

Rain and flood informed vehicle routing problem

Gabriel Silva Delgado

Instituto Tecnológico de Aeronáutica (ITA); Universidade Federal de São Paulo (UNIFESP)
São José dos Campos, SP
g.delgado@unifesp.br

Tiago Rodrigues Macedo

Universidade Federal de São Paulo (UNIFESP)
São José dos Campos, SP
tmacedo@unifesp.br

Leonardo Bacelar Lima Santos

Centro Nacional de Monitoramento e Alertas de Desastres Naturais (Cemaden/MCTI);
Universidade Federal de São Paulo (UNIFESP)
São José dos Campos, SP
santoslbl@gmail.com

Luiz Leduino Salles-Neto

Universidade Federal de São Paulo (UNIFESP)
São José dos Campos, SP
luiz.leduino@gmail.com

ABSTRACT

This article presents a method for obtaining routes that avoid rain and floods in a city. To achieve this, we model a city's road network as a weighted multigraph and use data on precipitation and flood occurrences. We employ well-known methods to solve the shortest path problem based on this data. To exemplify and validate the method presented in this article, we study cases in São Paulo in the Summer of 2019. For this purpose, we use rainfall prediction data from radar and flooding records confirmed by São Paulo's Climate Emergency Management Center. This information is employed to modify the weights of the edges (streets) of the multigraph. Using this modification, we show that the shortest paths between the chosen points avoid rain and floods.

KEYWORDS. Vehicle routing. Shortest path problem. Graph theory.

Logistics and transportation. Big data and analytics. Combinatorial optimization.

1. Introduction

Annually, cities worldwide are affected by weather events. Some of these events have more significant impacts on the economy and population's safety, and the development of new technologies can help mitigate some of these impacts. The main goal of this project is to help reduce harmful effects of rainfall on urban mobility.

More specifically, our goal is to propose a dynamic routing policy to minimize the impact of rain and resulting road flooding on the mobility of great cities. Note that, for this purpose, it is essential to have access to rainfall forecasts with high space-time resolution and good reliability, in addition to reliable data on the sensitivity of public roads to flooding.

The case study used to validate the model proposed in this article is from a region in the metropolitan area of the city of São Paulo. We employ RADAR precipitation estimates (from São Roque Radar, Departamento de Controle do espaço Aéreo) with $1\text{km} \times 1\text{km}$ of grid-spacing, available every 10 minutes. Flooding data was provided by the São Paulo's Climate Emergency Management Center (CGE-SP). While the street data was obtained from *Open Street Maps*.

The main objective of this article is to help the population of large cities to choose safer routes. For this, we propose a way to find routes between any two points in a city so that vehicles are least affected by the effects of precipitation.

2. Theoretical foundation

To model this problem, we can view the city as a graph, in which the intersections are the vertices and the streets are the edges. Thus, the problem of finding routes between two points in the city translates into finding a sequence of edges (a *path*) in the graph that connects the edge corresponding to the origin to the edge corresponding to the destination.

As we want to find routes that are least affected by the effects of rain, we need to consider one more ingredient: the effect of rain on each street. This translates into associating a numerical value (called *weight*) to each edge of the graph and finding a path that minimizes the sum of the weights of the edges that form the proposed path. This problem is known in the literature as the *Shortest Path Problem*.

In 1956, Dijkstra developed an efficient algorithm for finding a shortest path in a weighted graph [Dijkstra, 1959]. The idea of Dijkstra's algorithm is to build paths between two vertices of a graph greedily (that is, choosing the shortest edges between a vertex and its neighbors) and exhaustively (that is, considering all neighbors of a vertex whose total weight of the path does not exceed the minimum value already obtained). About a decade later, Hart, Nilsson and Raphael presented an extension of Dijkstra's algorithm, the *A* algorithm* [Hart et al., 1968]. In contrast to Dijkstra's algorithm, the A* algorithm uses a function that estimates the distances between the vertices of the path and the destination vertex (see Figure 1 and Figure 2). When this function satisfies certain conditions, it prevents the algorithm from testing some paths and makes searching for the shortest path more efficient. Note that both, Dijkstra's algorithm and A* algorithm (under certain conditions), find optimal paths [Dechter e Pearl, 1985, Corollary 1].

Since the precipitation data used in this project comes from a radar that estimates precipitation with a 10-minute frequency, the weights of the edges must be constantly updated. Thus, every 10 minutes, we run the A* algorithm with the weights of the edges updated and, if necessary, modify the vehicle's optimal route. Notice that to update these weights, we take into account the lengths and travel time of the edges, as well as the flooding data and the amount of rain predicted for the corresponding region.

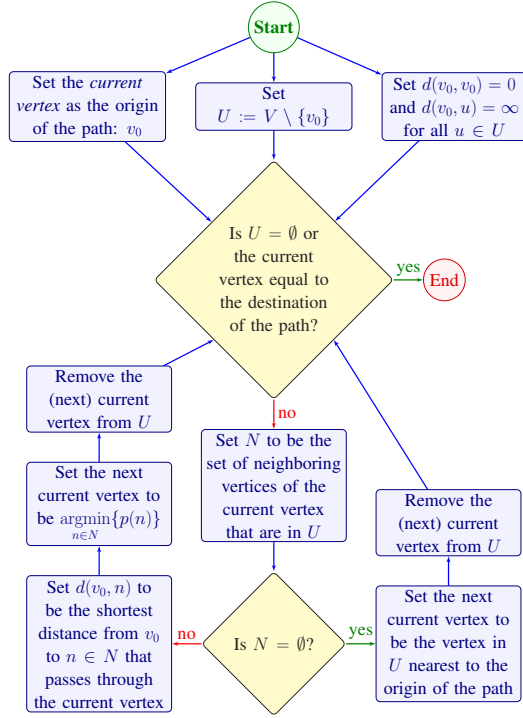


Figure 1: Dijkstra's algorithm

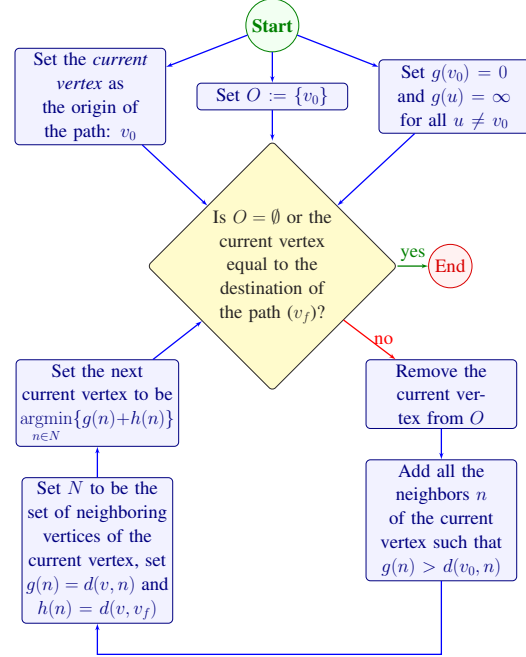


Figure 2: A* algorithm

3. Methods

Recall that, to model the main problem of this article, we are going to see a city map as a graph, more specifically, as an oriented weighted multigraph. Formally, this graph is a triple (V, E, p) , where V denotes the set of vertices, E denotes the set of edges and $p : E \rightarrow \mathbb{R}$ denotes the function that assigns a weight to each edge. Below we detail the data V , E and p that was used to validate our model.

To start with, we implemented all the steps of this project in *Python*. This gave us the possibility to use some of their libraries, for example, *osmnx*. Using this library, we download street layout data for the region bounded by latitudes -23.7748° and -23.2959° and longitudes -46.6807° and -46.3841° , directly from *Open Street Maps*, as a graph. Among other information, this dataset contains: a set of vertices V , which is formed by the intersections of streets in this region; a set of edges E , which is formed by the streets in this region; the lengths and estimates of free-flow travel times for each one of these edges, that is, functions $l, t : E \rightarrow \mathbb{R}$.

Now, to construct the function $p : E \rightarrow \mathbb{R}$, we use data from precipitation forecasts and flooding reports. Precipitation forecast datasets come from a radar that estimates the amount of rain that should fall in the subsequent 10 minutes in each subregion of $1\text{km} \times 1\text{km}$ within the analyzed region. Given an edge $e \in E$, denote by $r(e)$ the millimeters of rain forecasted for the subregion that contains e . If $r(e)$ is such that e is predicted to flood, then we set $p(e) = \infty$. Otherwise, we may set $p(e) = l(e) (1 + r(e)/100)$ or $p(e) = t(e) (1 + r(e)/100)$, depending on which property of the path we want to minimize (see Section 4 for more details). That is, for each edge $e \in E$, we will set $p(e)$ to be the length or travel time of the edge e increased by a factor of $r(e)\%$. To

differentiate between these two cases, we shall denote

$$\begin{aligned}
 p_l(e) &:= \begin{cases} \infty, & \text{if } e \text{ is predicted to flood,} \\ l(e) \left(1 + \frac{r(e)}{100}\right), & \text{otherwise,} \end{cases} \\
 &\text{and} \\
 p_t(e) &:= \begin{cases} \infty, & \text{if } e \text{ is predicted to flood,} \\ t(e) \left(1 + \frac{r(e)}{100}\right), & \text{otherwise.} \end{cases}
 \end{aligned} \tag{1}$$

Finally, we can describe our goal more explicitly. Given two vertices $v, w \in V$, we want to find a sequence of edges $e_1, e_2, \dots, e_n \in E$ such that: e_1 starts at v , e_n ends at w , and the value of $p(e_1) + p(e_2) + \dots + p(e_n)$ is a minimum. This problem is known as the *shortest path problem* and, amongst the existing algorithms, we have chosen the *A* algorithm* to solve this problem, as it returns an optimum in a timely manner.

4. Results

In this section, we will report on a few examples and results. To do that, we begin by describing the data that was used in more details. First, recall that we use the *Python* library *osmnx* to download street layout data for the region bounded by latitudes -23.7748° and -23.2959° and longitudes -46.6807° and -46.3841° , directly from *Open Street Maps*, as a graph.

Next, the precipitation predictions provided by Brazil's National Center for Natural Disaster Monitoring and Alerts (CEMADEN) are contained in text files which consist of 55 lines, with each line consisting of 31 integer numbers separated by spaces. Thus, the j th integer on the i th line of one of these files represents the amount of millimeters of rain that is predicted to fall in the $1\text{km} \times 1\text{km}$ region located i kilometers to the west and j kilometers to the south of the point $(-23.2959^\circ, -46.3841^\circ)$ at a certain period of time. For instance, if we assign to each edge of our graph the maximum amount of rain predicted to its points at a specific time, we may obtain a graph that looks like the one on Figure 3.

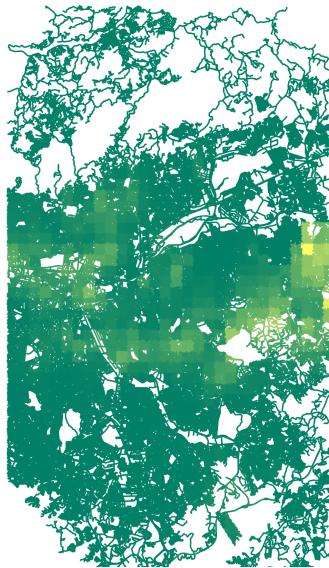


Figure 3: Graph showing precipitation intensity: greener means less and yellower means more rain.

X	Y	ADDRESS	DATE	START_T	END_T
333961.3570	7386903.0835	PC LEONOR KAUPA	1/4/2019	14:34:00	15:18:00
333666.1527	7395716.0954	R DA CANTAREIRA	1/4/2019	14:45:00	15:03:00
333795.6231	7386133.6740	AV PROF ABRAAO DE MORAIS	1/4/2019	14:48:00	15:19:00
333774.7752	7386172.2001	AV FAGUNDES FILHO	1/4/2019	14:55:00	15:21:00
333937.8519	7386816.4283	AV PROF ABRAAO DE MORAIS	1/4/2019	15:02:00	15:19:00
333859.7813	7395839.6785	R PAULA SOUSA	1/4/2019	15:04:00	15:53:00
333917.5214	7395412.4978	AV MERCURIO	1/4/2019	15:56:00	17:56:00
333560.2169	7395783.1148	AV SEN QUEIROS	1/4/2019	17:52:00	19:30:00
338242.6224	7391383.4413	R JOSE ZAPPI	1/4/2019	18:20:00	19:00:00
339084.9102	7390906.5483	AV PROF LUIZ IGNACIO ANHAIA MELLO	1/8/2019	16:15:00	18:15:00
338320.1863	7390923.5778	AV PROF LUIZ IGNACIO ANHAIA MELLO	1/8/2019	16:17:00	18:15:00
339905.8750	7391066.1873	AV PROF LUIZ IGNACIO ANHAIA MELLO	1/8/2019	16:21:00	18:15:00
332783.3766	7394601.3896	PC DA BANDEIRA	1/8/2019	16:22:00	17:47:00
334108.1104	7391411.2920	AV LACERDA FRANCO	1/8/2019	16:48:00	18:20:00
336990.2161	7389637.3741	R DAS JUNTAS PROVISORIAS	1/8/2019	17:04:00	17:30:00
332335.2142	7396128.4197	AV RIO BRANCO	1/8/2019	17:09:00	17:57:00
335788.6045	7391660.6369	AV TERESA CRISTINA	1/8/2019	17:09:00	17:49:00
336032.3902	7395411.1999	R BRESSER	1/8/2019	17:13:00	17:29:00
335484.8860	7393223.3621	AV DO ESTADO	3/11/2019	01:33:00	11:24:00
338115.4343	7390885.8703	AV PAES DE BARROS	3/11/2019	01:41:00	08:13:00
333960.9147	7396826.5193	R TIBIRICA	3/11/2019	04:45:00	08:44:00
334750.9597	7394053.3694	R OSCAR HORTA	3/11/2019	06:18:00	12:49:00
339566.8028	7391092.2596	AV PROF LUIZ IGNACIO ANHAIA MELLO	3/11/2019	06:24:00	09:57:00
337989.0360	7390778.1484	AV PROF LUIZ IGNACIO ANHAIA MELLO	3/11/2019	06:48:00	11:23:00
336743.6459	7391059.9899	AV DO ESTADO	3/11/2019	07:09:00	08:37:00
337224.2506	7391167.2432	AV HENRY FORD	3/11/2019	07:20:00	13:44:00
338471.7275	7389773.4490	AV DR FRANCISCO MESQUITA	3/11/2019	08:15:00	11:46:00
334234.6003	7394120.6241	AV DO ESTADO	3/11/2019	08:42:00	11:26:00
337907.6296	7390065.5609	R GUAMIRANGA	3/11/2019	08:47:00	12:08:00
334229.8188	7393632.5954	R OTTO DE ALENCAR	3/11/2019	09:15:00	09:29:00
339110.2889	7389131.1698	R IBITIRAMA	3/11/2019	15:46:00	17:41:00

Table 1: Table showing selected flooding events from the dataset used.

The flooding data that we use is available from the Github repository [Tomas, 2023]. Notice that this is a spreadsheet that contains, amongst other things, the coordinates (in UTM 23S system), dates, starting and ending times of several flooding events reported to and confirmed by Sao Paulo's Climate Emergency Management Center (CGE-SP). We show a few rows from this spreadsheet in Table 1.

Using this data and the A* algorithm, we are able to obtain paths between two vertices in the graph that minimize either: the total length of the path, the total travel time of the path, or one of the weight functions defined in Section 3, Equation (1). We will analyse a few examples below.

4.1. Short routes affected by rain

First, we consider a path in which a vehicle encounters no rain and no roads blocked by floods. In this scenario, one would expect that the route that minimizes the weight function p_l is the same as the one that minimizes the total length (since $p_l = l$) and the route that minimizes the weight function p_t is the same as the one that minimizes the total travel time (since $p_t = t$). This is indeed what happens when we set the coordinates of the origin of the route to be $(-23.5048^\circ, -46.4152^\circ)$, the coordinates of the destination to be $(-23.4986^\circ, -46.3864^\circ)$, and the initial time to be 4 pm on January 4, 2019 (see Table 2).

Next, we look at an example in which a vehicle encounters rain but no streets blocked

	Distance Traveled (km)	Travel Time (min:sec)	Rainfall on Route (mm)
Minimizing weight function p_l	4.007	6:34	0
Minimizing weight function p_t	4.509	5:55	0
Minimizing total distance	4.007	6:34	0
Minimizing total duration	4.509	5:55	0

Table 2: Routes from $(-23.5048^\circ, -46.4152^\circ)$ to $(-23.4986^\circ, -46.3864^\circ)$, at 4pm, on January 4, 2019

	Distance Traveled (km)	Travel Time (min:sec)	Rainfall on Route (mm)
Minimizing weight function p_l	4.007	6:34	58,856.700
Minimizing weight function p_t	4.603	6:01	49,073.700
Minimizing total distance	4.007	6:34	58,856.700
Minimizing total duration	4.509	5:55	51,406.700

Table 3: Routes from $(-23.5048^\circ, -46.4152^\circ)$ to $(-23.4986^\circ, -46.3864^\circ)$, at 4:20pm, on Jan. 4, 2019

by floods on its path. This is the scenario that one encounters when one sets the coordinates of the origin of our route to be $(-23.5048^\circ, -46.4152^\circ)$, the coordinates of the destination to be $(-23.4986^\circ, -46.3864^\circ)$, and the initial time to be 4:20 pm on January 4, 2019 (see Table 3).

Notice that, in this latter case, in order to minimize the function p_t and decrease the amount of rain to which a vehicle is exposed, one must choose a route that no longer minimizes the function t . Also notice that, in this case, the route that minimizes the total length is the same route that minimizes the function p_l .

As an illustration, in Figure 4 we show a few different scenarios of routes whose coordinates of the origins are $(-23.5048^\circ, -46.4152^\circ)$ and the coordinates of the destinations are $(-23.4986^\circ, -46.3864^\circ)$.

4.2. Longer routes affected by rain

The examples given in Section 4.1 show that the proposed method is able to deviate from routes which are affected by rain when the duration of travel is short, that is, less than 10 minutes. In this section, we will present examples that show that our method is also effective for longer routes.

We begin by confirming that, when there is no precipitation predicted for the routes that minimize the total distance and total time, the route that minimizes the weight function p_l coincides with the former and the route that minimizes the weight function p_t coincides with the latter (see Table 4).

These results are obtained by setting the coordinates of the origin of the route to be $(-23.5731^\circ, -46.5695^\circ)$, the coordinates of the destination to be $(-23.5814^\circ, -46.6385^\circ)$, and



Figure 4: Routes from $(-23.5048^\circ, -46.4152^\circ)$ to $(-23.4986^\circ, -46.3864^\circ)$ on January 4, 2019

	Distance Traveled (km)	Travel Time (min:sec)	Rainfall on Route (mm)
Minimizing weight function p_l	8.972	12:33	0
Minimizing weight function p_t	9.068	12:31	0
Minimizing total distance	8.972	12:33	0
Minimizing total duration	9.068	12:31	0

Table 4: Routes from $(-23.5731^\circ, -46.5695^\circ)$ to $(-23.5814^\circ, -46.6385^\circ)$, at 4:20pm, on Jan. 6, 2019

	Distance Traveled (km)	Travel Time (min:sec)	Rainfall on Route (mm)
Minimizing weight function p_l	8.972	12:33	2,155.000
Minimizing weight function p_t	9.194	12:39	1,991.200
Minimizing total distance	8.972	12:33	2,155.000
Minimizing total duration	9.068	12:31	2,617.600

Table 5: Routes from $(-23.5731^\circ, -46.5695^\circ)$ to $(-23.5814^\circ, -46.6385^\circ)$, at 4:40pm, on Jan. 6, 2019

the initial time to be 4:20 pm. Next, we consider the case where the coordinates of the origin are also $(-23.5731^\circ, -46.5695^\circ)$, the coordinates of the destination are also $(-23.5814^\circ, -46.6385^\circ)$, and the initial time is 4:40 pm (see Table 5).

Now, as a comparison, we present the results of a similar, static method. That is, even though an updated precipitation prediction is received at 4:50 pm, we *do not change* the routes accordingly (see Table 6). These results show that the proposed method is able to respond in real-time to the changes in the precipitation prediction and thus modify the optimal routes. In fact, in this case, the amount of time necessary to perform this change was 12 seconds.

Also notice in Table 6 that, when the vehicle leaves the origin, even though it is raining on the routes that minimize total distance and total time, these routes also minimize the weight functions p_l and p_t respectively. This is a consequence of the penalty that was chosen (namely, to increase the length and travel time by $r(e)\%$; cf. Section 5). However, by comparing it with the dynamical method, one notices that, once the weather prediction is updated, the route that minimizes the total duration no longer minimizes p_t .

4.3. Routes affected by rain and flooding

This section presents the analysis of a rain and flood situation for a route from Belenzinho $(-23.5367^\circ, -46.5898^\circ)$ to Teatro São Pedro $(-23.5319^\circ, -46.6536^\circ)$, which took place on January 4th, 2019, in the afternoon. This day, four floods were recorded by CGE-SP. These occurrences were identified in different locations and specific time periods. The first flood event occurred on Av. Senador Queiros, between 5:52 pm and 7:30 pm. Subsequently, Rua da Cantareira was affected by a flood from 2:45 pm to 3:03 pm. Shortly after, there was a flood on Rua Paula Sousa from 3:04 pm to 3:53 pm. Finally, Av. Mercúrio experienced a flood from 3:56 pm to 5:56 pm.

	Distance Traveled (km)	Travel Time (min:sec)	Rainfall on Route (mm)
Minimizing weight function p_l	8.972	12:33	3,110.600
Minimizing weight function p_t	9.068	12:39	3,075.600
Minimizing total distance	8.972	12:33	3,110.600
Minimizing total duration	9.068	12:31	3,075.600

Table 6: Routes from $(-23.5731^\circ, -46.5695^\circ)$ to $(-23.5814^\circ, -46.6385^\circ)$, at 4:40pm, on Jan. 6, 2019

	Distance Traveled (km)	Travel Time (min:sec)	Rainfall on Route (mm)
Minimizing weight function p_l	7.724	11:07	0.000
Minimizing weight function p_t	7.827	10:30	0.000
Minimizing total distance	7.724	11:07	0.000
Minimizing total duration	7.827	10:30	0.000

Table 7: Routes from $(-23.5367^\circ, -46.5898^\circ)$ to $(-23.5319^\circ, -46.6536^\circ)$, at 2pm, on January 4, 2019

	Distance Traveled (km)	Travel Time (min:sec)	Rainfall on Route (mm)
Minimizing weight function p_l	7.795	10:57	12.800
Minimizing weight function p_t	8.039	10:52	0.000
Minimizing total distance	7.724	11:07	0.000
Minimizing total duration	7.827	10:30	0.000

Table 8: Routes from $(-23.5367^\circ, -46.5898^\circ)$ to $(-23.5319^\circ, -46.6536^\circ)$, at 3pm, on January 4, 2019

In our initial simulation at 2 pm, we obtained similar results for two different objectives: minimizing the total length of the route with or without considering weight, p_l , and minimizing travel time with or without considering weight, p_t . These results are presented in Table 7 and show that the results were consistent with previous cases without rain.

At 3 pm, one observes changes in the penalized routes (see Table 8), which are caused by the brief flood that occurred at Rua da Cantareira $(-23.5417^\circ, -46.6284^\circ)$ between 2:45 pm and 3:03 pm.

By 4 pm, a flood had been reported to CGE-SP near Av. Mercurio $(-23.543^\circ, -46.627^\circ)$. In response to this flood, the routes that minimize the weight functions p_l and p_t avoid this area, ensuring a minimum distance of 200 meters from the center of the reported flood (see Table 9). Despite this detour, the total distance covered did not significantly increase.

In the simulation that starts at 4:30 pm, one observes that the route that minimized p_t at 4 pm encountered more intense rainfall. Hence, the route that minimizes p_t at 4:30 pm is changed as the data is updated (see Table 10).

Overall, the analysis demonstrates that the proposed method is able to dynamically respond to rain and flood events, adjusting the optimal routes accordingly. By considering the weight functions and penalty factors, the algorithm can effectively navigate through affected areas and minimize the impact of adverse weather conditions on travel time and distance.

4.4. Routes whose destination is flooded

In Section 4.3, we showed that the method proposed in this article is able to find routes that deviate from flooded streets. The reason why this happens is that, by setting $p_t(e), p_l(e)$ to

	Distance Traveled (km)	Travel Time (min:sec)	Rainfall on Route (mm)
Minimizing weight function p_l	7.795	10:57	6,427.200
Minimizing weight function p_t	7.799	10:53	6,245.100
Minimizing total distance	7.724	11:07	5,222.300
Minimizing total duration	7.827	10:30	4,936.900

Table 9: Routes from $(-23.5367^\circ, -46.5898^\circ)$ to $(-23.5319^\circ, -46.6536^\circ)$, at 4pm, on January 4, 2019

	Distance Traveled (km)	Travel Time (min:sec)	Rainfall on Route (mm)
Minimizing weight function p_l	7.795	10:57	1,587.600
Minimizing weight function p_t	8.039	10:52	1,640.000
Minimizing total distance	7.724	11:07	1,399.300
Minimizing total duration	7.827	10:30	1,348.300

Table 10: Routes from $(-23.5367^\circ, -46.5898^\circ)$ to $(-23.5319^\circ, -46.6536^\circ)$, at 4:30pm, on Jan. 4, 2019

	Distance Traveled (km)	Travel Time (min:sec)	Rainfall on Route (mm)
Minimizing weight function p_l	1.218	1:53	0
Minimizing weight function p_t	1.218	1:53	0
Minimizing total distance	0.764	0:56	0
Minimizing total duration	0.767	0:56	0

Table 11: Routes from $(-23.5390^\circ, -46.6459^\circ)$ to $(-23.5349^\circ, -46.6402^\circ)$, at 5 pm, on Jan. 8, 2019

be infinite whenever e is flooded, and $p_t(e), p_l(e)$ to be finite otherwise, a path on the graph that minimizes the weight functions p_t, p_l will, whenever possible, contain no edges that correspond to flooded streets.

However, it may happen that there exists no paths from the origin vertex to the destination vertex that contain no edges with finite p_t and p_l weights. In this section, we show that, this case, the proposed method is still able to find a path between the origin and destination. This justifies maintaining the edges corresponding to flooded streets on the graph (instead of deleting them), as well as the choice of weight functions p_t and p_l .

Consider the case where the coordinates of the origin are $(-23.5390^\circ, -46.6459^\circ)$, the coordinates of the destination are $(-23.5349^\circ, -46.6402^\circ)$, and the initial time is 5 pm, on January 8, 2019. Since there is a one-way street from the origin to the destination, the shortest and fastest paths linking them are the path that goes straight from origin to destination. However, on January 8, 2019, from 5:09 pm to 5:57 pm, there is a flood reported in an area crossed by this straight path (see Table 1). Thus, the paths that minimize p_l and p_t deviate from this area. This can be observed in the results on Table 11 as an increase in their distance and duration.

These results confirm that the proposed method is able to deviate from flooded areas. Furthermore, one can consider a case where the initial vertex and initial time are the same as above, but the destination lies within the flooded area, namely, by setting the coordinates of the destination to be $(-23.5368^\circ, -46.6429^\circ)$ (see Table 12).

Notice that, even though the algorithm is not able to find a path that passes through no flooded streets, it is still able to find a path from its origin to its destination. Also notice that in these

	Distance Traveled (km)	Travel Time (min:sec)	Rainfall on Route (mm)
Minimizing weight function p_l	0.359	0:26	0
Minimizing weight function p_t	0.359	0:26	0
Minimizing total distance	0.359	0:26	0
Minimizing total duration	0.359	0:26	0

Table 12: Routes from $(-23.5390^\circ, -46.6459^\circ)$ to $(-23.5368^\circ, -46.6429^\circ)$, at 5 pm, on Jan. 8, 2019

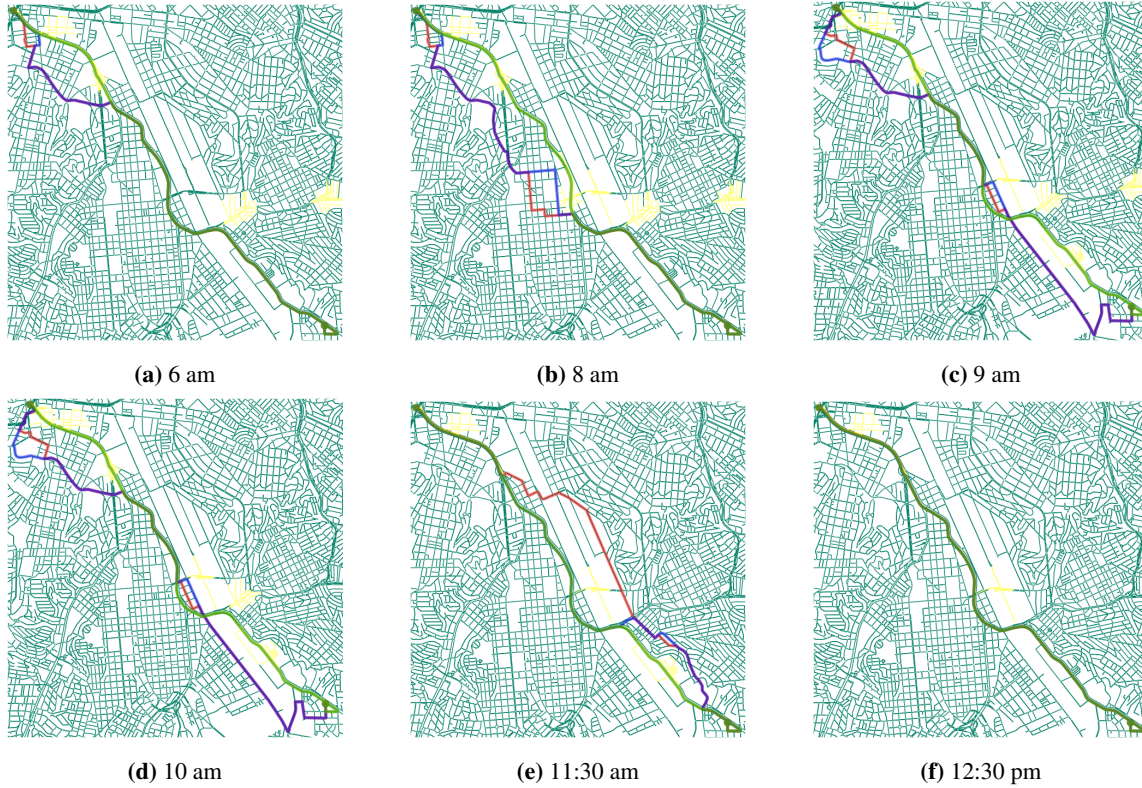


Figure 5: Routes around Avenida dos Estados on March 11th, 2019

examples, there is no predicted rain. In fact, $p_l = l$, $p_t = t$, and the flood event is a result of other factors (cf. Section 5).

4.5. Routes impacted by floods

Finally, we analyse a case of flooding that had a noticeable impact on the population, and show that the method proposed in the article would have been able to find routes that would avoid the reported flooded areas.

On the evening of March 10th, 2019, a rain storm struck the city of Sao Paulo [G1, 2019]. This storm resulted in several flooding events being reported to and confirmed by CEG-SP. In particular, an important road in Sao Paulo's transportation hub, *Avenida dos Estados*, was flooded in several different points at several different times (see Table 1). In this section, we simulate a situation in which a vehicle tries to cross a section of this avenue at certain specific times in the morning of March 11th, 2019, and propose routes that avoid the resulting floods.

First, set the coordinates of the origin of our routes to be $(-23.5523^\circ, -46.6263^\circ)$ and the coordinates of the destination to be $(-23.6031^\circ, -46.5720^\circ)$. Then, we compare routes that start at 6am, 8am and 12pm, on March 11th, 2019. As an illustration, we show the obtained routes in Figure 5.

If a vehicle started to cross this section of Av. dos Estados at 6am, then it would encounter a flooded area near the beginning of the route, resulting in a longer route (see Table 13). Notice that rain does not affect these routes. In fact, the only factor that disrupts these routes from $(-23.5523^\circ, -46.6263^\circ)$ to $(-23.6031^\circ, -46.5720^\circ)$ are flooding events. For instance, if a vehicle started to cross this section of Av. dos Estados at 8am, then it would encounter a larger flooded area

	Distance Traveled (km)	Travel Time (min:sec)	Rainfall on Route (mm)
Minimizing weight function p_l	9.578	12:04	0
Minimizing weight function p_t	9.605	11:56	0
Minimizing total distance	8.955	10:55	0
Minimizing total duration	9.038	10:55	0

Table 13: Routes from $(-23.5523^\circ, -46.6263^\circ)$ to $(-23.6031^\circ, -46.5720^\circ)$, at 6 am, on Mar. 11, 2019

	Distance Traveled (km)	Travel Time (min:sec)	Rainfall on Route (mm)
Minimizing weight function p_l	9.893	13:05	0
Minimizing weight function p_t	9.922	12:34	0
Minimizing total distance	8.955	10:55	0
Minimizing total duration	9.038	10:55	0

Table 14: Routes from $(-23.5523^\circ, -46.6263^\circ)$ to $(-23.6031^\circ, -46.5720^\circ)$, at 8 am, on Mar. 11, 2019

near the beginning of the route (see Table 14). Later, if a vehicle started to cross this section of Av. dos Estados at 12pm, then it would encounter no flooded area near the beginning, but a large flooded area towards the end of the route (see Table 15).

	Distance Traveled (km)	Travel Time (min:sec)	Rainfall on Route (mm)
Minimizing weight function p_l	9.331	12:39	0
Minimizing weight function p_t	9.558	11:54	0
Minimizing total distance	8.955	10:55	0
Minimizing total duration	9.038	10:55	0

Table 15: Routes from $(-23.5523^\circ, -46.6263^\circ)$ to $(-23.6031^\circ, -46.5720^\circ)$, at 12 pm, on Mar. 11, 2019

5. Conclusions

In this work, we propose a method for finding routes that avoid rain and flooding in a city. Using cases from the city of Sao Paulo during the Summer of 2019, we show that the proposed method is effectively able to find routes that deviate from rain and flooding, while optimizing the duration and length of the resulting routes.

The functions that are chosen as weight functions are deformations of the lengths and travel times by a linear factor given by the amount of rain that is predicted for each edge. They are standardized and do not differentiate between types of vehicles, such as motorcycles, cars and trucks. In order to improve the proposed model, one could acknowledge the vehicle type and adapt the weight functions accordingly.

Another aspect that our model does not presently consider is real-time traffic conditions for estimating travel times. Incorporating data on traffic flow would significantly improve the accuracy of the model in avoiding rain and floods, estimating travel time, and optimizing p_t .

It is important to note that for the practical deployment of this model, the ability to predict flooding events is crucial. The examples presented in Section 4 relied on historical flooding data. However, to effectively utilize this method in the future, it is necessary to predict flooding events in advance. A possible enhancement could involve integrating hydraulic models with a machine learning approach that utilizes recent rainfall data along with historical flooding information for training

purposes. It is also worth investigating the various causes of flooding events, as understanding these causes can offer valuable insights for improved prediction and mitigation strategies. Researchers working in this domain can explore alternative approaches to enhance flood forecasting, drawing inspiration from studies such as [Todini, 2007; Kratzert et al., 2019; Herath et al., 2021; Nevo et al., 2022].

To conclude, our proposed method demonstrates promising results in identifying optimal routes that avoid rain and floods, thus contributing to more efficient and safer transportation systems in urban areas. By incorporating additional data sources and predictive capabilities, we can further refine the model and apply it in practical scenarios, leading to tangible improvements in urban mobility.

References

- Dechter, R. e Pearl, J. (1985). Generalized best-first search strategies and the optimality of A*. *J. Assoc. Comput. Mach.*, 32(3):505–536. ISSN 0004-5411. URL <https://doi.org/10.1145/3828.3830>.
- Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numer. Math.*, 1: 269–271. ISSN 0029-599X. URL <https://doi.org/10.1007/BF01386390>.
- G1 (2019). Forte chuva isola áreas na grande são paulo e causa alagamentos. <http://g1.globo.com/sp/sao-paulo/noticia/2019/03/11/forte-chuva-isola-areas-na-grande-sao-paulo-e-causa-alagamentos.ghtml>.
- Hart, P. E., Nilsson, N. J., e Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics*, 4(2):100–107.
- Herath, H. M. V. V., Chadalawada, J., e Babovic, V. (2021). Hydrologically informed machine learning for rainfall–runoff modelling: towards distributed modelling. *Hydrology and Earth System Sciences*, 25(8):4373–4401. URL <https://hess.copernicus.org/articles/25/4373/2021/>.
- Kratzert, F., Klotz, D., Shalev, G., Klambauer, G., Hochreiter, S., e Nearing, G. (2019). Towards learning universal, regional, and local hydrological behaviors via machine learning applied to large-sample datasets. *Hydrology and Earth System Sciences*, 23(12):5089–5110. URL <https://hess.copernicus.org/articles/23/5089/2019/>.
- Nevo, S., Morin, E., Gerzi Rosenthal, A., Metzger, A., Barshai, C., Weitzner, D., Voloshin, D., Kratzert, F., Elidan, G., Dror, G., Begelman, G., Nearing, G., Shalev, G., Noga, H., Shavitt, I., Yuklea, L., Royz, M., Giladi, N., Peled Levi, N., Reich, O., Gilon, O., Maor, R., Timnat, S., Shechter, T., Anisimov, V., Gigi, Y., Levin, Y., Moshe, Z., Ben-Haim, Z., Hassidim, A., e Matias, Y. (2022). Flood forecasting with machine learning models in an operational framework. *Hydrology and Earth System Sciences*, 26(15):4013–4032. URL <https://hess.copernicus.org/articles/26/4013/2022/>.
- Todini, E. (2007). Hydrological catchment modelling: past, present and future. *Hydrology and Earth System Sciences*, 11(1):468–482. URL <https://hess.copernicus.org/articles/11/468/2007/>.
- Tomas, L. (2023). E-floods-2019. http://github.com/liviatomas/floods_saopaulo/raw/main/1_input/e_Floods_2019.xlsx.