# MSP Internship Technical Final Report

*Integration of intersection signage when automatically creating a road network from map data in a simulation context*

*Map of the island of Montreal and its surroundings, on the left, source: openstreetmap.org*
*Partial road and public transit networks in the same area, data from the Ministère des transports du Québec on the right*

Ecole Polytechnique de Montréal
Department of Civil, Geological and Mining Engineering

| | | |
|---|---|---|
| Intern, report author: | Noé Fillon | noe.fillon@entpe.fr |
| President of the jury, tutor ENTPE: | Ouassim Manout | ouassim.manout@entpe.fr |
| Host Institution Tutor: | Francesco Ciari | francesco.ciari@polymtl.ca |

Course dates: from 03/04 to 07/04 and from 17/04 to 11/08/2023
Report edited on August 16, 2023

## Summary

## Analytical note

Accurate input data is essential to obtain satisfactory results in modelling. In this report, we will look at MATSim's multi-agent modelling of transport demand and a more realistic representation of intersections on the road network. The work carried out is intended to be general, but some aspects must be adapted to the networks involved. In our case, we will look at the agglomeration of Montreal in Canada.

Just as in a real-life situation, for many intersections, a vehicle must move into a lane prior to a change of direction at a downstream intersection, we will seek to separate queues according to the direction the vehicles wish to take, in accordance with the actual layout in place. At the same time, we will be able to take into account steering restrictions (prohibitions on left turns, for example, which are particularly common in Montreal). This will lead us to look at the representation of lanes and directions in OpenStreetMap data, but also to seek to strengthen the reliability of the results by relying on another network provided by the Ministère des Transport du Québec.

We will then focus on a current research problem, *Geospatial data conflation*, which consists of associating, in this case automatically, the elements of two geographic databases representing the same area. Based on various indices such as geometric positioning, semantic data or topological measurements, it is a question of associating the elements of the two bases representing the same real object. Various methods exist, but none have been applied to MATSim networks. This work will therefore also consist of looking for a method appropriate to our particular situation and implementing it for MATSim networks.

## Keywords

MATSim, OpenStreetMap, Lane, Geospatial data conflation, Road network conflation

## Thanks

First of all, I would like to thank Ouassim Manout for helping me in my search for an internship and referring me to professors at the École Polytechnique de Montréal, including Francesco Ciari. I would also like to thank him for his attentiveness and advice during my internship, whether or not he is directly related to him. I would also like to thank Francesco Ciari for giving me the opportunity to join his research team for this internship.

I would also like to thank Ashraf Uz Zaman Patwary, a post-doctoral fellow in this team, for his help and advice, particularly in relation to IT development. Thanks to Theresa Ziemke, researcher at the Technische Universität Berlin for the tool she provided me, for her availability, and for the answers to the questions I was able to ask. Finally, I would like to thank the other members of the team, especially the other interns, for their human presence in a highly solitary work organization.

## 1. Introduction and overview

My internship is at the École Polytechnique de Montréal in the Department of Civil, Geological and Mining Engineering, in which I joined a research team currently composed of 24 people, working on MATSim (Multi-Agent Transport Simulation), an open-source multi-agent transport demand simulation model developed and maintained by researchers at the Technische Universität Berlin.

Based on the demand for transport at the scale of a geographical area (we will consider it to be an agglomeration below) and its characteristics (location of activities, characteristics of transport networks, speed and reliability of each mode, etc.), the objective of this model is to predict the mobility behaviour of the population as well as certain socio-economic effects of transport through an economic utility function specific to each agent.

MATSim simulates travel based on the transport networks of the study agglomeration (road or public transport). It is therefore necessary to provide them with data that accurately represents these networks. The objective of my internship is to implement or improve existing methods to create a network that is faithful to reality.

## 2. Background

### 2.1. MATSim, a multi-agent model of transport demand [1]

As mentioned earlier, MATSim is a multi-agent model simulating transport demand. The principle of an *agent-based model* is to individually simulate the behaviors of each agent as well as its interactions with other agents, in order to predict results at the level of a population. In the case of transportation demand modelling and MATSim, we are interested in the population of a geographic area, usually a metropolis, and an agent is usually an inhabitant of the geographic area in question. Trips are usually simulated over the course of a day.

MATSim is based on the activities of the agents, which are therefore an input to the model. For each agent, a list of the activities he or she plans to perform during the day is defined. An activity is the amount of time an agent spends in a location between trips. This list can be defined on the basis of various data, including socio-economic data, but its determination is not ensured by MATSim. For each activity, we specify its type (work, leisure, time spent at home, etc.), its location and in some cases the desired start time as well as the expected duration.

Based on this activity schedule, MATSim will choose a mode and estimate a departure time for each agent's trip, before starting the first iteration of its loop to optimize the agents' days. This loop will be performed a certain number of configurable times.

Each iteration of this loop will consist of three steps:

- Mobility simulation: All journeys with the mode and scheduled departure times are simulated on the city's network.
- Assigning the score: Each agent assigns a score to their day that corresponds to their "economic utility". The utility function is fully configurable, but typically it negatively evaluates the financial cost of transportation to the agent as well as the amount of time

spent in each mode of transportation. Characteristics of modes such as comfort or the ability to use travel time productively are usually taken into account. It also negatively assesses delays to activities where a start time was planned and positively assesses the time spent on each activity according to the type of activity.

- Planning update: Each agent attempts changes to their plan (e.g., by changing the mode of transportation, route, or departure time on one or more trips), or reuses a pre-existing plan that is kept in memory, prioritizing the plans that resulted in a better score, but without completely preventing a retry with a lower-performing plan. Indeed, the context regarding network congestion may have changed between the iteration in which this plan was evaluated and the one in which it could be evaluated again.

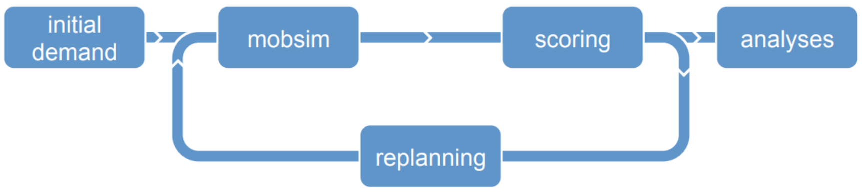


*Fig. 1: MATSim loop describing the operation as described in the previous paragraph. Spring: [1]*

The goal of this loop is to make agents' plans co-evolve so that they converge towards a balance where no agent can on their own significantly improve their score by changing their plan. This situation corresponds to the individual optimum that is assumed to be observed in real cases. The results make it possible to study the congestion of transport networks, the use of different modes, but also certain socio-economic effects, through utility scores.

## 2.2. Mobility Simulation and Traffic Flow Model [1]

Mobility simulation uses simplified models to simulate tens of thousands to several million trips for a day in reasonable computation times. It is performed differently in different modes.

Walking, and very often cycling, is considered a "teleported" mode. Their travel time is calculated from the distance as the crow flies, multiplied by a coefficient representing the tortuosity of the paths taken (generally 1.2 to 1.5), but their movements are not represented on the network.

Motor vehicles, on the other hand (including public transport), use the city's network. It is composed of nodes, points located in the plane, and links, each of which is assigned a start node and an end node. So they're a one-way street. Links for agents to move around the network. Among other characteristics, each link is also attributed:

- A list of authorized modes (car, public transport, bicycle, etc.)
- A free speed (usually in m/s): this allows you to deduce the minimum travel time of the link in the absence of a queue
- Throughput capacity (usually in vehicles/h): this defines the maximum throughput of vehicles that can leave the downstream link
- Link storage capacity (number of vehicles): This defines the maximum number of vehicles in a link's queue.

MATSim's mobility simulation therefore works on the principle of queues or *first in, first out* (FIFO): vehicles leave a link in the order in which they enter it. No overtaking is possible. A vehicle entering an upstream link will therefore only be able to leave it after these three conditions are met:

- A minimum travel time to travel the length of the link at free flow speed has elapsed
- All the vehicles that had entered this link before it have been evacuated and the vehicle in question is at the head of the queue, the evacuation rate being constrained in particular by the speed capacity of the link
- The link it's heading to isn't full (storage capacity)

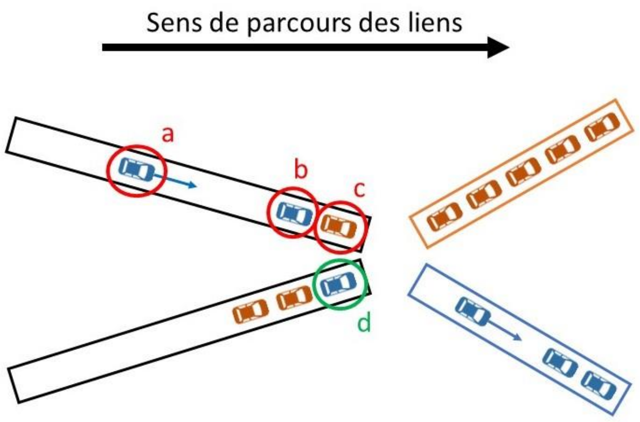


*Fig. 2: Constraints on vehicle traffic (the colour code corresponds to the direction desired by a vehicle: a blue vehicle is heading towards the blue link). In cases (a), (b) and (c), vehicles cannot leave the link, respectively because (a) it has not reached the downstream end of the link, (b) it is not at the head of the queue and (c) the link it is heading towards has reached its storage capacity. In case (d), it can leave the link because all the conditions are met (travel time at free speed, position at the head of the queue, next link not filled to the storage capacity).*

## 2.3. The network as input data

As mentioned above, the agglomeration network is an input that must be provided to MATSim. It can then come from official data or other data sources. The OpenStreetMap (OSM) collaborative platform [2] is often used, as it contains a wide variety of completely free and open source geographic data.

However, data from OSM cannot be read and used directly by MATSim. The road network must be extracted from all the data on the platform. In addition, the structure of the OSM databases does not correspond to that of MATSim.

The OSM road network is organized by nodes and paths (or route, translated from English "*Ways*»). Each path passes through multiple nodes and there is no limit on this number. In addition, a two-way road is registered as a single, two-way road. For each path, among other information, the speed limit and the number of lanes (in the sense of traffic lanes, delimited by road markings, translated from English "*Lanes*") are specified, the distribution of these lanes by direction of traffic is sometimes added. [3] [4]

On the contrary, MATSim-enabled networks are made up of nodes and links. Unlike OSM paths, each link has only one start node and one end node. A path in the sense of OSM will therefore usually be transformed into several links. Free speed and capacity are also not mentioned in the OSM tags (only the speed limit, and it may be missing). They must therefore be deduced from the characteristics of the OSM road: speed limit, number of lanes and category of road (motorway, primary, secondary, residential, etc.).

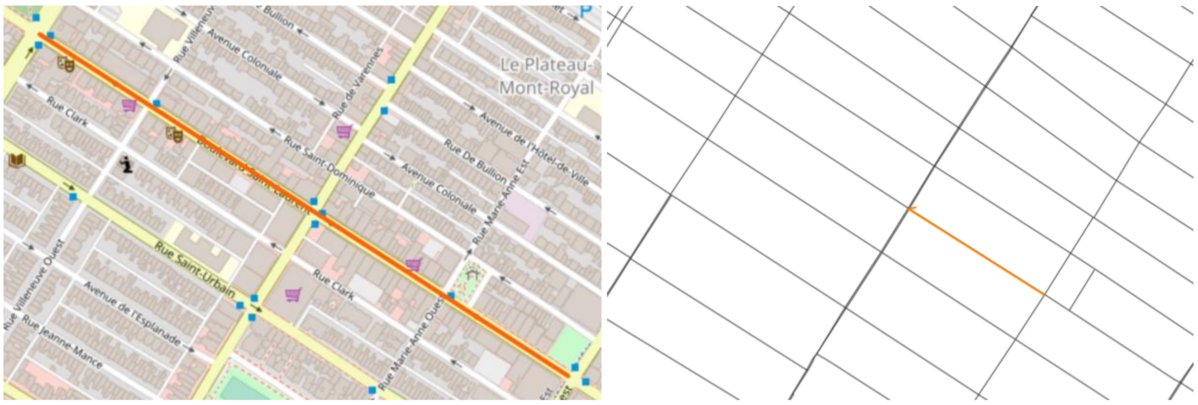


*Fig. 3: Screenshots of a portion of Saint Laurent Boulevard in Montreal, to the left of OSM Road [2], containing a large number of nodes, on the right, highlighted in orange, with one of the four links representing this path on the corresponding MATSim network. The MATSim link consists of one start node and one end node, and is necessarily a one-way street (MATSim represents a two-way road with two links).*

Several tools can be used to convert data from OSM into a road network compatible with MATSim, such as osm2matsim [5], a Java application published by Gustavo Covas in May 2018, OsmNetworkReader [6] [7], a Java class published in 2011 by Michael Zilske, Andreas Neumann and Kai Nagel and SupersonicOsmNetworkReader [8], a Java class published by Janek Laudan in July 2019.

## 3. Objectives of the internship

We'll take a special look at OsmNetworkReader (ONR in the following), however it seems that the other tools mentioned earlier have similar shortcomings. My internship topic focuses mainly on traffic lanes. Indeed, on most intersections, incident roads are equipped with several lanes on which the placement of vehicles is constrained by the direction the drivers wish to take. However, although these constraints are included in the OSM data, they are not taken into account by existing algorithms (in particular ONR) when creating the MATSim network [9].

The existence of lanes on a link then leads MATSim to separate the lanes: instead of one lane on the link, MATSim will create as many lanes as there are lanes and the vehicles traveling on the link will position themselves according to the direction they wish to take. This addition brings the simulation closer to the actual operation of an intersection in several ways:

- By differentiating the waiting times of vehicles according to the direction they are taking, these times can sometimes be very different
- By specifying the light planes: by separating the phases according to the direction
- Taking into account the rise in congestion caused by imbalances in the distribution of flow capacity by direction in relation to demand

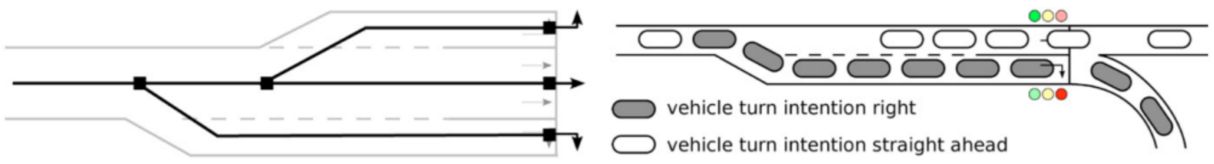

vehicle turn intention right
vehicle turn intention straight ahead

*Fig. 4: On the left, separation of the FIFO lanes as a result of the creation of tracks, on the right, overflow of the congestion of the right lane due to an imbalance between the distribution of demand and throughput capacity between the tracks. Spring: [9]*

My main objective during this internship is therefore to rely on the literature and existing algorithms to develop a method that will allow me to obtain a more satisfactory network of the city of Montreal, particularly regarding the steering constraints at intersections. The need is particularly acute in Montreal because left turns are prohibited at many intersections in the city. Taking these constraints into account can therefore greatly modify traffic in the city and it can be important to take them into account in the mobility simulation.

To do this, I work along two axes: taking into account the data concerning the routes on OSM and merging the OSM data with other data from official sources.

## 4. Methods used

### 4.1. Lane Accounting in OpenStreetMap Data

At the downstream end of OSM paths, the number of lanes and the authorized directions for each lane are usually specified under the tag "*turn:lanes* » [10]. Ten values are possible for this tag: seven of "*sharp_left*" to "*sharp_right*" as well as "*Reverse*», «*merge_to_left*" and "*merge_to_right*».

MATSim allows the creation of these channels in a file separate from the network, however the data structures for the channels on MATSim are, again, different. Unlike OSM, where directions are only indicated geometrically, in the manner of traffic signs or road markings where arrows often appear, each MATSim route requires the identifiers of the links from which you come and where you can go if you take it (we will call them inbound and outbound links). Inbound and outbound links must be directly connected by a common node.

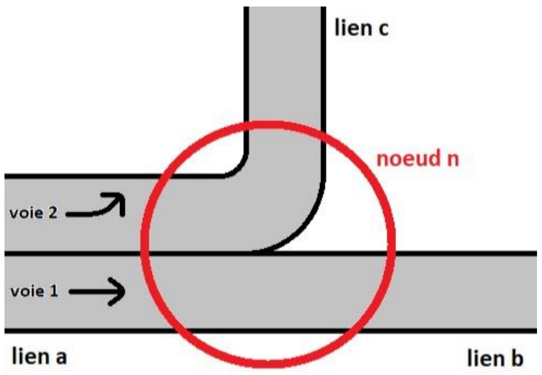


*Fig. 5: In this example, while OSM would only specify that channels 1 and 2 allow you to go straight and left respectively, MATSim requires that we specify that they allow you to go from link a to link b and from link a to link c, respectively.*

In an article published in 2021 [9], Theresa Ziemke and Söhnke Braun present a method to overcome two ONR shortcomings: taking into account the constraints on the possible directions mentioned earlier as well as traffic lights. Their article summarizes a master's thesis written by Nils Schirrmacher in 2017 [11].

In a nutshell, this method consists of identifying "complex" intersections (composed of several nodes), and simplifying them so that they consist of a single node. This is because the channels can only connect two links that follow each other (see Fig. 6). Subsequently, the MATSim channels are created according to the directions specified in the OSM data by comparing them with the angles formed by the links in and out of the intersection (for example, it is decided, if necessary, of a given inbound link, which link corresponds to the fact that a vehicle continues straight ahead). Finally, the algorithm detects compatible trajectories within an intersection and groups traffic lights before creating traffic light plans (OSM data only mentions the presence of a traffic light but does not specify the light plans).

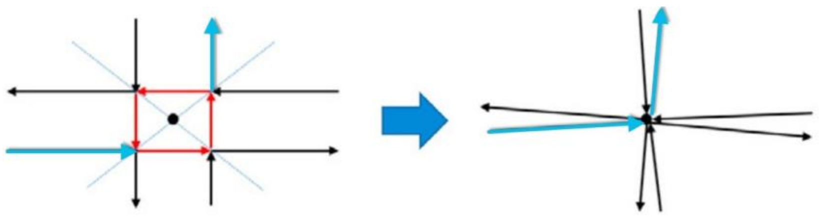


*Fig. 6: In the case on the left, it is not possible to add a lane on the incoming blue link to connect it to the outgoing blue link because they do not immediately follow each other, two intermediate links (here in red) separate them. The intersection is therefore simplified to one node (on the right) and all internal links to the intersection are removed, to allow the lanes to be added. Spring:* [9]

According to the researchers, the algorithm used in this method could be improved. At the moment, my work on this algorithm has essentially consisted of an analysis of how it works. I had to take note of the role of each of the methods in the class created as well as the relationships between them.

This analysis was mainly useful to understand how the program works and to access the right functions in order to use it for the construction of a MATSim network representing the city of Montreal. However, it can be seen that the critical step of simplifying intersections is often incomplete (see Fig.8), which also leads to errors in the restrictions on the directions. Unfortunately, I have not been able to look into the reasons for these errors which may be due to a bad configuration on my part, the use of a wrong version of the program (there are two different versions of the code, without the opinion of the authors, it is not possible to be certain that the second version is only the compiled version of the first and that there have been no changes in between), or errors in the program, even the most recent one. The work of analyzing the functioning of the class that I have done, although I have not personally used it afterwards, will then make it easier for another person to make the necessary corrections.

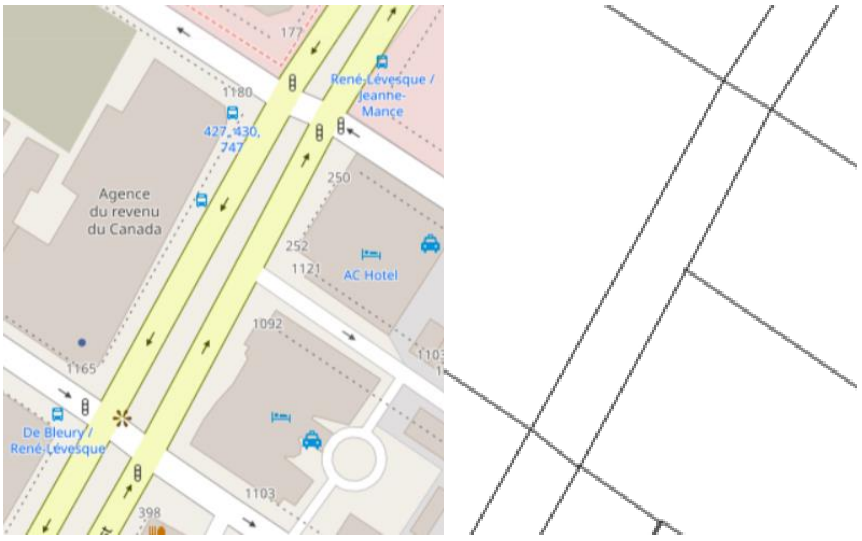

René-Lévesque / Jeanne-Mance
427, 430, 747
Agence du revenu du Canada
AC Hotel
De Bleury / René-Lévesque

*Fig. 8: On the left: a portion of Boulevard René-Lévesque in Montreal according to OSM, on the right the same portion in the MATSim network created with SALONR. It can be seen that the intersections composed of two nodes (with Jeanne Mance Street to the north and with de Bleury Street to the south) have not been simplified.*

## 4.2. Merging networks from different data sources

At the beginning of June, the team received a network from the Ministère des Transports du Québec (MTQ) that includes the main roads of the agglomeration of Montreal. The data from the MTQ contains the lanes and the direction restrictions associated with them, as well as, unlike OSM, the traffic light plans. However, not all streets in the agglomeration are represented in these data. In order to improve the quality of the data, at least on the main axes, my second task was to merge the networks. This consists of locating all the pairs of elements in the OSM network[1] and the other from the MTQ network, representing the same actual geographic feature. Once the connections have been identified, the geometrical layout of the OSM network, which is finer and more precise, could be retained, but the data on intersections, tracks, direction restrictions and light plans, from the MTQ, could complement or replace the data from OSM to ensure its reliability.

[1] From this point on, the MATSim-compatible network built from OpenStreetMap data using the SignalsAndLanesOsmNetworkReader tool is called the 'OSM network'

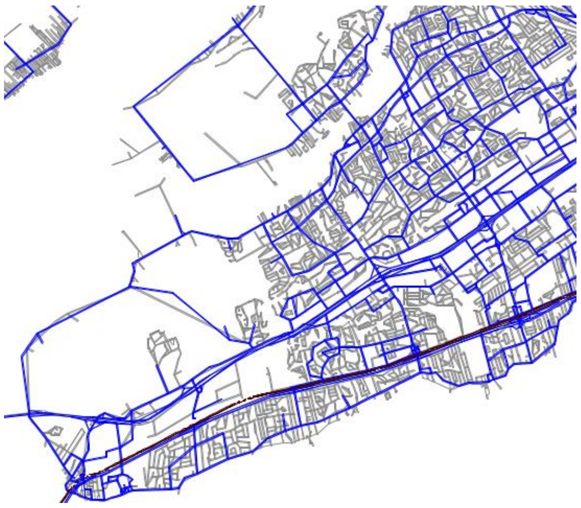


*Fig. 9: Southwest tip of the island of Montreal. In grey, the network built from OSM data, in blue the network provided by the MTQ, which is less fine.*

So I then had to go through research papers presenting different algorithms and methods for matching two networks. This problem is often referred to by the general name "*geospatial data conflation*".

### 4.2.1. Classification of Geographic Data Conflation Methods

Geographic data conflation is in fact a problem that has been the subject of ongoing research for several decades to improve the efficiency and accuracy of algorithms. However, this is not a one-size-fits-all type of problem. In fact, its nature can differ greatly, depending on the type of input data or the type of matches sought.

In his 2011 article [12], Juan J. Ruiz and colleagues propose a classification of geographic data linkage problems according to the criteria measured, the "categorization problem" (horizontal linkage: two databases (BDD) covering border areas, vertical: two databases covering the same area, temporal: two databases covering the same area at two different times), the mode of data representation (vector,  raster, elevation map) and the degree of automation. In our case, we will only be interested in the vertical conflation of vector data.

In 2016, Emerson M. A. Xavier and colleagues [13], also proposes to classify conflation problems according to the cardinality of association. This is to determine whether an object in database A should be associated with a single object in database B or can be associated with several objects and vice versa. A distinction is made between 1:1, 1:m (this case can be further divided into more precise sub-cases) and n:m classifications.

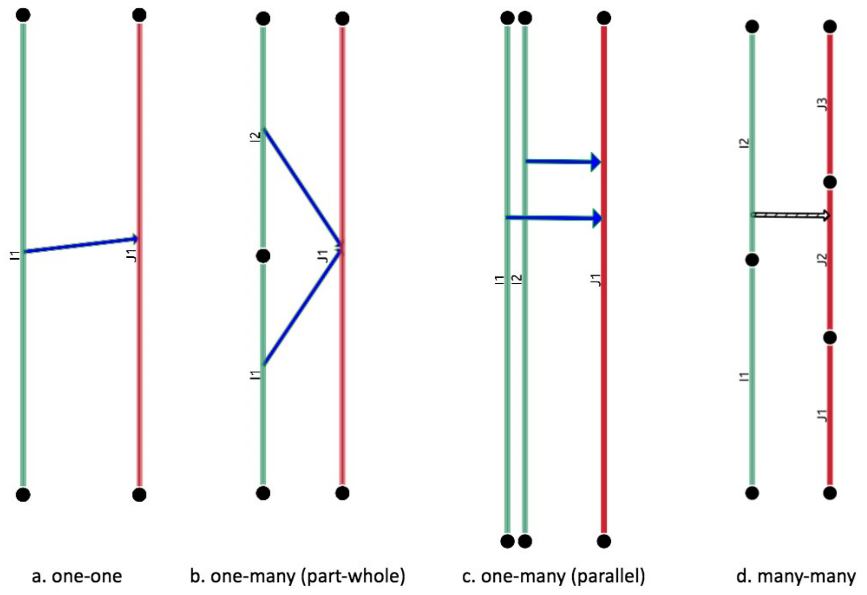


*Fig. 10: Different cardinalities of associations between two BDDs (green links and red links), the case discussed here is case (d) on the right. Spring:* [14]

Next, Ruiz distinguishes two main steps in solving a geographic data linkage problem:

- Determination of similar elements: This step requires determining a measure of the similarity between an element in base A and an element in base B. In most of the methods used, thresholds are set to determine whether two objects in base A and one in base B are sufficiently similar to form an acceptable match [13].
- Identification of matches: this step is based on measuring the similarity between the elements of the two databases. It is then a question of definitively determining the correspondences that will be retained between the elements of one database and the other.

Similarity measures:

The objective of linkage is to establish a correspondence between one or more elements belonging to two separate databases B and B' but representing the same geographical object. Similarity measures can be based on several characteristics of the data. They can be geometric, topological, based on the values of the objects' attributes, the context (other surrounding objects), or their semantic classification.

In the case of conflating two MATSim networks, the data are all of the same type so the semantic classification cannot be used as a measure of similarity. In addition, attributes that vary little throughout the network, such as the number of channels, or throughput capacity, or unreliable attributes such as free speed (which mainly results from the speed limit and the interpretation of the path classification on OSM for one of the networks). Similarity measures can therefore be geometric, topological or contextual. It is also possible to combine several similarity measures to improve their relevance based on the data present.

Geometric similarity measurements, especially for linear objects, can be based on different types of distance measurements, the area of intersection of surfaces, the shape of objects, or the orientation of segments [13].

According to Xavier, topological measurements are particularly applicable to networks, especially roads, which can be represented in the form of graphs. Instead of taking into account the absolute position of an object, they will consist of observing the connections between them. For example, we can measure the similarity of two nodes[2] by comparing their valence: the number of links (distinguishing between inbound and outbound links in the case of a directed graph) attached to a node [13], [15]. It is also possible to observe for two links the similarity (e.g. geometric) of their immediate neighbors, the links connected to them.

Ruiz [12] distinguishes between active and passive topology use. In the first case, topology is used as a measure in its own right, alongside other measures to determine how similar link pairs from different bases are. In the second case, it is used as a test when identifying matches, to verify a posteriori the validity of a linkage. For example, Lei Ting L. and Lei Zhen [16] propose the following rule for 1:m and 1:1 conflation:

"*A pair of connected features in one dataset should not be matched to disconnected target features in the other dataset* "

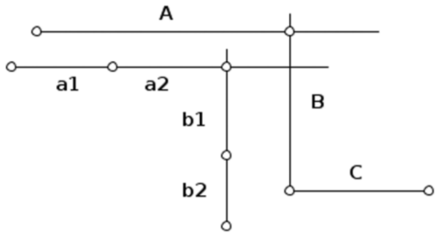


*Fig. 11: For a 1:m match the above rule allows matches such as {(a1, a2 ⇔ A) ; (b1 ⇔ B)} or {(a2 ⇔ A) ; (B1, B2 ⇔ B)} but does not allow some of them considered to be outliers, such as {(a1 ⇔ A) ; (a2 ⇔ C)}: a1 and a2 are connected, unlike A and C. Source:* [16]

According to Xavier [13], topological similarity measures are in practice more commonly used in addition to other measurements, particularly geometrical ones.

To match an object x in base A to an object y in base B, contextual measures take into account the relative position of x with other objects in base A and of y with other objects in base B. For example, we can choose a number of objects as reference points in base A as well as their corresponding points in base B and compare the position of x and y with these respective coordinate systems. It is also possible to take into account all the objects of the two bases by establishing compatibility scores between pairs of associations, responding to " To what extent the association (x1, y1*) is compatible with the association (x2, y2), taking into account in particular the position of x1 and x2 in base A compared to those of y1 and y2 in base B* ? ». These measures can be used to resolve contentious cases and are particularly useful in cases where there is a significant shift in the positioning of objects, but their implementation can be time-consuming and complex.

[2] The terms "nodes" and "links" specific to MATSim also refer here to the "vertices" and "arcs" of graph theory.

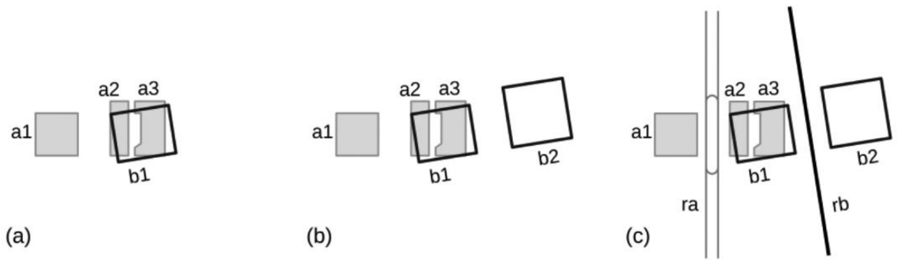


*Fig. 12: Illustration of how adding context can help a human observer: in case (a), an observer would tend to associate object b1 with objects a2 and a3. In cases (b) and (c), it can be seen that adding context and taking it into account corrects the effect of the geometric shift and makes the conflation more reliable.*

<u>Methods of Determining Correspondences:</u>

The goal of this step is to determine matches from similarity measures. Although this step has in many cases been carried out by heuristic methods, i.e. empirical methods whose optimality has not been demonstrated, it can be seen as an optimization problem in operations research [14]. It is a question of finding a set of associations that respect the constraints imposed on them. And allow you to achieve maximum overall similarity (which involves finding a satisfactory way to create a similarity score for each association made and combining them into a single value that you would like to maximize). By expressing the problem of geographic data linkage as an operations research problem, demonstrated domain-specific algorithms can be used.

Due to lack of time and expertise on the subject, I was not able to work on it in more detail.

### 4.2.2. Characteristics of the problem under study

Particular attention is paid to the conflation of two road networks. MATSim networks contain point (nodes) and linear (links) data. However, we will focus on the correspondence between the links, since the links contain the positioning information of the nodes. In addition, they associate the nodes in pairs, which makes it possible to add geometric information (orientation and length of the links), but also to model the network in the form of a graph, and to exploit its topological characteristics. Geometric, topological and contextual measures can therefore be exploited to couple these two networks (the links on the MATSim network contain few sufficiently reliable and heterogeneous attributes in the network for semantic or attribute-based identification).

In practice, we will use <u>*passive geometrical and topological measurements and why.*</u>

The important characteristic of the studied networks is that they are not bijection. In particular, the MTQ's network is highly incomplete. In addition, the OSM network generally has more precise geometric layouts than the MTQ network. However, the links are straight. The accuracy of the route therefore requires the definition of additional links to approximate the actual curves of the road network. As a result, for the same portion of the road, the two networks will have two different consecutive link splits and it will be impossible to associate a link from one network with a single link from the other. This will be an m:n conflation.

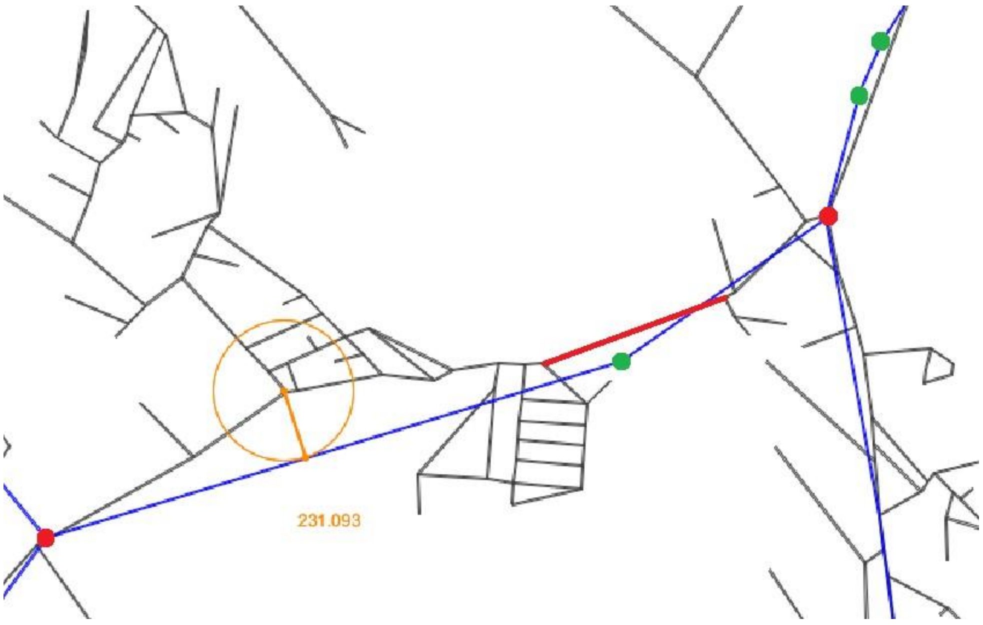


*FIg. 13: Capture of the two networks at a place called Lac-Echo, city of Saint-Prévost. The MTQ network is shown in blue, the OSM network is shown in grey. The links on the MTQ network are longer and less accurately represent the geometry of the road. The positioning offset between the two representations of the same road section can be up to about 230 m in this area.*
*Nodes marked in green on the MTQ network, which do not correspond to intersections common to the two networks, do not have a clear corresponding in the OSM network, and vice versa. It is therefore impossible to associate, for example, the red-marked segment of the OSM network with a single segment of the MTQ network, and vice versa, resulting in an m:n cardinality conflation.*

The layout of the MTQ network, which is less precise, also causes the removal of certain curves present on the OSM network. As a result, some segments of the OSM network may be at a great distance from a segment of the MTQ network that represents the same real object. This is particularly the case in sparsely populated areas where the network is less meshed and the links are therefore longer. Ideally, it would therefore be necessary to vary the distance tolerance of the conflation as a function of the local density of links or the length of the link being studied.

The relevance of the use of topology, particularly in active measurement (4.2.1, "Similarity measures"), is severely limited since many links are missing from the MTQ network, which affects active measures such as node valence or spider index [15], a measurement that is both topological and geometrical in which a node is cut the set of possible directions (the interval [0, 360°[) in angular sectors (usually 8 sectors of 45° each) and note the presence or absence of a link (incoming or outgoing of the node in question) in each of these sectors. However, it is possible to use the topology of the network as a passive measure, i.e. to set rules that make the identification of matches more reliable.

### 4.2.3. Method followed

I then looked for a conflation method that would satisfy the conditions discussed above. The most restrictive condition concerns cardinality, because it complicates the algorithms. The method that I found most compatible with the conditions of my work is the one that Gorisha

Agarwal proposes in her Master's thesis [17] "A Principled Approach to Automated Road Network Conflation," published in 2021. In it, she proposes and details a method that makes it possible to match two road networks represented in the form of a graph. This method allows m:n cardinality associations, while taking into account the network topology. The method is broken down into two steps, called "Map Merging" and "Map Matching", which correspond to the determination of similar elements and the identification of matches as described above. In addition, there is a step in the pre-processing of the databases.

Network conflation algorithms have already been computer-coded, however the format of the input data is not that of MATSim networks. It is therefore necessary, at a minimum, to adapt the reading of the data. However, I was unable to get in touch with Gorisha Agarwal. So I had to implement the whole method. I used the Java language, because the entire core and external contributions of MATSim is written in that language. For example, programs have already been written to read and write to the .xml files that contain the MATSim networks, which allowed me to free myself from some technical work foreign to the main subject of the internship.

## Network pre-processing:

This section describes my approach inspired by section 4.1 of Gorisha Agarwal's brief. In this section we will consider that we have to preprocess a single network, it is exactly the same task that must be done twice. The pre-processing of the network essentially consists of representing it in the form of a graph that is as simple as possible. The dissertation assumes that the input data is an OSM network. As detailed in section 2.3, the data structure of OSM networks does not correspond to a graph since a path in the sense of OSM contains several nodes, whereas the MATSim network has a graph structure. However, the dissertation also seeks to limit the number of arcs in the graph by distinguishing between terminal and intermediate nodes. Terminal nodes are the nodes that mark intersections in the network or the ends of dead ends. Intermediate nodes are those that exist only to ensure the geometric accuracy of the plot, without marking intersections. A segment is the sequence of links between two terminal nodes.

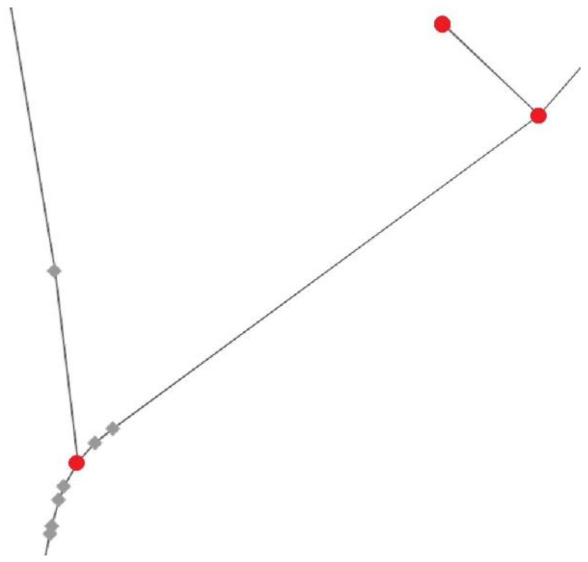


*Fig. 14: Screenshot of the OSM network in Montreal (converted to Intermediate and terminal nodes: Nodes marked in red are considered terminal since they mark intersections or ends of dead ends. The other nodes are considered intermediate.*

The submission does not specify the method used to locate the terminal nodes. The algorithm I wrote first goes through all the nodes and marks each one as terminal or not. Then I applied the algorithm of the depth scan of a graph (or *depth-first search*DFS*)*. The algorithm is initialized by choosing an exit link from a terminal node. Then, as long as it's possible, we move forward in the graph by following links one by one and marking each link we pass over as seen. When it is no longer possible to move forward without going through a link already seen, we go back to the last intersection where it is possible to take another direction not yet explored. As we go through the graph, we add the links that we pass over in a temporary list, until we come across a terminal node. This list is then saved as a segment, and the temporary list is reset to build the next segment.

In the case of MATSim networks, this pre-treatment is not essential. I've actually made it optional, by creating a parameter *allNodesTerminal* which makes it possible to distinguish between intermediate nodes and nodes, to consider all nodes as terminal, and to construct only segments composed of a single link. However, I have chosen, in the case where I have worked, to keep this simplification of the graph representing the network in longer segments.

This has several advantages. The first is the limitation of computation time, since it limits the number of segments in the network. Moreover, in the specific case of these two networks, it is assumed that the OSM network is comprehensive. The roads or streets represented in the MTQ network are therefore all also represented in the OSM network. This means that the vast majority of the terminal nodes of the MTQ network exist and are also terminal nodes in the OSM network (at least those that correspond to intersections, not necessarily the ends of dead ends). Since the ends of the MTQ network segments correspond to the ends of OSM network segments, in most cases, a segment of the MTQ network can therefore correspond to an integer number of segments of the OSM network (1:m correspondence).

## Finding Similar Segments and Building Candidates:

This section describes my approach inspired by sections 4.2.1 to 4.2.2 of Gorisha Agarwal's brief. From this point on, a distinction will be made between the "reference" network and the "target" network. I chose the MTQ network as my reference network and the OSM network as my target network. The objective of this part is first to find for each reference segment a set of sufficiently similar target segments, and then, from this set of segments, to construct "polylines", i.e. sequences of segments, that are "good candidates" for the reference segment.

The first part of this algorithm is to search for segments of the target network that are sufficiently similar to the reference segment. To do this, For a given reference segment, all nearby target segments are traversed. For each segment, we construct a vector from the start node of the segment to its end node. First, we measure the angular difference in direction between the vector representing the reference segment and the vector representing the target segment. Only those segments for which this difference is less than a certain threshold, which has been set at 45°, are selected. The uncertainty in the location of the segments for each of the networks is then estimated. This area of uncertainty is represented by a rectangle around the vector representing the segment, which will be called the tolerance zone. The target segments that have passed the angular criterion will then be sorted by measuring the area of intersection between the tolerance zones of the reference segment and the target segment. Noting respectively $S_R$ and the area of the tolerance zone of the reference segment and the target segment, as well as the area of a surface, the "overlap score" is then calculated as follows. This score must be below a certain threshold, which is not specified in the brief but which I have set at 0.2.$S_C A(S) S \frac{A(S_R \cap S_C)}{\min\left(A(S_R), A(S_C)\right)}$

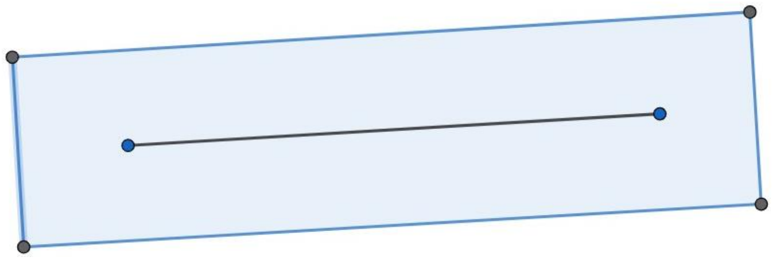


*Fig. 15: Tolerance zone around a segment, shown in black between the two blue dots.*

Some particularly long segments in the reference network may intersect curves that are materialized in the target network, due to the presence of certain intersections, i.e. terminal nodes, that are absent in the reference network. As a result, there may be segments in the target network that correspond well to a portion of the reference segment but are too far away from it to be included as candidates (see Figure 13). In this case, it may be worthwhile to increase the tolerance. However, increasing it too much could unnecessarily multiply the number of applicants for segments located in a dense area of the network. It can therefore be considered to vary the location uncertainty as a function of the length of the segment.

The pre-processing actually contains an additional step to speed up the calculation during the step described above, which will lead us to specify what the expression "nearby" means for two segments: the plane is broken up by a cadrillage (whose dimension *c* of the cells is configurable) and a table lists for each cell all the segments that cross it,  Each segment also contains a list of the squares it passes through. At the stage of finding similar segments, we search for a segment A of the reference network, the set of segments X of the target network for which the uncertainty zones of A and X intersect each other. If this is the case for A and X then there is a pair of points a on A, x on X such that where  and $d(a,x) < t_A + t_X t_A$ are the uncertainties for A and X. If we set the fineness of the cadrillage such that $t_X c \geq \max_{résRéf}(t) + \max_{résCible}(t)$, we then ensure that the X segments of the target network satisfying the condition previously defined for a segment A of the reference network pass through a cell crossed by A or a neighbor to a cell crossed by A. The construction of this cadrillage and the associated r-tree (a structure that keeps in mind the association between the segments and the squares they cross) therefore makes it possible to considerably reduce the search area for similar segments. Appendix 2 shows the variations in the calculation time as a function of the size of the shackle boxes.

## Construction of Candidate Polylines

This step consists of building "good"I am not sure if I am going to be able to do this for a segment A of the reference network, i.e. polylines consisting of segments of the target network selected in the previous step, and whose initial final nodes are sufficiently close to the initial and final nodes of segment A.

Initially, for this step, I wanted to be inspired by the method proposed by Gorisha Agarwal (section 4.2.3 of the thesis), however the application of this algorithm resulted in an obvious lack of candidates in the test phases without me being able to determine why. As the algorithm is complex and its implementation in Java is even more so, the lack of time before the end of the internship led me to choose a method that is certainly less efficient but easier to implement, test and debug.

The method proposed in the thesis consists in traversing the selected segments and constructing polylines either by extending them with other segments or by cutting them, depending on the length of the polyline relative to that of the reference segment and the proximity of the terminal nodes of the reference segment to any point on the polyline.

The method I used, on the other hand, consists in constructing the set of possible polylines, by applying a deep scan algorithm (DFS) on the subgraph formed by the segments of the target network selected in the previous step. The selected segments all have an orientation close to the reference segment so this subgraph does not have a cycle (for that, segments would have to be able to "go back"), so there is no need to worry about this case: the DFS only goes through each segment once, necessarily finishes and only constructs polylines. This method constructs more polylines than the method proposed in the thesis, however additional polylines constructed in addition are most likely less similar to the reference segment. These polylines are therefore unlikely to be retained later and affect the final result.

A final step in the construction of polylines is to verify that their initial and final nodes are close enough to the initial and final nodes of the reference segment. If this is not the case for the initial node, we look for the point on the first segment of the polyline closest to the initial node of the reference segment. The two points should be closer than the tolerance distance for the nodes, the sum of the location uncertainty of the nodes in the reference network and the uncertainty of the nodes in the target network. If such a point exists, it is defined as the new initial point of the polyline by creating a *Cut object* in that polyline. This makes it possible to keep in mind the fact that the first segment of the polyline has been amputated without altering the target network before having made the final choice of the polyline that will be retained. The same is done for the final node and the last segment of the polyline.

Once all the candidate polylines have been constructed, their similarity score is calculated (section 4.3 of the thesis). This is calculated in a similar way to the overlay score used to select the segments. For each polyline, a tolerance zone is drawn. To simplify the calculation and reuse code already produced, we actually build the tolerance zone around each segment. The lengths of the tolerance zone are constructed in exactly the same way as before. The widths, however, are constructed in the same way only at the endpoints of segments that correspond to the endpoints of the polyline. To limit overlaps between the tolerance zones of the different segments, at the other end nodes of the polyline, the widths pass exactly through the nodes instead of passing at a distance corresponding to the location uncertainty. It is then assumed that the angles between the segments are small, so that aberrations at the terminal nodes located inside the polyline do not have a decisive impact on its score.

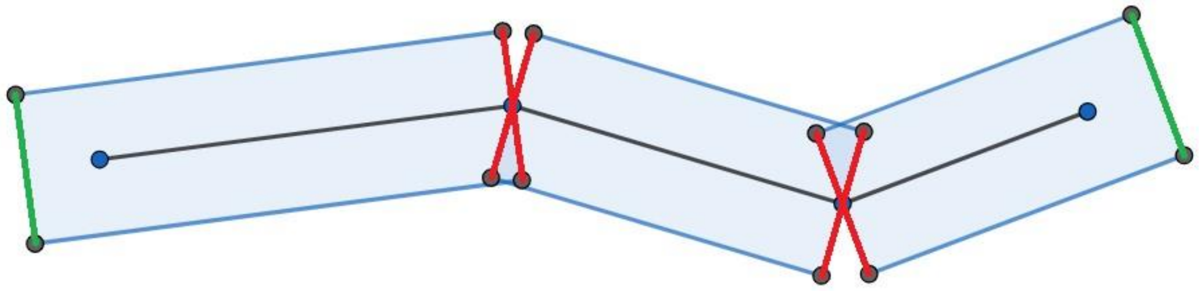
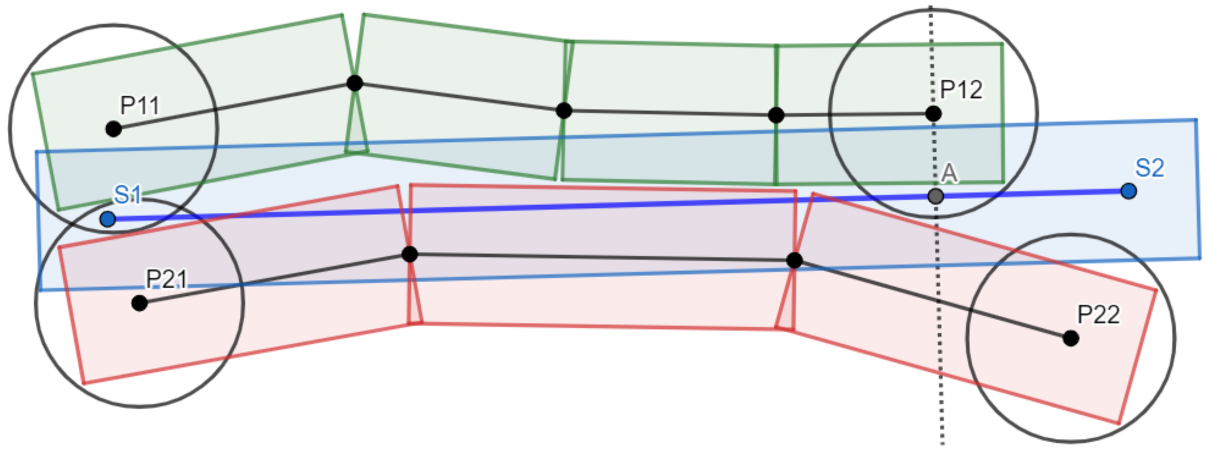


*Fig. 17: Tolerance zones of the segments of a polyline. The widths of these areas marked in green, located at the ends of the polyline, pass away from the nodes, while the other widths of the tolerance areas, in red, pass over the nodes.*

Once all the candidate polylines have been constructed, we distinguish two cases: the one where there are good candidates, whose initial and final nodes are close to the initial and final nodes of the reference segment, and the one where there are none.

In the event that there is a good candidate, only the correct candidates (and their score) are kept.

In the event that no good candidate is found, the polyline with the highest score is chosen from among those whose two ends are sufficiently close to the reference segment, i.e. at a distance from any point in this segment less than the sum of the location uncertainty of the nodes of the reference network and that of the nodes of the target network. If no candidate verifying this condition is found, then this segment is considered to have no counterpart in the target network. Then the reference segment is divided into two or three segments: for each end N *of the chosen polyline, if* N is too far from the corresponding end of the reference *segment, then the segment in question is cut so that the end common to the two new segments thus obtained is sufficiently close to* [3] *N* . We then repeat the entire procedure of finding similar target segments, constructing and selecting polylines for the newly obtained segments.



[3] We know that this will be the case for at least one of the two, otherwise this polyline would be a good candidate

*Fig. 18: Two candidate polylines for the S-segment shown in blue: The P2 polyline shown with a red tolerance area has a larger similarity score, however it will not be used to section the segment because the P22 point is too far away from the segment. The P12 end of the P1 polyline will be used to section S in the absence of a good candidate because it is the candidate whose two ends are close enough to S with the highest similarity score. The segment S will be sectioned into [S1,A] and [A,S2], where A is the closest point in S to P12.*

The result obtained by the program is a set of segments of the reference network with a set of polylines considered to be good candidates with their similarity score for the reference segment. The final determination of which polyline to associate with each segment remains to be programmed. Gorisha Agarwal proposes to apply the *rank-join Algorithm*, developed by Ihab F. Ilyas and colleagues in 2004 [18] to join two databases where each association between two elements of the databases has a score. The purpose of this algorithm is to determine the *k* best (in our case, ) joins, i.e. those that maximize the sum of the association scores. Agarwal proposes to adapt this algorithm to the problem of road network conflation. It offers optimizations for this specific problem, but also to take into account topological constraints, in particular the connectivity between network elements. $k = 1$

## 5. Critical analysis and perspective for the continuation of the work carried out

### 5.1. Analysis of the OSM Network Conversion Program

This analysis of the code produced by the team at the Berlin University of Technology was of little use to me personally. However, it may be useful for future improvements in the configuration of the code or for future updates of the code in collaboration with the original authors. The flow-chart will indeed make it possible to quickly clarify an organization of the code that seemed obscure to me for many weeks for a person who would like to continue the work in this field.

### 5.2. Road network conflation

Network conflation requires several choices and assumptions that need to be justified. In the first place, the choice of the MTQ network as the reference network and the OSM network as the target network is justified by the asymmetric treatment of the two networks. To match the networks, we tend to first extend the segments of the target network into polylines and then, if necessary, to intersect the ends of the constructed polylines, whereas if we have not found a polyline that runs through the reference segment from one end to the other, we cut the reference segment. The MTQ segment tends to have longer ties. In addition, the creation of segments (the fact of treating all the links between two terminal nodes as a single entity) allows for a higher proportion of 1:m matches (see section 4.2.3, network pre-processing). The association of multiple segments into polylines is done on the target network. The OSM network, which has shorter links, is therefore better suited for this role. This choice allows you to limit the number of breaks in segments and links.

Several thresholds have been set, in particular to consider a segment to be sufficiently similar to be acceptable. There is a condition on the direction and a condition on the area of intersection of the tolerance zones. In some cases, there are values that should not be exceeded. For example, the angle limit between the reference segment and the acceptable target segments should not be as high as 90° could be created, as this could create polylines that have cycles. However, the threshold was arbitrarily set at 45°. The same is true for the 0.2 threshold set for the overlay score (see section 4.2.3, Finding Similar Segments and Building Candidates). The value of the location uncertainty, parameterized and chosen at 20 m for the nodes and 50 m for the segments, was chosen empirically, by visualizing the overlapping networks in different locations and by measuring the distance between pairs of nodes and segments that I assumed to match.

In addition, as explained at the end of Section 4.2.3, the designed program is not completed and the right set of candidates is obtained without a final selection of segment associations. I therefore intend to provide, in addition to the programme, a sheet detailing the form and content of the input and output data of the programme that I have written, so that other programmes can be associated with it without difficulty, in particular to complete it.

### 5.3. Overall assessment of the internship

Defining goals and methods was an important challenge for me during this internship. Indeed, I was faced with a problem to be solved from data without necessarily a defined method or precise step-by-step instructions. In a way, it is a discovery of research working methods. However, this led me to spend a lot of time studying what was provided to me, including the code of the SignalsAndLanesOsmNetworkReader class, with no other goal than to understand how to use it and build a network. This analysis, followed by the production of the flow chart, in addition to other general research on graph theory or programming in Java kept me busy during the first five weeks of my internship, without having obtained any significant results or production, other than a (imperfect) MATSim network of the agglomeration of Montreal, produced from a code that I had not written myself. In the future, I will therefore try to focus more on the discussion with the tutor and the team members who supervise my project to be introduced to the subject in a more complete way and define more precise objectives, without restricting too much the subject which remains a research subject, therefore by nature, without a predefined method.

## Bibliography

[1] K. W. Axhausen and ETH Zürich, *The Multi-Agent Transport Simulation MATSim*. Ubiquity Press, 2016. DOI: 10.5334/BAW.

[2] "OpenStreetMap", *OpenStreetMap*. https://www.openstreetmap.org/

[3] "Way — OpenStreetMap Wiki". https://wiki.openstreetmap.org/wiki/Way (accessed June 15, 2023).

[4] "Highways - OpenStreetMap Wiki". https://wiki.openstreetmap.org/wiki/Highways

[5] G. Covas, "osm2matsim". November 30, 2022. [Online]. Available on: https://github.com/gustavocovas/osm2matsim

[6] "OsmNetworkReader (MATSim 12.0 API)". https://www.matsim.org/apidocs/core/12.0/org/matsim/core/utils/io/OsmNetworkReader.html

[7] M. Zilske, "OpenStreetMap For Traffic Simulation".

[8] J. Laudan, "super-sonic-osm-network-reader". December 5, 2019. Accessed: June 15, 2023. [Online]. Available on: https://github.com/Janekdererste/super-sonic-osm-network-reader

[9] T. Ziemke and S. Braun, "Automated generation of traffic signals and lanes for MATSim based on OpenStreetMap", *Procedia Comput. Sci.*, vol. 184, p. 745-752, 2021, doi: 10.1016/j.procs.2021.03.093.

[10] "Key:turn — OpenStreetMap Wiki". https://wiki.openstreetmap.org/wiki/Key:turn#Turning_indications_per_lane

[11] "Schirrmacher2017MasterThesisOSMSignalsLanesReader.pdf". Accessed: June 13, 2023. [Online]. Available on: https://svn.vsp.tu-berlin.de/repos/public-svn/publications/vspwp/2017/17-18/Schirrmacher2017MasterThesisOSMSignalsLanesReader.pdf

[12] J. J. Ruiz, F. J. Ariza, M. A. Ureña, and E. B. Blázquez, "Digital map conflation: a review of the process and a proposal for classification", *Int. J. Geogr. RN Sci.*, Vol. 25, No. 9, p. 1439-1466, Sept. 2011, doi: 10.1080/13658816.2010.519707.

[13] Emerson M. A. Xavier, "A Survey of Measures and Methods for Matching Geospatial Vector Datasets." https://dl.acm.org/doi/epdf/10.1145/2963147 (accessed June 9, 2023).

[14] T. L. Lei, "Geospatial data conflation: a formal approach based on optimization and relational databases," *Int. J. Geogr. RN Sci.*, Vol. 34, No. 11, p. 2296-2334, Nov. 2020, doi: 10.1080/13658816.2020.1778001.

[15] Barbara Rosen and Alan Saalfeld, "Match criteria for automatic alignment", *Proc. 7th Int. Symp. Comput.-Assist. Cartogr.*, p. 456-462, 1985.

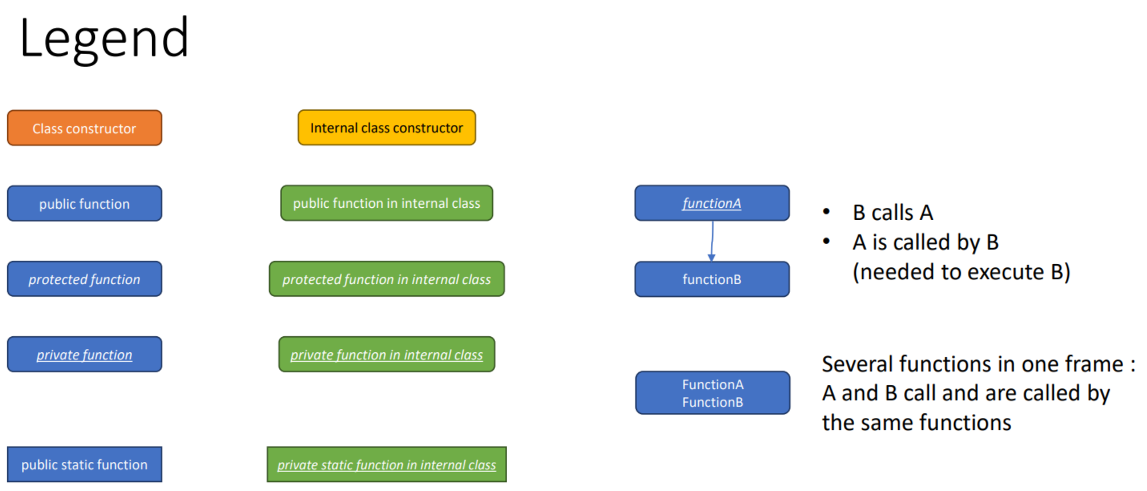
[16] T. L. Lei and Z. Lei, "Linear feature conflation: An optimization-based matching model with connectivity constraints," *Trans. GIS*, Vol. 27, No. 4, p. 1205-1227, 2023, doi: 10.1111/tgis.13062.

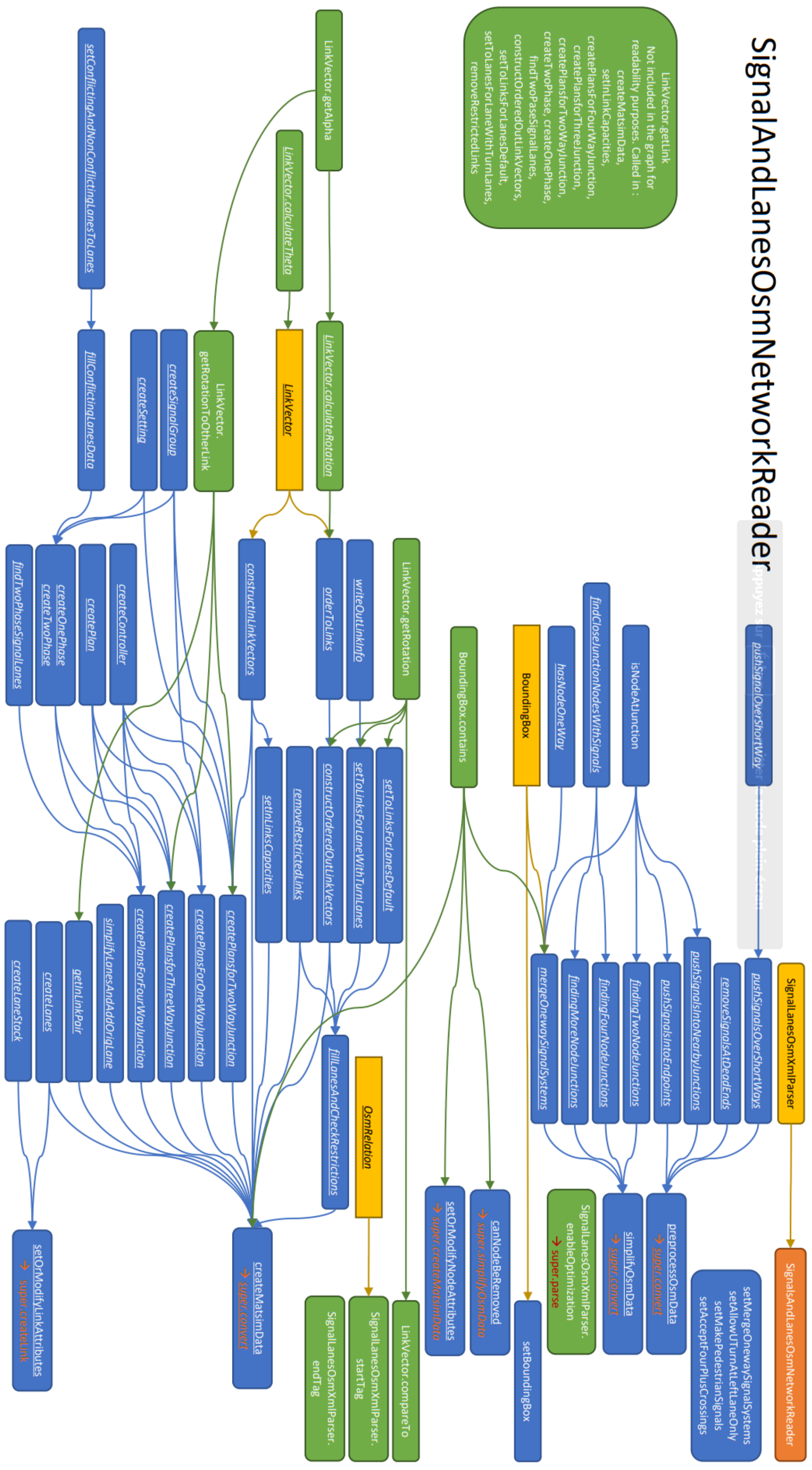
[17] Gorisha Agarwal, "A principled approach to automated road network conflation," University of British Columbia, 2021. [Online]. Available on: https://open.library.ubc.ca/soa/cIRcle/collections/ubctheses/24/items/1.0398182

[18] I. F. Ilyas, W. G. Aref, and A. K. Elmagarmid, "Supporting top-k join queries in relational databases," *VLDB J.*, Vol. 13, No. 3, p. 207-221, Sept. 2004, doi: 10.1007/s00778-004-0128-2.

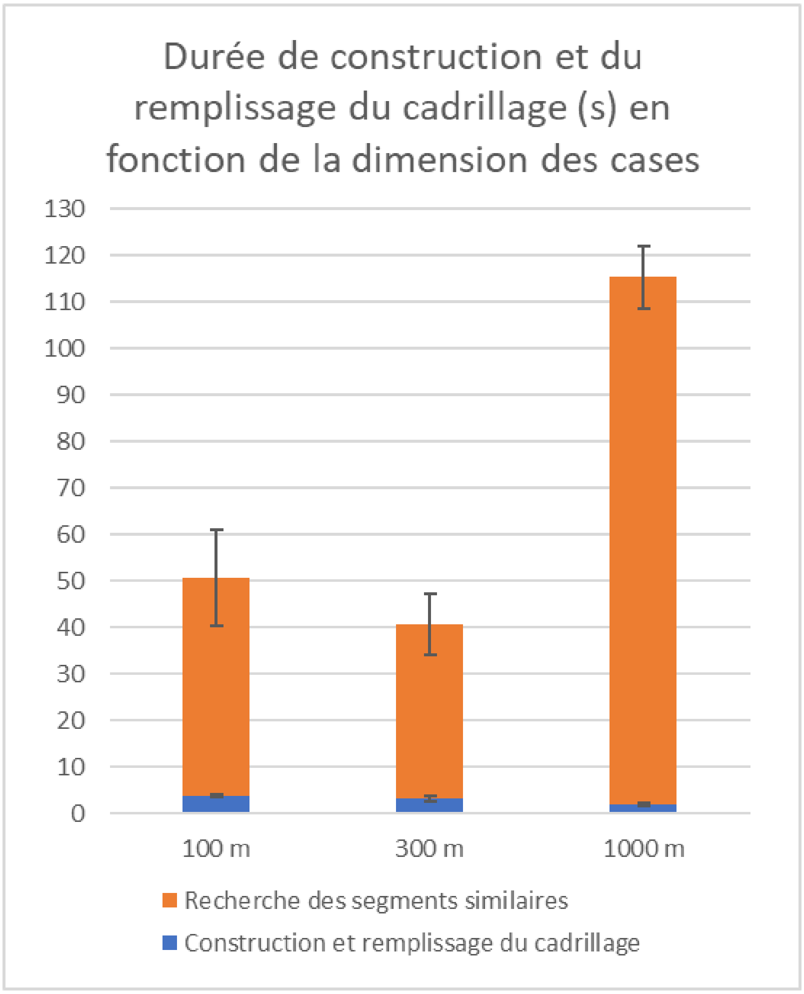
## Annexes

### Appendix 1: Flow chart representing the functions of the SignalsAndLanesOsmNetworkReader class


Legend
Class constructor
Internal class constructor
public function
public function in internal class
protected function
protected function in internal class
private function
private function in internal class
public static function
private static function in internal class
functionA
functionB
B calls A
A is called by B (needed to execute B)
FunctionA FunctionB
Several functions in one frame : A and B call and are called by the same functions


SignalAndLanesOsmNetworkReader
LinkVector.getLink
Not included in the graph for readability purposes. Called in : createMatsimData, setInLinkCapacities, createPlansForFourWayJunction, createPlansforTwoWayJunction, createPlansForThreeJunction, createTwoPhase, createOnePhase, findTwoPaseSignalLanes, constructOrderedOutLinkVectors, setToLinksForLaneDefault, setToLanesForLaneWithTurnLanes, removeRestrictedLinks
SignalAndLanesOsmNetworkReader
setMergeOnewaySignalSystems
setAllowUTurnAtLeftLaneOnly
setMakePedestrianSignals
setAcceptFourPlusCrossings
SignalLanesOsmXmlParser
pushSignalOverShortWay
pushSignalsOverShortWays
removeSignalsAtDeadEnds
pushSignalsIntoNearbyJunctions
pushSignalsIntoEndpoints
findingTwoNodeJunctions
findingFourNodeJunctions
findingMoreNodeJunctions
mergeOnewaySignalSystems
isNodeAtJunction
findCloseJunctionNodesWithSignals
hasNodeOneWay
BoundingBox
BoundingBox.contains
preprocessOsmData
→ super.convert
simplifyOsmData
→ super.convert
SignalLanesOsmXmlParser.enableOptimization
→ super.parse
setBoundingBox
canNodeBeRemoved
→ super.simplifyOsmData
setOrModifyNodeAttributes
→ super.createMatsimData
LinkVector.getRotation
LinkVector.compareTo
SignalLanesOsmXmlParser.startTag
SignalLanesOsmXmlParser.endTag
OsmRelation
fillLanesAndCheckRestrictions
setToLinksForLanesDefault
setToLinksForLaneWithTurnLanes
constructOrderedOutLinkVectors
removeRestrictedLinks
setInLinksCapacities
writeOutLinkInfo
orderToLinks
constructInLinkVectors
LinkVector.getAlpha
LinkVector.calculateRotation
LinkVector.calculateTheta
LinkVector
LinkVector.getRotationToOtherLink
createSignalGroup
createSetting
fillConflictingLanesData
setConflictingAndNonConflictingLanesToLanes
createController
createPlan
createOnePhase
createTwoPhase
findTwoPhaseSignalLanes
createPlansforTwoWayJunction
createPlansForOneWayJunction
createPlansforThreeWayJunction
createPlansForFourWayJunction
simplifyLanesAndAddOrigLane
getLinkPair
createLanes
createLaneStack
createMatsimData
→ super.convert
setOrModifyLinkAttributes
→ super.createLink

## Appendix 2: Optimization Effi ciency: Time to Find Similar Segments as a Function of Shackle Size



The time it takes to populate the table representing the shack and to find similar segments based on the length of a slab in the shack. The gray bars correspond to the standard deviation. Although the construction time of the table is extended by decreasing the length of the squares, its value and variations remain negligible compared to the time it takes to find similar segments and its significant decrease with the size of the squares. With 10,000 m squares on each side, two hours was not enough time to search for similar segments.