is used to describe the process of finding the shortest path between a start and an end location (nodes) on a map (graph). The contraction hierarchies algorithm that Simacan uses to determine the fastest route of a vehicle from the start to the end location is time-independent (TID), meaning the time it takes to drive a road segment (edge) is assumed to be static. However, in reality, the time it takes differs per time of the day and day of the week. Time-independent routing, therefore, results in inaccurate routes visualized on the platform, inaccurately estimated times of arrival (ETAs), and ultimately customer dissatisfaction. This leads to the following research objective:

"Find an extension to the contraction hierarchies method that supports time-dependent routing."

Current situation

By conducting interviews and meetings with Simacan employees, it is observed that routing is the core of the Simacan platform since all customers use routing in one way or another. Routing within the platform is based on map data (such as speed profiles), real-time and historic traffic data. The calculated routes are used in for example route visualizations and ETA calculations. The ETAs are used in different applications such as to see whether the delivery of the truck is within the time-window promised. Given how frequently routes are calculated, the newly developed time-dependent (TD) method must be fast and able to deal with large road networks.

Literature review

First of all, contraction hierarchies (CH) is a routing algorithm that consists of two phases to solve the shortest path problem. The first phase is the pre-processing phase, in which shortcuts are created to reduce the search space. During the second phase, the query phase, bidirectional search is used (based on the shortcuts from the previous phase) to find the shortest path between the start and end location. Bidirectional search is a method in which the route is made by searching the shortest path from the start node and the end node simultaneously.

In the literature research, multiple approaches are found that incorporate time-dependency in CH. Some are exact methods and others are heuristics. Between the two types, there is a trade-off between the relative error on the one hand, and on the other the preprocessing times, query times and index size. Based on the comparison of the approaches and applicability to the Simacan context, the approach chosen is time-dependent sampling (TDS).

TDS is a heuristic that finds an approximation of a time-dependent route, by searching for multiple time-independent routes for every different time interval. These routes are then combined into a subgraph on which the time-dependent route is calculated. What and how many time intervals are chosen affects how close to optimality the time-dependent sampling heuristic performs. Limited research is conducted on the number of time intervals to use in TDS and how the time intervals are chosen. Therefore, the contribution of this research is to create a method to determine how many intervals and what time intervals should be used to avoid user-defined or random time intervals.

Model description

In this research, TDS is combined with CH into TDS-CH. The six general steps of TDS-CH are (1) decide on the number and width of time intervals, (2) average the travel time for each edge for each time interval, (3) create a time-independent graph for each time interval using contraction hierarchies, (4) run time-independent routing on each time-independent graph, (5) form a subgraph by taking the union of all time-independent routes, and (6) run time-dependent Dijkstra (which is an exact method) on the subgraph edges with time-dependent edge costs. In this 6-step process, steps 1 to 3 are part of the preprocessing phase, which is only conducted when updating the map on which the routes need to be found. Steps 4 to 6 are part of the query phase, which is used every time a route is requested.

The research gap is worked out by designing the interval selection method. The method consists of four steps, namely: (1) obtaining speed profile information, (2) creating the weighted speed profile, (3) dividing the weighted speed profile into periods of similar speed, and (4) selecting the most diverse time intervals. To complete the model, the interval selection method is added as the new interval selection phase before the preprocessing and the query phase.

Experimental setup

Experiments are set up to test the performance of the newly created interval selection method and to find the best number and width of intervals to use in the TDS-CH method. All experiments are conducted on the same dataset, consisting of start and end locations in the Netherlands from the Simacan database split up into four operation types: 1. *retail operation* (DC to shop), 2. *Post/parcel operation* (DC to DC), 3. *Home delivery operation* (hubs and houses in urban areas), and 4. *Random locations*. Each operation consists of 250 combinations, resulting in 1000 scenarios. Next, these scenarios are run every 10 minutes in each hour between 02:00 and 22:00 resulting in 120 departures per scenario, therefore 1000 * 120 = 120.000 routes per experiment. The number of intervals in the experiments varies between 2 and 9. Apart from experiments with different intervals, the way of selecting intervals is also experimented with. Two variations are used, namely the interval selection method and an approach that uses equal width intervals throughout the day. This results in $8 * 2 = 16$ experiments. Next to the 16 experiments, two benchmarks are created, namely the current TID algorithm and the exact time-dependent Dijkstra algorithm. Furthermore, several KPIs such as preprocessing, average query times, and travel times are formulated.

Results & conclusion

Comparing the different experiments with each other and with the time-independent and exact benchmark based on the KPIs leads to various insights and results. Firstly, the results show that for almost 90% of the scenarios, time-dependency is relevant, therefore showing the potential of time-dependent routing. Secondly, implementing TDS-CH results in a longer preprocessing time and more required preprocessing space but the increase is linear and the preprocessing can be parallelized. Thirdly, TDS-CH with 9 time intervals also takes on average 2.6 times more time for a single query than the time-independent benchmark but is on average 9.9 times faster than the exact benchmark.

Fourthly, the average optimality percentages, which shows the potential travel time difference between the time-independent and the exact benchmark, of the experiments showed promising results. The experiment using 9 time intervals that are found with the interval selection method results in an average optimality percentage of 92.14%. On the contrary, the experiment using 9 equal width intervals results in an average optimality percentage of 78.67%. Not only for 9 time intervals but also for all other experiments with a different number of time intervals used, the interval

selection method outperforms the equal width interval selection approach. Other KPIs, such as the average subgraphs size and the number of different time-dependent routes found, also confirm that the interval selection method always outperforms the equal width interval selection approach.

The decision on how many time intervals to use is more subjective. The tradeoff between the query times and average optimality percentage shows that based on the used dataset, 4 or 8 time intervals should be used depending on the application of TDS-CH. In these cases, the average optimality percentages are 78.04% and 92.13% respectively, with average single query times of 0.11 and 0.19 seconds. The decision should be made based on what additional query time is acceptable to further increase the average optimality percentage. This research fulfills the research objective of finding an extension to the contraction hierarchies method that supports time-dependent routing and shows that time-dependent routing adds value compared to time-independent routing.

Recommendations & discussion

The four recommendations for Simacan are as follows: implement the TDS-CH method to include time-dependency in their route calculations, experiment with 9 intervals or more on a larger dataset, implement the in the research left out Simacan-specific factors in TDS-CH, and check the predicted routes of the TDS-CH to the realized routes of customers.

The contribution of this research to theory is the well-performing newly created interval selection method, which makes the choice of time intervals no longer user-defined but fully data-driven. The limitations of this research include the lack of accurate computational times because of the code and machine used, which limits the size of the dataset. Another limitation is the number of design choices that are made in the interval selection method. This is also a suggestion for future research as other design choices could be experimented with in order to reach even better results. Finally, another topic for further research is the implementation of the interval selection method in other routing techniques.

Acknowledgments

This finished research marks the end of my thesis project at Simacan and my time as an Industrial Engineering & Management student at the University of Twente. I worked on this research for the last 12 months and looking back I think achieved a lot. I think I conducted a well-structured and relevant research that not only helped Simacan with making their routing time-dependent but also with the contribution of this research to science. Conducting this research has not always been easy for me but in the end, I can say that it was worth it and that I am proud of the result.

There are several people I would like to thank for helping me during this research. First of all, I want to thank my girlfriend Eveline who not only supported me mentally but also helped me with brainstorming, proofreading and providing valuable insights throughout the entire thesis writing process. This has been a great help. Secondly, I want to thank my 1st UT supervisor, Alessio Trivella, for his guidance, insightful discussions, feedback and comments. His expertise and eye for detail made it possible to improve the overall quality of my research. Thirdly, I would like to thank my 2nd UT supervisor, Eduardo Lalla, for his critical evaluation of my work and his ability to find improvements at a high level. Fourthly, I thank Bart Toersche, my supervisor at Simacan, for his valuable support and endless assistance with my programming work, which had a great contribution to the results and completion of this research. I also want to thank my friends, family and other colleagues at Simacan for their support and interest in my work. Lastly, I want to thank Simacan for providing me with the opportunity to do my thesis at their company.

Now that I have finished not only my research but also my time as a student, I think that I can proudly say that I learned a lot and have become a better version of myself as a person. For now, I am looking forward to what the future will bring.

List of tables

List of figures

Abbreviations

ATA	Actual Time of Arrival
ATCH	Approximated Time-dependent contraction hierarchies
CH	Contraction hierarchies
DC	Distribution Centre
ETA	Estimated Time of Arrival
FIFO	First In, First Out
FMS	Fleet Management System
FRC	Functional Road Class
GPS	Global Positioning System
KPI	Key Performance Indicator
PTA	Planned Time of Arrival
SSSP	Single Source Shortest Path
TCH	Time-dependent contraction hierarchies
TDS	Time-dependent Sampling
TDS-CH	Time-dependent Sampling with contraction hierarchies
TDSPP	Time-Dependent Shortest Path Problem

Definitions

Bidirectional search	An algorithm that solves the shortest path problem that searches simultaneously from start node s on the original graph and from end node t on the reversed graph. When a node is both visited by the forward search and the backward search, the algorithm is stopped and the shortest path between s and t is obtained.
Contraction hierarchies	A routing algorithm that consists of two phases that solves the shortest path problem. The first phase is the pre-processing phase, in which shortcuts are created to reduce the search space. During the second phase, the query phase, bidirectional search is used (based on the shortcuts from the previous phase) to find the shortest path.
Cost profiles	Penalties for specific route types. This penalty is used to calculate the travel time which is used in the routing. The penalty depends on how comfortable the road is for driving and depends on the vehicle type and road type.
Dijkstra's algorithm	A routing algorithm to solve the shortest path problem to optimality. It visits all available nodes and therefore takes long query times on large maps.
Interval selection method	The 4-step method designed in this research that finds time intervals based on a given map that can be used in the TDS-CH method.
Pre-processing phase	A first, and relatively long, phase of the contraction hierarchies method in which shortcuts are derived for combinations of edges that seem irrelevant because of the high costs of the combined edges compared to other options. This reduces the graph and therefore query time. This phase is conducted only once in a long time.
Query phase	The second phase of the contraction hierarchies method in which bidirectional search or Dijkstra's algorithm is used to find the shortest path. The query phase is conducted whenever a route is requested.
Shortest path problem	The problem of finding a path between nodes (points) on a graph (map) that minimizes the sum of all edge (roads) costs in the graph.
Speed profiles	A speed profile is a representation of a speed value for every road segment for all times of day, every day of the week and is based on anonymously gathered GPS data.
Time-dependent routing	Finding the shortest path between a start and an end location where the time of the day and day of the week is taken into account.
Time-dependent sampling	A heuristic to find the shortest path based on *k* time intervals. For each time interval, a time-independent route is added to a subgraph. An exact routing method is used to find the best route on that subgraph.

Contents

1 Introduction

In this thesis, research on including time-dependency in routing using contraction hierarchies at Simacan in Amersfoort is done. In this research, the term routing is used to describe to process of finding the shortest path between a start and an end location. This research is conducted as part of the graduation assignment for the master's program Industrial Engineering and Management at the University of Twente.

This chapter is constructed as follows. Section 1.1 introduces the company Simacan, the relevance of the research topic is described in Section 1.2, and the problem statement is given in Section 1.3. Furthermore, the research objective is described in Section 1.4 and the research approach is described in Section 1.5.

1.1 Introduction to Simacan

Simacan is a young but rapidly developing company based in Amersfoort. Simacan created an intelligent platform that helps its customers to achieve high delivery performance within supply chains. It gives their customers insights into their complete delivery process, for example, from the bakery to the distribution center and up to a consumer's home. This platform helps all parties in the delivery process to work together and achieve a common goal: a good customer experience.

Simacan strives for a high delivery experience not only for customers such as Ahold, but also for the customers of their customers (such as a consumer that orders their weekly groceries from Ahold). Important is that Simacan's customers are continuously informed about the estimated time of arrival (ETA) at distribution centers, shops, and consumers' homes. This is achieved by linking all transport management systems and IT systems of all parties to the Simacan platform (GPS location, traffic information, etc.). The platform adds real-time information like traffic conditions and optimal access routes to final destinations to create "one source of truth." In this way, when a truck is stuck, all relevant parties will instantly be updated on the situation. This creates time to come up with actions that can help to "save" the delivery. Currently, the Simacan platform already supports 100.000 deliveries daily. The goal is to further develop the platform in the upcoming years and to support more deliveries.

1.2 Relevance

In the last decade, delivery in food retail is growing rapidly and this growth is still expected to continue in the coming years. Amongst other software solutions, Simacan can provide products that can help customers like Jumbo and Ahold with the delivery of groceries to consumers' houses. Simacan uses routes (in particular shortest paths) for many applications. For example, when travel times need to be predicted, the routes should be known first. Routes are therefore critical in many applications of the software of Simacan. For example, during a scenario where a truck is in the process of delivering groceries for Ahold, there might be situations where the estimated time of arrivals (ETAs) for certain stops along the route, as determined by Simacan's predicted routes, fall outside the prearranged time window agreed upon for home deliveries. This could occur due to unexpected and unforeseen circumstances. In that case, operations could decide that an extra (or half-empty) truck close by can be used to help the delayed truck by taking over some deliveries from that truck. These cases are supported by Simacan through its software. To quickly recalculate ETAs, it is critical to have a fast algorithm to determine the routes.

The example illustrates that it is very important for employees that work for the planning & operations departments of Simacan's customers to clearly see when and where a potential problem can occur. A good understanding of the traffic situation is therefore important.

Simacan uses contraction hierarchies (CH) to efficiently calculate shortest paths. Contraction hierarchies is an algorithm that consists of two phases that are used to solve shortest path problems. The first phase is the pre-processing phase in which shortcuts are created to reduce the search space. This phase takes relatively much time. In the second phase, the query phase, bidirectional search is used (based on the shortcuts from the previous phase) to find the shortest path.

Before using contraction hierarchies to find the shortest paths, Simacan used different approaches such as bidirectional Dijkstra and after that bidirectional A*. These approaches were however not fast enough when the graph becomes too big, especially with routing in Europe. In the last couple of years, real-time rerouting became more important to better serve the wishes of the customers. And since contraction hierarchies work fast, using this technique is important.

The current implementation of contraction hierarchies on the current Simacan platform supports multiple cost profiles. However, the cost per link is static. With a static cost per link, it means that for every road segment, only one value is used during the routing. In real life, however, the costs per link are not static but vary depending on the time of the day and day of the week (e.g., during rush hour, delays are expected to be bigger than during the night). The routing of Simacan is therefore currently not time-dependent. The route corresponds therefore supposedly not to the optimal in reality at some times of the day. Because of the potential suboptimal route, the ETAs will also not correspond with reality. This means that the information on the Simacan platform of a customer is not always accurate.

At the moment, this time-dependency is not included in the routing with contraction hierarchies and therefore the routing is not optimal. As an effect, the customers of Simacan are presented with routes that the driver is not likely to take in some cases (for example in rush hour) and have therefore not the correct information. This can lead to inadequate actions being taken or that no action is taken when this could be useful.

The drivers of a truck do not receive an actual route from Simacan that they should drive. They decide for themselves how they are going to drive (based on experience or navigation systems like TomTom and Google Maps). Simacan tries to predict the route a driver takes (and corrects this route based on GPS data and when a driver is too far away from the expected route). This route is essential for obtaining accurate ETAs, among other objectives. For retailers, it is very important to see what we think where the driver is going to drive, so they can easily inform the driver about potential problems.

1.3 Problem statement

As the previous examples and context illustrate, it is very important to get the most accurate routes in the Simacan software. At the moment, both Simacan but also their customers know that due to not including time-dependency, the routes are not always optimal. For example, during rush hour sometimes routes are shown that are clearly not the fastest (based on for example experience). This is a problem and is difficult to explain to the customer. Also, Simacan is always trying to improve, and this is a known flaw.

The available data for Simacan that can be used to include time-dependency in their processes are speed profiles from a well-established map provider. Speed profiles represent average speeds for all

road segments for all times of the day on each day of the week. Speed profiles are therefore considered dynamic as opposed to the currently used average speeds per road segment which are static. Currently, Simacan uses speed profiles to calculate travel times for routes and therefore the travel times are time-dependent. However, route calculations are not based on speed profiles and therefore time-independent. The current routing process of Simacan is explained in more detail in Chapter 2.

The fact that calculated routes depend on static information has several effects. Other Simacan products like the time-distance matrix which can be used as an input for creating a planning can be improved by making routes time-dependent. Furthermore, there is an increased chance of determining a suboptimal route, ETAs are not accurate, and the chance that the route does not comply with the driver's route is increasing. All these effects of routes being calculated based on static information lead to customer dissatisfaction. Therefore, the core problem of not using speed profiles in route calculations (light blue) ultimately results in the action problem of customer dissatisfaction (dark blue). In Figure 1.1, all problems encountered and relevant to this research are visualized. The cause-effect relations are represented by arrows.

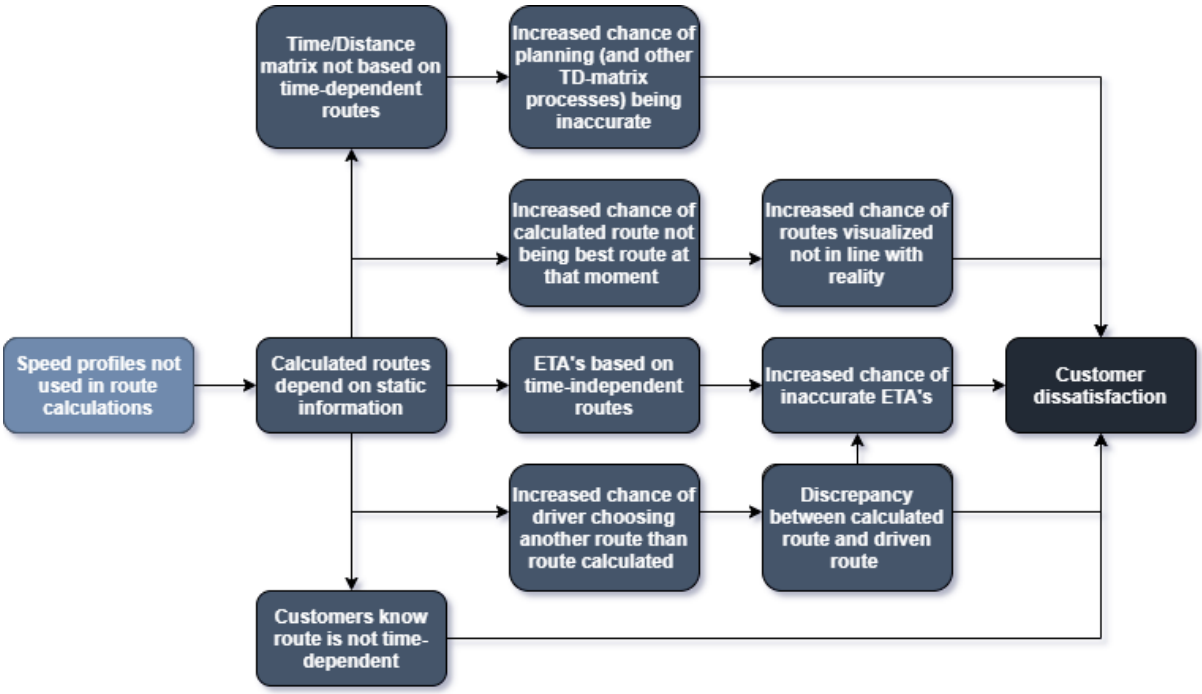


Figure 1.1 Problem cluster

From Figure 1.1 we can conclude that not using speed profiles in route calculations, which makes that routing time-independent, is the core problem that should be addressed. Including speed profiles in route calculations can be beneficial to multiple problems from Figure 1.1 and can in the end have a positive effect on the customer satisfaction. Taking this all into consideration, the problem statement is as follows:

"The software of Simacan lacks an algorithm including time-dependent routing, leading to a situation in which routes are suboptimal and ETAs are incorrect, and therefore customer dissatisfaction is higher than Simacan would like to."

1.4 Research Objective

The objective of this research is then defined as follows:

"Find an extension to the contraction hierarchies method that supports time-dependent routing."

When looking for ways on how these can be achieved two constraints play an important role. Firstly, the preprocessing phase should not take too much time. It already takes several hours. Secondly, the result of the preprocessing phase should still have a limited size, as it should fit in the RAM for the query phase.

One of the results of the research could be a model in Python that shows the suggested solution.

To achieve the research objective, a research approach is made. This plan states the research questions to gradually come to a solution. The plan of approach is discussed in the next section.

1.5 Research approach

The research approach consists of the different research questions and research sub-questions. This approach also structures the different chapters in this thesis. In this section, the research questions and the research sub-questions are introduced. In each chapter one research question and the underlying sub-questions are covered.

In Chapter 2 of this thesis, the current situation at Simacan is researched. The focus of this chapter is on how routing is used and managed by Simacan, understanding the problem context, and looking at other processes within Simacan that influence the research. This is done based on the following questions:

1) How does the current routing process look like within Simacan?
    a) How are the different components in the architecture related to each other?
    b) What is the exact use of routing for customers?
    c) How are routes currently created?
    d) How is the performance of the model measured?

To answer the questions above, information is gathered by interviewing Simacan employees from different departments and attending relevant meetings. Other ways for data gathering in this part of the research include reading relevant topics on Slab (an online platform where the knowledge database of Simacan is stored), analyzing previously written theses at Simacan and watching recordings of relevant topics of work that Simacan does.

The main goal of Chapter 3 is to learn about time-dependency in routing and especially in combination with contraction hierarchies. This leads to the following questions for this chapter:

2) How can literature regarding time-dependency in relation to contraction hierarchies be used to achieve the research objective?
    a) How are routing problems modeled in literature?
    b) What implementations of time-dependent routing can be found in the literature?
    c) How is time-dependency modeled in contraction hierarchies?
    d) How do the different contraction hierarchies methods including time-dependency compare to each other?
    e) What method is best applicable to the Simacan-specific context?

In Chapter 4, all knowledge obtained from the literature is used to create the model that is used to realize the research objective. Furthermore, the research gap of this research is addressed, which is a

method to determine the number and width of time intervals using time-dependent sampling. To do so, the following research questions are relevant:

3) How is time-dependent sampling constructed and extended?
    a) What assumptions are made?
    b) How can speed profile information be used?
    c) What is the logic of time-dependent sampling with contraction hierarchies?
    d) How can the time intervals for time-dependent sampling be determined?
    e) How are the time intervals included in time-dependent sampling?

In Chapter 5, the dataset and experiments used to test the performance of the interval selection method are explained. Furthermore, the KPIs to measure performance are presented. To do this, the following research questions are answered:

4) What experiments can be used to test the performance of the interval selection method?
    a) How is the dataset on which the experiments are conducted constructed?
    b) What variations are made in the different experiments?
    c) What indicators are used the measure the performance of the experiments?

In Chapter 6, the potential of using time-dependent routing is analyzed. Next, the results of the experiments are analyzed using the KPIs of the previous chapter and the best configuration is determined. This results in the following research questions:

5) What is the best configuration for the intervals that can be used for the TDS-CH method?
    a) What is the potential of using time-dependent routing on the dataset?
    b) What is the effect of time intervals on the preprocessing times and the required preprocessing space?
    c) How do the time intervals affect the creation time and size of the subgraph?
    d) How do the time intervals affect the query times and paths used?
    e) How close to optimality are the TDS-CH experiments in terms of travel times compared to the exact method?
    f) How can all the observed effects of the time intervals in the experiments be combined in selecting the best configuration?

Based on the previous chapters, conclusions and recommendations are formulated in Chapter 7. Finally, Chapter 8 contains the discussion.

2 Current situation

In this chapter, the current situation at Simacan is explained and the problem introduced in the previous chapter is explained in detail. The focus of this chapter is on how routing is used and managed by Simacan but also other processes within Simacan that are relevant for this research are touched upon. The first research question and the corresponding sub-questions are answered in this chapter. These are the following:

How does the current routing process look like within Simacan?

a) How are the different components in the architecture related to each other?
b) What is the exact use of routing for customers?
c) How are routes currently created?
d) How is the performance of the model measured?

To answer these questions, interviews and meetings with employees of Simacan were conducted. Other information has been gathered by analyzing data provided by Simacan such as previously recorded meetings and information available in the Simacan knowledge database. Each sub-question is discussed in a separate section and this chapter ends with a conclusion in which the answers to these questions are summarized.

2.1 The Simacan platform and its architecture

The Simacan platform is a smart collaboration platform and the core product of Simacan that allows stakeholders in time-critical delivery networks to provide customers with a 5-star delivery experience (Simacan, 2022). The Simacan platform can be used by different types of parties within the supply chain. These different roles in a supply chain include for example shippers and carriers. The platform is cloud-based which makes it easy to share and connect data and the platform can be used to manage multiple operations at the same time. Already used IT systems by customers of Simacan can be used alongside the Simacan platform and can be connected to the platform. APIs are used to exchange data between the platform and other data sources. The platform is offered to customers as a Software as a Service (SaaS) solution in which different options are available.

Different options are available since the wishes of customers and users differ. Therefore, the Simacan platform is customizable. Customers can start small and acquire more options and integrations when needed. This fits the current market where there are many different people/roles in a supply chain who need to work as one.

Within Simacan, routing is one of the many components that together make up the Simacan platform. In Appendix A, a complete overview of the Simacan component architecture is shown. This overview consists of many different components such as map data, planning and analytics. The most important component for this research is the routing component. In Figure 2.1, the components that are directly connected to the routing component are visualized.

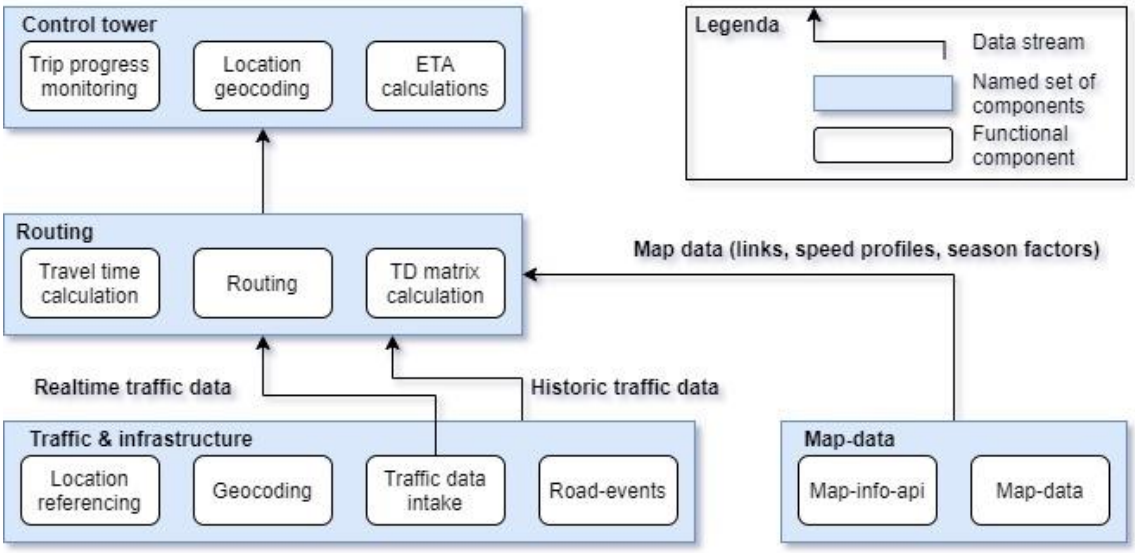


Figure 2.1: Routing component overview (Simacan, 2022)

2.2 Routing within the Simacan platform

The routing component of the Simacan platform consists of several services, namely: the actual routing, the travel time calculation and the time/distance matrix (TD-matrix) calculation. The important data streams for this component are real-time traffic data, historic traffic data and map data. In this research, the routing-related components are taken into account and the corresponding product owners are treated as a stakeholder.

As explained, the exact features of the Simacan Platform that the customer uses vary, however, routing is relevant for all customers since it is part of the main core of the Simacan platform which can also be seen in Figure 2.1 and Appendix A, where the component structure is shown. How the routing component is used differs per user and therefore in the next sections, a distinction between routing for Simacan and routing for customers is made.

2.2.1 Routing for Simacan

For Simacan, routing is very important as an input for several services and products. The most important and visible service that routing is needed for, is the travel time calculation on which the ETA is based. Other products/services for which routing is needed are calculating the first and last miles, the TD-matrix (which is both used by Simacan and sold to customers as input for planning) and the *Roads-API* which is used to look at traffic jams and matrix signs above the highway.

2.2.2 Routing for customers

Since routing is part of the main core of the Simacan platform, all customers of Simacan make use of routing in some way or another. Customers of Simacan profit directly from what Simacan does with the routing as explained in the previous section. Customers especially benefit from the ETAs. These ETAs are based on calculated routes and are updated every time a new location (GPS) is received and when a vehicle is stopped.

The expected routes that are calculated are shown in the Simacan platform. An example of a trip with multiple stops of a home delivery vehicle is shown in Figure 2.2. Figure 2.2 visualizes the order of the stops within a trip and shows the calculated routes between subsequent stops. Part of the corresponding stops of this trip is shown in Figure 2.3. In Figure 2.3 the bars on the right side indicate the status of an order. The blue color is the time window that was chosen by the person who ordered

the home delivery, the grey color shows the planned time of the delivery, and the green arrow shows the expected arrival (ETA). Insights into the expected routes can help customers to make interventions in their operations, like rescheduling certain stops to prevent delays or moving freight from one vehicle to another.

Figure 2.2: Example of the expected route in the Simacan platform

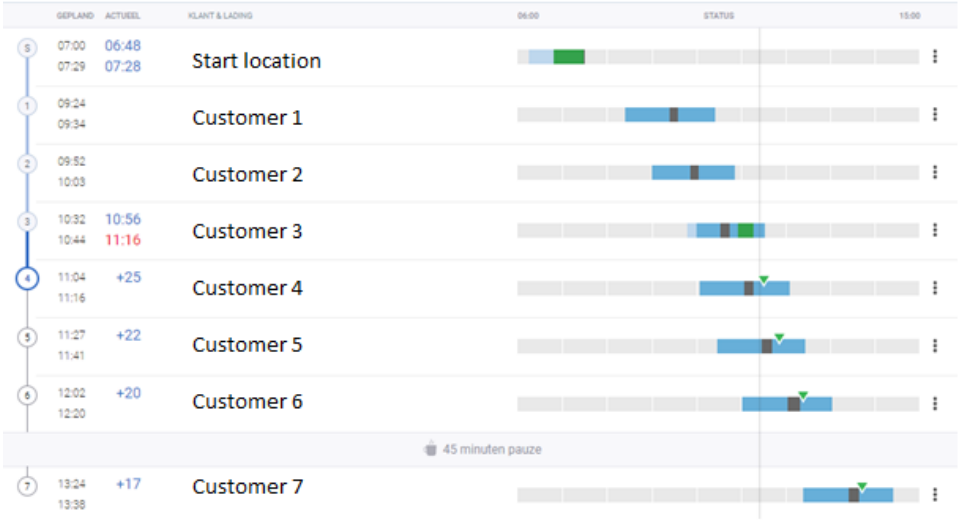


Figure 2.3: Stops in a trip

2.3 The current routing process

At the moment, the routing between two points is based on the contraction hierarchies algorithm. Contraction hierarchies consist of two steps. In the pre-processing step, the map (from a map provider) is changed in such a way that the search space becomes smaller and by doing so the actual routing can be done faster. Currently, the pre-processing step from contraction hierarchies is done only when a map update is available (which is a few times per year). The second step, the query step is the actual routing that gives the optimal routes as an output. In Chapter 3, contraction hierarchies are further explained. The second step, calculating the expected route, is done when one of the following situations apply:

1) The planning is received and imported.
2) A vehicle is located too far away from its expected routes. Too far away depends on settings set by a customer in consultation with Simacan.
3) The order of stops in a trip changes.

A new route calculation automatically triggers a recalculating of the travel time and corresponding ETA. The routing is dependent on several input parameters such as the chosen cost profile (the

vehicle type) and the latitude and longitude coordinates for both the source and target location. During the routing process restrictions such as the height of a vehicle are considered by setting the costs of relevant edges to infinity. When the cost of an edge is infinite, it can never be part of a route and the restriction is therefore upheld. Because of the many route calculations, a fast routing algorithm is necessary.

The routes between all stops in a trip are currently created simultaneously since the route between for example stops 3 and 4 does not depend on the route (and time taken) of stops 2 and 3 since time-dependency is not considered. It does for example therefore not matter if the previous route between two stops took place during rush hour since we expect the route to be the same (which is the research topic of this research).

2.4 Measuring route performance

The success of this research can be expressed in two ways. Firstly, this research is considered successful if the research objective is reached. To be more concrete, this research benefits Simacan by finding an extension to the contraction hierarchies method that supports time-dependent routing. Secondly, from the problem, cluster inaccurate ETAs, which are based on the travel time of traversing a route, are considered a big problem that would benefit from including time-dependency in routing.

To measure the actual progress and the effect of time-dependent routing, the following measuring method is used. Ideally, including time-dependency in routing would sometimes result in a route that is considered better than the time-independent route that is computed in the current situation between the same start and end point. We consider a time-dependent route to be better than a time-independent route when the expected travel time is shorter for the time-dependent route than for the time-independent route. This travel time reduction can be expressed as a percentage of the original travel time of the time-independent route and can by doing make progress measurable. How often routes differ between the current situation and the desired situation, where time-dependency is included in the routing, and what the average percentage of travel time reduction is, should be tested. These tests are performed on a dataset that consists of many start and end locations. In Chapter 5, where the experiments are explained, the exact calculations to measure performance are explained.

2.5 Conclusion

In this chapter, the research question *"How does the current routing process look like within Simacan?"* is answered. In this context, routing describes the process of finding the shortest path between a start and an end location. Combining the knowledge obtained in this chapter with the problem description from Chapter 1, this chapter provides a clear direction of what should be further researched. By conducting interviews and meetings with Simacan employees, it was observed that routing is the core of the Simacan platform since all customers use routing in one way or another. Routing within the platform is based on map data (such as speed profiles), real-time and historic traffic data. The calculated routes are used in for example route visualizations and ETA calculations. The ETAs are used in different applications such as to see whether the delivery of the truck is within the time-window promised. Given how frequently routes are calculated, the newly developed time-dependent method must be fast and able to deal with large road networks.

In the next chapter, fast routing algorithms that can deal with large road networks are researched. To do that, first the general concepts of routing and time-dependent routing are examined based on the literature. Also, the currently used contraction hierarchies algorithm is further researched and time-dependent variations of contraction hierarchies are looked at.

3 Literature review

In this chapter, extensive literature research is conducted. The literature research is centered around the research subject and provides an overview of existing methods and approaches for tackling related problems. Existing routing methods and especially methods that include time-dependency in routing are focused upon. Based on the findings, the research gap is defined. In this chapter, the following questions are answered:

How can literature regarding time-dependency in relation to contraction hierarchies be used to achieve the research objective?

a) How are routing problems modeled in literature?
b) What implementations of time-dependent routing can be found in the literature?
c) How is time-dependency modeled in contraction hierarchies?
d) How do the different contraction hierarchies methods including time-dependency compare to each other?
e) What method is best applicable to the Simacan-specific context?

In each of the following sections, one of the sub-questions is discussed. This chapter ends with Section 3.6, the conclusion, where the answer to the research question of this chapter is formulated.

3.1 Time-independent routing

In this section, the basic principles of routing using graphs, the shortest path problem and several models and algorithms are discussed. Furthermore, the relation between the models discussed in the literature and their applicability to the Simacan context is explained. In the next sections, the concept of routing and different time-independent routing techniques are discussed. These techniques can be seen as predecessors of contraction hierarchies, which is the algorithm that Simacan currently uses for routing. The contraction hierarchies algorithm is explained in Section 3.1.4.

3.1.1 Routing using graph theory

In this research, the term routing is used to describe the process of finding a path between two locations. These locations can be consumers of home delivery products but can also be distribution centers or hubs. The roads and locations form a network that can be modeled as a graph. In graph theory, a graph consists of two sets of elements, a set of n nodes and m pairs of nodes that form edges E (Saha Ray, 2013). Depending on the problem, edges can be directed or undirected. Each edge is the connection between two nodes and has a certain cost that is incurred if that edge needs to be used. In routing, the costs of edges are for example the expected travel time when that edge is used. The point where an edge is connected to other edges is called a node. All the edges E and nodes N form a network represented in a graph $G = (N, E)$ of a certain area.

One of the applications of graphs is routing problems. In routing problems, the roads are split up into road segments which can be modeled as edges. In routing, the graph is a directed graph because road costs between two nodes can be different in both directions. Also, an edge from node a to node b does not mean that there is also an edge from node b to node a since some road segments are one-way streets. A directed graph means that all edges in the graph have a direction (Bang-Jensen & Gutin, 2009). In this research, the weight of an edge is the time taken to travel between the two connected nodes and is assumed to always be a positive real number. Note that there are also other types of problems where non-negative edge weight can be considered but that is outside the scope of this research. If nodes are not directly connected by a single edge, a path is needed to travel from one node to another. A path is therefore a sequence of connected edges. Note that, even when two

nodes are connected by a single edge, a path consisting of more than one edge can have a lower cost than the single edge if triangular inequality does not hold. This is because the summed weights of the edges of the path in this case are lower.

The problem of finding the shortest path between a start node and the other nodes in a graph is called the single-source shortest path problem (Bang-Jensen & Gutin, 2009) and is explained in the next section.

3.1.2 Single-source shortest path problem

In this section, the single-source shortest path (SSSP) problem is explained. In routing, every edge between a pair of nodes has a certain cost c associated with it. For example, the cost of the edge $(a, b) \in E$ between nodes a and b can be denoted as $c(a, b)$. In the SSSP problem, the goal is to find paths from a certain start node in a graph to all other connected nodes, in which the sum of the costs in a single path is minimized (Ahuja et al., 1990).

There are numerous ways how the SSSP problem can be solved. In small graphs with only a few nodes and edges, finding the optimal path is not difficult and can be found by complete enumeration of all paths. However, when more nodes and edges are added to a graph, this becomes impossible since the number of paths grows exponentially with the network size. The SSSP problem can then be solved using numerous algorithms of which the Dijkstra algorithm is the best known (Ahuja et al., 1990). In the next sections, several algorithms used to solve the SSSP problem are introduced. The focus is on the algorithms Simacan currently uses and where these algorithms originate from.

3.1.2.1 Dijkstra's algorithm

In 1959, Dijkstra proposed a solution to the problem of finding a path with minimal length between two nodes in a graph. The solution found was an algorithm that consists of several steps, and works as follows (Dijkstra, 1959).

1) First, the distance from the start node s to all other nodes is initialized to infinity except for the distance from the start node s to itself, which is set to 0. Also, a set is created that is initialized with all nodes. This set contains all nodes that are not yet visited.
2) In step 2, a node from the unvisited nodes set that has at that moment the lowest tentative distance from the start node s to itself is chosen. For this chosen node c, the distance from to start node s via node c to each of the direct neighbors of node c is calculated. If for a neighbor of c, the calculated tentative distance is lower than the current saved tentative distance of the start node to the neighbor of node c, the shortest distance to that node is updated. To later be able to see what the shortest path is, node c should be saved as a predecessor of the neighbor with a short distance to start node s in the list. The last part of step 2 is removing node s from the unvisited nodes set.

Step 2 is repeated until a stopping criterion is reached. In the SSSP problem, the stopping criterion is reached when all nodes are visited. If only the shortest path between a start node s and an end node t is relevant, the stopping criterion is reached when the end node is visited. When all nodes are visited, the shortest distance from the start node s to every other node in the graph is known. The shortest path taken can be derived by backtracking the predecessors in the list. Dijkstra's algorithm can be used to solve the SSSP problem to optimality. The time complexity of the original Dijkstra's algorithm is equal to $O(N^2)$ which makes it a polynomial time algorithm. However, when the number of nodes increases, for example when we consider the road network of the Netherlands that consists of millions of nodes and edges, Dijkstra's algorithm takes a lot of computational time to solve the shortest paths in large graphs. Therefore, more advanced algorithms are needed.

3.1.2.2 Bidirectional Dijkstra

Bidirectional Dijkstra is a modification of the original algorithm by Dijkstra. In bidirectional Dijkstra, the original Dijkstra's algorithm is performed simultaneously from start node s on the original graph and from end node t on the reversed graph. Bidirectional Dijkstra is only used for single-source single-destination scenarios. When a node is both visited by the forward search and the backward search, the algorithm is stopped and the shortest path between s and t can be obtained from the gathered results (Dantzig, 1963). Compared to the original Dijkstra's algorithm, the search space is reduced and for networks, such as road networks that are considered in this research, approximately half of the nodes are visited before the optimal path between s and t is found (Bast et al., 2015). On its own, bidirectional Dijkstra is not fast enough the solve routing problems in large networks but it is commonly used in combination with other speedup techniques that are discussed in the next sections (Sanders & Schultes, 2007).

3.1.3 Hierarchical structures in routing algorithms

Many speedup techniques use the concept of hierarchical structures. The idea behind a hierarchical structure is that in a longer path (among many edges) the path converges to more important roads. For routing algorithms, this means that when the path converges there will be only looked at the edges that represent the more important roads (Bast et al., 2015). The highest level of the hierarchy represents roads such as highways whereas the lowest level in the hierarchy represents roads in small neighborhoods. For short distances roads of a lower hierarchy are used but when the distance between two locations increases, roads of higher hierarchy such as highways will be used most of the time, except close to the start and end location. Concerning graph theory, this means that only certain edges are likely to be used and that only a small portion of all nodes and edges are relevant for this routing problem. In the next sections, Highway Hierarchies (HH) and Highway Node Routing (HNR), which are predecessors of contraction hierarchies (CH), are explained.

3.1.3.1 Highway Hierarchies

The previously described algorithms were directly applied to a graph. These led to good results for small graphs but for larger graphs that represent large road networks such as the Netherlands, the previously mentioned algorithms are insufficient because the search space is too big. To achieve fast results, Highway Hierarchies (HH) consists of two steps: the "contraction" step (or preprocessing step) and the "query" step (Sanders & Schultes, 2006).

1) In the contraction step, the original graph is transformed into a new graph with different layers that represent the different levels of the hierarchy that depend on the distance edges from the start and end nodes. These different hierarchical levels are computed by the algorithm itself based on a chosen neighborhood parameter (Delling et al., 2009).
2) In the query step, the shortest path is searched for with bidirectional Dijkstra on the new graph where edges are only expanded if they are close enough to the start and end node (Delling et al., 2009).

This exact method finds optimal shortest paths and has as an important advantage in that the search space is largely reduced in the pre-processing step and that the search will therefore be much faster (Sanders & Schultes, 2007). A disadvantage is that a change in cost for a certain edge results in that the entire preprocessing step should be redone, which is time-consuming (Schultes & Sanders, 2007).

3.1.3.2 Highway Node Routing

Highway Node Routing (HNR) continues on the idea of a separate preprocessing step and query step from the HH approach, but this technique can deal with small edge weight changes which make this technique dynamic (Schultes & Sanders, 2007). An edge weight change, like when a traffic jam occurs

on a certain road segment, causes the preprocessed information to be updated, instead of recalculated, which saves computation time. This approach is however not suited for a time-dependent application where the weights of all edges change depending on the time of the day (Schultes & Sanders, 2007).

3.1.4 Contraction hierarchies

Geisberger et al. (2008) extended the concept of node contraction from HH and HNR and created the contraction hierarchies (CH) algorithm. Like the previously described methods that include hierarchical structures, CH consists of two main steps which are the preprocessing step and the query step. In contrast to HH and HNR, every contracted node belongs to a unique hierarchical layer in the new graph which represents the importance of each node (Geisberger et al., 2008). By using CH, the search space is reduced which results in less computational time and caused the query time to be 5 times faster than other hierarchical speed-up techniques such as HH and HNR at that time (Geisberger et al., 2008). These advantages are the reason that Simacan uses CH in their routing operations and therefore the two steps of the algorithm are explained in more detail below:

Preprocessing step:
Firstly, all nodes in the graph $\mathrm{G} = (\mathrm{N}, \mathrm{E})$ are ordered. There are multiple ways to order nodes, and the order of the nodes is important because the order has a substantial effect on the length of the preprocessing step. The order has however no effect on the correctness of the algorithm (Geisberger et al., 2008). Secondly, iteratively all nodes are contracted based on this order. In each iteration, the nodes are reordered and when this is done the next non-contracted node is selected to be contracted. When a node is contracted, the node and the connected edges are removed from the graph and a shortcut is created and added to the graph if necessary. A shortcut is created if the removal of a node causes a current shortest path in the neighborhood of the removed node to be removed. Whether or not the removed node and edges are part of the shortest path is checked using a local bidirectional Dijkstra algorithm. A newly created shortcut is also added to the overlay graph G^*. The graph G^* is used in the query step. If this is done, the next node can be contracted until all nodes are contracted.

Query step:
When all nodes are contracted, the preprocessing step is completed and the shortest routes can be found using the query step. In the query step, the bidirectional Dijkstra algorithm is used. A forward search is conducted on the upward graph starting at the start node s and a backward search is conducted on the downward graph starting at the end node t. A restriction is that in the forward search, only nodes of a higher hierarchical level can be used and in the backward search only nodes of a lower hierarchical level can be used.

3.1.5 Other speed-up techniques

Apart from the techniques mentioned in this paper up to now, there are other speed-up techniques. Sanders and Schultes (2007) provide an overview in which also goal-oriented speed-up techniques are mentioned that in general focus on reducing the search space by focusing on the goal. In this research, these techniques are considered to be outside the scope since the focus is on techniques that work with contraction hierarchies.

3.2 Time-dependent routing

Until this point in this chapter, all approaches and algorithms described depend on static information. This means that the weights of the edges in a graph are fixed and do not change depending on the time of the day the shortest/fastest route is needed for. As introduced in chapters

1 and 2 this does not fit the way Simacan wants to deal with routing. Therefore, in this section, we have looked at different approaches and models where time-dependency is taken into account in routing.

In time-dependent routing, travel times, so the costs of the edges, may vary over the planning horizon. In this research, the variation in travel times is considered to be known beforehand and is based on historical data.

3.2.1 Time-dependent Shortest Path Problem

The problem looked at in this research is referred to by different names such as time-dependent Quickest Path Problem (TDQPP) and point-to-point shortest path. In this research, the problem where the shortest path between two points is computed and where time-dependency is included is referred to as Time-Dependent Shortest Path Problem (TDSPP) which is the same as the TDQPP.

Gendreau et al. (2015) state that, when including time-varying travel times, each edge $i, j \in E$ in a graph $G = (N, E)$ has an associated function $\tau_{ij}(t)$ that represents the travel time between two points i and j when entering the edge at point i at time t**.** The total travel time $z_r(t)$ of a route r that goes through nodes $n_1, n_2, \dots, n_e$ where n_1 is start node and n_e is the end node, can then be calculated recursively as follows:

$$z_{(n_1,\dots,n_i)}(t) = z_{(n_1,\dots,n_{i-1})}(t) + \tau_{(n_{i-1},n_i)}(t + z_{(n_1,\dots,n_{i-1})}(t))$$

$z_{(n_1,n_2)}(t)$ is initialized as $\tau_{(n_1,n_2)}(t)$

3.2.1.1 The FIFO-property

An important assumption in time-dependent modeling is the FIFO-property. The FIFO-property is a non-overtaking property which means that on all edges $i, j \in E$ if a vehicle x_1 leaves from node i at time t_1 and a vehicle x_2 leaves from node i at time t_2 where $t_1 < t_2$, it is impossible for x_2 to arrive earlier at node j than that x_1 arrives at node j when using edge i, j. The TDSPP would be NP-hard in a network without the FIFO-property whereas in a network with the FIFO-property, TDSPPs are polynomially solvable (Gendreau et al., 2015).

3.2.1.2 Different queries

Batz et al (2013) distinguish two, for this research relevant, time-dependent routing queries which are the earliest arrival (EA) queries and profile queries. In EA queries, the earliest arrival time and the corresponding route are computed based on the start location, end location and departure time. In profile queries, a travel time profile for a start and end location is computed based on a departure time interval. This profile query is especially useful when a good departure time needs to be chosen (Batz et al., 2013)**.** In the Simacan case, both queries are relevant. In Chapter 6, both queries are experimented with.

3.2.2 Solving the TDSPP

Research on the TDSPP is not quite as extensive as on the time-independent variant. Most research is focused on making adaptions to time-independent techniques to solve the TDSPP. In this research, we consider the TDSPP where the travel times are known before the route is calculated. In the next sections, some techniques that were previously explained as predecessors of CH and have a time-dependent variant solution are introduced.

3.2.2.1 Time-dependent Dijkstra

The Dijkstra algorithm described in Section 3.1.2.1 can be extended to a variant that can deal with time-dependent lengths of edges by using known travel time functions (Dreyfus, 1969). Just like the

original Dijkstra, this algorithm is, on its own, too slow and requires too much memory. Therefore, time-dependent Dijkstra on its own is not suited for large road networks (Van den Eynde et al., 2020). In Section 3.4, a comparison between the time-dependent Dijkstra and other time-dependent techniques is shown.

3.2.2.2 Time-dependent Bidirectional search

To speed up queries on time-dependent networks, a time-dependent bidirectional search can be used. The main problem in bidirectional search is that for the backward search, the arrival time at the destination is unknown. This unknown arrival time problem can be tackled in multiple ways such as by using lower bounds of travel times to make an initial smaller relevant graph that can later be explored by a forward search (Nannicini et al., 2012).

According to Gendreau et al. (2015), time-dependent route planning in road networks is not yet used because the algorithms used to solve the quickest path problem at the time of their research are too slow and use a lot of data (Gendreau et al., 2015). The above time-dependent methods are, although relevant, not in line with the current situation at Simacan. Therefore, including time-dependency in combination with contraction hierarchies is addressed in the next section.

3.3 Time-dependency in contraction hierarchies applications

In this section, different techniques that combine time-dependency with contraction hierarchies are introduced. The different time-dependent techniques will be compared in Section 3.4.

3.3.1 Time-dependent contraction hierarchies (TCH)

Batz et al (2009) developed an exact model that generalizes the original CH to a variant that uses time-dependent edge weight, called the time-dependent contraction hierarchies (TCH). Their model uses travel time functions (TTF) which are piecewise linear functions that have the FIFO-property. The TCH variant differs from the original CH in both the preprocessing and the query steps.

1) In the preprocessing step, the node order is often done similarly as in CH since it is expected that the importance of a node is not influenced by edge use. In the contraction phase of the preprocessing step, the decision of whether a shortcut is created depends on if the contracted node is part of a shortest path at all points in time. If so, a new shortcut is created and the new TTF of the new shortcut is then computed by chaining the TTF of the related edges.
2) Like time-independent CH, the query phase uses bidirectional Dijkstra. The forward search uses time-dependent Dijkstra and for the backward search, an approach like the one described in Section 3.2.2.2 is used.

However promising, this approach is considered to be not yet suited for real-world applications due to space consumption, preprocessing time, and query time (Batz et al., 2009). Therefore, Batz et al (2013) implemented TCH, this exact implementation is called KaTCH.

3.3.2 Time-dependent contraction hierarchies and approximation (ATCH)

Batz et al (2010) continued the TCH idea and developed a technique that uses an approximation of the travel time functions, but still has an exact solution. The travel time functions are represented by piecewise linear functions. This approximated time-dependent contraction hierarchies (ATCH) technique uses approximated shortcuts, based on exact time-dependent edge weights. The approximations are used to determine a list of interesting roads on which later a time-dependent search is performed. By doing so, much less space was needed and query exactness is still guaranteed, with the only drawback that query speed is a little slower. Despite a slower query speed

than TCH, ATCH is still fast. In the test instance of the German road network, exact time-dependent routes could be found in 2 milliseconds (Batz et al., 2010).

3.3.3 Inexact time-dependent contraction hierarchies (Inexact TCH)

Until now, only exact techniques were discussed. Next to these exact techniques, there are also heuristics that can be used when exactness is not guaranteed or possible, and a small error is allowed (Batz et al., 2010). Batz et al. (2013) developed an inexact version of TCH (Inexact TCH) that replaces the TTF with inexact TTF. Inexact TCH can be used for both EA queries and profile queries but is especially useful for profile queries because of realized speed-ups (Batz et al., 2013).

3.3.4 Time-dependent sampling (TDS)

Another heuristic is Time-dependent sampling (TDS). This heuristic was developed by Strasser (2017) and can be combined with contraction hierarchies to solve the TDSPP. Extensions on this heuristic made it possible to also solve the profile query (TDS+P) and the dynamic version (TDS+D) that could on-trip travel congestions. In TDS, a subgraph is computed using the following steps:

1) A constant number k time intervals are defined
2) In each time interval, an average of the time-dependent travel times for each edge is calculated and so a time-independent graph for each interval is created
3) For each time-independent graph, the shortest path is calculated using a speed-up technique such as CH
4) The union of the k time-independent shortest paths is the subgraph

When the subgraph is created, a time-dependent version of Dijkstra's algorithm is run on this subgraph. The relative error is calculated to show how the computed route differs from the optimal route computed with exact methods. To lower the error, more time intervals can be used. This has as a drawback that the preprocessing and query times become longer and more space needs to be used. Strasser (2017) recommends the use of time intervals below 10. The more time intervals are used the higher the chance that the optimal path is found. However, no matter the number of time intervals, the optimal path is never guaranteed because creating the subgraph still depends on static time-dependent travel times. For the experiments, Strasser (2017) used 9 time intervals, which is abbreviated as TDS+9. TDS is built upon CH but can also be used in combination with other techniques (Strasser, 2017).

3.3.5 Time-dependent contraction hierarchies through Unpacking (CATCHUp)

To overcome problems of the other two-phase approaches (preprocessing and query phase) such as slow running times and large index size, Strasser et al (2021) developed an approach called customizable approximated time-dependent contraction hierarchies through Unpacking (CATCHUp). In this technique, instead of storing the travel times at shortcuts, paths are stored at shortcuts. This overcomes the problem of the complex travel time functions of which computing and storing are expensive. The idea of storing paths, and obtaining travel times later, is based on the idea that paths do not change that often and that even when there is a lot of traffic on the highway, the highway is still faster than routes around it. Using this idea, CATCHUp stores not all breakpoints of the travel time functions but only the points in time when the fastest path changes (Strasser et al., 2021)**.** In Section 3.4, a comparison between the different time-dependent techniques is shown including the error of the heuristics is also included in Section 3.4.

3.4 Comparison of methods to solve TDSPP

In order to determine which method is best to use for Simacan, it is necessary to compare them. Strasser et al (2021) compared their CATCHUp technique to other time-dependent methods. For

CATCHUp, TD-Dijkstra, KaTCH and TDS+9, open code was available and, in that case, Strasser et al (2021) evaluated the models using the same map instances. TCH, ATCH and Inexact TCH had no open code available, so Strasser et al (2021) used numbers as reported in the different papers. The size of the map instances used in terms of nodes and edges in shown in Table 3.1.

Map instance	Nodes (*1000)	Edges (*1000)
Ger06	4688.2	10795.8
Ger17	7247.6	15752.1
Eur17	25758.0	55503.8
Eur20	28510.0	60898.8

Table 3.1: Size of map instances (Strasser et al., 2021)

In Table 3.2, a comparison of the different discussed methods is shown. The methods are compared based on preprocessing time, index size, query time and relative error (in case of heuristics). The queries run are earliest arrival queries. OOM means that the program crashed because it was out of memory. The value 1.0 in brackets behind ATCH and Inexact TCH means that a maximum value of 1.0 was allowed for these methods.

Map instance	Method	Preprocessing		Index	Query		
		Time (s)	Cores	Size (GB)	Time (ms)	Relative error Avg. (%)	Max. (%)
Ger06	TD-Dijkstra	-	-	-	719.26	-	-
	KaTCH	169	16	4.66	0.64	-	-
	TCH	378	8	4.66	0.75	-	-
	ATCH (1.0)	378	8	1.12	1.24	-	-
	Inexact TCH (1.0)	378	8	1.00	0.69	0.270	1.010
	TDS+9	542	1	3.61	2.07	0.001	1.523
	CATCHUp	52	16	1.06	0.72	-	-
Ger17	TD-Dijkstra	-	-	-	814.60	-	-
	KaTCH	859	16	42.81	1.26	-	-
	TDS+9	601	1	5.28	2.61	0.001	0.963
	CATCHUp	142	16	1.50	2.02	-	-
Eur17	TD-Dijkstra	-	-	-	2929.72	-	-
	KaTCH	3066	16	146.97	OOM	-	-
	TDS+9	3149	1	18.84	4.70	0.002	1.159
	CATCHUp	747	16	5.47	4.92	-	-
Eur20	TD-Dijkstra	-	-	-	3784.11	-	-
	KaTCH	7149	16	239.78	OOM	-	-
	TDS+9	3352	1	20.65	4.23	0.006	1.733
	CATCHUp	1249	16	6.32	5.60	-	-

Table 3.2 Method comparison, based on Strasser et al (2021)

With this comparison, it is shown that the CATCHUp technique can be very useful since it outperforms the other techniques on preprocessing time and index size. Also, the query time is very good. The TDS (with 9 time intervals) also looks promising if errors are allowed.

3.5 The chosen method for Simacan

In the previous section, a comparison between different methods regarding time-dependent routing was made based on preprocessing speed, index size, query speed and, in the case of heuristics, the relative error. This comparison showed that the CATCHUp technique can be very useful since it outperforms the other techniques on preprocessing time and index size. Also, the query time is very good.

However, the query speed of time-dependent sampling (with 9 time intervals) is comparable to that of CATCHUp and shows also very limited preprocessing time and index size and looks therefore very promising if errors are allowed. Additionally, the preprocessing time in TDS can be further decreased when the preprocessing of the different time-independent graphs with contraction hierarchies is parallelized. Another important advantage of time-dependent sampling is that time-dependent sampling is the most realistic to implement and experiment with because it is more of an extension of the current situation at Simacan than a complete revision of the current process. Taking this all into consideration, time-dependent sampling is chosen as the approach to reach the research objective. In the next chapter, the time-dependent sampling heuristic and how it can be used within Simacan is explained in more detail.

3.6 Conclusion

In this chapter, the research question *"How can literature regarding time-dependency in relation to contraction hierarchies be used to achieve the research objective?"* was answered. We can conclude that in the literature there are multiple ways described to include time-dependency in routing in general and that there are also approaches/models that incorporate time-dependency in contraction hierarchies. A distinction between exact methods and heuristics is made and it is shown that there is a tradeoff between the relative error and preprocessing times, query times and index size.

Based on the comparison between the different methods and their applicability to the Simacan context, time-dependent sampling was chosen as the approach to continue with and to build further upon. An important element of time-dependent sampling is the time intervals. Not much research has been performed on how time intervals are exactly chosen and with which time intervals time-dependent sampling performs best. Until now, this is done by manual trial and error. This research, however, contributes to the literature by further researching and experimenting with the time intervals and how the time intervals can best be used by a method instead of user-defined time intervals. In the next chapter, the time-dependent sampling heuristic and how it can be used in the Simacan environment is explained in detail.

4 Model description

Chapter 3 ended with the conclusion that time-dependent sampling is chosen as the method to extend the contraction hierarchies algorithm to make the routing time-dependent. In this chapter, the time-dependent sampling heuristic is explained in more detail. This chapter starts with the assumptions that are made in the solution approach in Section 4.1. In Section 4.2, the speed profiles, which contain important data necessary to include time-dependency are explained. In Section 4.3**,** time-dependent sampling is explained in detail and an explanation of how the heuristic can be adapted to fit the Simacan architecture and work with the available data is explained. In Section 4.4, a new method of determining time intervals that are used in time-dependent sampling is introduced. Section 4.5 shows how the new method of determining time intervals fits in the time-dependent sampling approach. This chapter ends with a conclusion that answers the following research question:

How is time-dependent sampling constructed and extended?

a) What assumptions are made?
b) How can speed profile information be used?
c) What is the logic of time-dependent sampling with contraction hierarchies?
d) How can the time intervals for time-dependent sampling be determined?
e) How are the time intervals included in time-dependent sampling?

4.1 Assumptions

In the remainder of this research, several assumptions are made. Firstly, we assume that to model time-dependency we can only use the speed profiles and map data from a well-established map provider. These speed profiles are the input for making the edge costs time-dependent. How these speed profiles exactly look like is explained in Section 4.2. Secondly, we assume that the FIFO-property, which was explained in Section 3.2.1.1**,** holds. Thirdly, we assume that all edge costs of the graphs used are positive real numbers, this is a requirement for the time-dependent sampling approach discussed in this chapter. Fourthly, only the map of the Netherlands is used due to hardware limitations. Lastly, Tuesday is chosen as the day to use in all examples regarding the method to select intervals. Tuesday is used because it is the day with the most speed fluctuations in the Netherlands. Using the day with the most speed fluctuations leads to finding the most extreme intervals which results in time-dependent sampling to perform as best as possible.

4.2 Speed profiles

Before going more into how time-independent graphs can be constructed, we first need to examine the data that can be worked with further. The data available to use to include time-dependency in routing are speed profiles**.** In this section, speed profiles are further explained and visualized.

4.2.1 Speed profiles explained

A speed profile is a representation of a speed value for every road segment for all times of day, every day of the week (TomTom, 2023). These speed profiles are based on anonymously gathered GPS data and are available in over 85 countries. A single speed profile consists of a list of speed profile percentages for every 5-minute interval in a day. Each percentage represents the speed as a percentage of the free flow speed which is the speed normally used and represent the speed if the vehicle is not hindered by other traffic. This means that each speed profile consists of a list of 288 percentages starting at 00:00 until 23.55. As an example, Figure 4.1 shows the percentages for speed

profile 47. From Figure 4.1 we can conclude that speed profile 47 has two major periods in time where the speed is significantly lower.

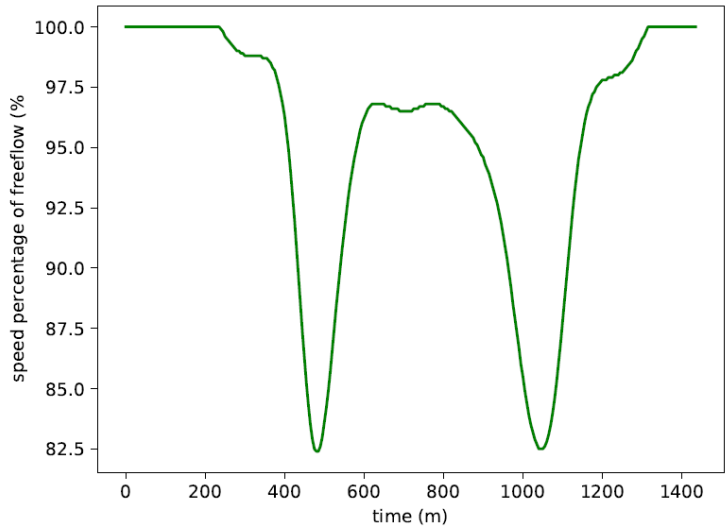


Figure 4.1: Speed percentages for speed profile 47

Analysis of the map of the Netherlands showed that there are 293 different speed profiles used. Every road segment in the graph that is time-dependent is matched to one of these 293 speed profiles. A road segment can have a different speed profile for every day of the week. The matched speed profile is used to calculate the exact speed in km/h depending on the time day and day of the week by multiplying the free flow speed of the road segment with the corresponding speed percentage from the matched speed profile. By using speed profiles in the calculation of speed, the road segments are now considered time-dependent as opposed to the currently used average speeds per road segment which are static and therefore not time-dependent.

The time-dependent speeds are visualized for a random road segment in Figure 4.2**.** The legend in Figure 4.2 shows the speed profile on which the speed per day of that road segment is based. For example, on Monday, speed profile 47 is used. From Figure 4.2 it can be included that five different speed profiles are used for this road segment.

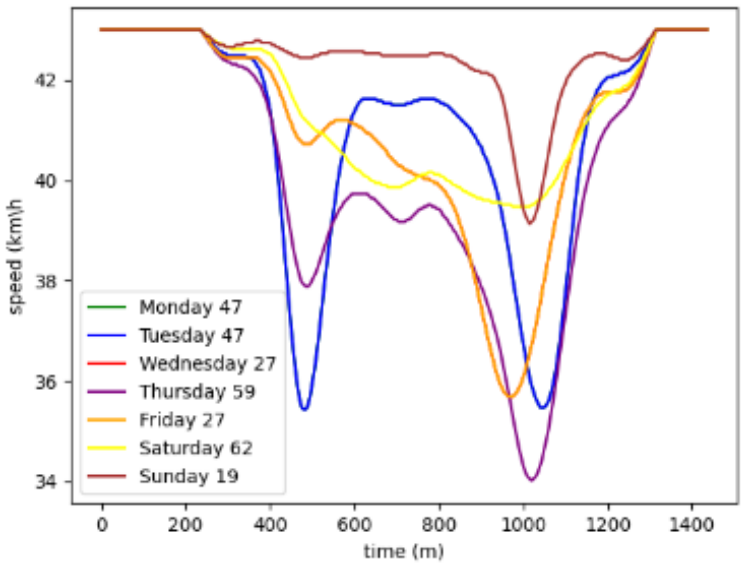


Figure 4.2: Speeds in road segment per day

Speed profiles are not available for every road, but higher hierarchical roads are covered. The exact coverage is discussed in Section 4.2.2.2**.** If there is no speed profile available for a road segment, the average speed of that road segment is used in computing the shortest route.

The speed profiles are used to calculate the cost of the road segment, which is an important input for the time-dependent Dijkstra routing technique from Section 3.2.2.1 combined with the time-dependent sampling. The road segment cost is the time taken to traverse a specific road segment and depends on the start time and day at the beginning of the road segment and uses the speed profile information explained in this section as input. In the next section, other insights obtained from the speed profile analyzed are shared.

4.2.2 Analyzing the speed profiles in a graph

As input for determining what time intervals and time-independent graphs in time-dependent sampling could be useful, further analysis of the speed profiles is done. These insights are later in this chapter when the method for determining time intervals and time-independent graphs is formulated. The speed profile analysis is split up into two parts: impact and frequency.

4.2.2.1 Speed profile impacts

A metric that can be used to show the effect of a speed profile is the impact. The impact of a speed profile is expressed as a percentage of the free flow speed of which an example was shown in **Error! Reference source not found.**. A distinction between the maximum impact and the average impact can be made.

The maximum impact of a speed profile is the lowest percentage of the corresponding speed profile. The lower this percentage the higher the impact. For speed profile 47 in Figure 4.1**,** the maximum impact is 82.4%. This maximum impact is measured at 08:00. The average impact for speed profile 47 is 94.4%. Figure 4.3**,** shows an overview of the maximum impact of all 293 speed profiles. Figure 4.4 shows at what times the maximum impact usually occurs. A table with the maximum and average impact for each speed profile can be found in Appendix B.

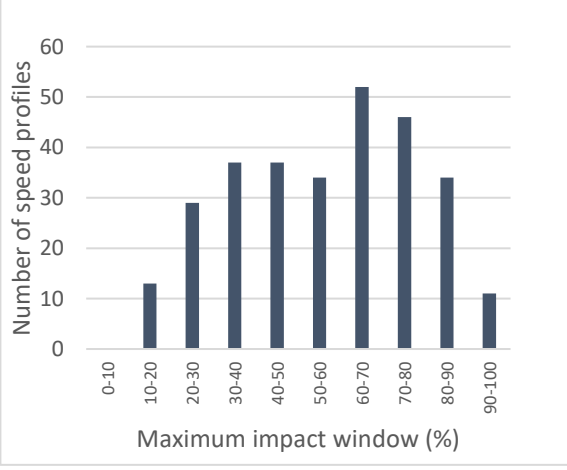


Figure 4.3: Frequency of maximum impact

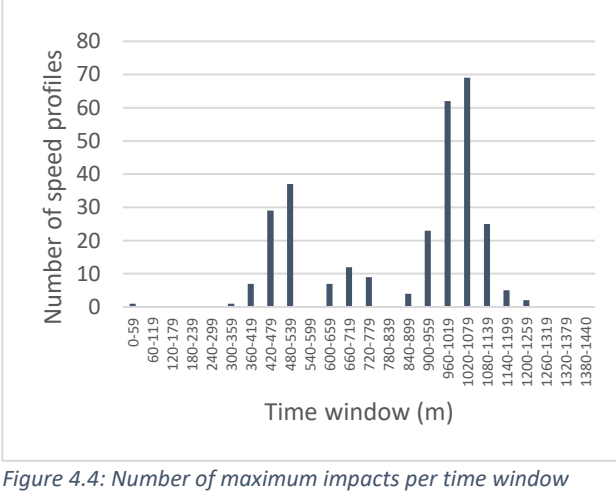


Figure 4.4: Number of maximum impacts per time window

4.2.2.2 Speed profile frequencies

A second analysis is conducted on the frequency at which speed profiles occur. For this analysis, the graph of the Netherlands was used. The graph of the Netherlands consists of 4052562 edges which can be classified using Functional Road Class (FRC) values. Low FRC values are used for larger roads and high FRC values are used for smaller roads. A complete overview of the different FRC values and

their exact definitions can be found in Appendix C. Table 4.1 shows how the edges of the graph of the Netherlands are distributed among the different FRC values.

FRC value	0	1	2	3	4	5	6	7
Number of edges	29006	18230	89557	138562	354997	160672	863690	2397848

Table 4.1: Number of edges per FRC value

On average 2171513 of these 4042562 edges in the Netherlands have speed profiles available. This means that, on average, for 46.42% of the edges in the Netherlands, there are no speed profiles available. As mentioned before, if an edge does not have a speed profile, average speeds are used. If looked more closely at the edges with no speed profile and split them up per FRC value, the results in Table 4.2 are obtained.

FRC value	0	1	2	3	4	5	6	7
Edges with no speed profile (average)	183	347	876	1366	4399	1807	107117	1764952
% of total edges with no speed profiles	0.01%	0.02%	0.05%	0.07%	0.23%	0.10%	5.69%	93.83%
% of total edges with FRC value	1%	2%	1%	1%	1%	1%	12%	74%
% of total edges	0.00%	0.01%	0.02%	0.03%	0.11%	0.04%	2.64%	43.55%

Table 4.2: No speed profile edges examined

From Table 4.2**,** we can conclude that most edges with no speed profiles are the edges of high FRC values which are considered the smaller and less important roads.

To get a better understanding of what speed profiles are the most important, a table with all speed profiles is created. In this table, found in Appendix D, the distribution per FRC value, the total number of edges and the percentage of the total edges with a speed profile are shown. In Table 4.3, the top 10 speed profiles which occur most often in the graph of the Netherlands are shown.

Profile-id	0	1	2	3	4	5	6	7	sum	Percentage of total edges
90	599	2530	6707	10304	38415	16501	96150	76715	247922	11.42%
62	1660	2213	15347	25049	51341	16291	28430	7987	148318	6.83%
8	19	181	1552	3232	18705	6752	37632	18682	86756	4.00%
216	1040	1553	8516	10672	22767	6623	17601	6905	75676	3.48%
141	372	1218	5568	7943	19125	7944	18113	7212	67494	3.11%
72	10	22	174	518	5694	2640	26330	21135	56524	2.60%
215	55	332	1415	3061	10064	4082	21512	14551	55073	2.54%
19	108	198	1175	2151	8493	3586	21255	14671	51636	2.38%
22	268	607	2033	2619	7519	3947	18591	13282	48865	2.25%
144	3	30	140	246	1685	1178	16368	17010	36660	1.69%

Table 4.3: Top 10 most occurring edges in the graph of the Netherlands

Finally, we can combine the impact analyses and frequency analyses above and show how many edges suffer what impact. These results can be found in Table 4.4.

Impact window (%)	Number of speed profiles	Number of edges	Percentages of total edges with speed profile
0-10	0	0	0%
10-20	13	4903	0.23%
20-30	29	21022	0.97%
30-40	37	47403	2.18%
40-50	37	62383	2.87%
50-60	34	100181	4.61%
60-70	52	231846	10.68%
70-80	46	285864	13.16%
80-90	34	640579	29.50%
90-100	11	777333	35.80%
Total	**293**	**2171513**	**100%**

Table 4.4: Number of edges per impact window

How the insights from this section are used in determining the method is explained later in this chapter.

4.3 Time-dependent sampling

In Chapter 3, the time-dependent sampling (TDS) heuristic by Strasser (Strasser, 2017) was introduced. This section starts with the most basic version of time-dependent sampling and extends to the version where the contraction hierarchies algorithm is included. Also, an example of the basic version of TDS in practice is visualized in Section 4.3.2.

4.3.1 Time-dependent sampling basics

The most basic version of the time-dependent sampling heuristic by Strasser (2017) consists of 6 steps and is shown in Figure 4.5.

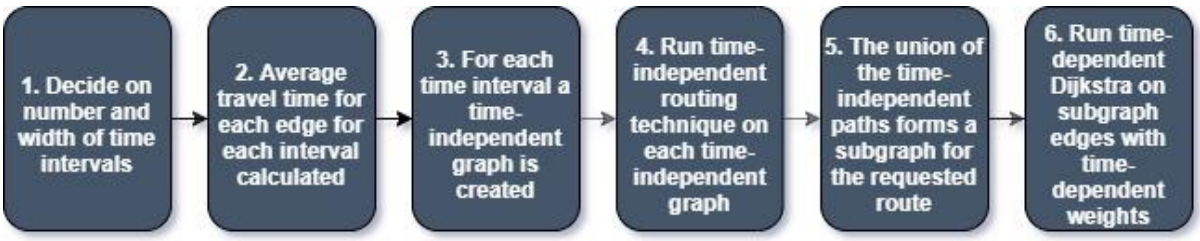


Figure 4.5: Time-dependent sampling in 6 steps

In more detail, the steps are as follows:

1. *Decide on number and width of time intervals:* For this section, the assumption is that the time intervals are known and that they were constructed using experimentation and no additional method was used in determining both the number and the exact contents of the time intervals. This additional method is a contribution to theory and is discussed later in this chapter. It is important to understand how the number and contents of the time intervals impact the rest of the TDS heuristic. In this context, the content of a time interval means what times of day are included in that interval. For example, a time interval can range from 00:00 until 07:00 and another time interval can range from 07:00 until 07:30. The following aspects of time intervals are important to take into account:
    a. Time intervals do not necessarily have to be of the same width.

b. Not all times of a day or week have to fit into the time intervals. If not in a time interval, these times are not used. It is not necessary for every minute of a day to be part of a time interval. If a time does not fall within a specific interval its information is not used in the next steps.
c. The more time intervals are used, the more memory is needed.
d. The more time intervals are used, the longer the query times are.
e. In the case of a preprocessing step. The more time intervals are used, the longer the preprocessing takes. However, Increasing the number of time intervals by a factor of two does not lead to a proportional doubling of processing duration due to parallelization in the preprocessing.

This means that the general idea is to choose time intervals in such a way that the number of time intervals is minimized while at the same the time-dependent routes are as accurate as possible. This tradeoff is experimented with in Chapter 5.

2. *Average travel time of each edge for each interval is calculated.* When the number and contents of the time intervals have been defined, the known time-dependent travel times for each edge in each time interval are averaged. These time-dependent travel times are obtained from the speed profiles which were explained in Section 4.2. The average travel times calculated are the costs that are later used in the routing. Each edge has now an average travel time costs c_i in time interval $i = 1,2, \dots, k$ where k is the number of time intervals.

3. *For each time interval, a time-independent graph is created.* The new travel time costs c_i are used to create a time-independent graph for each time interval. These time-independent graphs G_i are denoted as: $G_i(N, E, c_i)$ with nodes N and edges E being the same in every time-independent graph.

Steps 1 to 3 are part of the preprocessing and therefore do not depend on a specific route to compute and are also not repeated for every route request. Steps 4 to 6 of TDS instead depend on the requested route and these steps are therefore repeated for each route request.

4. *Run time-independent routing technique on each time-independent graph.* In step 4, a time-independent routing technique is used to find the shortest path on each time-independent graph G_i from step 3. This leads to at most k routes, one for every time interval.

5. *The union of the time-independent paths forms a subgraph for the requested route.* The k routes, which are equal to the number of time-independent graphs are combined into a new subgraph, forming subgraph G_s for the requested route. Subgraph G_s consists of all visited nodes and all used edges in the route from step 4. If some of the paths from step 4 overlap with each other, the overlapping edges and nodes are not added again to the subgraph. This makes sure that all edges and nodes in the subgraph are unique.

6. *Run time-dependent Dijkstra on subgraph edges with time-dependent weights.* In the last step, a time-dependent routing technique is performed on the subgraph G_s. The costs of the edges in G_s are time-dependent and based on the speed profiles and are denoted as c_{td}. This subgraph G_s is denoted as $G_s(N, E, c_{td})$. Only the edges in the subgraph are used for this time-dependent search. This time-dependent routing technique returns the time-dependent shortest route and is the result of TDS. The time-dependent routing technique used in this research is the exact time-dependent Dijkstra technique from Section 3.2.2.1.

This general 6-step process does cover the complete process but is not specific enough for the Simacan context. Therefore, in Section 4.3.3, we explain how to combine TDS with the contraction hierarchies algorithm. In the next section, an example of the effect of TDS in practice is visualized.

4.3.2 TDS visualization

The 6-step process from the previous sections showed the general steps of the TDS technique. In this section, a step for step visual example of the effect of TDS in practice is visualized. For this example, an own implementation of TDS was built and used. This implementation uses bidirectional Dijkstra (from Section 3.1.2.2) in step 4 and time-dependent Dijkstra (from Section 3.2.2.1) in step 6.

The first 3 steps of the TDS process, which are determining time-independent graphs $G_i(N, E, c_i)$, averaging travel time costs c_i and creating the time-independent graphs is in this section still assumed to be known and is not visualized. In the implementation used for this example, the edge costs in the time-independent graphs are not based on averages of the costs in the chosen interval. Instead, the edge costs are static costs based on different times of a day, chosen in such a way that it resulted in different time-independent routes for 4 different time-independent graphs. This process deviates from the 6-step process described in Section 4.3.1 but was chosen for demonstration purposes.

From step 4 onwards TDS depends on route request. In this implementation, a route request consists of a start and end location which are specified as coordinates (latitude and longitude) and the departure time (time of the day and day of the week). In this example, the route request is from Simacan headquarters to a location in Utrecht on a Monday at 17:00. Running bidirectional Dijkstra on the 4 time-independent graphs results in 2 different routes (blue and green). These routes are visualized in Figure 4.6.

Figure 4.6: Example of fastest routes on different time-independent graphs

The two routes from Figure 4.6 combined form the subgraph (step 5 from the TDS process). In step 6 of TDS, time-dependent Dijkstra is used on the subgraph. For this time-dependent search, the departure time and day are used. The search space for this search is very limited since only the edges part of the subgraph are considered in the time-dependent search. Figure 4.7 shows the shortest time-dependent path in blue and the visited edges of the subgraph to find the shortest time-dependent path in red. The number is visited edges to come up with this path is 628.

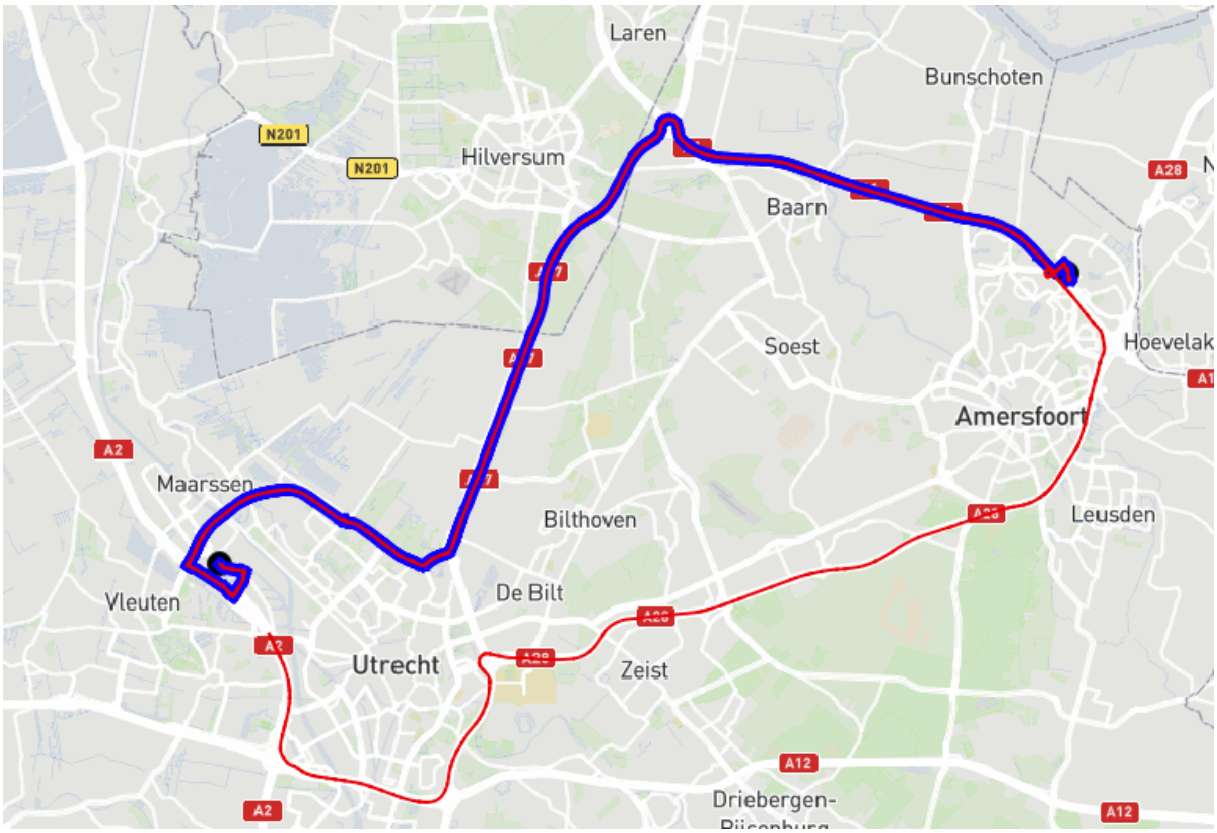


Figure 4.7: Example of time-dependent route found on subgraph

Instead of finding this time-dependent route with TDS, time-dependent Dijkstra could also be used as an exact approach. For this specific route request, the exact approach leads to the same route as was found using TDS. Figure 4.8 shows the exact route in blue and the visited edges to find this exact route with time-dependent Dijkstra in red. The edges visited for this exact solution (57634) are much more than the edges visited in the time-dependent step of TDS (628). The approximately 90 times more edges visited in the exact approach than in TDS results in higher query times. In Chapter 6, query time comparisons between the different approaches are conducted.

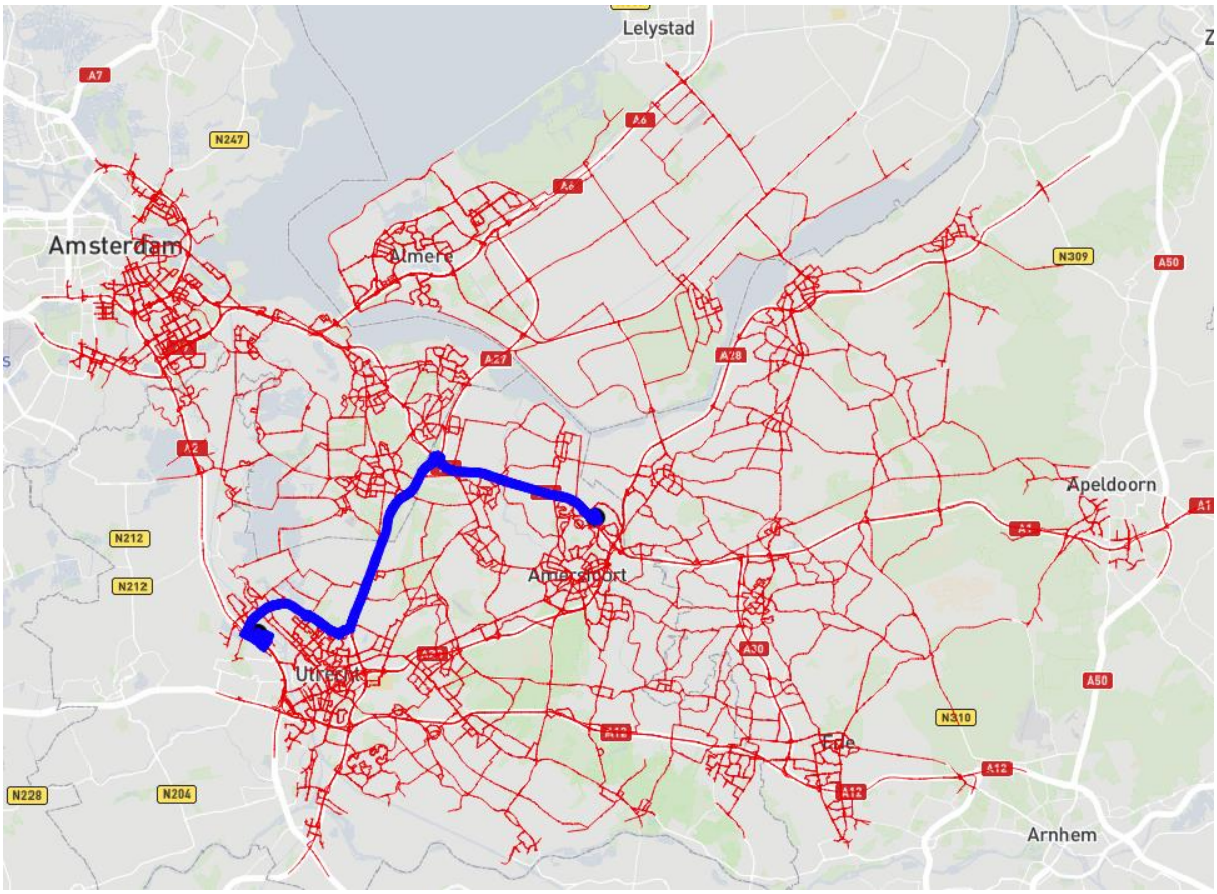


Figure 4.8: Example of an exact time-dependent routing technique (time-dependent Dijkstra)

If the time-dependent route for multiple departure times is wanted, only step 6 of the TDS process needs to be repeated in this case. This can be used to get information about at what time of the day a certain route is preferred in terms of shortest paths and is considered a profile query as described in Section 3.2.1.2. The background color in Figure 4.9 shows for this specific route request at what time which route is considered shortest on a Monday. The north (blue) and the south route (green) correspond to the routes from Figure 4.6. Figure 4.9 also shows a comparison between travel time during a Monday if the regular time-independent route is always used versus the travel time if the route found with TDS is used. The difference between the two travel time lines shows the effect of using TDS in this example route request.

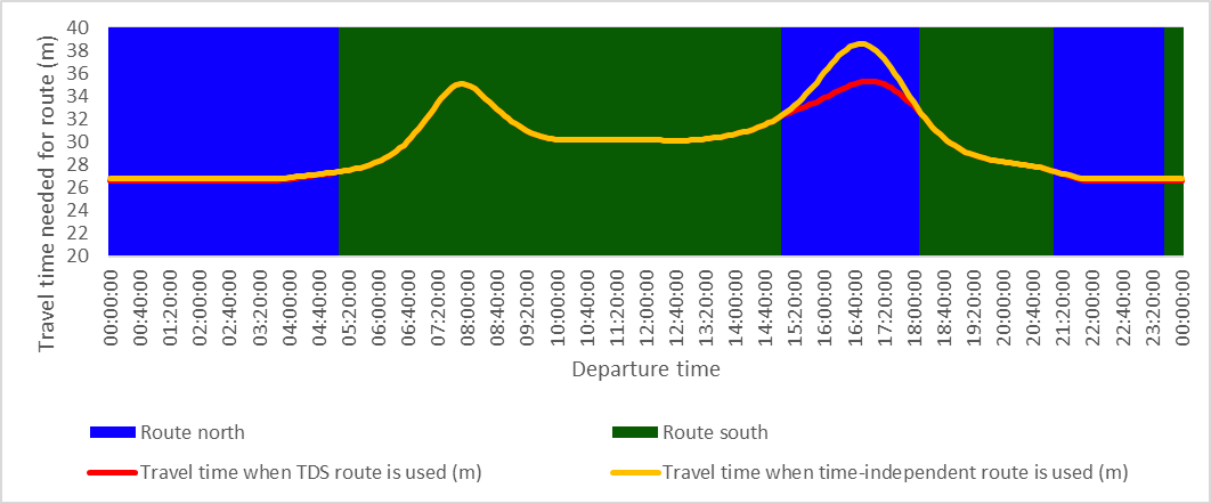


Figure 4.9: Travel time comparison of the TDS route used versus the time-independent route used

This example and result show that time-dependent routing matters because of the difference between the regular and time-dependent paths. The maximum difference occurs at 16:40 and is 3.44 minutes which is 8.93% of the time-independent route travel time. In the experimentation phase of this research, comparisons such as this one are made on a large dataset.

4.3.3 TDS including contraction hierarchies

In step 4 of the TDS process explained in Section 4.3.1**,** it was explained how a time-independent routing technique is used to find the shortest path on each time-independent graph G_i from step 3. Because of the reasoning from Chapter 3, we have chosen to use the contraction hierarchies (CH) algorithm as the time-independent routing technique in the implementation and extension of TDS that is used in this research.

As was explained in Section 3.1.4 in the literature chapter, the CH algorithm consists of two phases. The preprocessing phase and the query phase. The overview figure from Section 4.3.1 is not CH-specific, therefore Figure 4.10 is created. Figure 4.10 shows a more detailed version of the TDS technique that includes CH. For this figure, it is still assumed that the method of making the different time-independent maps is known upfront.

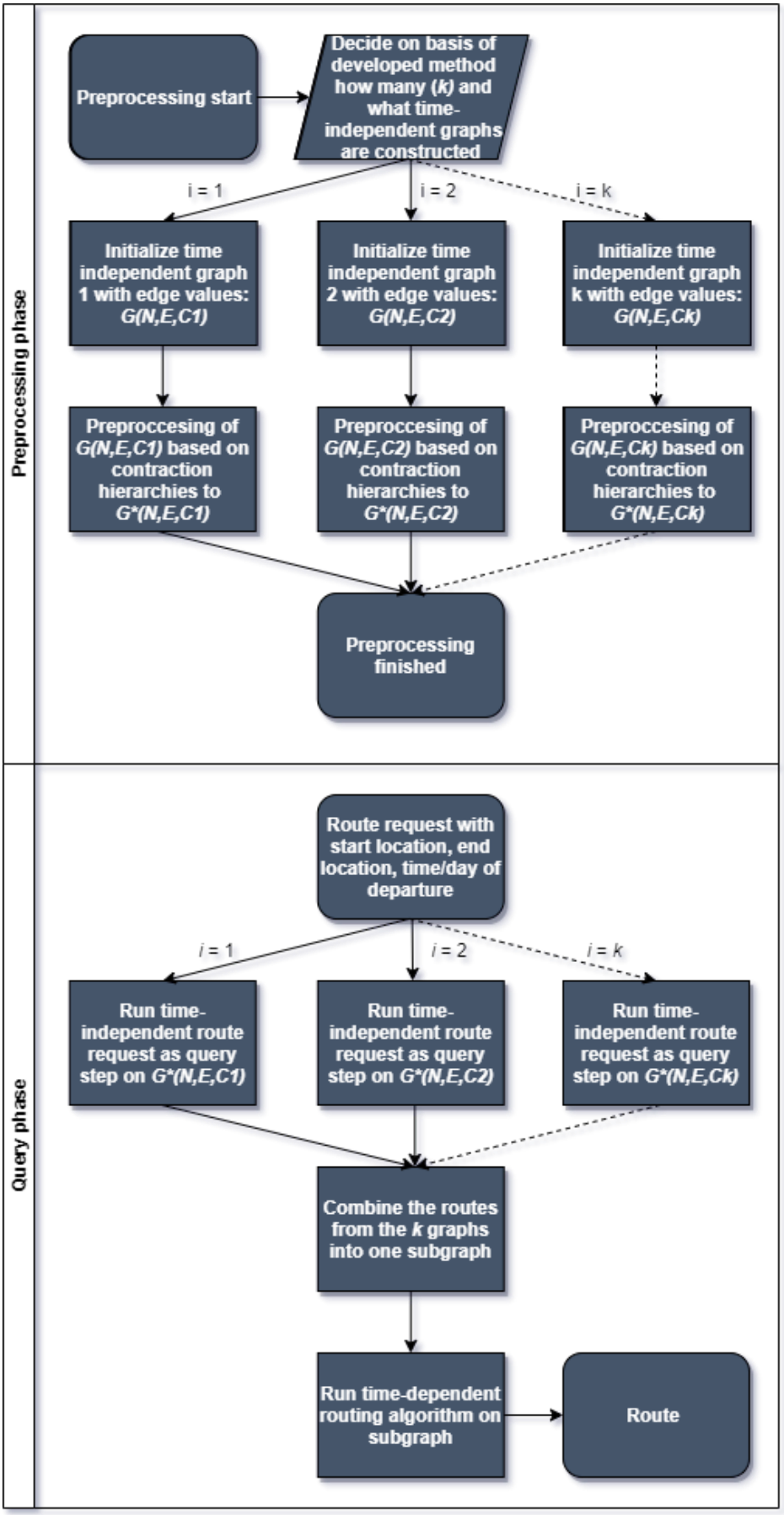


Figure 4.10: TDS-CH process flow

The two phases of CH in combination with TDS work as follows:

1. *Preprocessing phase*: Since TDS works with multiple time-independent graphs, TDS-CH (TDS with CH) also works with multiple time-independent graphs. The number of time-independent graphs is equal to the number of time intervals (k). Just as the standard TDS technique, each time-independent G_i has costs per edge that depend on the time intervals. This results in k time-independent graphs: $G_i(N, E, c_i)$ with $i = 1,2, \dots, k$. In CH, a graph is preprocessed similarly, the k graphs in TDS-CH are preprocessed which results in k preprocessed graphs: $G_i^*(N, E, c_i)$ with $i = 1,2, \dots, k$.

2. *Query phase*: Before the query phase can be performed, the preprocessing phase should be finished. The query phase is conducted whenever a route is requested. Similar to the standard TDS technique, TDS-CH performs a time-independent search on k time-independent graphs. In the case of TDS-CH, this is done using the CH query step from Section 3.1.4. once on each of the preprocessed graphs $G_i^*(N, E, c_i)$. This results in k routes which are combined into a new subgraph, G_s for the requested route. Finally, A time-dependent routing technique is performed on the subgraph G_s. This time-dependent routing technique returns the time-dependent shortest route and is the result of TDS-CH.

TDS-CH can be used for both the earliest arrival query and the profile query. In the case of the earliest arrival query, the query phase is completely executed. On the contrary, for a profile query where multiple departure times for the same route are considered, only a part of the query phase is executed since the same subgraph can be used for the different departure times. For both types of queries, the preprocessing phase is only repeated if the map where the graphs are based on changes or when other time intervals are going to be used.

4.4 The interval selection method

Until now, what time intervals to use in TDS-CH was assumed to be known. In previous research, the time intervals in time-dependent sampling were just based on visual observation and not on a data-driven approach. In this section, the decision on how many and what time intervals to use is examined more closely and worked out. A method of selecting intervals based on speed profile information is proposed. By doing so, this section contributes to the research gap which is developing a method that creates good time-independent graphs.

The proposed method, from now called the *interval selection method*, consists of several steps that are always conducted in the same order:

1. Obtaining speed profile information of the relevant region
2. Creating the weighted speed profile from the speed profile information
3. Dividing the weighted speed profile into periods of similar speed
4. Selecting the most diverse time intervals for time-dependent sampling

These steps are explained in more detail in Sections 4.4.1 to 4.4.4. The interval selection method is designed in such a way that it is generic enough so that it can be applied to the Simacan case but also to other cases. In all cases, the method however always requires several inputs such as information about changing speeds (in this research, speed profiles are used) on road segments. To better understand the interval selection method in practice, an example of the interval selection method is applied to the region of the Netherlands in these sections.

4.4.1 Obtaining speed profile information of the relevant region

The first step of the interval selection method is the gathering of speed profile information of the relevant region. With relevant region, the geographical area in which locations around the start and end node are meant. This relevant region is equal to the region of the graph that is used for the routing.

Once the relevant region is determined, the speed profile information from this relevant region is obtained by analyzing the graph of the relevant region. An introduction to this analysis was visualized and summarized for the graph of the Netherlands in Section 4.3.2. The speed profile information is obtained for every speed profile separately and consists of three attributes:

- *Maximum speed impact percentage*: this is a percentage that represents the maximum impact of a speed profile and was introduced in Section 4.2.2.1. This percentage is the lowest percentage of all speed percentages during the day for a certain speed profile. A low percentage means therefore a higher impact.
- *Speed profile frequency percentage:* refers to the proportion, expressed as a percentage, of how frequently a particular speed profile is used in a graph within a given time frame (such as a single day like a Tuesday in this research). It is determined by dividing the number of times a specific speed profile is used during that period by the total number of speed profile usages across the same duration.
- *Average FRC value:* the average of the FRC values of all road segments of a certain speed profile.

These three attributes were chosen because they provide a good reflection of the importance of a speed profile in relation to the other profiles in the relevant region based on data analysis. The level of importance for each attribute is explained in the next section where weights are linked to the attributes. As an example, the obtained speed profile information for a speed profile is shown in Table 4.5 below. The example is a speed profile with a relatively high impact (low percentage), low frequency and high FRC value average.

Speed profile id	Maximum speed impact percentage	Speed profile frequency percentage	Average FRC value
0	31.7%	0.09%	5.06

Table 4.5: Example of a speed profile with attributes

4.4.2 Creating the weighted speed profile from the speed profile information

After all speed profile information is gathered, the second step is to create a weighted speed profile. From Strasser (2017) we know that time intervals work best when they are adapted to the expected speed fluctuations on the road segments in the graph. In this research, speed fluctuations are captured in the different speed profiles as was explained in Section 4.3.2. TDS-CH is therefore expected to best work with time intervals that are adapted to the speed profiles. However, the time intervals for TDS-CH need to be the same for all road segments and the time intervals are therefore not specifically tailored for every individual speed profile. Therefore, the time intervals should be tailored to one speed profile out of the 293 speed profiles available. This would however lead to overfitting the time intervals to this specific speed profile and that is considered not wanted as well.

The proposed solution is therefore to combine all speed profiles into one overlapping speed profile. Combining all speed profiles into one by averaging the speed percentage would cause all speed profiles to be equally important. However, this is not wanted because from the speed profile analysis, we know that the impact and frequency of each speed profile differ greatly. The idea is

therefore to link weights to the speed profile attributes obtained in the first step of the interval selection method. Next, these weights are used to reflect a difference in relative importance between the different speed profiles. The weights will then be used to create one overlapping speed profile (from now on called the weighted speed profile) that reflects the importance.

In the previous section, three attributes of speed profiles were introduced. Set rules are assigned for what weight should be used per attribute depending on the value. Every attribute is scored based on a 5-pointed scale where a high assigned weight (5) means that that attribute is relatively important and a lower weight (1) means that that attribute is relatively unimportant.

What weight is linked to an attribute depends on the value of the attribute. For each attribute, five weight intervals were calculated. Each weight interval corresponds to a weight between 1 and 5. Then, the weight per speed profile depends on in which weight interval each attribute falls. The widths of the weight intervals are chosen in such a way that every interval fits the same number of values. This means that for the 293 speed profiles in the Netherlands, every weight interval contains $293/5 = 58.6$, so 58 or 59 speed profiles. The borders between the weight intervals for each attribute were found using the 0, 20th, 40th, 60^{th,} 80th and 100th percentiles. The weight intervals for the three attributes are shown in Table 4.6.

Weight	1	2	3	4	5
Maximum speed impact percentage	100.00% ≤ x < 77.32%	77.32% ≤ x < 64.62%	64.62% ≤ x < 50.10%	50.10% ≤ x < 35.22%	35.22% ≤ x ≤ 14.80%
Speed profile frequency percentage intervals bounds	0.00% ≤ x < 0.04%	0.04% ≤ x < 0.07%	0.07% ≤ x < 0.14%	0.14% ≤ x < 0.41%	0.41% ≤ x ≤ 10.08%
Average FRC value intervals bounds	6.73 ≤ x < 6.24	6.24 ≤ x < 5.84	5.84 ≤ x < 5.37	5.37 ≤ x < 4.75	4.75 ≤ x ≤ 1.62

Table 4.6: Interval bounds per attribute with linked weights

When the maximum speed impact percentage of a speed profile has the value of 31.7% like in the example, that value would get weight 5. After a weight is linked to each of the three speed profile attributes for a speed profile, the total speed profile weight can be calculated by multiplying the linked weight per speed profile with each other. For the example from the previous section, this is done in Table 4.7. In Table 4.7 the linked weights per attribute and the resulting speed profile weight are shown.

Speed profile	Maximum speed impact percentage		Speed profile frequency percentage		Average FRC value		Speed profile weight
	Attribute value	Attribute weight	Attribute value	Attribute weight	Attribute value	Attribute weight	
0	31.1%	5	0.09%	3	5.06	4	60

Table 4.7: Example of a speed profile with attribute values and weights

The next step is calculating the weighted speed profile. This is done by multiplying the speed profile percentages for every time period per speed profile with the corresponding speed profile weight. Multiplying the attribute weights makes sure that speed profiles that have a relatively high combination of weights are featured much more prominent in the weighted speed profile. This makes the weighted speed profile more extreme which makes periods in the third step of the interval selection method more distinctive. Once this is done for all speed profiles, all speed

percentages per time period are added together and then divided by the sum of the total weight per speed profile. The result is the overlapping speed profile.

For the relevant region of the Netherlands on a Tuesday, this results in the weighted speed profile shown in Figure 4.11.

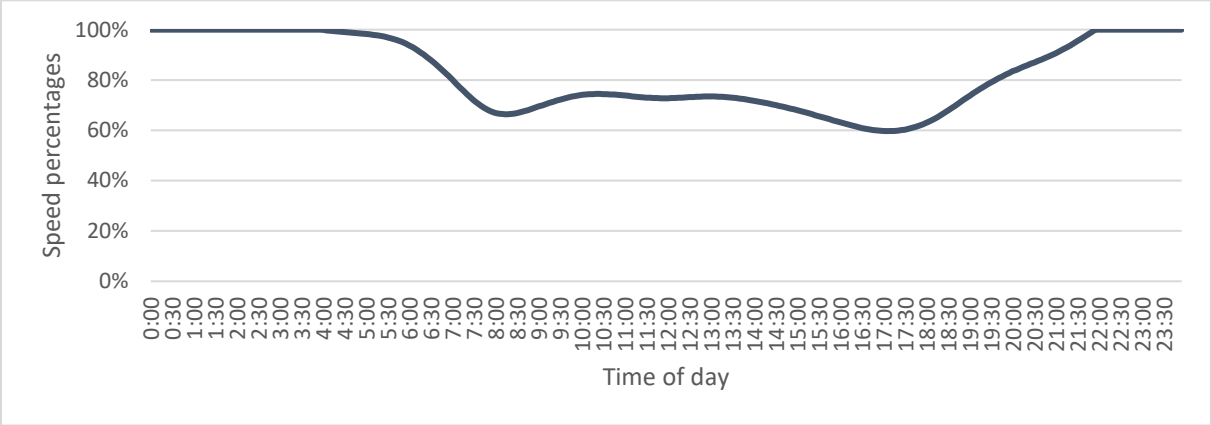


Figure 4.11: Weighted speed profile of the Netherlands

4.4.3 Dividing the weighted speed profile into periods of similar speed

In the third step, the actual time intervals are obtained from the weighted speed profile. The goal is to create the time intervals in such a way that when the intervals are later used in the query phase of TDS-CH, the time-independent routes for every route, between location A and location B are as different as possible while as few as possible time intervals are used. In Chapter 5, the experiments will also be analyzed on this fact.

To find different useful time intervals, the weighted speed profile is first split up into three types of periods, namely: periods of increasing speed, periods of stable speed and periods of decreasing speed. This division is made because we are interested in the extremes and fluctuations because that is what causes different time-independent paths to be found in TDS-CH. The division of the weighted speed profile into these three periods depends on the speed percentage difference between the current and previous 5-minute time periods which are used as bounds for the periods and are calculated as follows:

$$speed\ percentage\ difference_i = speed\ percentage_i - \ speed\ percentage_{i-1} \ \ for\ all\ i$$

A relatively low speed percentage difference indicates that speed does not change much and is therefore stable, whereas a relatively higher speed percentage difference indicates more fluctuating speeds. Based on the now-known speed percentage differences for all time periods, the upper and lower bounds of what is considered a stable period are chosen. These bounds are based on the number of different periods wanted. Based on Strasser (2017), where experiments were performed with a maximum of 9 time intervals, here 9 is chosen as the number of periods wanted. The decision to choose 9 periods is unrelated to the final selection of the time intervals. If a number lower than 9 is chosen it would result in periods that are too wide and contain too many speed percentages which would cause the average speed percentage of those periods to be too generic and extremes would be ignored. This is unwanted since data research has shown that extremes make it possible to find new time-independent routes in TDS-CH.

The exact values of the bounds are based on a newly created algorithm that starts with relatively high bounds (i.e. speed percentage differences of -5% and 5%) and then checks if the bounds are low

enough to split the weighted speed profile into 9 periods. If there are 9 periods found, the algorithm stops, otherwise the algorithm decreases the bounds and continues this process until 9 periods are found. This algorithm results in a list of periods that can be used as time intervals. The periods found using this algorithm are shown in Table 4.8. It shows the type per period, the length per period and the average speed percentage per period.

Period number	Period start time	Period end time	Length of period	Period type	Average speed percentage
1	0:00	5:30	05:30	Stable	99.60%
2	5:30	7:55	02:25	Decreasing	83.11%
3	7:55	8:35	00:40	Stable	66.70%
4	8:35	9:30	00:55	Increasing	69.94%
5	9:30	15:20	05:50	Stable	72.36%
6	15:20	16:00	00:40	Decreasing	64.60%
7	16:00	17:35	01:35	Stable	60.60%
8	17:35	21:50	04:15	Increasing	80.04%
9	21:50	0:00	02:10	Stable	100.00%

Table 4.8: Period overview

In Figure 4.12**,** the 9 periods are visualized in the weighted speed profile.

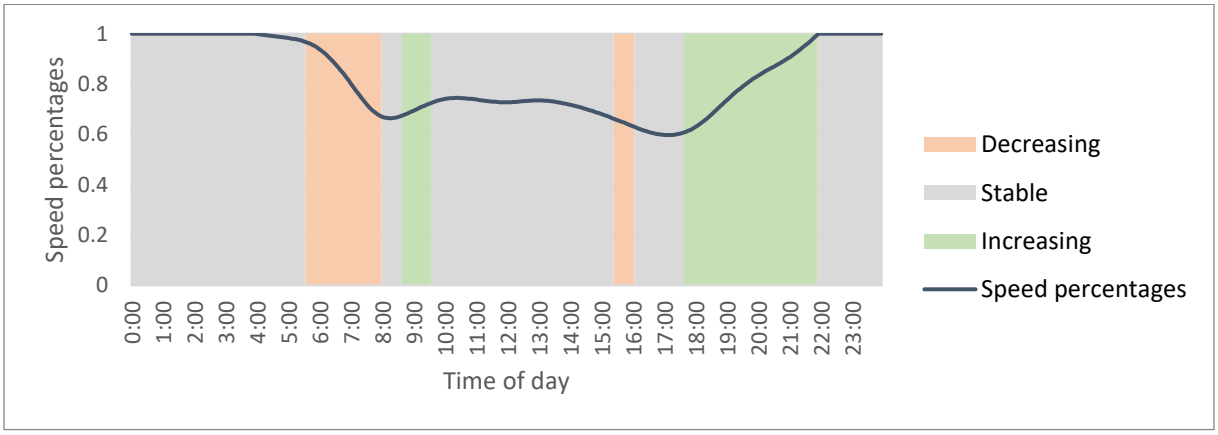


Figure 4.12: Weighted speed profile with periods

What periods are used as time intervals is discussed in the fourth step.

4.4.4 Selecting the most diverse time intervals for time-dependent sampling

Finally, the found periods from the previous step can be chosen as time intervals. What periods are chosen as time intervals depends on the number of time intervals desired. For every additional time interval wanted, an additional period is picked from a list. The list where the time interval is picked from however is first ordered. The order of the list is based on the average speed percentage shown in the last column of Table 4.8 and is determined by selecting the period with the highest average speed first and then iteratively selecting the period in which the average speed is furthest away from the previously selected periods until all values are picked. This ensures that the periods in the final ordered list are arranged based on their distance from each other.

This ordered list of periods can now be used to pick time intervals from. For the found periods in Table 4.8, this means that the period from 21:50 to 00:00 is selected first, and the period from 16:00 to 17:35 is selected second. In Table 4.9, the complete order in which the periods are picked is shown, in which the last column (picking order) is added compared to Table 4.8.

Period number	Period start time	Period end time	Average speed percentage	Picking order
1	0:00	5:30	99.60%	9
2	5:30	7:55	83.11%	6
3	7:55	8:35	66.70%	8
4	8:35	9:30	69.94%	4
5	9:30	15:20	72.36%	7
6	15:20	16:00	64.60%	5
7	16:00	17:35	60.60%	2
8	17:35	21:50	80.04%	3
9	21:50	0:00	100.00%	1

Table 4.9: Ordered periods

The picking order from Table 4.9 means that if four time intervals are wanted in TDS-CH, periods 9, 7, 8 and 4 are used. As stated before, the picked periods do not have to cover a whole day because that is not necessary for TDS-CH.

4.5 TDS-CH process flow with the interval selection method

Now that the interval selection method is explained, it can be included in the TDS-CH from section 4.3.3. The interval selection method can be seen as a separate phase before the preprocessing phase and therefore the TDS-CH process flow from Figure 4.10 can be extended with a new phase, the *interval selection phase.* The process flow of the interval selection phase is visualized in Figure 4.13 below:

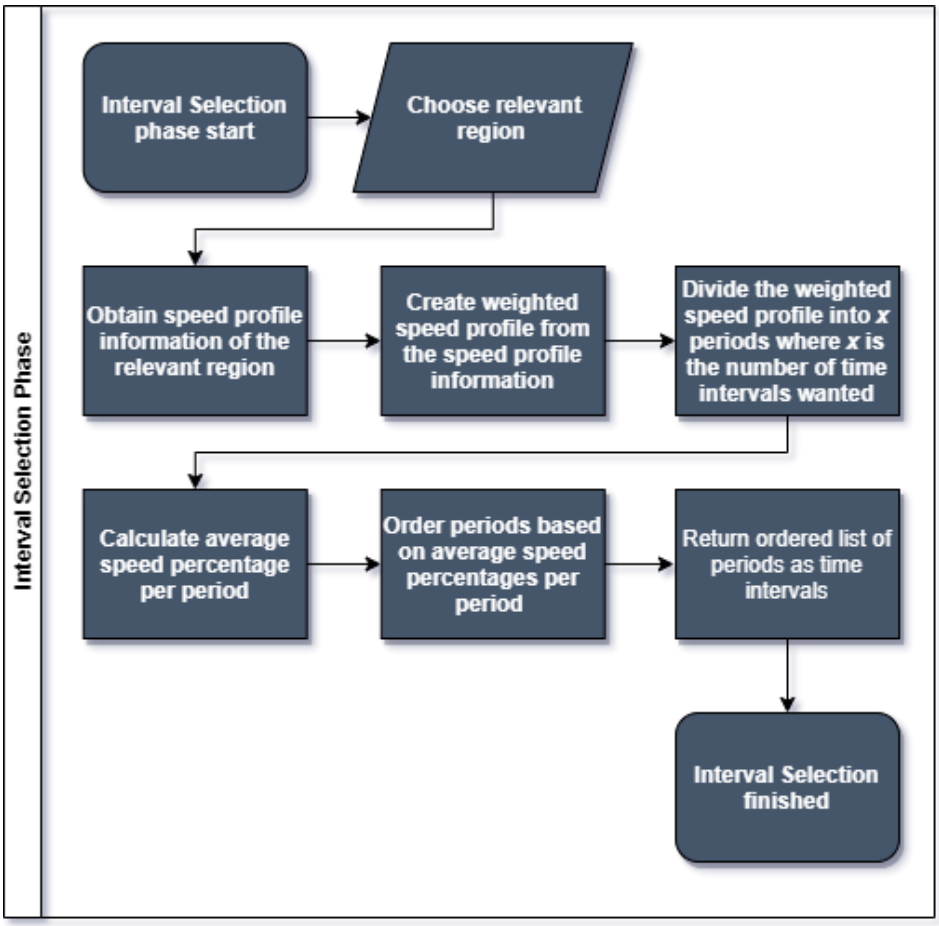


Figure 4.13: Process flow of the Interval Selection Phase

4.6 Conclusion

In this chapter, the time-dependent sampling technique is described that determines the shortest route depending on the day of the week, and the time of the day. The six general steps of time-dependent sampling are explained which are (1) decide on the number and width of time intervals, (2) average the travel time for each edge for each time interval, (3) create a time-independent graph for each time interval using contraction hierarchies, (4) run time-independent routing on each time-independent graph, (5) form a subgraph by taking the union of all time-independent routes, and (6) run time-dependent Dijkstra on the subgraph edges with time-dependent weights.

Also, the general 6-step process was extended with the contraction hierarchies algorithm which showed that the TDS technique could be split up into a preprocessing phase that covers steps 1 to 3 of TDS and the query phase that covers steps 4 to 6. Before the query phase (and the rest of the method) can be performed, the preprocessing phase should have taken place.

Using an illustrated example, it was shown that implementing time dependency in routing is beneficial. How well the time-dependent sampling technique with contraction hierarchies works depends on the chosen time intervals. Therefore, in this chapter, the research gap was worked out by developing and explaining the proposed interval selection method. The method consists of four steps, namely: (1) obtaining speed profile information, (2) creating the weighted speed profile, (3) dividing the weighted speed profile into periods of similar speed, and (4) selecting the most diverse time intervals. To complete the model, the interval selection method was added as the new interval selection phase before the preprocessing and the query phase.

5 Experimental setup

In this chapter, the experiments that are used to test the performance of the newly created interval selection method and to find the best number and width of the time intervals to use are explained. Furthermore, two benchmarks are introduced which are used to compare the experiments to and that are used to show the effect of time-dependency in general. In Section 5.1 the dataset that will be used for the experiments is explained. Next, in Section 5.2, the different experiments and motivations behind them are described. In Section 5.3, the way the results are compared and how these are interpreted is explained. This chapter ends with a conclusion in Section 5.4 that answers the following research questions:

What experiments can be used to test the performance of the interval selection method?

a) How is the dataset on which the experiments are conducted constructed?
b) What variations are made in the different experiments?
c) What indicators are used the measure the performance of the experiments?

5.1 Dataset selection

All experiments are conducted on the same dataset. This is done to make sure that the effect of the control variables (the changeable factors and variations in the method) that are experimented with are comparable.

The dataset consists of a list of start and end locations between which the time-dependent routes must be found.

To better match the different types of customers in the Simacan operation and potentially find different results for different types of locations, the dataset is split into three types of operations with different types of start and end locations:

- *Retail operation*: the start locations are distribution centers (DCs) and the end locations are shops such as supermarkets. Roads used are quite diverse and depend on exact locations.
- *Post/parcel operation:* both start and end locations are mostly DCs located all around the country. Roads used include mostly motorways, freeways and other major roads.
- *Home delivery operation:* the start locations are mostly hubs and end locations are mostly houses in urban areas, such as individuals who order groceries from an online supermarket to their house. Roads used are mostly local connecting roads and roads in urban areas.

For each of the three operations, start and end locations were picked randomly from the Simacan database. Each start and end location has a latitude and longitude. To prevent privacy and ethical issues, the locations used are publicly available in the case of the DCs and the hubs. For the individual customer locations in the home delivery operation, random locations in cities close to the related hub were chosen. The locations are chosen in such a way that they are scattered across the whole relevant region (the Netherlands) and that both locations inside rural and urban areas are included. For each of these sets, 250 start and end location combinations are included in the dataset. The number of 250 combinations per set is chosen because otherwise the total number of computed routes becomes too large and the system used to conduct this research is not able to compute all experiments in a reasonable time.

In addition to these three different sets, another set is added to test whether it matters what type or locations are used in time-dependent routing. The fourth set also consists of 250 start and end

location combinations. The start and end locations in this set are randomly chosen from the relevant region and have a road nearby. In total, the number of start and end locations for which routes are calculated adds up to 1000 among the 4 different sets. Each of these start and end location combinations have a scenario number s. The set of all scenarios is denoted as:

$$s = scenario\ number\ \{1,2,\dots,1000\}$$

To test the effect of time-dependency during a day, for every start and end location combination, a set of departure times is chosen. The set of chosen departure times is the same for every scenario. Since all speed profiles do not fluctuate between 04:00 and 22:00, only departure times between 02:00 and 22:00 are used. 02:00 is chosen because when the route takes longer than 2 hours, time-dependency is still relevant in this case. Ideally, every minute between 02:00 and 22:00 is chosen as a departure time. But because of both computational time and relevance, departure times with 10 minutes between each other are chosen. In total, this results in $20\ hours * 6\ departures\ per\ hour = 120$ departures per scenario. The departure time in minutes is denoted as d:

$$d = departure\ moment\ in\ 10\ minute\ intervals\ \{0,1,\dots,120\}$$

For each scenario also the operation type number o is specified. The set of all different operations is denoted as o:

$$o = operation\ type\ \{1,2,3,4\}$$

This results in $1000 * 120 = 120.000$ route calculations per experiment.

5.2 Experimental setup

The goal of the experiments is to test the performance of the interval selection method and find the ideal number of intervals to use in the TDS-CH method. Therefore, experiments are conducted in which two different factors are changed. The first factor is the number of time intervals used in TDS-CH and the second factor is the way of selecting the intervals. For the first variable factor, Strasser (2017) experimented with a maximum of 9 different time intervals and this is the maximum number of time intervals used in this research. Since experimenting with 1 time interval would have no use in relation to time-dependent sampling, the number of intervals used in all the experiments differs between 2 and 9.

For the second variable factor, the way of selecting intervals, two variations are experimented with:

- *Equal width interval approach:* The day is divided into equal segments. For instance, in the case of three intervals, each interval covers 8 hours of a day. Namely from 00:00 to 08:00, from 08:00 to 16:00, and from 16:00 to 24:00.
- *Interval selection method:* This is the in Section 4.4 created and explained method of selecting time intervals.

The equal width interval approach variation was chosen because this relatively random variation can prove whether the in this research proposed interval selection method has an effect at all.

In each experiment, only one factor is changed to make sure the possible effects of the different factors are isolated. The total number of experiments is therefore equal to the multiplication of the number of variations per factor which results in $2 * 8 = 16$ experiments. The set of all experiments is denoted as e:

$$e = experiment\ number\ \{1,2,\dots,16\}$$

The experiments conducted in this research are summarized in Table 5.1 below:

Experiment number E	Number of time intervals used	Method of selecting time intervals
1	2	Equal width interval approach
2	3	Equal width interval approach
3	4	Equal width interval approach
4	5	Equal width interval approach
5	6	Equal width interval approach
6	7	Equal width interval approach
7	8	Equal width interval approach
8	9	Equal width interval approach
9	2	Interval selection method
10	3	Interval selection method
11	4	Interval selection method
12	5	Interval selection method
13	6	Interval selection method
14	7	Interval selection method
15	8	Interval selection method
16	9	Interval selection method

Table 5.1: Experiments summary

5.3 Performance measurement

Before the actual experiments from Section 5.2 are conducted using the dataset from Section 5.1, indicators are defined. The goal of these indicators is that they are used to compare the different experiments with each other and indicate what makes one experiment perform better than another.

The research objective is to find an extension to the contraction hierarchies method that supports time-dependent routing. By choosing and expanding the time-dependent approach, this objective is reached but the next step is to finetune the in the previous chapter explained approach and examine how well it performs. By finetuning the approach, we mean searching for a balance between the number of intervals used and the way intervals are selected. We are searching for the solution that comes closest to the exact method in terms of used routes and corresponding travel times while using as few as possible time intervals. This is desired because the more time intervals, the bigger the preprocessing time, query time and space required.

Therefore, the experiments have two goals. The first goal is to show if the proposed interval selection method performs better than the approach that uses equal width intervals. The second goal is to find the best number and width of the time intervals to use in TDS-CH.

For each experiment, multiple results are measured as indicators that are used in comparing the experiments. The following key performance indicators (KPIs) are measured per experiment:

- *Preprocessing time (m):* the time it takes to preprocess the time-independent graphs in minutes where a lower preprocessing time is considered favorable.
- *Preprocessing space needed (GB):* the total space needed to store the preprocessed time-independent graphs in gigabytes. A lower preprocessing space needed is considered favorable.
- *Average subgraph creation time (s):* the average time it takes to create a subgraph for all scenarios in an experiment. A lower average subgraph creation time is favorable.

- Average *subgraph size (km):* the average length of all road segments part of the subgraph over all scenarios. A bigger subgraph results in more road segments that can be used in the time-dependent search over the subgraph.
- *Average number of unique TID paths in the subgraph:* a number that shows how many different time-independent routes were found on the time-independent graphs. This number is at most equal to the number of time intervals.
- *Average profile query time (s):* the average time it takes to finish the profile query in seconds. In the experiments, the profile query time is the time it takes to find the routes for all 120 departure times used in this research. A faster profile query time is considered favorable because of the enormous number of queries that are required in real-time applications.
- *Average single query time (s):* the average time it takes to finish computing a single route query. A faster single query time is considered favorable because of the enormous number of queries that are required in real-time applications.
- *Subgraph utilization (%):* a percentage that shows how much of the road segments of the subgraph are used by all time-dependent routes calculated on that subgraph. A higher subgraph utilization is considered favorable because when only a small part of the subgraph is used, the subgraph is unnecessarily large.
- *Average number of different TD paths in the subgraph:* a number that shows how many different time-dependent routes (out of the 120 computed routes per scenario) were found.
- *Travel time (s):* the time it takes in seconds to travel from a start to an end location based on a computed route. In TDS-CH experiments, the computed routes and therefore the travel times are approximations. A lower travel time is favorable.

We are searching for a solution that comes closest to the exact method in terms of used routes and corresponding travel times and therefore, an exact method is used as a benchmark to compare all experiments to. Time-dependent Dijkstra, which was explained in Chapter 3, is used as the exact method. Comparison is done based on the above KPIs, apart from the preprocessing KPIs, as preprocessing is not part of the time-dependent Dijkstra method. In addition to the exact benchmark, the time-independent CH algorithm is also used as a benchmark. All the abovementioned KPIs are also calculated for this method, and comparison is based on this. In the next chapter, the two benchmarks are used to compare the experiments with, but also to show the effect and potential of using time-dependency in routing.

Based on these two benchmarks, two additional times KPIs are formulated. Unlike the previous KPIs which are calculated per experiment, the following KPIs are calculated per scenario in each experiment:

- *Total time difference with the TID route (m):* the sum in minutes of all time differences between the time-dependent route and the time-independent route per experiment per scenario. For experiment e and scenario s the equation is as follows:

$$Total\ time\ difference\ with\ TID\ route_{e,s} = \sum_{d=1}^{d=120} TID\ travel\ time_{e,s,d} - TD\ travel\ time_{e,s,d}$$

- *Maximum time difference with the TID route (m):* is the biggest difference in minutes between the TID travel time minus the TD travel time per experiment per scenario.

These two KPIs are calculated for all scenarios in all experiments but also for the exact benchmark. When for an experiment the result of the KPI is equal to the KPI calculated for TID compared to the exact benchmark, this means the result of the experiment is optimal.

5.4 Conclusion

In this chapter, the experiments that are conducted in the next chapter are explained. All experiments are conducted on the same dataset, consisting of start and end locations retrieved from the Simacan database and split into four operation types: 1. *Retail operation* (DC to shop), 2. *Post/parcel operation* (DC to DC), 3. *Home delivery operation* (hubs and houses in urban areas), and 4. *Random locations*. Each operation consists of 250 combinations, resulting in 1000 scenarios. Next, these scenarios are run every 10 minutes in each hour between 02:00 and 22:00 resulting in 120 departures per scenario, therefore $1000 * 120 = 120.000$ routes per experiment.

To test the performance on this dataset, two factors are changed, namely the number of intervals used and the way of selecting intervals. The number of intervals should vary to measure the difference, resulting in the decision to test between 2 and 9 intervals. This results in 8 experiments. Next, the way of selecting intervals is experimented with two variations, namely the interval selection method of the previous chapter, and secondly, by using equal width intervals throughout the day. This results in $8 * 2 = 16$ experiment numbers, of which for each of them 120.000 routes are constructed.

The experiments have two goals. The first goal is to show if the proposed interval selection method performs better than the approach that uses equal width intervals. The second goal is to find the best number and width of the time intervals to use in TDS-CH. To do this, the 16 experiments are compared with each other and additionally also to two benchmarks, namely the exact time-dependent Dijkstra and the TID CH. For comparison between the experiments to the benchmarks and comparison of experiments to other experiments, ten KPIs are defined. Furthermore, two KPIs are used to compare the results to the optimal situation, namely 1. *Total time difference with the TID route*, and 2. *Maximum time difference with the TID route*. The benchmarks are also compared with each other to show the effect and potential of using time-dependency in routing.

6 Results

In this chapter, the results of the experiments are presented and analyzed, based on the KPIs of the previous chapter. In Section 6.1, the benchmarks of the scenarios are presented and the potential of using time-dependent routing on this dataset is shown. Furthermore, in Section 6.2, the effect of using time intervals on the preprocessing time and the required preprocessing space is studied. Next, the effect of using different time intervals on the size of the subgraph, the subgraph utilization and the number of time-dependent routes used is examined in Section 6.3. Fourthly, Section 6.4 presents the effects of the different experiments on the query times and time-dependent paths used. Section 6.5 describes the effect of the experiments on the travel times and the results are compared to the exact benchmark. In Section 6.6, all insights from the previous sections are combined and the best setup in terms of the number of intervals and how the time-intervals are best determined is discussed. This chapter ends with a conclusion in Section 6.7 which answers the following research questions:

What is the best configuration for the intervals that can be used for the TDS-CH method?

a) What is the potential of using time-dependent routing on the dataset?
b) What is the effect of time intervals on the preprocessing times and the required preprocessing space?
c) How do the time intervals affect the creation time and size of the subgraph?
d) How do the time intervals affect the query times and paths used?
e) How close to optimality are the TDS-CH experiments in terms of travel times compared to the exact method?
f) How can all the observed effects of the time intervals in the experiments be combined in selecting the best configuration?

6.1 Benchmark results

In the previous chapter, two benchmarks are explained, namely the exact method and the time-independent method. The results of the experiments are compared to these two in this chapter. The results in terms of routes and travel time per departure time per scenario for the first benchmark are presented in Section 6.1.1 and the results of the second benchmark in Section 6.1.2. Based on that, Section 6.1.3 discusses the comparison of the two and provides insight into the potential of using time-dependent routing on the dataset.

Similarly to the experiments, the routes for both benchmark methods were computed for 120 departure times in 1000 scenarios. And similar to the experiments, not only do the actual travel times of calculated routes matter but also the precise location of the routes is looked at because this also affects the action problem, customer dissatisfaction.

In analyzing the results of the preprocessing and query times of the benchmarks and the experiments, it is important to understand that all timings depend on what machine the code is running on. In this research, the machine used is a laptop with the Windows 11 operating system. The laptop features an Intel(R) Core(TM) i7-8750H CPU clocked at 2.20GHz (base frequency) and 2.21GHz (max turbo frequency), accompanied by 16 GB of RAM.

6.1.1 Exact method

As explained in Chapter 5, the routes for the exact method were computed using time-dependent Dijkstra. This method requires no preprocessing phase and therefore the only relevant timing aspect that can be looked at is the query time. In Table 6.1, the query times of the exact method are shown.

In all results reported in this chapter, average results are presented but all results are also split up into the four different scenario groups representing the four different types of operations. Firstly, the average path length is presented, which is the average of all routes for all scenarios. Secondly, the total calculation time, which is the sum of all route calculations for all scenarios. Thirdly, the average profile query calculation time, which represents the average calculation time per scenario that consists of all 120 departure times. And fourthly, the average route calculation time, which represents the average calculation time of a single route.

Scenarios	Operation Type	Average path length (km)	Total calculation time (s)	Average profile query calculation time (s)	Average single query calculation time (s)
1 – 1000	**All scenarios**	78.35	221282.59	221.28	1.84
1 – 250	**Retail**	62.46	43320.36	173.28	1.44
251 – 500	**Post/parcel**	101.59	81863.39	327.45	2.73
501 – 750	**Home delivery**	27.16	8389.16	33.56	0.28
751 – 1000	**Random locations**	122.18	87709.68	350.84	2.92

Table 6.1: Exact query times

From Table 6.1, we see that the presented timings vary greatly throughout the different operations. This is a result of the composition of the different operations. As explained in Chapter 3, the route calculation time of the exact (time-dependent) Dijkstra method largely depends on the number of road segments that are visited in determining the exact route. The distance between the start and end locations in scenarios of the post/parcel operations are on average much longer than the distances between the location in the retail operation and the home delivery operations. In Table 6.1, this is reflected in the average route calculation times.

The most important results gathered from the exact method benchmark are the actual travel times per departure time per scenario. The total 120.000 calculated travel times are considered the optimal travel times since these are calculated with the exact method. On their own, the calculated travel times have no meaning. Travel times are only interesting when compared to other travel times. Therefore, later in this chapter, all travel times per departure time per scenario for the different experiments are compared to these benchmark travel times and are used to calculate gaps.

In addition to the travel times, another important insight from this exact method is the different fastest paths that are found. In theory, for every scenario 120 different fastest paths are possible because of the 120 departure times. In practice however, it becomes clear that for each scenario, the number of paths used at least once differs greatly, which is shown in Figure 6.1. TID only produces one path. Based on this, it is clear that in almost 90% of the scenarios multiple routes are found, therefore, the TID is lacking other routing options. However, it should be noted that in some cases the multiple routes only differ in only one or two road segments.

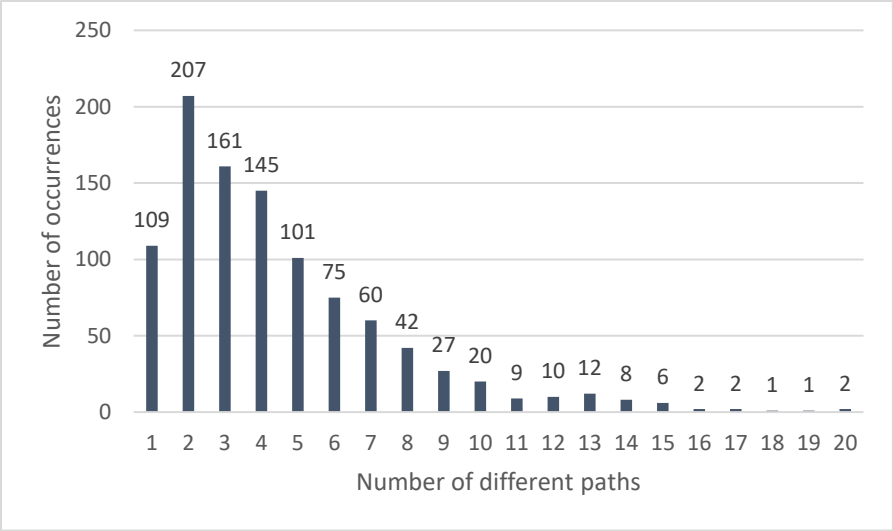


Figure 6.1: Frequency of the number of different paths used

In Table 6.2, the average number of different paths for all scenarios and the average path length are presented.

Scenarios	Operation Type	Average number of different paths	Average path length (km)
1 – 1000	**All scenarios**	4.46	78.35
1 – 250	**Retail**	3.58	62.46
251 – 500	**Post/parcel**	4.77	101.59
501 – 750	**Home delivery**	3.29	27.16
751 – 1000	**Random locations**	6.20	122.18

Table 6.2: Exact path results

From Table 6.2, the conclusion can be drawn that the longer the average path length is, the higher the average number of different paths. Furthermore, when combining the insight from both Table 6.1 and Table 6.2, it can be concluded that a longer average path length corresponds to higher route calculation times.

6.1.2 Time-independent method

For the time-independent method that is used as the second benchmark, the contraction hierarchies algorithm is used. As explained in Chapter 3, this is a fast method of finding routes, but it is time-independent and therefore for this method, the route is always the same in each scenario no matter the departure time.

Since the time-independent method uses contraction hierarchies, preprocessing is necessary. All start and end locations in the dataset are locations in the Netherlands, therefore only the map of the Netherlands was preprocessed. The preprocessing took 6.08 minutes and the locally saved preprocessed map requires 0.14 gigabytes of storage. Similarly to the exact method, for the time-independent method query timings are calculated and the results are presented in Table 6.3. Table 6.3 shows that the query times are higher for longer paths.

Scenarios	Operation Type	Average path length (km)	Total calculation time (s)	Average profile query calculation time (s)	Average single query calculation time (s)
1 – 1000	**All scenarios**	78.35	8830.74	8.83	0.07
1 – 250	**Retail**	62.46	2493.37	9.97	0.08
251 – 500	**Post/parcel**	101.59	2367.33	9.47	0.08
501 – 750	**Home delivery**	27.16	1209.47	4.84	0.04
751 – 1000	**Random locations**	122.18	2760.57	11.04	0.09

Table 6.3: Time-independent path length and query times

6.1.3 Exact and time-independent method comparison

In this section, a comparison between the exact and time-independent methods is made, which is important because it provides insight into the effect of time-dependency in general. The comparison is made based on two KPIs from the previous chapter, namely the total time difference with the TID route and the maximum time difference with the TID route. The KPIs show how much time difference there is between the exact and time-independent methods in terms of travel time for all departure times for every scenario.

To illustrate these two KPIs and how these are used to show the potential of time-dependent routing, an example is created. In Figure 6.2, the exact travel time and the TID travel time are plotted for every departure time for scenario 405. The time difference between the exact and TID travel time is indicated by the shaded area. The sum of all time differences in this scenario is 37.40 minutes. For this specific scenario, the maximum time difference is 7.09 minutes and occurs at departure time 430 (07:10). During rush hours, it can be seen that the result of the time-independent method is substantially different from the exact method. This is the result of the time-independent method always using the same route while there is a faster route available.

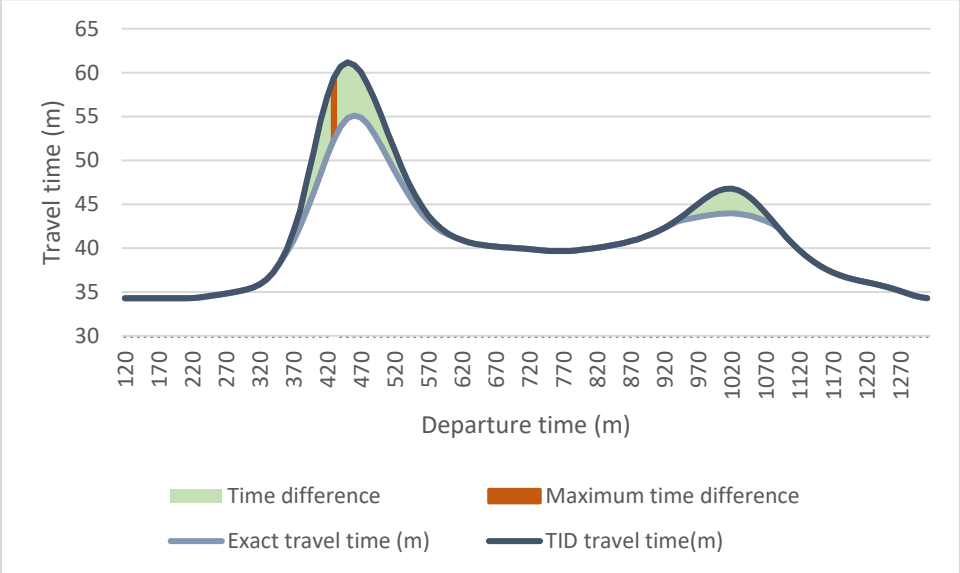


Figure 6.2: Exact and TID travel time comparison

Some insights into the effect of time-dependency in general are based on these KPIs. Out of the 1000 scenarios, 116 scenarios have a total time difference of 0 which means that in those scenarios, the travel time on the time-independent route is the same as the travel time of the exact route.

Therefore, for those 116 scenarios, time-dependency plays no role, and using a time-dependent routing technique, no matter how exact, has no effect. This means that, for the remaining $1000 - 116 = 884$ scenarios (almost 90%), time-dependent routing has an effect.

To provide better insight into how big the effect per scenario is, the maximum time difference is used to calculate the maximum improvement percentage per scenario as follows:

$$Maximum\ improvement\ percentage_{e,s} = \frac{Maximum\ time\ difference_{e,s}\ (m)}{TID\ travel\ time\ at\ time\ of\ maximum\ improvement_{e,s}\ (m)} * 100\%$$

For the example scenario 405 from Figure 6.2 where the TID travel time at departure time 430 is 59.41 minutes, this calculation looks as follows:

$$Maximum\ improvement\ percentage_{exact,405} = \frac{7.09}{59.41} * 100\% = 11.93\%$$

This means that the maximum travel time improvement for this scenario while using the exact method instead of the TID method is 11.93%. In Figure 6.3, leaving out the 116 scenarios where time-dependent routing has no effect, the maximum improvement percentage frequencies are shown.

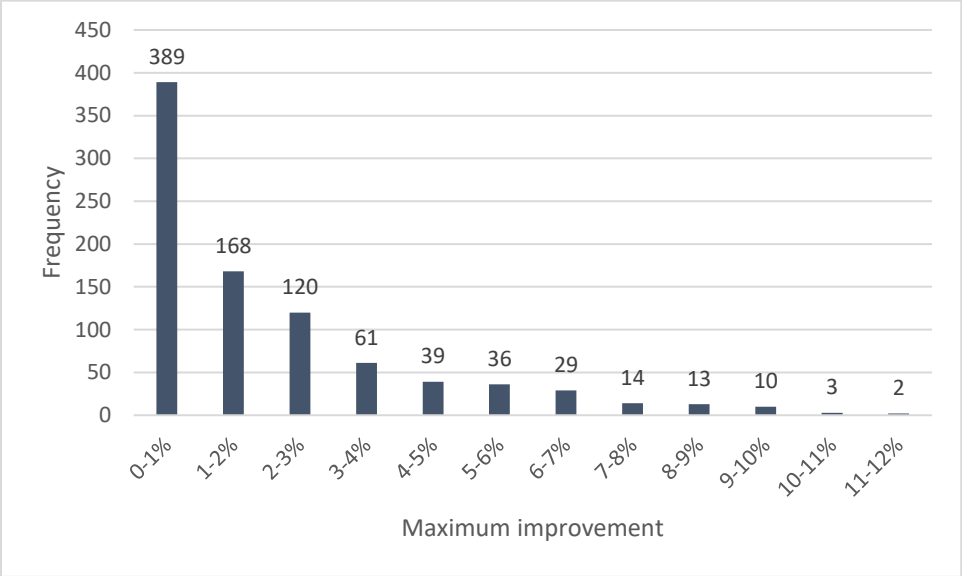


Figure 6.3: Maximum improvement frequency

To provide insight into the effect of time-dependent routing, the maximum improvement percentages are averaged in Table 6.4. The 116 scenarios where time-dependent routing has no effect at all are included in the averages and are split into the different operation types.

Scenarios	Operation Type	Average maximum travel time improvement (%)	Percentage of scenarios without time-dependent routing effect (%)
1 – 1000	**All scenarios**	1.85%	11.60%
1 – 250	**Retail**	1.76%	15.20%
251 – 500	**Post/parcel**	1.64%	9.20%
501 – 750	**Home delivery**	2.44%	16.00%
751 – 1000	**Random locations**	1.54%	6.00%

Table 6.4: Average travel time improvement per operation

Table 6.4 shows that the effect of time-dependent routing in terms of travel time improvement is the biggest for the home delivery operation because the average maximum travel time improvement percentage is highest for that operation. The other operations seem quite similar.

Although the average maximum travel time improvement is not very high, time-dependent routing is still very important for the scenarios with a higher maximum travel time improvement percentage because it leads to lower travel times. Also, time-dependent routing for scenarios with lower maximum travel time percentages is still relevant because while the travel times may not change very much, the time-dependent route can be different than the time-independent route.

6.2 Preprocessing time and required space comparison

In this section, the effect of using time intervals on the preprocessing time and required preprocessing space is presented. As stated before, on their own, the preprocessing speed and space required do not mean much. But in comparison with each other, time and space differences are relevant. Especially the differences in time and space required when an additional time interval is used are interesting.

In the different experiments conducted, the number of intervals varies between 2 and 9. In the TDS-CH method, adding a time interval means that another time-independent graph needs to be processed. For this research, the code of the current contraction hierarchies algorithm used by Simacan was altered in such a way that the preprocessing of multiple time-independent graphs is done simultaneously. Furthermore, the preprocessed time-independent graphs are saved in one file that can be used for the query part of contraction hierarchies. In Table 6.5, the preprocessing time and required preprocessing space depending on the number of time-independent graphs are shown. Just like for the time-independent method, the preprocessed graphs are graphs of the Netherlands. In Figure 6.4, the increasing preprocessing time and required space are visualized.

Number of time intervals	Used in experiments	Preprocessing time (m)	Required preprocessing space (GB)
1	**TID benchmark**	6.08	0.14
2	**1, 9**	7.32	0.16
3	**2, 10**	9.41	0.18
4	**3, 11**	11.45	0.21
5	**4, 12**	12.35	0.23
6	**5, 13**	14.28	0.25
7	**6, 14**	15.92	0.28
8	**7, 15**	17.90	0.30
9	**8, 16**	19.33	0.32

Table 6.5: Preprocessing time and required preprocessing space per number of time intervals used

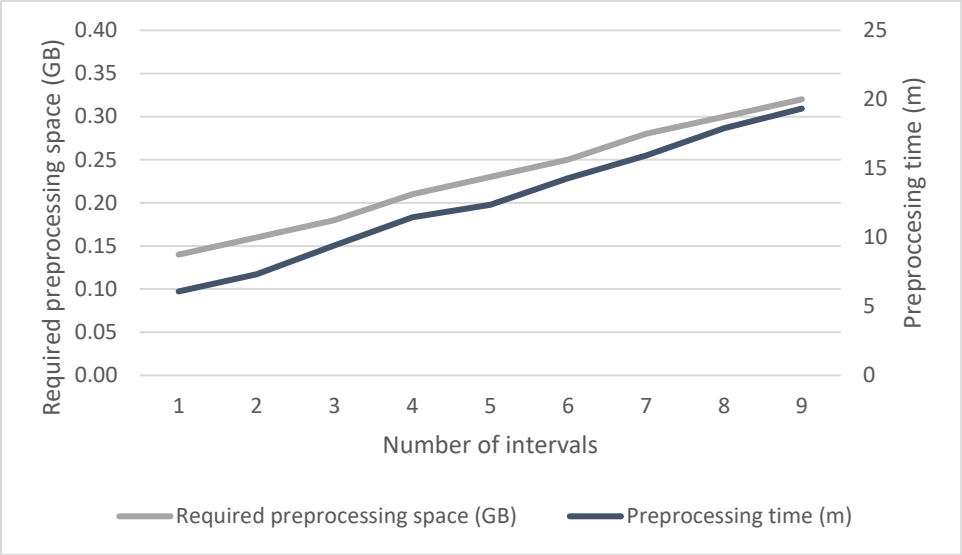


Figure 6.4: Required preprocessing time and space per number of time intervals used

From Table 6.5 and Figure 6.4, it is concluded that preprocessing every additional time-independent graph takes on average 1.66 minutes and requires 0.02 gigabytes more space. Although the growth in both required space and preprocessing time is linear, the growth is not proportional to the number of intervals used. This is because the preprocessing of multiple graphs is done in parallel in these experiments.

Later in this chapter, the insights obtained regarding the required preprocessing time and space versus the number of time intervals used are compared with the experiments' performance in choosing the optimal configuration in terms of what and how many time intervals to use.

6.3 Subgraph comparison

The TDS-CH method used in the experiments consists of the preprocessing phase and the query phase which were explained in Chapter 4. In the previous section, the comparison of the KPIs for the preprocessing phase was discussed. From this section onwards, the KPIs that are part of the query phase are presented.

The query phase of TDS-CH starts when a route is requested until the route is fully calculated. When a route is requested, a subgraph needs to be made. How the subgraph looks depends not only on the start and end location but also on the number of time intervals and which time intervals are used. In Table 6.6 below, the subgraph-related KPIs for all scenarios per experiment are presented. These are the average subgraph creation time, the average subgraph size and the average number of unique TID paths.

	Average subgraph creation time (s)	Average subgraph size (km)	Average number of unique TID paths in the subgraph
Experiment 1	0.04	80.21	1.26
Experiment 2	0.05	84.51	1.73
Experiment 3	0.07	86.47	1.89
Experiment 4	0.09	88.89	2.14
Experiment 5	0.11	88.71	2.17
Experiment 6	0.13	90.47	2.26
Experiment 7	0.15	92.59	2.45
Experiment 8	0.16	93.28	2.53
Experiment 9	0.04	91.16	1.72
Experiment 10	0.06	91.63	1.96
Experiment 11	0.08	93.89	2.27
Experiment 12	0.10	94.56	2.41
Experiment 13	0.13	95.48	2.62
Experiment 14	0.15	95.98	2.71
Experiment 15	0.17	97.99	3.01
Experiment 16	0.15	98.00	3.01

Table 6.6: Average subgraph creation, average subgraph size and average number of unique TID paths in subgraph KPIs per experiment

Based on Table 6.6, multiple conclusions between the equal width interval approach (experiments 1 to 8) and the interval selection method used (experiments 9 to 16) can be derived. Firstly, a correlation between the number of time intervals and the subgraph creation time. Specifically, as the number of time intervals increases, the subgraph creation time also increases. Figure 6.5 shows the comparison of the average subgraph creation time.

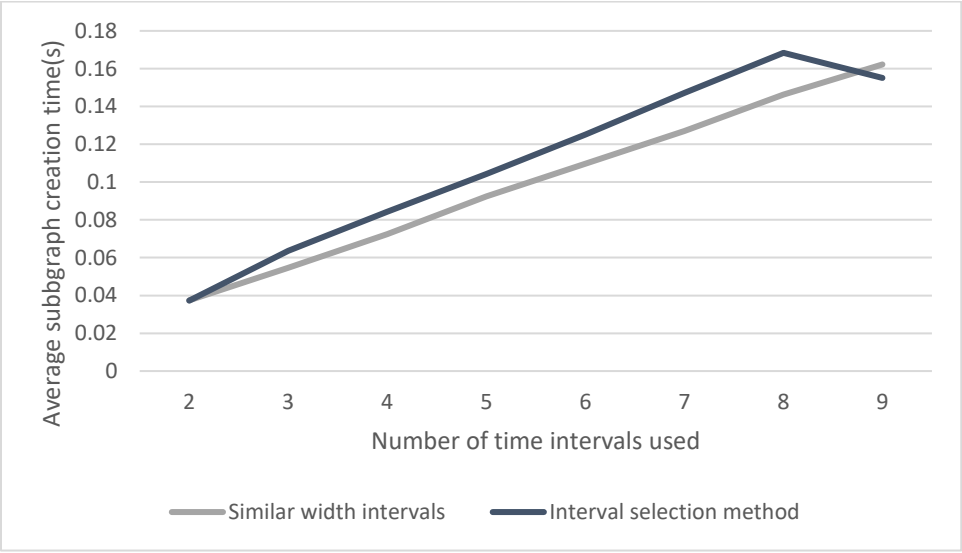


Figure 6.5: Average subgraph creation time versus the number of time intervals used

Another insight worth mentioning is the difference between the equal width approach and the interval selection method. The latter requires, on average, 10.71% more time. This is caused by the higher number of unique routes found that need to be saved for the subgraph. However, it is again worth mentioning that certain time fluctuations could arise from the speed of the code which is influenced by the system used and what tasks the system is performing simultaneously. Therefore,

the time difference results in the research are less important than other results such as the fluctuations in travel times and the routes used among the different experiments.

From Table 6.6 it is also concluded that the more time intervals used leads to bigger subgraphs. When a subgraph is bigger its means that more road segments can be used while finding the time-dependent route on the subgraph. An important observation is that the number of unique TID paths in the experiments based on the interval selection method is always higher than the number of unique paths in the experiments where the equal width interval approach is used. A direct effect of a higher number of unique TID paths being used in the subgraph is that the subgraph size is bigger which is also a result reported in Table 6.6. To what extent the subgraph size being bigger influences the time-dependent route is analyzed later in this section when the results of the subgraph utilization KPI are discussed.

6.4 Query time and path comparison

In this section, the results of KPIs that illustrate the usage of the subgraph are presented. These are the average subgraph utilization, the average number of different TD paths in the subgraph and the average query times. First, the subgraph utilization is addressed. Interestingly, the results for all experiments show similar results, namely that the subgraph utilization for all experiments is 100%. This KPI shows that for the chosen time intervals in all experiments, it results in subgraphs with road segments that are all used at least once in the 120 time-dependent routes searched for on the subgraph. This result shows that it is important to choose the time intervals in such a way that they result in a bigger subgraph because all road segments are used in the optimal routes. The effect that this has on the optimality of travel times is addressed in the next section.

As stated in the first chapter, including time-dependency in routing is not only relevant for predicting the travel times, but also to provide insight into what exact routes are predicted to be driven. In the time-independent situation, the same route is always driven no matter the departure time but by using the exact method in Section 6.1, it was shown that there are many different time-dependent routes used. To show the number of different time-dependent paths used in the experiment, the average number of different TD paths in the subgraph KPI is used. In Table 6.7 the average number of different TD paths in the subgraph per experiment is split between the two different ways of determining the time intervals.

	2	3	4	5	6	7	8	9
Equal width interval approach	1.33	1.94	2.15	2.49	2.50	2.68	3.04	3.12
Interval selection method	2.39	2.55	2.91	3.00	3.22	3.27	3.63	3.63

Table 6.7: Average number of different TD paths per time interval (for two ways of selecting time intervals)

Comparing the two methods shows that the average number of different TD paths is always higher for the interval selection method. This already shows that the results of the interval selection method in terms of travel time will be better than the travel times in experiments with a similar number of intervals because those extra paths are only used when those paths are optimal for a certain departure time. In Section 6.1 however, the exact average number of different TD paths was calculated as 4.46. So, the experiments are never optimal. How close to optimality the experiments are, is discussed in Section 6.5.

Finally, the query time KPI is addressed. Based on the query time KPI, the experiments are compared to each other and the benchmarks. In Section 6.1, the average query times for the benchmarks were presented and those can easily be compared with each other. In the experiments however, comparisons are more difficult because of the TDS-CH method that was used. In this method, two

factors determine the query time, namely: the creation of the subgraph and the time-dependent search on the subgraph. In the experiments conducted, the same subgraph was used 120 times (for every departure time) and was therefore created only once per scenario. By doing this, the time of creating the subgraph was incurred only once per profile query (120 route requests). If in TDS-CH only a single route needs to be computed (single query), however, the subgraph still needs to be created but it then is used only once. Comparison based on query times is therefore done based on the two different query types: the profile query and the single query. In Table 6.8 the average profile query times between the two different ways of determining the time intervals are presented.

	2	3	4	5	6	7	8	9
Equal width interval approach	2.83	2.90	2.89	2.98	2.97	3.00	3.05	3.30
Interval selection method	2.91	3.43	3.51	3.02	3.15	4.26	3.11	3.21

Table 6.8: Average profile query times (s) per time interval (for two ways of selecting time intervals)

Based on Table 6.8, it can be concluded that using more time intervals results in a slightly higher average query time. The differences between the two ways of selecting intervals are not significant. Furthermore, the query times are not very reliable because certain time fluctuations could arise from the speed of the code which is influenced by the system used and what tasks the system is performing simultaneously.

In Section 6.1, the average query time for the exact benchmark was 221.28 seconds which is more than 65 times longer than any results from the experiments. Therefore, the TDS-CH method outperforms the exact method in terms of query speed.

Instead of average profile queries also average single queries are calculated. A single query time consists of the subgraph creation time plus the time-dependent search on the subgraph time. In Table 6.9 the average single query times are presented.

	2	3	4	5	6	7	8	9
Equal width interval approach	0.06	0.08	0.10	0.12	0.13	0.15	0.17	0.19
Interval selection method	0.06	0.09	0.11	0.13	0.15	0.18	0.19	0.18

Table 6.9: Average single query times (s) per time interval (for two ways of selecting time intervals)

Table 6.9 shows that more intervals lead to higher single query times. Similarly to the profile query time averages, the differences between the two ways of selecting intervals are not significant. The average single query time from the exact benchmark was 1.84 seconds which is almost 10 times more than the average single query time when using 9 time intervals. Therefore, it can be included that also for the average single query times the TDS-CH method outperforms the exact method.

Comparing the average profile query times from Table 6.8 with the single query profile times from Table 6.9 shows that the increase in average query times differs between the two different types of queries. Table 6.8 shows that using 9 time intervals instead of 2 time intervals leads to average profile query times that are 1.2 times longer. For the average single query times in Table 6.9, using 9 time intervals instead of 2 time intervals leads to average single query times that are 3.1 times longer. This difference is the result of the subgraph creation time being included in every query of the single query times, while for every profile query, the subgraph creation time is included only once. This shows that when multiple routes with different departure times but for the same start and end location are requested, profile queries are more efficient.

6.5 Travel times comparison

In this section, the actual effects of the different experiments on the travel times are compared to each other and to the benchmark travel times. The KPIs used for this travel time comparison are the total time difference and the maximum time difference and are used to show how close to optimality each of the experiments performs. This section is split up into two sections in which in each one, the comparison based on one KPI is conducted. For all 16 experiments, the results are compared. The results are split up into results for the equal width interval approach (experiments 1 to 8) and the interval selection method (9 to 16).

6.5.1 Comparing based on the total time difference to the TID route

For every scenario in each of the experiments, the total time difference with the TID route is calculated. If only a single scenario in an experiment is considered, the experiment is optimal when the total time difference is equal to the exact benchmark for that scenario. This is illustrated by continuing the example of scenario 405 introduced at the start of this chapter. Figure 6.6 shows, apart from the exact and TID benchmark travel times, also the travel time results for experiments 9, 11, & 15.

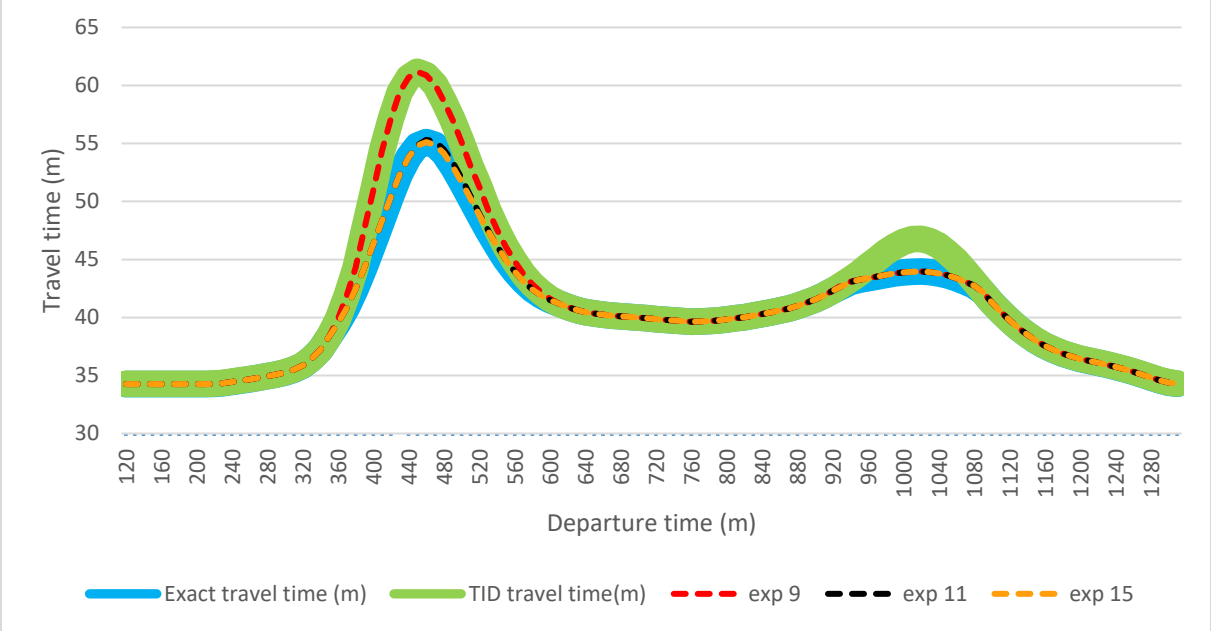


Figure 6.6: Example of travel time comparison between experiments and benchmarks

Figure 6.6 shows the same example as Figure 6.2 in Section 6.1.3, showing the line for the exact travel time in blue and the TID travel time in green. From the visualization of the travel times in Figure 6.6, experiment 9 follows the TID travel time in the morning rush hour, so the route used during that part of the day is not optimal. However, in the evening rush hour, the experiment performs the same as the exact travel time benchmark. Furthermore, experiment 11 is almost similar to the exact benchmark and experiment 15 matches the exact benchmark completely.

Instead of using visual comparison to see the performance, the total time difference with the TID route KPI can be used to check how close to optimality the experiment results are. In Table 6.10, the total time differences with the TID route for the same three experiments of scenario 405 are presented. In the last column of Table 6.10, the *scenario optimality percentage* is introduced. This percentage per scenario is calculated as follows:

$$Scenario\ optimality\ percentage_{e,s} = \frac{Total\ time\ difference\ with\ the\ TID\ route_{e,s}}{Total\ time\ difference\ between\ exact\ and\ TID\ benchmark_s} * 100\%$$

Method	Total time difference with the TID route (m)	Scenario optimality percentage
Exact benchmark	105.89	-
TID benchmark	0	0%
Experiment 9	26.53	25.05%
Experiment 11	103.91	98.13%
Experiment 15	105.89	100.00%

Table 6.10: Example of total time difference with the TID route and scenario optimality percentage for scenario 405

If the scenario optimality percentage is 100% for a certain scenario in an experiment, it means that this experiment results in an optimal solution since it is the same as the exact benchmark. The closer the scenario optimality percentage is to the TID route, the further away from optimality the experiment is. For the example of scenario 405, the scenario optimality percentage of experiment 15 is 100% and therefore experiment 15 yields optimal results for scenario 405. This corresponds with the visualization that was made in Figure 6.6.

The scenario optimality percentage is also used to assess the overall performance of an experiment rather than just a single scenario. To accomplish this, the *average optimality percentage* is calculated:

$$Average\ optimalty\ percentage_e = (\sum_{s=1}^{1000} Scenario\ optimality\ percentage_{e,s})\ /1000$$

Table 6.11 presents the average optimality percentages for the equal width interval approach per experiment. Additionally, Table 6.12 presents the average optimality percentages for the interval selection method. Both tables also include the number of scenarios that achieve optimality, represented by a scenario optimality percentage of 100%. Additionally, the tables present the average optimality percentages and the count of scenarios solved to optimality, both as absolute values and as percentages when considering only the scenarios (884) affected by time-dependent routing. Lastly, the final row of the tables indicates the number of scenarios that have poorer travel time results compared to the TID route.

	Exp 1	Exp 2	Exp 3	Exp 4	Exp 5	Exp 6	Exp 7	Exp 8
Number of time intervals used	2	3	4	5	6	7	8	9
Average optimality percentage[1]	25.06%	53.09%	57.08%	66.47%	66.45%	69.49%	79.66%	81.14%
Number of optimal scenarios[1]	149	248	265	319	333	339	393	426
Average optimality percentage[2]	15.22%	46.94%	51.45%	62.07%	62.04%	65.49%	76.99%	78.67%
Number of optimal scenarios[2]	33	132	149	203	217	223	277	310
Percentage of optimal scenarios[2]	3.73%	14.93%	16.86%	22.96%	24.55%	25.23%	31.33%	35.07%
Number of scenarios worse than the TID route	93	9	1	1	1	1	1	1

[1]out of all 1000 scenarios

[2]out of the 884 scenarios that are affected by time-dependent routing

Table 6.11: Equal width interval approach experiments results

	Exp 9	Exp 10	Exp 11	Exp 12	Exp 13	Exp 14	Exp 15	Exp 16
Number of time intervals used	2	3	4	5	6	7	8	9
Average optimality percentage[1]	66.43%	71.52%	80.59%	81.47%	84.77%	85.12%	93.05%	93.05%
Number of optimal scenarios[1]	284	303	360	374	446	458	573	574
Average optimality percentage[2]	62.03%	67.78%	78.04%	79.04%	82.78%	83.17%	92.13%	92.14%
Number of optimal scenarios[2]	168	187	244	258	330	342	457	458
Percentage of optimal scenarios[2]	19.00%	21.15%	27.60%	29.19%	37.33%	38.69%	51.70%	51.81%
Number of scenarios worse than the TID route	20	5	1	1	1	1	0	0

[1]out of all 1000 scenarios

[2]out of 884 scenarios that are affected by time-dependent routing

Table 6.12: Interval selection method experiments results

Based on Table 6.11 and Table 6.12**,** several comparisons are conducted between the two factors that are experimented with: the way of selecting intervals and the number of intervals used. Firstly, based on the average optimality percentage, the performance of the interval selection method (experiments 9 to 16) compared to the equal width interval approach (experiments 1 to 8) performs better for all variations on the number of time intervals used. The interval selection method also needs fewer time intervals to perform better. For example, experiment 9 shows that the intervals selection method needs only 2 time intervals to reach an average optimality percentage of 66.43% whereas experiment 4 shows that when using the equal width interval approach, 5 time intervals are needed to reach an average optimality percentage of 66.47%.

Secondly, when only scenarios that are affected by time-dependent routing are included, the same pattern can be seen. To provide a more visual comparison between the two different ways of selecting intervals, the average optimality percentage and the number of optimal scenarios results for all scenarios that are affected by time-dependent routing are plotted in Figure 6.7 and Figure 6.8**.** These figures show that the interval selection method outperforms the equal width interval approach.

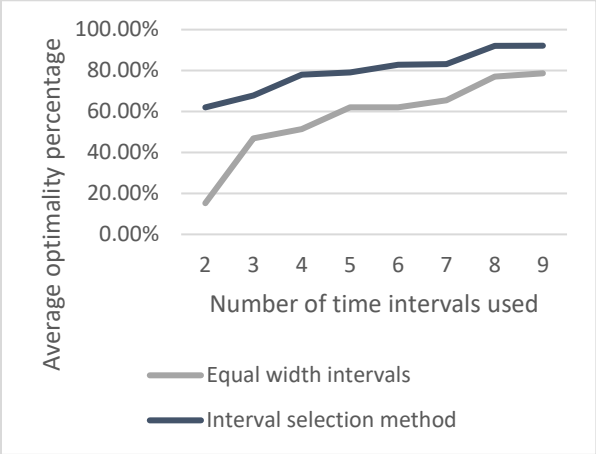


Figure 6.7: Average optimality percentage comparison

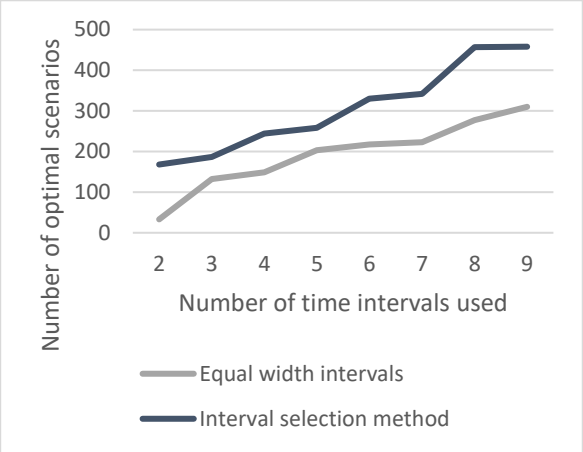


Figure 6.8: Number of optimal scenarios comparison

Another conclusion is that the effect of using more time intervals in the interval selection method leads to a high increase in average optimality percentage when using 2 and 3 time intervals but that after the increase in optimality percentage is more stable.

A third insight from these results is that the number of scenarios that are worse than the TID route in terms of the total time difference with the TID route is smaller for the interval selection method than for the equal width interval approach. A worse result regarding the total time difference with the TID route is caused when the chosen time intervals create a subgraph that does not contain the TID route. The number of scenarios worse than the TID route should therefore be as low as possible.

The tables and figures in this section that are based on comparing the experiments with total time difference with the TID route do show that the interval selection method outperforms the equal width interval approach. However, the results also show that none of the experiments lead to a situation where for all the scenarios optimal routes are found compared to the exact method.

6.5.2 Comparing based on maximum time difference to the TID route

The second KPI used to address the performance of the experiments is the maximum time difference with the TID route. If the total time difference is equal to the exact benchmark, the maximum time difference is automatically also equal to the exact benchmark. Another possibility is that only the maximum time difference is equal to the exact benchmark. In that case, there are still departure times in the scenario that are not yet optimal but the biggest difference between the TID method and the exact method in terms of travel times is then covered.

Similarly to the comparison using the total time difference with the TID route KPI, the maximum total time difference with the TID route in the experiments is also expressed as a percentage of the maximum time difference between the exact and the TID benchmark, this is the scenario maximum time difference percentage:

$$Scenario\ maximum\ time\ difference\ percentage_{e,s} = \frac{Maximum\ time\ difference\ with\ the\ TID\ route_{e,s}}{Maximum\ time\ difference\ between\ exact\ and\ TID\ benchmark_s} * 100\%$$

In the context of this experiment, when the maximum time difference percentage reaches 100% for a specific scenario, it indicates that the experiment has achieved the highest potential time difference. However, it does not imply that all travel time differences compared to the exact benchmark are optimal, as was the case with the scenario optimality difference percentage. When the percentage is lower than 100%, it serves as an indication of how closely the experiment has approached the maximum time difference using the TID route.

To illustrate this, the example of scenario 405 is used again. Table 6.13 shows for three experiments the maximum time difference with the TID route and the scenario maximum time difference percentage.

Method	Maximum time difference with the TID route (m)	Scenario maximum time difference percentage
Exact benchmark	7.09	-
TID benchmark	0	0%
Experiment 1	0	0%
Experiment 2	0.02	0.34%
Experiment 3	7.09	100.00%

Table 6.13: Example of maximum time difference with the TID route for scenario 405

From Table 6.13, it is concluded that for the experiments for this scenario, the highest potential time difference is reached in experiment 3. The maximum time difference percentage is also used to assess the overall performance of an experiment rather than just a single scenario. To accomplish this, the *average maximum time difference percentage* is calculated:

$$Average\ maximum\ time\ difference\ percentage_e = (\sum_{s=1}^{1000} Scenario\ maximum\ time\ differnce\ percentage_{e,s})\ /1000$$

Table 6.14 presents the average maximum time difference percentage achieved by the equal width interval approach in each experiment. Table 6.15 presents the average maximum time difference percentage obtained through the interval selection method. In these tables, only the scenarios (884) affected by time-dependent routing are considered. Both tables also include the number of scenarios that achieve optimality based on the maximum time difference percentage being equal to 100%.

	Exp 1	Exp 2	Exp 3	Exp 4	Exp 5	Exp 6	Exp 7	Exp 8
Number of intervals used	2	3	4	5	6	7	8	9
Average maximum time difference percentage	30.37%	46.17%	49.84%	60.24%	60.01%	63.81%	75.20%	77.05%
Number of scenarios with optimal maximum time difference	170	301	321	402	401	433	520	542

Table 6.14: Equal width interval approach maximum time difference results

	Exp 9	Exp 10	Exp 11	Exp 12	Exp 13	Exp 14	Exp 15	Exp 16
Number of intervals used	2	3	4	5	6	7	8	9
Average maximum time difference percentage	69.77%	70.11%	78.28%	78.88%	81.95%	82.15%	91.79%	91.79%
Number of scenarios with optimal maximum time difference	479	484	557	567	609	611	739	739

Table 6.15: Interval selection method maximum time difference results

The results presented in Table 6.14 and Table 6.15 show that the interval selection method needs fewer time intervals to achieve the same average maximum time difference percentage as the equal width interval approach. For example, the equal width interval approach needs at least 8 time intervals to achieve an average maximum time difference percentage of 70% whereas the interval selection method only needs 3 intervals to achieve similar results. For the number of scenarios with optimal maximum time difference, similar results can be seen. With 9 time intervals, the equal width interval approach has 542 scenarios as the number of scenarios with optimal maximum time difference out of the 884 scenarios possible. The interval selection method achieves already a higher number of scenarios with an optimal maximum time difference after 4 time intervals. The results of Table 6.14 and Table 6.15 are also plotted in Figure 6.9 and Figure 6.10 for visual comparison.

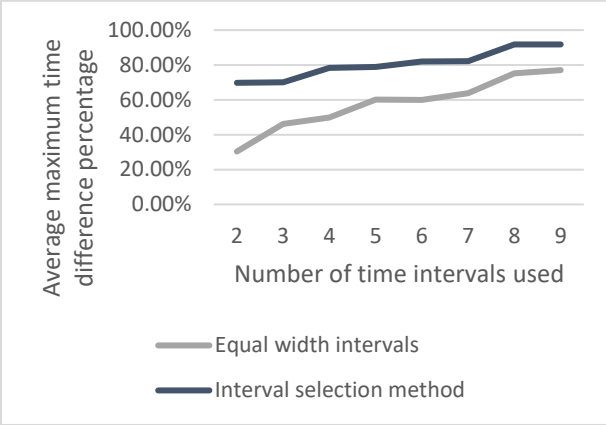


Figure 6.9: Average maximum time difference percentage comparison

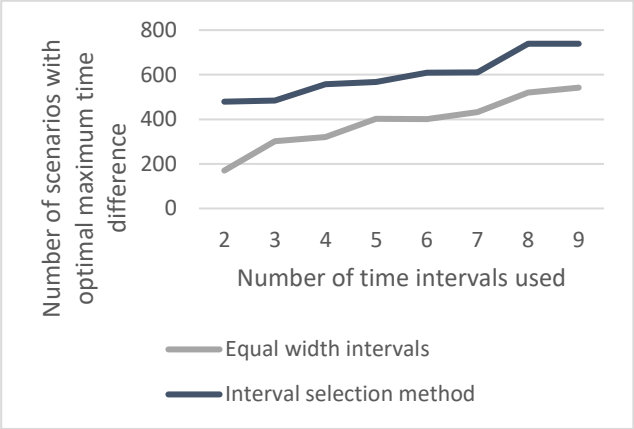


Figure 6.10: Number of scenarios with optimal maximum time difference comparison

Figure 6.9 and Figure 6.10 demonstrate that the interval selection method outperforms the equal width interval approach.

6.6 Choosing the best TDS-CH configuration

Until now, this chapter provided various insights into the effects of the experiments on several KPIs. This section aims to find the best configuration for the time intervals that can be used for the TDS-CH method. For the best configuration, two decisions are made: the way of selecting time intervals and the number of time intervals used.

In this chapter, the equal width interval approach and interval selection method were compared with each other based on several KPIs. In Table 6.16 these KPIs are presented and the method that has favorable results is indicated by “X”.

	Equal width interval approach (experiments 1 to 8)	Interval selection method (experiments 9 to 16)
The average subgraph creation time	X	
The average subgraph size		X
Average number of unique TID paths in the subgraph		X
Average profile query time	X	
Average single query time	X	
Average number of different TD paths in the subgraph		X
Total time difference with the TID route		X
Maximum time difference with the TID route		X

Table 6.16: Ways of selecting time intervals comparison summary

Table 6.16 shows that the only KPIs on which the equal width interval approach performs better are the timing-related KPIs. This is no coincidence because these three KPIs are all related to creating and using the subgraph and the average subgraph is smaller for the equal width interval approach than for the interval selection method. However, a smaller subgraph means that there are fewer options for time-dependent routes to be found which is reflected in the subgraph utilization always being 100%. Therefore, that the equal width interval approach is performing better on those three KPIs does not mean that this approach is favored. Also, the difference between these KPIs is tiny. Taking the KPIs comparison from Table 6.16 and earlier comparisons in this chapter into account, it can be concluded that the interval selection method outperforms the equal width interval approach.

The second decision in finding the best TDS-CH configuration is about the number of time intervals to use and is a much more subjective decision. In this research, it was stated that the general idea is to choose time intervals in such a way that the number of time intervals is minimized while at the same time, the time-dependent routes are as accurate as possible. The intention of minimizing the time intervals originate from the idea that using more time intervals results in higher preprocessing times, more required preprocessing space and higher query times. The KPIs in the experiments confirmed this idea in Section 6.2 and Section 6.4. Despite the preprocessing and query times depending on the machine used, a general trend was still visible. The growth in preprocessing and query times was linear. Section 6.5 confirmed that increasing the number of time intervals indeed leads to more accurate time-dependent routes.

For the application for a company such as Simacan, it can be argued that the preprocessing time and the required space are relatively less important to consider than the other KPIs because those preprocessing KPIs are only relevant when the map is updated and that happens not that often as was explained in Chapter 2. With this considered only the average optimality percentage and the average single query time remain as factors in the tradeoff.

In Figure 6.11 below, the average optimality percentage and the average single query time are plotted for the interval selection method experiments. The labels of each point indicate the number of intervals used.

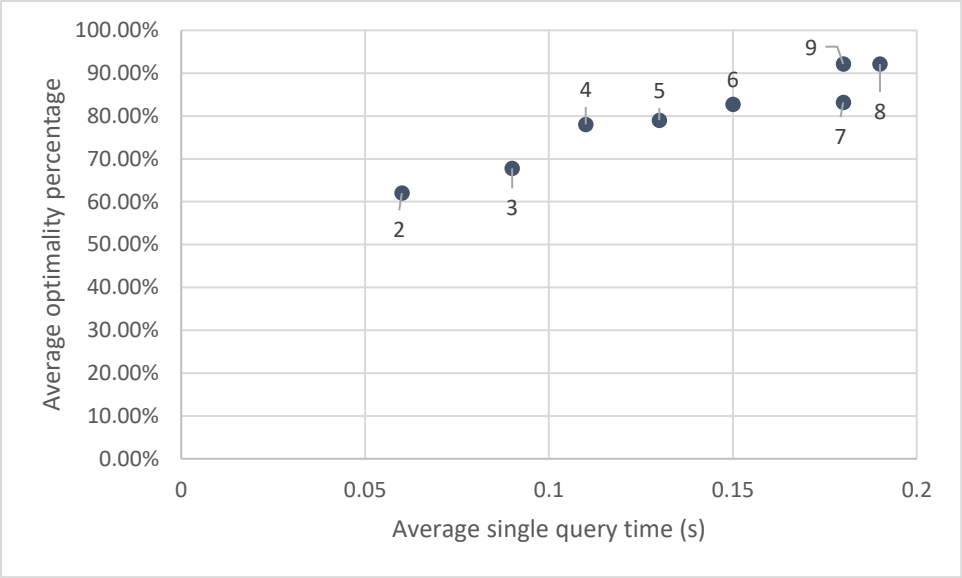


Figure 6.11: Tradeoff between average optimality percentage versus average single query time

From Figure 6.11 it can be concluded that every additional time interval leads to both better optimality and higher query time. However, the increase in average optimality stagnates a bit after the 4th interval is added. After that, the biggest jump in average optimality percentage is when the 8th interval is added. Meanwhile, the increase in average single query time stays quite linear. Logic therefore dictates that the ideal number of time intervals to use is either 4 or 8 for this dataset.

The decision between those two options still depends on the exact application and more specifically on the maximum accepted average single query time. If it is not essential that the average query time is as low as possible, it would be best to use 8 time intervals or even more. On the contrary, if the average query time is considered essential, the better number of time intervals would be 4. In the end, the final decision of what number of intervals is used in the TDS-CH with the interval selection method is up to the user but can be made based on the in this research presented KPIs.

6.7 Conclusion

In this chapter, the results of the experiments are presented and analyzed based on the KPIs formulated in Chapter 5. For the results that included the timing of speeds such as the preprocessing speed and the query speed, it should be noted that the machine used affects the timed results. These timing-related KPIs were still presented because they revealed relevant time differences and trends among the different methods and experiments.

First of all, a comparison between the exact and time-independent benchmarks on the total time difference and maximum time difference showed that in 884 scenarios the time-independent route was different, and therefore not optimal, to the exact time-dependent route. Furthermore, in 891 scenarios, it was shown that two or more paths were found, whereas the time-independent route always only finds one path. Therefore, we conclude that time-dependency plays a role in routing for most of the scenarios in the dataset and this shows that time-dependent routing adds value for almost 90% of the scenarios.

Secondly, the experiments were compared to the two benchmarks in various ways. Since the time-independent method only has to preprocess the map for one route, the required preprocessing time and preprocessing space are smaller than that of the experiments. The times vary between 1 and 3

times as small, and the space between 1 and 2.5 times as small, which can be plotted as a linear function.

The comparison of the different subgraph KPIs which were the subgraph creation time, subgraph size and the number of unique TID paths in the subgraph, showed that using the interval selection method always resulted in larger subgraphs than when using the equal width interval approach. These larger subgraphs made it possible to find more optimal routes in the query phase because no matter the size of the subgraph, the subgraph utilization was always 100%. This means that all road segments in the subgraph were used at least once in the 120 time-dependent routes searched for. Therefore, a bigger subgraph provides, on average, better results.

When comparing the number of different time-dependent paths found, the average for the equal width approach was on average between 1.33 and 3.12, and between 2.39 and 3.63 for the interval selection method. This shows that the interval selection method outperforms the equal width interval approach on this KPI. However, with 4.46 different TD paths for the exact method, the interval selection method is still not optimal. When looking at the profile and single query times, these showed only minor differences between the equal width interval approach and interval selection method. However, compared to the exact benchmark, TDS-CH can be 100 times faster.

Finally, a comparison was made based on the travel times. First through the scenario optimality percentage, which is the total time difference with the TID route divided by the total time difference between the exact and TID benchmark, where 100% is an optimal score. For the 884 scenarios, the percentages ranged from 15% to 79% for the equal width interval approach and between 62% and 92% for the interval selection method. Furthermore, the interval selection method always needs fewer time intervals than the equal width interval approach to reach similar results in terms of the average optimality percentage and the number of optimal scenarios. For the maximum time difference percentage, the equal width interval approach was between 30% and 77% and the interval selection method was between 70% and 92%. Therefore, the interval selection method again outperforms the equal width interval approach.

Taking this all into account, the interval selection method outperforms the equal width interval approach on all relevant KPIs. This makes the interval selection method the better choice. Nevertheless, the results also indicate that for none of the experiments, all scenarios were optimal. The decision on how many time intervals to use is a more subjective decision. The tradeoff between the query times and average optimality percentage showed that based on the used dataset, 4 or 8 time intervals should be used depending on the application of TDS-CH. In these cases, the average optimality percentages are 78.04% and 92.13% respectively, with average single query times of 0.11 and 0.19 seconds. The decision should be made based on what additional query time is acceptable to further increase the average optimality percentage.

7 Conclusion & Recommendations

In this chapter, the conclusion of this research and the recommendations for Simacan are presented.

7.1 Conclusion

From the analysis of the current situation at Simacan, it became clear that routing within the Simacan platform is essential for many different services offered. In the Simacan context and in this research, the term routing is used to describe to process of finding the shortest path between two locations. Currently, Simacan uses the contraction hierarchies algorithm to calculate routes but although very fast, the calculated routes are time-independent. Using time-independent routes has several effects such as inaccurate ETAs and visualizing the wrong routes which in the end all result in customer dissatisfaction. It therefore became clear that time-dependency should be included in the current routing process. To do so, research questions were formulated to reach the following research objective.

"Find an extension to the contraction hierarchies method that supports time-dependent routing."

For the method to be applicable for Simacan, it is required that this method is fast and able to deal with large road networks. Additionally, the method should be able to work together with the currently used contraction hierarchies algorithm. In literature, exact methods and heuristics are found that include time-dependency within contraction hierarchies. Based on a comparison between the different potential solutions found and their applicability to the Simacan context, the Time-Dependent Sampling heuristic is chosen. Time-Dependent Sampling relies on time intervals that are used to calculate time-independent routes which are combined into a subgraph. This subgraph can then be used to find the time-dependent route on. The Time-Dependent Sampling heuristic and the contraction hierarchies algorithm are combined into TDS-CH.

The contribution to theory of this research is to find a way to determine the number and what time intervals should be used based on a method to avoid user-defined or a random number of time intervals. Having the preprocessing phase and query phase of contraction hierarchies, a step before this is designed, called the interval selection phase. In this phase, the interval selection method is used which is a four-step method designed to find the time intervals that should be used:

1) obtaining speed profile information,
2) creating the weighted speed profile,
3) dividing the weighted speed profile into periods of similar speed,
4) selecting the most diverse time intervals

To test the method, experiments are designed for scenarios between 02:00 and 22:00, varying the number of intervals between 2 and 9, for four different operation types. The interval selection method is compared to the same experiments that used the equal width interval approach, a time-independent route, and the exact Dijkstra algorithm.

Comparing the different experiments with each other and with the time-independent and exact benchmark based on the KPIs leads to various insights and results. Firstly, the results show that for almost 90% of the scenarios, time-dependency is relevant. Secondly, implementing TDS-CH results in a longer preprocessing time and more required preprocessing space but the increase is linear and the preprocessing can be parallelized. Thirdly, TDS-CH with 9 time intervals also takes on average 2.6 times more time for a single query than the time-independent benchmark but is on average 9.9 times faster than the exact benchmark.

Fourthly, the average optimality percentages, which show the potential travel time difference between the time-independent and the exact benchmark, of the experiments show promising results. The experiment using 9 time intervals that are found with the interval selection method results in an average optimality percentage of 92.14%. On the contrary, the experiment using 9 equal width intervals results in an average optimality percentage of 78.67%. Not only for 9 time intervals but also for all other experiments with a different number of time intervals used, the interval selection method outperforms the equal width interval selection approach. Other KPIs, such as the average subgraphs size and the number of different time-dependent routes found, also confirm that the interval selection method always outperforms the equal width interval selection approach.

The decision on how many time intervals to use is more subjective. The tradeoff between the query times and average optimality percentage showed that based on the used dataset, 4 or 8 time intervals should be used depending on the application of TDS-CH. In these cases, the average optimality percentages are 78.04% and 92.13% respectively, with average single query times of 0.11 and 0.19 seconds. The decision should be made based on what additional query time is acceptable to further increase the average optimality percentage.

Taking this all into account, this research fulfills the research objective of finding an extension to the contraction hierarchies method that supports time-dependent routing and shows that time-dependent routing adds value compared to time-independent routing. Nevertheless, the method has some limitations and there are suggestions for further research, as explained in the next chapter.

7.2 Recommendations

Based on this research, several recommendations are formulated for Simacan. Firstly, I recommend Simacan to implement the Time-Dependent Sampling heuristic as their method to include time-dependency in their route calculations. Namely, this research shows that including time-dependency in routing has a positive effect on the travel times and using time-independent routing leads to unnecessary high travel times. Time-Dependent Sampling can be used together with the current implementation of the contraction hierarchies algorithm and is proven to perform well in terms of relative error, preprocessing times, query times and required preprocessing space.

Secondly, the results of the experiments show that the interval selection method achieves close to optimal results on the used dataset. I therefore advise Simacan to implement the interval selection method combined with the TDS-CH. However, some new experiments should be done. I recommend running an experiment with 9 time intervals but for a larger number of scenarios (e.g. 10000) and checking whether the KPIs have similar results. Also, experiments on bigger maps such as Europe are recommended. The most important KPI for these experiments is the single query time. This time should be fast enough to be applicable in practice. If the results prove fast enough, experiments with even more time intervals are advised to check whether even better results in terms of optimality can be realized. Note that more time intervals will increase the preprocessing time, and it is up to Simacan whether an even better result is worth it.

Thirdly, I recommend Simacan to further investigate the usage of the proposed TDS-CH method with the interval selection method for their current operation. In this research, many Simacan-specific factors that are used in their current routing method were simplified and not used because it would lead to results not being generic enough. An example is the different vehicle types used in the current routing process that are used to set preferences for certain road types. Simacan could test how these factors would affect the results of the TDS-CH from this research.

The final recommendation for Simacan is that the implementation of TDS-CH can be used to check to what extent the predicted time-dependent routes are similar to the realized route of Simacan's customers. Insights on the similarity could help Simacan to further tailor their routing processes.

8 Discussion

In this chapter, the contribution to theory, the research limitations and suggestions for future research are discussed.

8.1 Contribution to theory

The contribution to theory of this research is the newly created interval selection method in which intervals are chosen using a 4-step method based on map data and the importance of road segments. This interval selection method makes it possible to select intervals that can be used for the TDS-CH heuristic, but it can also be applied in other methods that depend on interval selection. By using the interval selection method, the decision regarding the time intervals to be used is no longer based on intuition but is now fully data-driven. Additionally, the performance of the interval selection method was evaluated by comparing multiple KPIs with those of another method, as well as time-independent and exact benchmarks. The results demonstrated the good performance of the interval selection method.

8.2 Limitations and suggestions for future research

In this research, there were also limitations. First of all, the machine and programming language (Python) used caused the computational times of the methods used to become quite long. The long computational time of the experiments affected some experimental design decisions such as the size of the dataset, the number of experiments, the size of the map used and the departure in the scenarios being ten minutes apart instead of for example one minute, which would have made the experiments more accurate.

A second limitation of this research are the design choices that were made in the interval selection method. The interval selection method was created from scratch based on the ideas of the researcher, literature research, data analysis and experimentation. This method is completely new and therefore many design choices such as what makes a speed profile important, the construction of the weighted speed profile and the ordering of the found periods had to be made and worked out. All these decisions affect to some extent the outcomes of the experiments. Instead of the decisions made, also other decisions or approaches could have been thought of.

The last limitation also leads to future research that could be done. The design choices made in the interval selection method could be further researched, extended and isolated to test the individual effect of each choice. Also, the experiments showed that although the results were quite good, the results while using 9 time intervals were not optimal compared to the exact method. Therefore, experiments with even more time intervals could be studied.

Another topic for future research is to implement the interval selection method in other routing techniques. In this research, the interval selection method functioned as a separate phase before the preprocessing phase of TDS-CH. TDS sampling can also be used with other routing algorithms instead of CH, and it could be interesting to see how the interval selection method would fit into these other approximation methods.

References

Ahuja, R. K., Mehlhorn, K., Orlin, J., & Tarjan, R. E. (1990). Faster algorithms for the shortest path problem. *Journal of the ACM*, *37*(2), 213–223. https://doi.org/10.1145/77600.77615

Bang-Jensen, J., & Gutin, G. Z. (2009). *Digraphs*. Springer. https://doi.org/10.1007/978-1-84800-998-1

Bast, H., Delling, D., Goldberg, A., Müller-Hannemann, M., Pajor, T., Sanders, P., Wagner, D., & Werneck, R. F. (2015). *Route Planning in Transportation Networks* (arXiv:1504.05140). arXiv. http://arxiv.org/abs/1504.05140

Batz, G. V., Delling, D., Sanders, P., & Vetter, C. (2009). Time-Dependent Contraction hierarchies. In I. Finocchi & J. Hershberger (Red.), *2009 Proceedings of the Eleventh Workshop on Algorithm Engineering and Experiments (ALENEX)* (pp. 97–105). Society for Industrial and Applied Mathematics. https://doi.org/10.1137/1.9781611972894.10

Batz, G. V., Geisberger, R., Neubauer, S., & Sanders, P. (2010). Time-Dependent Contraction hierarchies and Approximation. In P. Festa (Red.), *Experimental Algorithms* (pp. 166–177). Springer. https://doi.org/10.1007/978-3-642-13193-6_15

Batz, G. V., Geisberger, R., Sanders, P., & Vetter, C. (2013). Minimum time-dependent travel times with contraction hierarchies. *ACM Journal of Experimental Algorithmics*, *18*, 1.4:1.1-1.4:1.43. https://doi.org/10.1145/2444016.2444020

Dantzig, G. B. (1963). *Linear Programming and Extensions*. 224.

Delling, D., Sanders, P., Schultes, D., & Wagner, D. (2009). Engineering Route Planning Algorithms. In J. Lerner, D. Wagner, & K. A. Zweig (Red.), *Algorithmics of Large and Complex Networks: Design, Analysis, and Simulation* (pp. 117–139). Springer. https://doi.org/10.1007/978-3-642-02094-0_7

Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, *1*(1), 269–271. https://doi.org/10.1007/BF01386390

Dreyfus, S. E. (1969). An Appraisal of Some Shortest Path Algorithm. *Oper. Res.*, *17*, 395–412. https://doi.org/10.1287/opre.17.3.395

Geisberger, R., Sanders, P., Schultes, D., & Delling, D. (2008). Contraction hierarchies: Faster and Simpler Hierarchical Routing in Road Networks. In C. C. McGeoch (Red.), *Experimental Algorithms* (pp. 319–333). Springer. https://doi.org/10.1007/978-3-540-68552-4_24

Gendreau, M., Ghiani, G., & Guerriero, E. (2015). Time-dependent routing problems: A review. *Computers & Operations Research*, *64*, 189–197. https://doi.org/10.1016/j.cor.2015.06.001

Nannicini, G., Delling, D., Schultes, D., & Liberti, L. (2012). Bidirectional A* Search on Time-Dependent Road Networks. *Networks*, *59*, 240–251. https://doi.org/10.1002/net.20438

Saha Ray, S. (2013). *Graph Theory with Algorithms and its Applications*. Springer. https://doi.org/10.1007/978-81-322-0750-4

Sanders, P., & Schultes, D. (2006). Engineering Highway Hierarchies. In Y. Azar & T. Erlebach (Red.), *Algorithms – ESA 2006* (pp. 804–816). Springer. https://doi.org/10.1007/11841036_71

Sanders, P., & Schultes, D. (2007). Engineering Fast Route Planning Algorithms. In C. Demetrescu (Red.), *Experimental Algorithms* (pp. 23–36). Springer. https://doi.org/10.1007/978-3-540-72845-0_2

Schultes, D., & Sanders, P. (2007). Dynamic Highway-Node Routing. In C. Demetrescu (Red.), *Experimental Algorithms* (pp. 66–79). Springer. https://doi.org/10.1007/978-3-540-72845-0_6

Strasser, B. (2017). Dynamic Time-Dependent Routing in Road Networks Through Sampling. In G. D'Angelo & T. Dollevoet (Red.), *17th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS 2017)* (Vol. 59, p. 3:1-3:17). Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. https://doi.org/10.4230/OASIcs.ATMOS.2017.3

Strasser, B., Wagner, D., & Zeitz, T. (2021). Space-Efficient, Fast and Exact Routing in Time-Dependent Road Networks. *Algorithms*, *14*(3), Art. 3. https://doi.org/10.3390/a14030090

TomTom. (2023). *Speed Profiles*. TomTom. https://www.tomtom.com/products/speed-profiles/

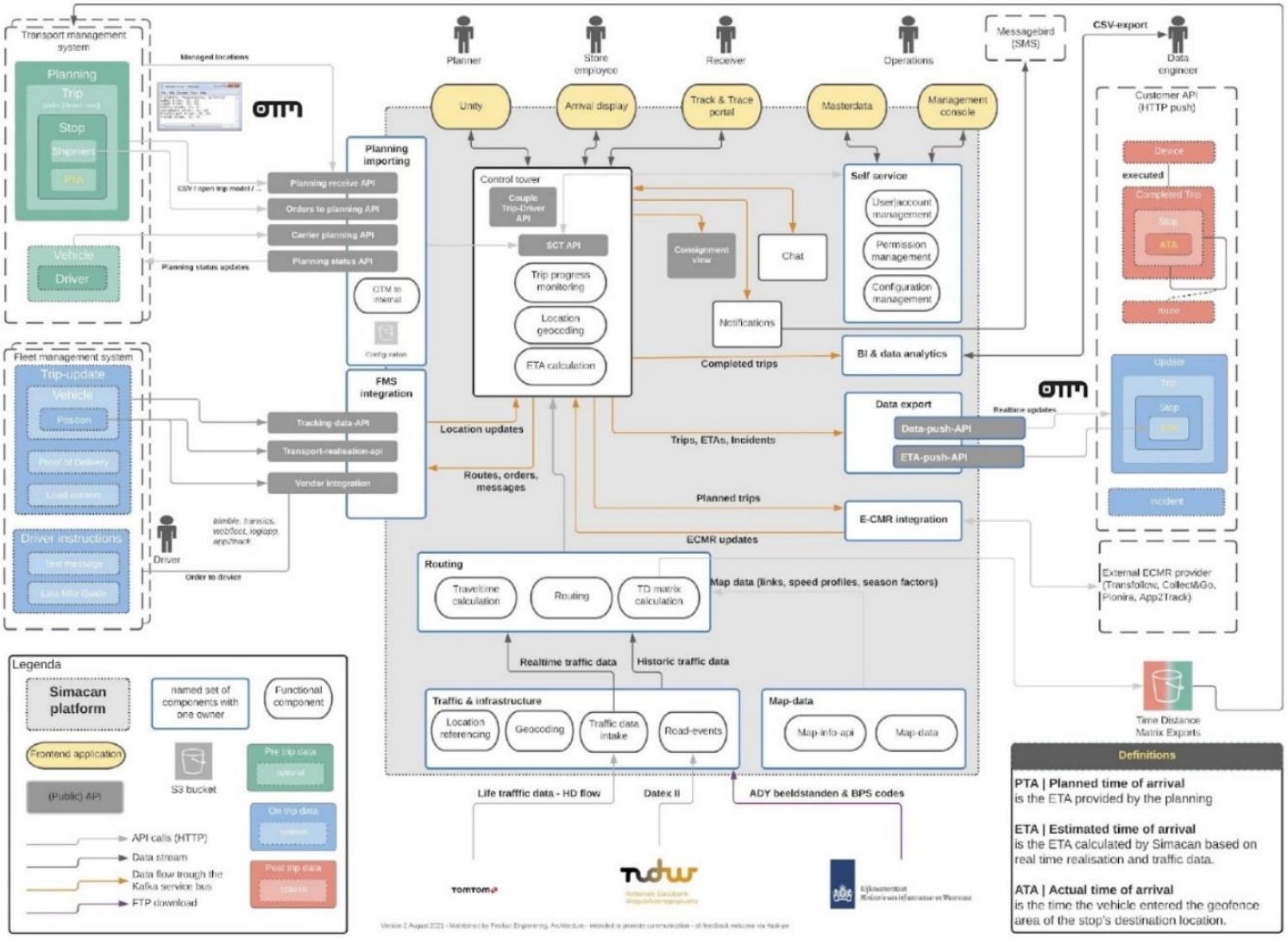
Traffic Stats API FAQ. (2022, augustus 25). [TomTom]. Traffic Stats API FAQ. https://developer.tomtom.com/traffic-stats/documentation/product-information/faq

Van den Eynde, S., Audenaert, J. V. P., Derudder, B., Colle, D., & Pickavet, M. (2020). A low-memory alternative for time-dependent Dijkstra. *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, 1–6. https://doi.org/10.1109/ITSC45102.2020.9294640

Appendices

Appendix A: Component overview

In the figure below, the Simacan component overview is shown. This figure illustrates how all the different components are connected. The Simacan platform (grey box) is shown in the middle, the inputs of the platform are shown on the left and the outputs are shown on the right. On the bottom, real-time traffic data that is bought from other organizations and used as input is shown. In this data flow, a distinction between pre-trip (green boxes), on-trip (blue boxes) and post-trip data (red boxes) are made. Examples of data that are used as inputs include planning data (trips, stops, planned time of arrival) which is pre-trip data. Both tracking data (from fleet management systems) and real-time traffic data are examples of on-trip data. Outputs can be updates (trip, stops, estimated time of arrivals (ETAs)) and incidents which are both on-trip data. This data stream is continuously updated to provide real-time information to the customers. The last outputs, which are provided as post-trip data, include for example the actual time of arrival.


Transport management system
Planning
Trip
Stop
Shipment
PTA
Vehicle
Driver
Managed locations
Planning receive API
Orders to planning API
Carrier planning API
Planning status API
Planning status updates
Planning importing
OTM to internal
Configuration
Fleet management system
Trip-update
Vehicle
Position
Proof of Delivery
Driver instructions
Text message
Last Mile Guide
Driver
Order to device
Tracking-data-API
Transport-realisation-api
Vendor integration
FMS integration
Location updates
Routes, orders, messages
Planner
Store employee
Receiver
Operations
Messagebird (SMS)
CSV-export
Data engineer
Unity
Arrival display
Track & Trace portal
Masterdata
Management console
Control tower
Couple Trip-Driver API
SCT API
Trip progress monitoring
Location geocoding
ETA calculation
Consignment view
Chat
Notifications
Completed trips
Self service
User/account management
Permission management
Configuration management
BI & data analytics
Data export
Data-push-API
ETA-push-API
Realtime updates
Trips, ETAs, Incidents
Planned trips
E-CMR integration
ECMR updates
Routing
Traveltime calculation
Routing
TD matrix calculation
Map data (links, speed profiles, season factors)
Realtime traffic data
Historic traffic data
Traffic & infrastructure
Location referencing
Geocoding
Traffic data intake
Road-events
Map-data
Map-info-api
Map-data
Life trafffic data - HD flow
Datex II
ADY beeldstanden & BPS codes
Customer API (HTTP push)
Device
executed
Completed Trip
Stop
ATA
Route
Update
Trip
Stop
ETA
Incident
External ECMR provider (Transfollow, Collect&Go, Pionira, App2Track)
Time Distance Matrix Exports
Legenda
Simacan platform
named set of components with one owner
Functional component
Frontend application
(Public) API
S3 bucket
Pre trip data
optional
On trip data
optional
Post trip data
optional
API calls (HTTP)
Data stream
Data flow trough the Kafka service bus
FTP download
Definitions
PTA | Planned time of arrival
is the ETA provided by the planning
ETA | Estimated time of arrival
is the ETA calculated by Simacan based on real time realisation and traffic data.
ATA | Actual time of arrival
is the time the vehicle entered the geofence area of the stop's destination location.

Appendix B: Speed profile info

In this appendix, the maximum impact, the time of maximum impact and the average impact for each speed profile is shown. An explanation of this data is included in Section 4.2.2.1.

Speed profile ID	Maximum impact	Time of maximum impact	Average impact
0	31.7%	8:15	74.4%
1	80.1%	18:15	94.1%
2	80.3%	7:25	96.8%
3	17.6%	16:25	76.2%
4	89.2%	20:50	96.3%
5	44.0%	17:00	80.5%
6	21.6%	17:45	69.2%
7	78.2%	17:00	92.9%
8	87.0%	5:55	98.5%
9	28.1%	16:55	68.6%
10	76.6%	18:20	88.6%
11	63.1%	12:10	87.1%
12	90.3%	6:05	95.5%

Speed profile ID	Maximum impact	Time of maximum impact	Average impact
74	51.3%	16:50	80.3%
75	23.2%	7:35	87.6%
76	24.6%	17:55	67.5%
77	69.3%	16:55	95.1%
78	77.2%	17:00	90.6%
79	56.2%	17:10	93.2%
80	27.2%	15:15	59.0%
81	68.7%	8:00	96.6%
82	79.5%	11:20	87.7%
83	81.4%	8:10	91.5%
84	58.9%	17:15	83.4%
85	77.5%	20:40	94.4%
86	40.9%	7:40	84.1%

Speed profile ID	Maximum impact	Time of maximum impact	Average impact
147	37.1%	16:45	75.3%
148	40.5%	17:30	72.4%
149	74.2%	10:25	90.3%
150	50.9%	18:35	79.6%
151	84.8%	14:00	93.6%
152	30.2%	15:15	66.4%
153	21.0%	17:05	69.5%
154	70.5%	17:20	87.8%
155	55.3%	18:25	90.0%
156	85.5%	10:50	94.3%
157	32.0%	8:10	88.5%
158	50.4%	15:15	73.1%
159	26.9%	17:25	76.2%

Speed profile ID	Maximum impact	Time of maximum impact	Average impact
220	73.5%	16:50	90.7%
221	54.7%	16:35	88.5%
222	16.7%	15:45	60.3%
223	75.2%	7:40	88.3%
224	53.0%	8:00	80.3%
225	82.0%	6:50	92.1%
226	14.8%	16:25	55.6%
227	29.6%	16:50	59.4%
228	19.4%	16:40	63.6%
229	88.0%	17:00	94.6%
230	45.2%	16:40	81.3%
231	36.2%	15:45	66.1%
232	19.6%	17:05	63.1%

13	68.6%	8:20	85.5%
14	91.5%	6:55	98.8%
15	39.7%	16:45	77.9%
16	42.4%	17:05	76.0%
17	72.6%	19:20	91.5%
18	38.6%	19:05	71.4%
19	91.0%	16:55	98.6%
20	64.1%	17:30	91.0%
21	42.5%	17:05	82.0%
22	94.0%	17:50	99.1%
23	27.8%	18:30	67.9%
24	32.5%	17:05	69.7%
25	17.1%	16:10	49.2%
26	72.8%	12:15	88.5%
27	83.0%	16:05	95.4%
28	79.6%	14:45	93.0%

87	50.1%	18:40	88.2%
88	50.1%	17:25	85.5%
89	21.2%	18:15	61.3%
90	100.0%	0:00	100.0%
91	29.6%	8:10	89.2%
92	36.5%	17:20	72.8%
93	32.5%	18:25	83.1%
94	72.5%	11:05	86.3%
95	52.0%	16:50	92.5%
96	68.5%	7:25	91.5%
97	73.1%	7:55	93.6%
98	35.7%	7:40	75.9%
99	41.3%	16:20	76.3%
100	48.8%	17:00	86.9%
101	42.7%	11:20	72.5%
102	52.5%	12:15	82.1%

160	89.4%	7:25	94.9%
161	70.4%	17:35	93.1%
162	64.5%	16:00	83.0%
163	45.1%	8:05	87.3%
164	69.8%	7:45	85.7%
165	70.4%	17:05	87.6%
166	38.7%	16:25	62.3%
167	55.3%	17:05	81.3%
168	72.6%	15:40	85.7%
169	62.8%	16:20	85.7%
170	82.4%	18:20	91.5%
171	77.1%	17:40	90.5%
172	63.1%	7:35	95.1%
173	78.4%	16:50	90.2%
174	34.3%	16:45	66.4%
175	60.7%	16:20	76.3%

233	38.8%	18:50	75.9%
234	28.8%	17:00	74.4%
235	76.3%	8:10	93.7%
236	41.9%	8:05	69.6%
237	26.4%	17:50	78.4%
238	77.4%	16:50	89.3%
239	54.6%	8:15	84.6%
240	53.8%	8:10	93.4%
241	33.5%	17:10	88.6%
242	26.9%	18:40	80.6%
243	66.2%	16:55	86.2%
244	37.2%	16:55	82.9%
245	25.5%	17:05	78.6%
246	54.4%	7:55	85.9%
247	42.0%	17:20	80.2%
248	39.4%	18:00	73.5%

29	28.9%	16:50	87.7%
30	64.5%	8:10	89.8%
31	34.9%	15:05	70.5%
32	62.4%	7:40	90.9%
33	46.9%	15:55	82.1%
34	69.3%	18:20	85.2%
35	82.2%	6:55	96.2%
36	80.3%	8:10	93.3%
37	60.1%	17:50	88.0%
38	50.1%	17:35	74.4%
39	80.7%	8:30	96.8%
40	61.6%	10:35	84.5%
41	90.1%	6:10	97.6%
42	47.8%	17:55	79.4%
43	47.2%	11:45	75.1%
44	66.7%	17:50	93.8%

103	67.8%	14:45	87.8%
104	57.1%	17:20	94.7%
105	80.7%	8:50	94.9%
106	55.1%	8:15	85.1%
107	62.0%	16:35	85.1%
108	67.8%	19:15	87.7%
109	56.7%	8:15	89.2%
110	36.9%	15:20	69.1%
111	55.4%	17:40	79.2%
112	17.3%	16:45	66.2%
113	43.5%	15:55	72.0%
114	44.6%	17:50	80.6%
115	77.4%	7:55	96.6%
116	34.5%	17:20	91.0%
117	25.4%	18:05	72.8%
118	26.9%	17:35	60.6%

176	61.3%	17:55	83.9%
177	57.0%	16:40	78.9%
178	51.9%	17:00	84.4%
179	24.0%	11:45	62.2%
180	60.5%	6:10	93.2%
181	81.9%	8:20	91.8%
182	94.4%	8:20	98.8%
183	89.3%	10:05	93.8%
184	68.9%	16:50	88.6%
185	52.7%	16:50	73.5%
186	38.9%	15:55	73.2%
187	53.2%	8:10	87.5%
188	77.4%	7:40	94.0%
189	34.0%	8:15	81.6%
190	73.4%	17:30	96.7%
191	65.2%	11:05	89.3%

249	29.4%	12:15	71.7 %
250	21.9%	17:35	85.2 %
251	42.4%	7:55	87.5 %
252	67.4%	8:10	90.2 %
253	40.1%	16:25	84.5 %
254	45.0%	17:35	90.9 %
255	72.3%	7:30	83.9 %
256	81.0%	11:50	93.3 %
257	57.5%	15:00	81.6 %
258	29.9%	8:00	68.1 %
259	63.9%	15:10	84.1 %
260	83.8%	10:30	90.5 %
261	53.2%	15:20	78.6 %
262	68.8%	8:15	93.5 %
263	69.5%	17:20	92.2 %
264	39.9%	10:30	74.3 %

45	54.9%	16:10	82.2%
46	44.0%	7:35	92.0%
47	82.4%	8:00	95.4%
48	58.6%	15:40	78.7%
49	24.2%	16:35	83.7%
50	74.3%	18:15	91.6%
51	79.6%	12:05	91.3%
52	42.2%	16:40	76.1%
53	65.1%	17:20	87.2%
54	72.8%	8:25	90.9%
55	70.8%	16:35	85.2%
56	69.8%	8:20	87.7%
57	79.6%	7:55	91.5%
58	67.3%	16:35	86.0%
59	79.1%	17:00	93.6%
60	84.4%	18:40	94.0%

119	47.1%	15:40	75.9%
120	35.8%	16:00	79.9%
121	44.2%	18:15	74.5%
122	38.0%	17:50	79.5%
123	15.3%	16:50	74.1%
124	50.3%	16:55	75.2%
125	71.6%	17:50	87.4%
126	82.2%	8:00	95.8%
127	79.4%	7:55	94.5%
128	37.2%	15:45	66.3%
129	32.0%	16:15	71.9%
130	62.1%	17:30	83.3%
131	40.9%	18:10	68.1%
132	82.5%	16:00	90.5%
133	84.8%	7:55	98.3%
134	62.7%	15:15	77.6%

192	65.7%	16:25	93.1%
193	40.5%	15:45	64.2%
194	74.2%	17:35	90.6%
195	76.1%	18:35	95.0%
196	58.2%	16:05	88.0%
197	67.9%	16:50	81.6%
198	43.0%	16:10	74.9%
199	54.1%	16:50	80.5%
200	15.5%	17:45	53.5%
201	31.2%	16:45	68.7%
202	59.2%	15:15	80.8%
203	82.8%	11:05	92.2%
204	86.8%	17:25	97.8%
205	60.6%	19:05	82.5%
206	73.1%	17:05	95.9%
207	63.9%	11:20	83.3%

265	32.2%	8:15	75.2%
266	86.2%	8:50	97.1%
267	77.9%	17:05	89.2%
268	36.5%	16:35	69.2%
269	68.7%	11:45	86.1%
270	70.7%	16:15	88.3%
271	40.1%	17:05	71.3%
272	39.1%	7:50	88.5%
273	22.0%	8:05	79.1%
274	86.6%	10:35	93.2%
275	61.0%	7:50	93.4%
276	30.1%	8:10	79.4%
277	78.0%	11:00	89.6%
278	65.7%	16:00	92.7%
279	41.8%	12:05	78.5%
280	80.5%	18:35	90.6%

61	76.6%	16:45	87.0%
62	91.8%	16:15	96.3%
63	15.2%	16:00	51.9%
64	31.5%	7:55	76.3%
65	91.3%	8:00	97.5%
66	75.3%	16:05	91.1%
67	63.3%	8:05	81.4%
68	70.6%	18:15	88.9%
69	38.5%	6:10	88.1%
70	66.1%	7:35	81.9%
71	74.4%	11:15	87.9%
72	86.5%	14:55	97.1%
73	22.1%	16:45	70.3%

135	42.4%	17:45	75.6%
136	66.4%	17:05	91.3%
137	66.2%	17:45	89.2%
138	41.2%	17:05	88.6%
139	23.7%	16:00	71.6%
140	62.4%	18:15	80.4%
141	94.1%	18:30	97.9%
142	24.0%	15:30	63.2%
143	63.7%	17:05	89.5%
144	83.0%	19:50	95.7%
145	46.0%	7:50	72.2%
146	27.1%	16:55	61.7%

208	37.5%	17:45	67.7%
209	63.7%	17:45	85.5%
210	18.5%	17:05	82.3%
211	60.7%	18:55	85.5%
212	48.6%	18:05	83.3%
213	19.6%	7:55	79.8%
214	32.6%	17:35	67.1%
215	82.7%	12:05	93.6%
216	93.6%	7:10	97.6%
217	70.3%	12:05	88.1%
218	22.9%	7:50	58.2%
219	58.2%	7:35	85.7%

281	73.8%	12:25	91.0%
282	31.8%	16:35	80.7%
283	49.6%	17:50	86.8%
284	19.5%	8:00	68.1%
285	86.7%	15:20	92.6%
286	67.6%	17:15	82.9%
287	47.4%	16:35	89.8%
288	63.3%	17:20	88.4%
289	42.7%	15:15	73.8%
290	42.3%	8:00	79.9%
291	71.7%	17:55	93.1%
292	61.9%	16:45	82.4%

Appendix C: FRC classification

In this appendix, a description of the different FRC values is given (*Traffic Stats API FAQ*, 2022).

FRC Value	Short description	Long description
0	Motorways, freeways, major roads	All roads that are officially assigned as motorways.
1	Major roads less important than motorways	All roads of high importance, but not officially assigned as motorways, that are part of a connection used for international and national traffic and transport.
2	Other major roads	All roads used to travel between different neighboring regions of a country.
3	Secondary roads	All roads used to travel between different parts of the same region.
4	Local Connecting roads	All roads making all settlements accessible or making parts (north, south, east, west, and central) of a settlement accessible.
5	Local roads of high importance	All local roads that are the main connections in a settlement. These are the roads where important through traffic is possible e.g.,: • arterial roads within suburban areas, industrial areas or residential areas • a rural road, which has the sole function of connecting to a national park or important tourist attraction
6	Local roads	All roads used to travel within a part of a settlement or roads of minor connecting importance in a rural area.
7	Local roads of minor importance	All roads that only have a destination function, e.g., dead-end roads, roads inside a living area, alleys: narrow roads between buildings, in a park or garden.
8	Other roads	All other roads that are less important for a navigation system: • a path: a road that is too small to be driven by a passenger car • bicycle paths or footpaths that are especially designed as such • stairs • pedestrian tunnel • pedestrian bridge • alleys that are too small to be driven by a passenger car

Appendix D: Speed profile distribution

In this table, the distribution per FRC value, the total number of edges and the percentage of the total edges with a speed profile are shown. Edges with no speed profile are excluded. The explanation of this data is included in Section 4.2.2.2.

profile-id	FRC 0	FRC 1	FRC 2	FRC 3	FRC 4	FRC 5	FRC 6	FRC 7	sum	Percentage of total edges
0	2	16	80	107	161	103	300	386	1156	0.05%
1	1	14	55	145	621	391	3664	4742	9633	0.44%
2	4	16	89	135	993	748	5021	3951	10958	0.50%
3	9	1	4	10	14	7	15	19	79	0.00%
4	4	22	246	422	2437	1323	8436	5241	18132	0.83%
5	86	26	163	121	120	36	198	363	1112	0.05%
6	2	1	3	8	12	4	14	31	75	0.00%
7	4	18	109	269	744	433	1127	631	3335	0.15%
8	19	181	1552	3232	18705	6752	37632	18682	86756	4.00%
9	4	8	86	81	98	36	75	70	458	0.02%
10	0	6	16	34	85	66	463	470	1139	0.05%
11	13	55	233	538	1203	860	5720	7155	15777	0.73%
12	802	557	3069	3341	9134	3202	11043	4734	35882	1.65%
13	3	16	89	144	444	359	4057	5591	10704	0.49%
14	47	136	795	1206	5446	3091	15650	10107	36478	1.68%
15	1	1	8	42	52	39	118	161	422	0.02%
16	2	17	98	145	239	123	222	178	1023	0.05%
17	0	5	21	40	145	137	3439	5280	9067	0.42%
18	1	0	2	6	8	6	149	474	646	0.03%
19	108	198	1175	2151	8493	3586	21255	14671	51636	2.38%
20	1	3	13	29	73	53	403	502	1077	0.05%
21	19	4	27	56	75	47	141	188	556	0.03%
22	268	607	2033	2619	7519	3947	18591	13282	48865	2.25%
23	0	0	0	0	1	1	15	60	77	0.00%
24	3	28	191	221	314	161	248	145	1311	0.06%

25	0	3	10	19	17	8	12	36	104	0.00%
26	10	50	244	642	1551	1012	2428	1477	7413	0.34%
27	120	91	735	1210	3933	1762	10001	6814	24667	1.14%
28	15	111	645	1490	3488	1654	8116	8207	23727	1.09%
29	125	20	80	100	83	29	170	269	877	0.04%
30	10	3	12	28	68	51	518	617	1306	0.06%
31	0	12	56	94	119	79	218	366	945	0.04%
32	124	33	113	126	287	161	1285	1202	3332	0.15%
33	15	21	100	276	387	264	930	1189	3182	0.15%
34	0	4	19	42	71	48	352	439	975	0.04%
35	11	15	217	429	2635	1384	5481	2542	12715	0.59%
36	99	102	718	1145	1848	973	997	306	6189	0.28%
37	64	7	22	36	65	39	390	484	1106	0.05%
38	0	2	21	36	46	25	149	295	574	0.03%
39	74	54	173	216	847	820	7606	7345	17134	0.79%
40	0	6	18	36	126	117	2340	3606	6250	0.29%
41	7	47	440	621	3153	1715	9240	5738	20961	0.97%
42	0	3	9	14	32	21	141	222	443	0.02%
43	1	20	134	227	280	208	510	597	1977	0.09%
44	29	12	27	53	122	100	1791	2194	4330	0.20%
45	2	29	154	357	453	325	977	1515	3811	0.18%
46	12	26	29	39	216	206	4367	6000	10895	0.50%
47	60	36	212	214	342	188	1383	1272	3708	0.17%
48	1	29	197	232	332	190	915	1225	3121	0.14%
49	16	6	22	28	30	8	33	53	197	0.01%
50	1	5	12	35	133	93	651	550	1479	0.07%
51	24	88	453	1161	3117	1610	3490	1517	11461	0.53%
52	0	4	22	86	134	126	247	236	854	0.04%
53	23	3	22	46	85	52	229	250	710	0.03%
54	35	34	203	463	914	583	1239	711	4182	0.19%
55	15	24	119	116	156	64	149	125	769	0.04%
56	4	22	127	268	510	323	592	335	2181	0.10%

57	630	59	400	392	487	166	419	232	2786	0.13%
58	1	9	47	142	254	195	661	637	1946	0.09%
59	393	145	995	1220	1565	613	1139	556	6626	0.31%
60	1	9	30	67	297	215	1718	1345	3681	0.17%
61	1	15	94	160	241	121	361	251	1243	0.06%
62	1660	2213	15347	25049	51341	16291	28430	7987	148318	6.83%
63	0	1	11	33	54	38	18	12	167	0.01%
64	4	3	17	22	31	19	398	761	1255	0.06%
65	94	112	834	1071	2174	1140	2170	899	8494	0.39%
66	118	69	557	972	1731	777	1761	1033	7018	0.32%
67	1	18	120	147	256	165	1070	1462	3239	0.15%
68	0	0	5	6	31	35	715	1072	1864	0.09%
69	4	3	28	45	482	351	4594	5125	10630	0.49%
70	0	31	134	127	203	88	521	652	1756	0.08%
71	1	16	54	125	292	197	704	488	1877	0.09%
72	10	22	174	518	5694	2640	26330	21135	56524	2.60%
73	2	7	41	77	123	64	75	52	441	0.02%
74	6	11	104	106	176	74	282	357	1117	0.05%
75	22	6	15	12	80	67	2498	3616	6316	0.29%
76	0	1	5	8	9	4	20	47	94	0.00%
77	69	25	119	115	263	159	2914	4007	7671	0.35%
78	0	5	19	44	235	148	1204	1075	2731	0.13%
79	87	10	63	71	73	32	399	798	1532	0.07%
80	6	16	118	146	129	53	63	117	647	0.03%
81	11	23	85	115	531	504	6202	7562	15033	0.69%
82	3	21	152	196	389	183	1083	948	2975	0.14%
83	141	34	307	587	2188	1259	7299	5770	17585	0.81%
84	1	3	9	21	66	48	440	592	1179	0.05%
85	1	15	74	139	1196	876	13809	14565	30675	1.41%
86	113	9	46	40	118	61	1102	1686	3177	0.15%
87	0	0	5	11	43	43	1170	2413	3686	0.17%
88	33	7	35	68	104	63	256	328	892	0.04%

89	0	0	1	2	2	1	12	49	66	0.00%
90	599	2530	6707	10304	38415	16501	96150	76715	247922	11.42%
91	28	20	63	62	102	59	1122	1397	2853	0.13%
92	0	1	6	10	14	15	94	160	299	0.01%
93	0	2	0	5	9	12	284	598	910	0.04%
94	14	22	103	197	380	266	878	647	2507	0.12%
95	198	23	132	121	129	59	1007	1796	3465	0.16%
96	51	18	57	79	280	177	1025	890	2577	0.12%
97	2	3	33	74	806	583	7830	8470	17801	0.82%
98	2	16	88	68	124	42	501	917	1759	0.08%
99	9	21	114	257	338	237	421	390	1788	0.08%
100	292	55	307	298	301	113	391	549	2307	0.11%
101	0	6	33	66	67	61	229	399	861	0.04%
102	8	58	232	489	812	573	2316	3032	7522	0.35%
103	6	64	326	699	1155	709	2686	3161	8806	0.41%
104	97	43	197	186	285	127	2542	4607	8084	0.37%
105	72	55	268	754	2715	2013	7359	4318	17554	0.81%
106	4	6	15	20	48	35	424	651	1203	0.06%
107	33	42	274	427	721	371	752	538	3159	0.15%
108	0	7	22	39	157	126	2680	4133	7164	0.33%
109	210	40	200	291	465	346	1753	1607	4911	0.23%
110	0	7	52	130	133	98	155	180	755	0.03%
111	0	4	8	19	42	36	332	524	965	0.04%
112	1	0	4	6	19	8	9	13	60	0.00%
113	1	30	113	202	264	191	327	266	1393	0.06%
114	0	2	5	11	24	14	92	177	326	0.01%
115	8	13	69	84	332	292	1759	1174	3731	0.17%
116	214	42	185	145	184	60	538	961	2329	0.11%
117	0	1	0	3	4	4	38	94	146	0.01%
118	0	4	21	23	25	16	26	58	174	0.01%
119	2	14	52	144	196	184	493	503	1588	0.07%
120	10	12	38	103	101	68	268	461	1062	0.05%

121	0	0	4	16	16	14	127	278	454	0.02%
122	2	3	3	12	13	9	123	210	377	0.02%
123	6	0	3	4	7	3	4	7	33	0.00%
124	0	11	72	182	234	189	417	317	1422	0.07%
125	0	4	5	9	31	21	225	264	558	0.03%
126	72	58	653	826	2130	1248	2739	1063	8789	0.40%
127	3	14	142	258	1447	613	3064	1398	6938	0.32%
128	0	26	156	210	251	158	140	95	1036	0.05%
129	1	4	31	102	110	75	87	123	533	0.02%
130	0	7	20	24	42	24	128	175	420	0.02%
131	0	5	47	92	102	71	139	262	718	0.03%
132	26	49	401	674	1210	536	667	260	3823	0.18%
133	14	44	129	220	1463	1204	9210	7863	20147	0.93%
134	0	23	118	141	135	69	363	639	1489	0.07%
135	0	1	6	21	27	27	109	175	366	0.02%
136	343	69	454	567	660	261	868	678	3898	0.18%
137	0	5	28	78	178	142	1371	2481	4283	0.20%
138	20	5	18	25	21	8	74	202	373	0.02%
139	9	5	25	77	83	58	88	106	451	0.02%
140	0	8	37	70	99	72	382	619	1286	0.06%
141	372	1218	5568	7943	19125	7944	18113	7212	67494	3.11%
142	0	9	33	88	96	65	66	86	443	0.02%
143	9	7	49	99	175	102	306	232	979	0.05%
144	3	30	140	246	1685	1178	16368	17010	36660	1.69%
145	2	32	159	172	203	66	291	463	1388	0.06%
146	0	9	81	170	211	125	104	61	760	0.04%
147	2	15	119	172	281	133	267	202	1192	0.05%
148	0	9	51	71	99	58	133	132	554	0.03%
149	0	13	26	63	377	317	3942	4135	8874	0.41%
150	0	1	0	2	6	7	251	600	867	0.04%
151	1	7	64	132	1046	563	5958	5102	12873	0.59%
152	0	14	44	96	127	85	138	187	692	0.03%

153	31	11	38	36	33	7	24	50	230	0.01%
154	0	7	34	106	221	154	555	484	1561	0.07%
155	0	5	5	17	70	69	1918	2789	4872	0.22%
156	97	206	1175	2387	6638	2897	11143	6201	30743	1.42%
157	36	13	49	57	89	65	1442	2116	3868	0.18%
158	3	26	131	153	192	71	363	653	1591	0.07%
159	6	2	11	17	32	22	48	63	201	0.01%
160	1591	562	3845	4084	5206	1636	1787	547	19257	0.89%
161	252	35	197	217	259	120	839	727	2645	0.12%
162	1	21	126	240	409	287	650	469	2203	0.10%
163	68	8	22	17	40	34	748	1176	2113	0.10%
164	35	7	36	42	70	38	247	311	787	0.04%
165	1	1	6	18	83	58	686	772	1624	0.07%
166	0	13	75	83	93	33	93	185	575	0.03%
167	3	21	158	227	356	188	415	321	1689	0.08%
168	0	19	123	240	374	233	539	340	1869	0.09%
169	10	26	128	381	751	545	1136	838	3815	0.18%
170	2	21	102	268	804	380	1610	1006	4194	0.19%
171	1	7	10	30	77	53	244	205	627	0.03%
172	2	24	39	62	404	353	5056	6497	12438	0.57%
173	1	19	119	284	594	336	1231	982	3565	0.16%
174	0	14	94	229	261	196	164	95	1053	0.05%
175	2	13	92	113	127	54	324	665	1391	0.06%
176	0	1	4	7	15	17	260	436	740	0.03%
177	0	22	109	295	339	237	535	333	1870	0.09%
178	26	9	51	117	199	122	266	219	1008	0.05%
179	1	6	27	72	89	54	56	90	394	0.02%
180	2	11	88	223	1673	1126	8041	6701	17865	0.82%
181	0	6	36	83	251	179	1142	864	2562	0.12%
182	219	966	2176	2891	8019	4069	11511	5756	35607	1.64%
183	1483	548	1685	2046	4320	1539	2525	742	14886	0.69%
184	125	38	297	259	363	124	575	568	2349	0.11%

185	0	21	166	272	308	199	392	495	1854	0.09%
186	0	9	46	130	189	151	247	142	914	0.04%
187	176	28	151	109	91	39	463	887	1945	0.09%
188	116	71	301	293	762	433	1625	879	4479	0.21%
189	13	18	68	81	137	91	753	1082	2244	0.10%
190	48	37	144	169	422	311	6424	9758	17314	0.80%
191	5	8	41	96	671	504	9491	11542	22359	1.03%
192	24	10	55	76	164	102	1364	2097	3893	0.18%
193	0	23	111	129	129	49	93	173	708	0.03%
194	14	11	31	100	203	90	312	226	987	0.05%
195	2	8	25	71	457	417	8178	10238	19395	0.89%
196	19	20	105	266	468	306	2065	2744	5993	0.28%
197	0	18	117	147	148	91	322	399	1241	0.06%
198	1	21	158	206	335	190	422	334	1667	0.08%
199	1	9	47	111	196	138	340	253	1094	0.05%
200	0	0	3	9	14	5	8	11	50	0.00%
201	0	4	24	57	78	48	97	59	366	0.02%
202	1	18	118	233	330	269	834	830	2632	0.12%
203	68	103	564	1320	2980	1420	2268	808	9531	0.44%
204	110	72	332	497	1870	1086	8242	6714	18924	0.87%
205	0	7	20	35	65	53	1129	2175	3485	0.16%
206	37	20	90	104	249	143	2242	2999	5883	0.27%
207	0	15	82	120	222	160	887	1029	2514	0.12%
208	0	1	9	13	15	10	33	72	152	0.01%
209	0	4	12	38	80	54	195	211	595	0.03%
210	14	2	19	29	28	9	15	29	146	0.01%
211	0	2	2	6	17	17	755	1609	2408	0.11%
212	0	3	5	12	28	29	272	433	781	0.04%
213	24	3	5	5	22	15	821	1386	2281	0.11%
214	0	0	3	10	6	5	33	113	169	0.01%
215	55	332	1415	3061	10064	4082	21512	14551	55073	2.54%
216	1040	1553	8516	10672	22767	6623	17601	6905	75676	3.48%

217	1	9	30	43	94	79	2201	3588	6044	0.28%
218	5	15	74	87	85	34	53	111	465	0.02%
219	90	15	94	92	191	88	692	945	2208	0.10%
220	3	21	113	330	828	485	1106	765	3652	0.17%
221	79	20	132	203	301	136	663	764	2300	0.11%
222	1	3	19	51	78	52	30	29	263	0.01%
223	4021	39	266	320	727	321	1368	1064	8125	0.37%
224	1	5	31	43	59	39	249	497	923	0.04%
225	440	131	312	432	1606	754	3009	1725	8408	0.39%
226	0	3	19	44	83	54	26	16	246	0.01%
227	0	16	126	204	229	133	106	97	912	0.04%
228	0	1	11	22	36	23	22	21	136	0.01%
229	33	347	1758	3725	9026	3467	7270	2749	28375	1.31%
230	14	24	128	211	368	191	413	323	1672	0.08%
231	2	20	160	185	198	89	232	342	1228	0.06%
232	2	11	101	138	165	85	74	39	616	0.03%
233	0	0	0	1	1	2	105	298	407	0.02%
234	2	3	23	49	78	43	73	55	327	0.02%
235	53	17	72	108	231	170	850	573	2073	0.10%
236	2	27	135	152	158	82	358	778	1692	0.08%
237	1	1	1	1	4	3	24	78	112	0.01%
238	0	32	139	357	738	412	551	201	2430	0.11%
239	18	28	152	218	364	223	718	714	2434	0.11%
240	104	47	191	191	451	371	5344	6830	13527	0.62%
241	111	14	67	68	71	20	84	144	579	0.03%
242	0	0	1	3	4	3	122	343	476	0.02%
243	0	5	10	33	89	61	361	349	908	0.04%
244	4	1	6	18	24	11	45	77	185	0.01%
245	89	26	132	127	130	39	76	87	705	0.03%
246	121	5	39	51	123	81	710	890	2019	0.09%
247	3	10	44	80	129	72	191	226	755	0.03%
248	0	0	2	2	4	5	41	116	170	0.01%

249	2	14	78	157	189	108	252	412	1213	0.06%
250	40	10	50	43	57	13	88	153	453	0.02%
251	41	6	7	12	58	48	1613	2614	4399	0.20%
252	97	15	154	245	452	280	856	597	2696	0.12%
253	5	4	12	30	32	21	128	321	555	0.03%
254	135	24	72	94	104	47	539	892	1907	0.09%
255	0	18	135	135	212	105	601	779	1985	0.09%
256	1	11	69	85	534	388	8996	11307	21392	0.99%
257	2	19	88	250	501	397	1560	1755	4572	0.21%
258	4	6	35	41	43	26	79	204	438	0.02%
259	4	22	113	276	571	438	1252	1048	3724	0.17%
260	3616	45	268	356	949	422	1191	649	7495	0.35%
261	2	24	99	250	344	288	1074	1215	3296	0.15%
262	200	42	256	347	659	511	2328	1703	6046	0.28%
263	89	34	191	160	185	83	757	1068	2568	0.12%
264	0	1	8	19	34	35	910	1988	2996	0.14%
265	0	2	8	8	13	9	88	242	369	0.02%
266	49	52	217	480	2590	2002	10899	7200	23489	1.08%
267	2264	22	119	142	189	81	218	152	3186	0.15%
268	0	20	97	213	265	189	228	94	1106	0.05%
269	10	25	101	258	650	500	1900	1762	5207	0.24%
270	95	53	322	618	1108	533	942	580	4252	0.20%
271	0	4	33	81	109	87	193	214	720	0.03%
272	33	20	52	43	86	60	1018	1211	2523	0.12%
273	14	3	6	9	10	10	307	660	1020	0.05%
274	2103	341	1770	2355	4916	1810	3134	1094	17523	0.81%
275	32	17	70	71	182	130	1293	1125	2920	0.13%
276	86	18	80	50	42	10	97	228	610	0.03%
277	44	32	206	437	1223	796	2765	1826	7328	0.34%
278	27	12	81	142	454	319	5748	7041	13824	0.64%
279	3	32	111	226	361	240	1434	2440	4847	0.22%
280	2	12	56	92	236	137	931	648	2114	0.10%

281	14	81	400	871	2389	1375	9282	9966	24377	1.12%
282	26	12	44	87	107	50	105	150	580	0.03%
283	0	0	4	5	12	9	183	357	571	0.03%
284	1	4	14	20	27	8	183	468	723	0.03%
285	12	87	636	1441	3129	1338	1850	589	9083	0.42%
286	0	6	67	121	154	85	265	230	928	0.04%
287	41	12	46	49	56	26	410	771	1410	0.06%
288	286	20	117	209	337	146	371	282	1768	0.08%
289	2	12	58	108	139	102	327	441	1189	0.05%
290	7	5	30	28	57	41	654	1239	2062	0.09%
291	0	1	6	16	71	59	642	888	1684	0.08%
292	0	8	51	126	211	164	580	582	1723	0.08%