F

final-assignment

Forked from COM490 / final-assignment

fix Dockerfile
Eric Pierre Bouillet authored 1 week ago

Name	Last commit	Last update
.renku	WIP: final assignment	1 week ago
data	dslab2022-renku.epfl.ch: init final-assignment	1 week ago
figs	WIP: with figures	1 week ago
notebooks	dslab2022-renku.epfl.ch: init final-assignment	1 week ago
.dockerignore	dslab2022-renku.epfl.ch: init final-assignment	1 week ago
.gitignore	dslab2022-renku.epfl.ch: init final-assignment	1 week ago
.gitlab-ci.yml	dslab2022-renku.epfl.ch: init final-assignment	1 week ago
.renkulfsignore	dslab2022-renku.epfl.ch: init final-assignment	1 week ago
Dockerfile	fix Dockerfile	1 week ago
README.md	fix README	1 week ago
environment.yml	WIP: final assignment	1 week ago
requirements.txt	WIP: final assignment	1 week ago

README.md

Final Assignment: Robust Journey Planning

Executive summary: build a robust SBB journey planner for the Zürich area, and make a short video presentation of it - to be done in **groups of 4 or 5**, before **midnight of May 29**.

Content

- References
- FAQ

HOW-TO

This section intentionally blank. Students must complete it with instructions on how to use the code in their project.

top

Important Dates

The assignment (clear, well-annotated notebook and/or code; report-like), **with a short, 7-minute video of your presentation** is due on **Sunday May 29th, 23:59 (noon) CEST**.

For the oral defense, we will organize short Q&A discussions of 8 minutes per group. These discussions will be scheduled on **Wednesday June 1st, 13:00 - 18:00 CEST** - tentatively, actual times to be discussed on a case by case basis.

top

Problem Motivation

Imagine you are a regular user of the public transport system, and you are checking the operator's schedule to meet your friends for a class reunion. The choices are:

1. You could leave in 10mins, and arrive with enough time to spare for gossips before the reunion starts.
2. You could leave now on a different route and arrive just in time for the reunion.

Undoubtedly, if this is the only information available, most of us will opt for option 1.

If we now tell you that option 1 carries a fifty percent chance of missing a connection and be late for the reunion. Whereas, option 2 is almost guaranteed to take you there on time. Would you still consider option 1?

Probably not. However, most public transport applications will insist on the first option. This is because they are programmed to plan routes that offer the shortest travel times, without considering the risk factors.

top

Problem Description

In this final project you will build your own *robust* public transport route planner to improve on that. You will reuse the SBB dataset (See next section: Dataset Description).

Given a desired arrival time, your route planner will compute the fastest route between departure and arrival stops within a provided confidence tolerance expressed as interquartiles. For instance, "what route from *A* to *B* is the fastest at least *Q*% of the time if I want to arrive at *B* before instant *T*". Note that *confidence* is a measure of a route being feasible within the travel time computed by the algorithm.

The output of the algorithm is a list of routes between *A* and *B* and their confidence levels. The routes must be sorted from latest (fastest) to earliest (longest) departure time at *A*, they must all arrive at *B* before *T* with a confidence level greater than or equal to *Q*. Ideally, it should be possible to visualize the routes on a map with straight lines connecting all the stops traversed by the route.

In order to answer this question you will need to:

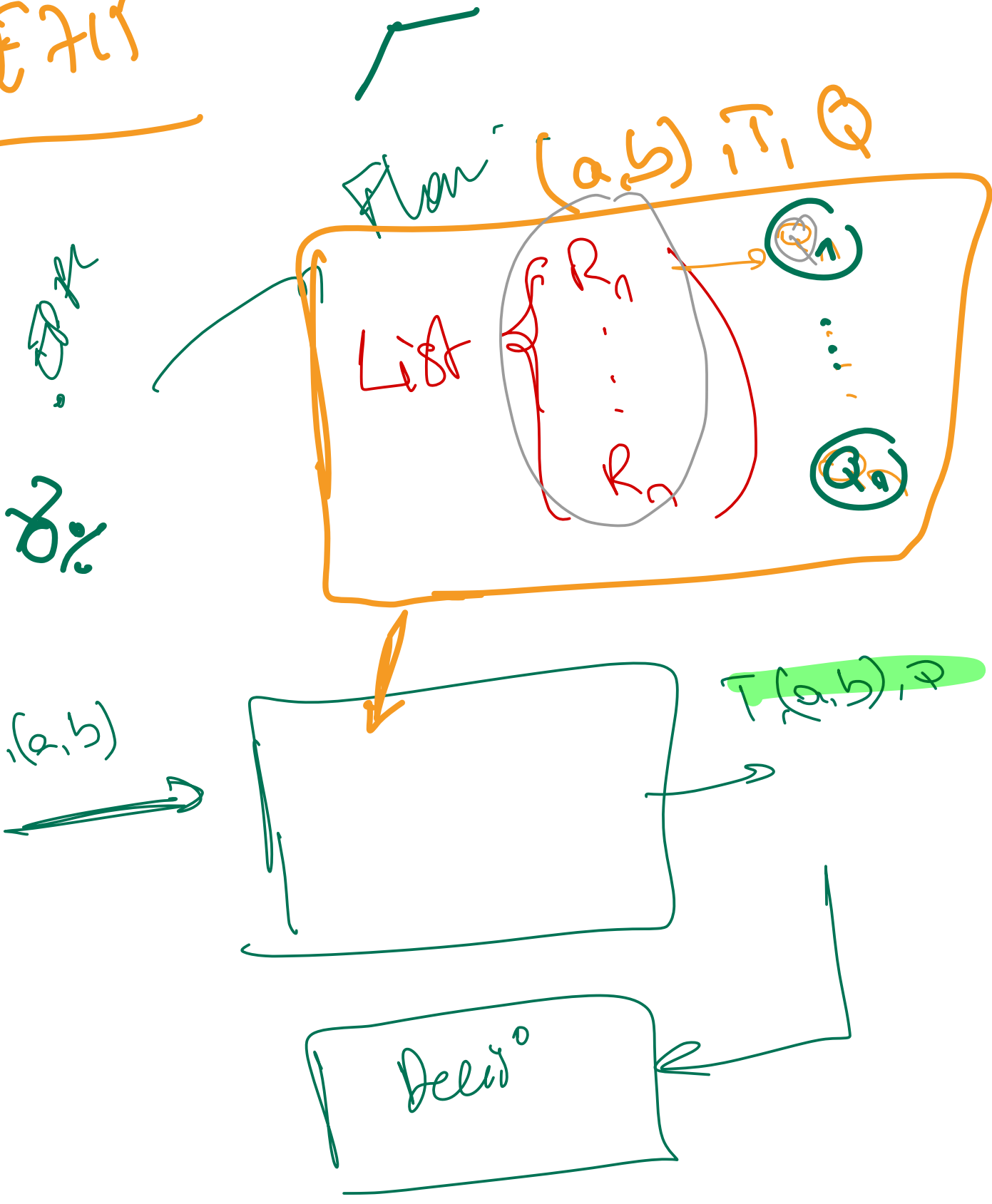
- Model the public transport infrastructure for your route planning algorithm using the data provided to you.
- Build a predictive model using the historical arrival/departure time data, and optionally other sources of data.
- Implement a robust route planning algorithm using this predictive model.
- Test and validate your results.
- Implement a simple Jupyter-based visualization to demonstrate your method, using Jupyter dashboard such as Voilà or ipywidgets.

Solving this problem accurately can be difficult. You are allowed a few **simplifying assumptions**:

- We only consider journeys at reasonable hours of the day, and on a typical business day, and assuming the schedule of May 13-17, 2019.
- We allow short (total max 500m "As the Crows Flies") walking distances for transfers between two stops, and assume a walking speed of *50m/1min* on a straight line, regardless of obstacles, human-built or natural, such as building, highways, rivers, or lakes.
- We only consider journeys that start and end on known station coordinates (train station, bus stops, etc.), never from a random location. However, walking from the departure stop to a nearby stop is allowed.
- We only consider departure and arrival stops in a 15km radius of Zürich's train station, `Zürich HB (8503000)`, (lat, lon) = `(47.378177, 8.540192)`.

Route = $\int$ (a;b) : points
T : instant d'arrivée
Q : confidence level.

RETOUR



d_{ij} : ⟶ ① passengers

↳ Route

we already have? train-specific

- Delays of each ~~station~~

VSN ⟶ GNN ⟶ BERN.

⟶ patterns.

Last 20 mondays -

loop over n days,

consider a train

↳ compute its delay

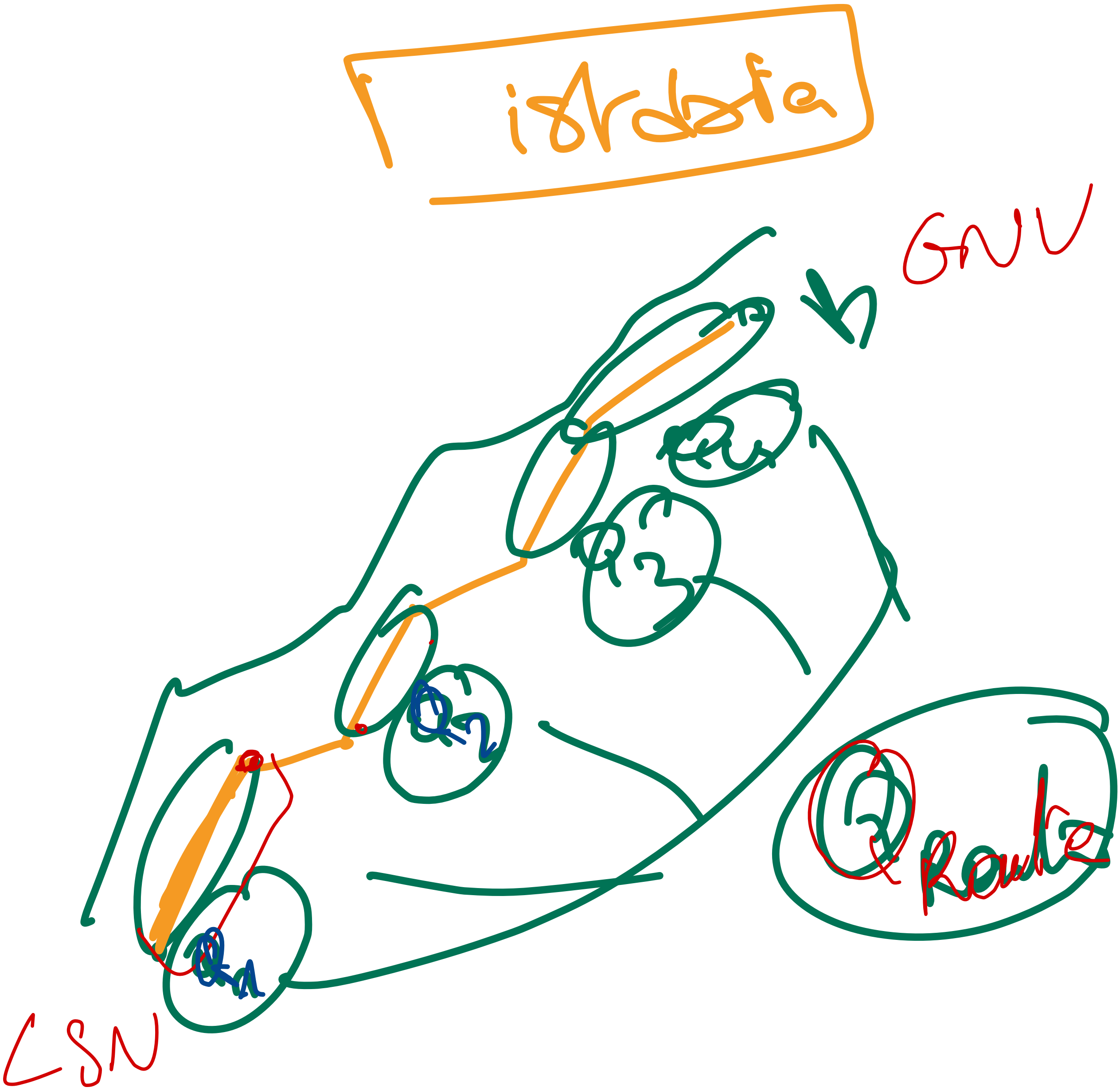
↳ q

R1 R2

A,B

actual ans → expected

↳ Δ



2min : 20 min

Hip A :

	actual	expected
	.	.

$\Delta = 0$

$\Delta > 0$.

Std

count + 1

① $\Delta > 0$ →

② Set $\{\Delta_1 \dots \Delta_n\}$

Mean, Std, Median

+ std

mean

↳ Aggregat

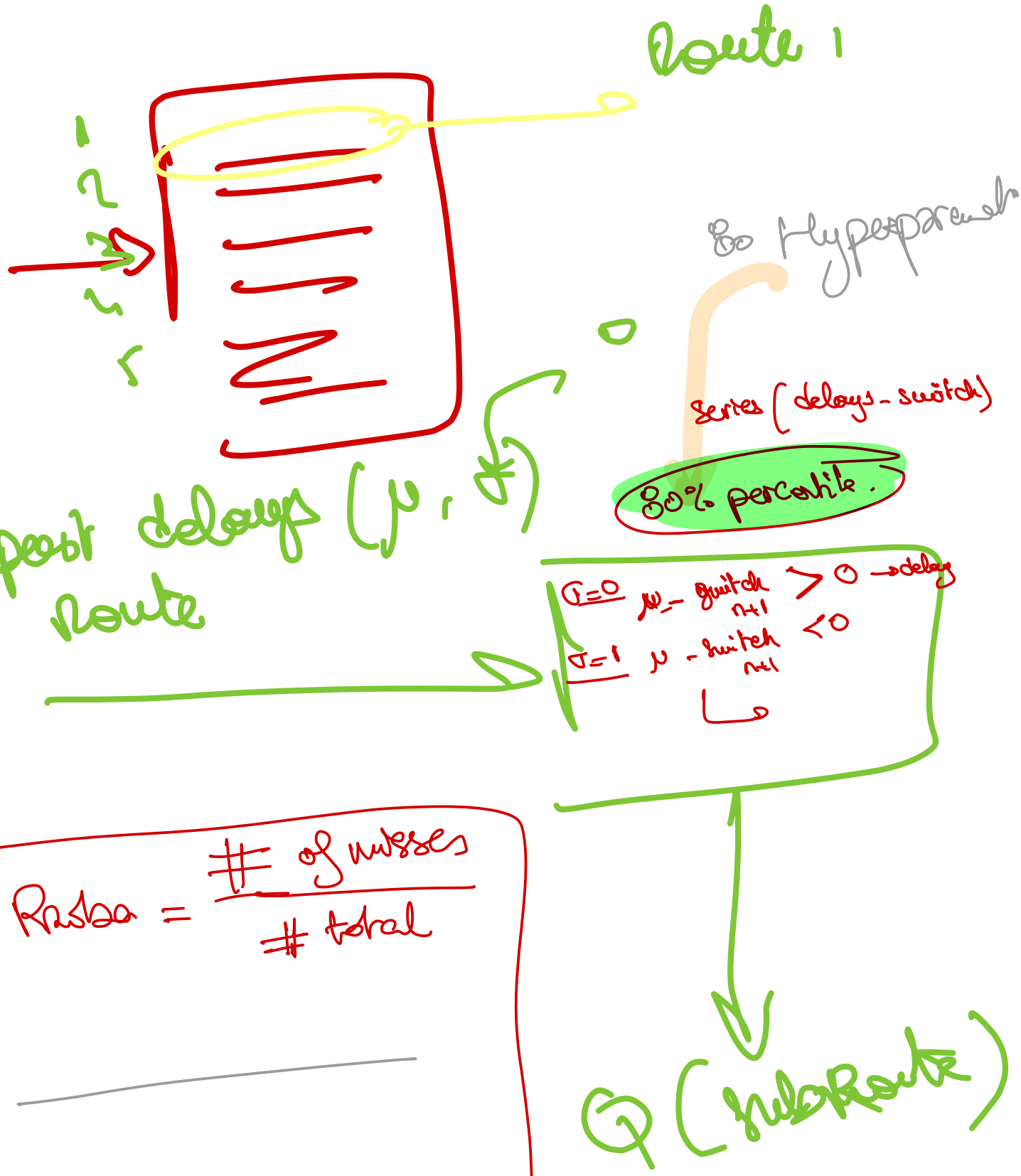
Compare this delay to the switch time - walking time

↳ Q :

with : $A \in$ hole

with this confidence Level Q

$Q = 70\%$ ⟹ Factor ↑ $Q = 70\%$



$$\text{Rasba} = \frac{\#\ \text{of misses}}{\#\ \text{total}}$$

$Q_1 = 0{,}3$

$Q_2 = 0 = Q_3$

$Q_i =$

$Q_1 = 0{,}3 \neq Q_2 = Q_3$

wait = 15 min.

↳ How many times

∴ 1

- Quadratic in positive values.

↳

- Testing our delay predict°

↳ splitting the data.

- We only consider stops in the 15km radius that are reachable from Zürich HB. If needed stops may be reached via transfers through other stops outside the 15km area.
- There is no penalty for assuming that delays or travel times on the public transport network are uncorrelated with one another.
- Once a route is computed, a traveller is expected to follow the planned routes to the end, or until it fails (i.e. miss a connection). You **do not** need to address the case where travellers are able to defer their decisions and adapt their journey "en route", as more information becomes available. This would require us to consider all alternative routes (contingency plans) in the computation of the uncertainty levels, which is more difficult to implement.
- The planner will not need to mitigate the traveller's inconvenience if a plan fails. Two routes with identical travel times under the uncertainty tolerance are equivalent, even if the outcome of failing one route is much worse for the traveller than failing the other route, such as being stranded overnight on one route and not the other. ⟹ Pas besoin de considérer l'cas où le voyage a été raté et ss conséq
- All other things being equal, we will prefer routes with the minimum walking distance, and then minimum number of transfers.
- You do not need to optimize the computation time of your method, as long as the run-time is reasonable.
- You may assume that the timetables remain unchanged throughout the 2018 - 2020 period.

Upon request, and with clear instructions from you, we can help prepare the data in a form that is easier for you to process (within the limits of our ability, and time availability). In which case the data will be accessible to all.

top

Project Submission Checklist

- Project and 7 minute (max) video are due before midnight of May 30th.
- The final assignment is to be done in **groups of 4 or 5**, remember to update your group member list if needed.
- All projects must be submitted on Renku, as a group project.
- Project must contain `final` in the name, or you can fork this final-assignment project.
- Provide instructions on how to test your project in the **HOW TO** section of the `README.md` file. Include a link to your video presentation.
- Project sizes, including history, must not exceed 100Mb. Use git-lfs for your larger data sets, or keep as much data as possible on HDFS.

Note: use `git lfs migrate import --fixup --include-ref=refs/heads/master` if you accidentally push a large data set on gitlab. See using git lfs responsibly in the renku documentation. Since you will be rewriting history, you will need to unprotect your branch in gitlab and force `git push -f`, and coordinate with your peers to make sure that you are all working off the same history.

top

Video Presentations

Instruction for video presentations:

1. Use Zoom (or other tools) to record your group video.
2. Save the video as an mp4 file.
3. Upload your video to moodle under `Final assignment - video presentation`.
4. Include the link to the video in the **HOW TO** section, at the top of the `README.md` file of your final assignment

Please, **DO NOT** load the video as part of your project, send a video embedded in a PowerPoint presentations, or use any format other than mp4 videos. We must be able to stream the videos in our web browsers.

top

Grading Method

At the end of the term you will provide a 7-minute video, in which each member of the project presents a part of the project.

After reviewing your videos, we will invite each group for a 8 mins Q&A. Before the Q&A, we will validate your method on a list of pre-selected departure arrival points, and times of day.

Think of yourselves as a startup trying to sell your solution to the board of a public transport company. Your video is your elevator pitch. It must be short and convincing. In it you describe the viability of the following aspects:

1. Method used to model the public transport network
2. Method used to create the predictive models
3. Route planning algorithm
4. Validation method

Your grades will be based on the code, videos and Q&A, taking into account:

1. Clarity and conciseness of the video presentation, code and Q&A
2. Team work, formulation and decomposition of the problem into smaller tasks between team members
3. Originality of the solution design, analytics, and presentation
4. Functional quality of the implementation (does it work?)
5. Explanation of the pro's and con's / shortcomings of the proposed solution

top

Dataset Description

For this project we will use the data published on the Open Data Platform Mobility Switzerland.

We will use the SBB data limited around the Zurich area, focusing only on stops within 15km of the Zurich main train station.

Actual data

Students should already be familiar with the istdaten.

The 2018 to 2021 data is available as a Hive table in partitioned ORC format on our HDFS system, under `/data/sbb/orc/istdaten` .

See assignments and exercises of earlier weeks for more information about this data, and methods to access it.

As a reminder, we provide the relevant column descriptions below. The full description of the data is available in the opentransportdata.swiss data istdaten cookbooks. If needed you can translate the column names and descriptions from German to English with an automated translator, such as DeepL.

- `BETRIEBSTAG` : date of the trip
- `FAHRT_BEZEICHNER` : identifies the trip
- `BETREIBER_ABK` , `BETREIBER_NAME` : operator (name will contain the full name, e.g. Schweizerische Bundesbahnen for SBB)
- `PRODUCT_ID` : type of transport, e.g. train, bus
- `LINIEN_ID` : for trains, this is the train number
- `LINIEN_TEXT` , `VERKEHRSMITTEL_TEXT` : for trains, the service type (IC, IR, RE, etc.)
- `ZUSATZFAHRT_TF` : boolean, true if this is an additional trip (not part of the regular schedule)
- `FAELLT_AUS_TF` : boolean, true if this trip failed (cancelled or not completed)
- `HALTESTELLEN_NAME` : name of the stop
- `ANKUNFTSZEIT` : arrival time at the stop according to schedule
- `AN_PROGNOSE` : actual arrival time (see `AN_PROGNOSE_STATUS`)
- `AN_PROGNOSE_STATUS` : method used to measure `AN_PROGNOSE` , the time of arrival.
- `ABFAHRTSZEIT` : departure time at the stop according to schedule
- `AB_PROGNOSE` : actual departure time (see `AN_PROGNOSE_STATUS`)
- `AB_PROGNOSE_STATUS` : method used to measure `AB_PROGNOSE` , the time of departure.
- `DURCHFAHRT_TF` : boolean, true if the transport does not stop there

Each line of the file represents a stop and contains arrival and departure times. When the stop is the start or end of a journey, the corresponding columns will be empty (`ANKUNFTSZEIT` / `ABFAHRTSZEIT`). In some cases, the actual times were not measured so the `AN_PROGNOSE_STATUS` / `AB_PROGNOSE_STATUS` will be empty or set to `PROGNOSE` and `AN_PROGNOSE` / `AB_PROGNOSE` will be empty.

Timetable data

We have copied the timetable to HDFS.

We are in the process of converting the files in an easy to query table form, and will keep you updated when the tables are available.

You will find there the timetables for the years 2018, 2019 and 2020. The timetables are updated weekly. It is ok to assume that the weekly changes are small, and a timetable for a given week is thus the same for the full year - use the schedule of a recent week that is a typical week for the year.

Only GTFS format has been copied on HDFS, the full description of which is available in the opentransportdata.swiss data timetable cookbooks. The more courageous who want to give a try at the Hafas Raw Data Format (HRDF) format must contact us.

We provide a summary description of the files below. The most relevant files are marked by (+):

- stops.txt(+):
  - `STOP_ID` : unique identifier (PK) of the stop
  - `STOP_NAME` : long name of the stop
  - `STOP_LAT` : stop latitude (WGS84)
  - `STOP_LON` : stop longitude
  - `LOCATION_TYPE` :
  - `PARENT_STATION` : if the stop is one of many collocated at a same location, such as platforms at a train station

- stop_times.txt(+):
  - `TRIP_ID` : identifier (FK) of the trip, unique for the day - e.g. *1.TA.1-100-j19-1.1.H*
  - `ARRIVAL_TIME` : scheduled (local) time of arrival at the stop (same as DEPARTURE_TIME if this is the start of the journey)
  - `DEPARTURE_TIME` : scheduled (local) time of departure at the stop
  - `STOP_ID` : stop (station) identifier (FK), from stops.txt
  - `STOP_SEQUENCE` : sequence number of the stop on this trip id, starting at 1.
  - `PICKUP_TYPE` :
  - `DROP_OFF_TYPE` :
- trips.txt:
  - `ROUTE_ID` : identifier (FK) for the route. A route is a sequence of stops. It is time independent.
  - `SERVICE_ID` : identifier (FK) of a group of trips in the calendar, and for managing exceptions (e.g. holidays, etc).
  - `TRIP_ID` : is one instance (PK) of a vehicle journey on a given route - the same route can have many trips at regular intervals; a trip may skip some of the route stops.
  - `TRIP_HEADSIGN` : displayed to passengers, most of the time this is the (short) name of the last stop.
  - `TRIP_SHORT_NAME`: internal identifier for the trip_headsign (note TRIP_HEADSIGN and TRIP_SHORT_NAME are only unique for an agency)
  - `DIRECTION_ID` : if the route is bidirectional, this field indicates the direction of the trip on the route.
- calendar.txt:
  - `SERVICE_ID` : identifier (PK) of a group of trips sharing a same calendar and calendar exception pattern.
  - `MONDAY` .. `SUNDAY` : 0 or 1 for each day of the week, indicating occurence of the service on that day.
  - `START_DATE` : start date when weekly service id pattern is valid
  - `END_DATE` : end date after which weekly service id pattern is no longer valid
- routes.txt:
  - `ROUTE_ID` : identifier for the route (PK)
  - `AGENCY_ID` : identifier of the operator (FK)
  - `ROUTE_SHORT_NAME` : the short name of the route, usually a line number
  - `ROUTE_LONG_NAME` : (empty)
  - `ROUTE_DESC` : *Bus*, *Zub*, *Tram*, etc.
  - `ROUTE_TYPE` :

Foreign Key = link two tables

Note: PK=Primary Key (unique), FK=Foreign Key (refers to a Primary Key in another table)

The other files are:

- *calendar-dates.txt* contains exceptions to the weekly patterns expressed in *calendar.txt*.
- *agency.txt* has the details of the operators
- *transfers.txt* contains the transfer times between stops or platforms.

Figure 1. better illustrates the above concepts relating stops, routes, trips and stop times on a real example (route *11-3-A-j19-1*, direction *0*)

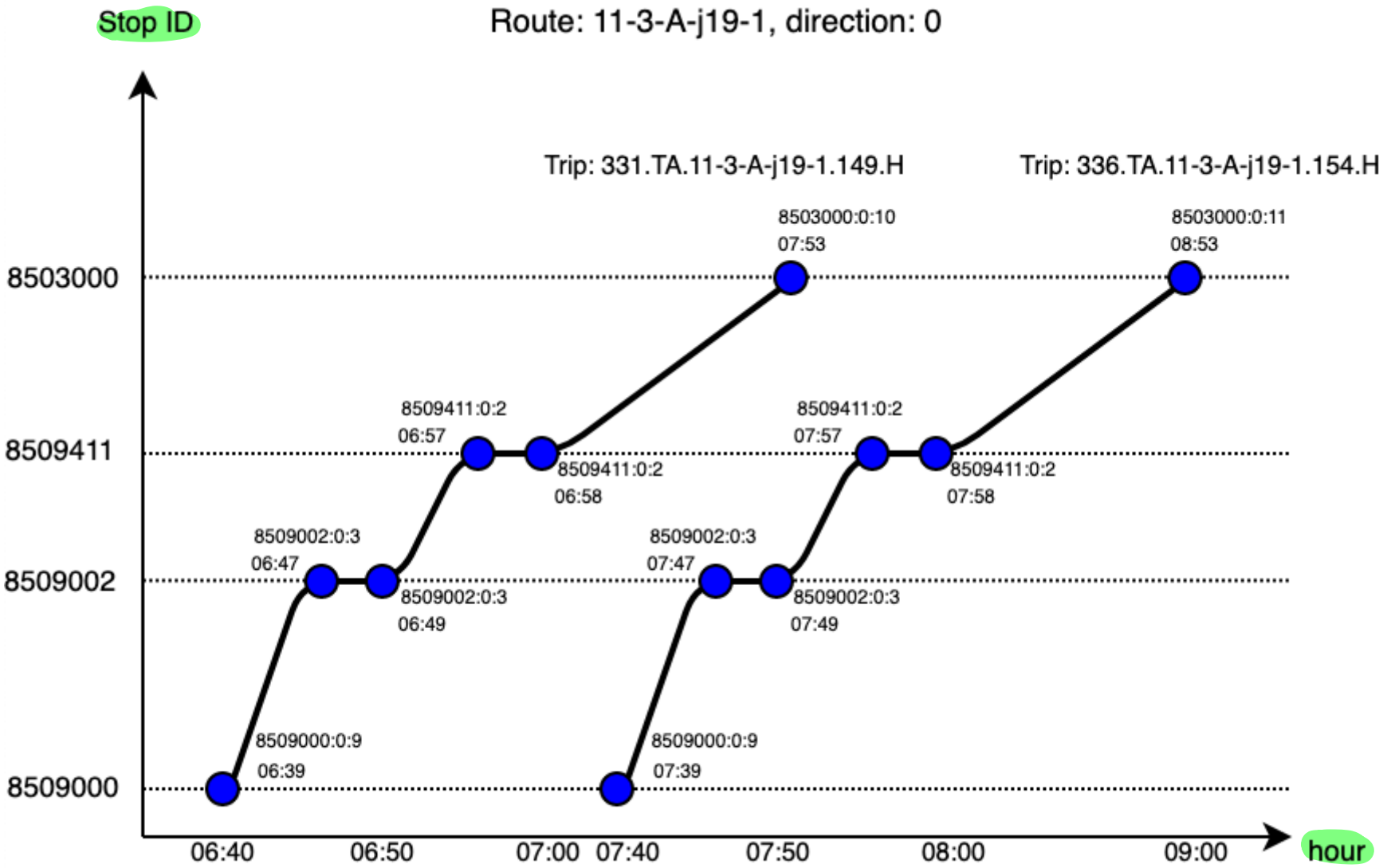


Figure 1. Relation between stops, routes, trips and stop times. The vertical axis represents the stops along the route in the direction of travel. The horizontal axis represents the time of day on a non-linear scale. Solid lines connecting the stops correspond to trips. A trip is one instances of a vehicle journey on the route. Trips on same route do not need to mark all the stops on the route, resulting in trips having different stop lists for the same route.

Stations data

For your convenience we also provide a consolidated liste of stop locations in ORC format under `/data/sbb/orc/allstops` . The schema of this table is the same as for the `stops.txt` format described earlier.

Finally, you may also find additional stops data in BFKOORD_GEO. Note however that this list has not been updated since 2017, and it is not as complete as the stops data from the GTFS timetables. It has the altitude information of the stops, which is not available from the timetable files, in case you need that.

It has the schema:

- `STATIONID` : identifier of the station/stop
- `LONGITUDE` : longitude (WGS84)
- `LATITUDE` : latitude (WGS84)
- `HEIGHT` : altitude (meters) of the stop
- `REMARK` : long name of the stop

Misc data

Althought, not required for this final, you are of course free to use any other sources of data of your choice that might find helpful.

You may for instance download regions of openstreetmap OSM, which includes a public transport layer. If the planet OSM is too large for you, you can find frequently updated exports of the Swiss OSM region.

Others had some success using weather data to predict traffic delays. If you want to give a try, web services such as wunderground, can be a good source of historical weather data.

top

Hints

Before you get started, we offer a few hints:

- Reserve some time to Google-up the state of the art before implementing. There is a substantial amount of work on this topic. Look for *time-dependent*, or *time-varying networks*, and *stochastic route planning under uncertainty*.
- You should already be acquainted with the data. However, as you learn more about the state of the art, spend time to better understand your data. Anticipate what can and cannot be done from what is available to you, and plan your design strategy accordingly. Do not hesitate to complete the proposed data sources with your own if necessary.

- Start small with a simple working solution and improve on it. In a first version, assume that all trains and buses are always sharp on time. Focus on creating a sane collaborative environment that you can use to develop and test your work in team as it evolves. Next, work-out the risk-aware solution gradually - start with a simple predictive model and improve it. In addition you can test your algorithm on selected pairs of stops before generalizing to the full public transport network under consideration.

top

References

We offer a list of useful references for those of you who want to push it further or learn more about it:

- Adi Botea, Stefano Braghin, "Contingent versus Deterministic Plans in Multi-Modal Journey Planning". ICAPS 2015: 268-272.
- Adi Botea, Evdokia Nikolova, Michele Berlingerio, "Multi-Modal Journey Planning in the Presence of Uncertainty". ICAPS 2013.
- S Gao, I Chabini, "Optimal routing policy problems in stochastic time-dependent networks", Transportation Research Part B: Methodological, 2006.

top

FAQ

This section will be updated with the Frequently Asked Questions during the course of this project. Please stay tuned.

1 - Q: Do we need to take into account walking times at the connections?

- **A**: Yes, but since we do not have the details of the platforms at each location, we can use a universal formula to come up with a reasonable walking time. We must also allow time for transfers between different modes of transports, such as from bus to tramways. You can use the transfer time information available from `transfers.txt` from the timetables. Otherwise, we assume that `2min` mininum are required for transfers within a same location (i.e. same lat,lon coordinates), to which you add *1min per 50m* walking time to connect two stops that are at most *500m* appart, on a straight line distance between their two lat,lon.

2 - Q: Can we assume statistical independence between the observed delays?

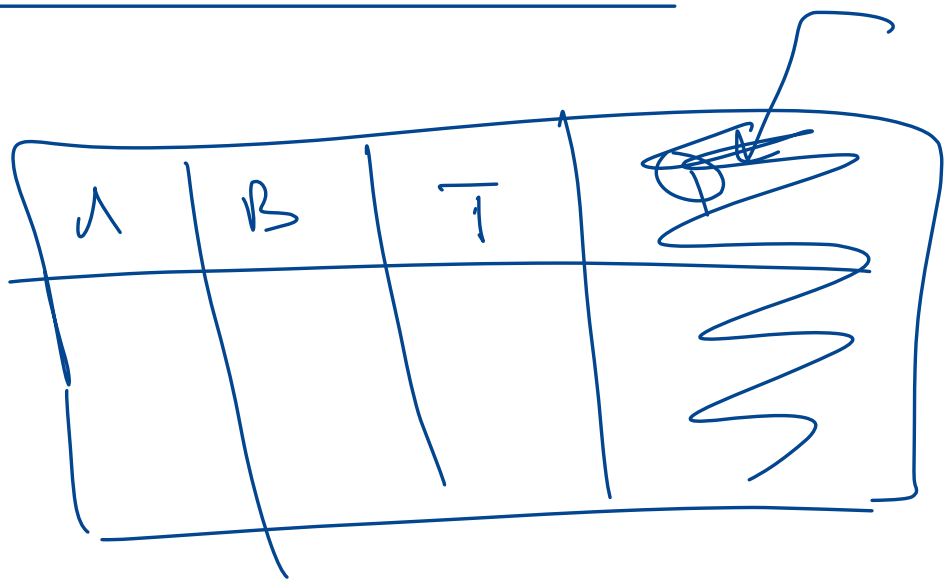
- **A**: Yes, see simplifying assumptions in **Problem Description**. You will incur no penalty for assuming that the delay of a given train (or other mode of transport, ...), at a given location and time is independent of the delays for all other trains, locations, and times. Even if our experience tells us that this is most of the time not the case. Also, you must assume that you have no real-time delays information at the time you plan your journey, which limits the benefits you could gain by assuming such a dependency.

3 - Q: Can I take advantage of the fact that a connection departs late most of the time to allow a plan that would otherwise not be possible according to the official schedule.

- **A**: You may discover that you could take advantage of connections that have a high probability of departing late. However, this is not recommended, or it should come with a warning. Imagine from a user experience perspective, how would you react if you are being proposed an impossible plan in which a transfer is scheduled to depart before you arrive? Furthermore, who would you blame if the plan fails: the planner that came up with a theoretically infeasible plan, or the operator who respected their schedule?

top

Meeting 22 Mai : delay;



① ②

② Sort by Q.

X=

if (switch - waiting - delay > 0)

↳ missing connect-

{-10; -9; - - -

50% | 0; 1; ...}

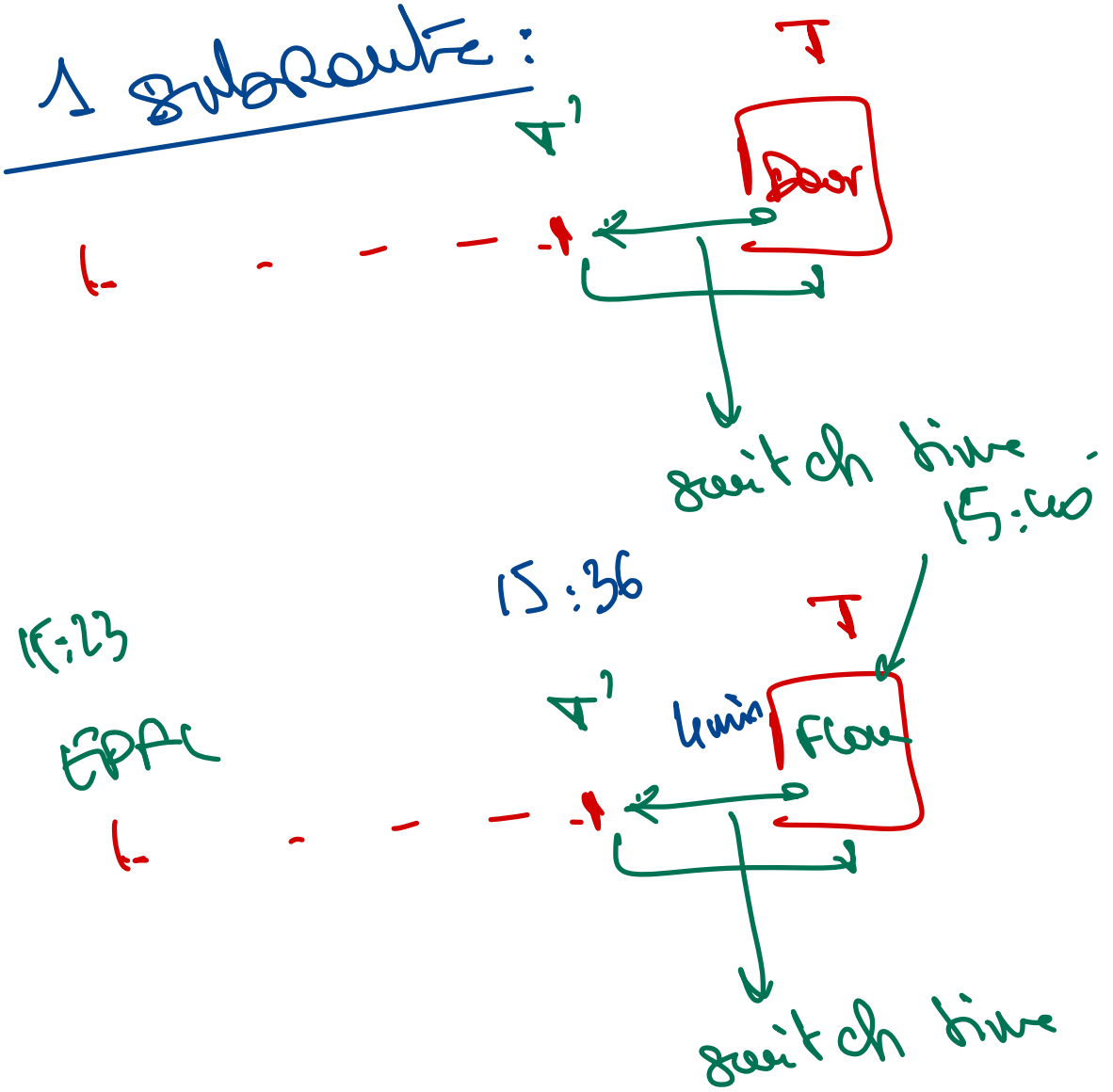
50% missing connect

Ratio = $\frac{\text{missing}}{\text{All.}}$

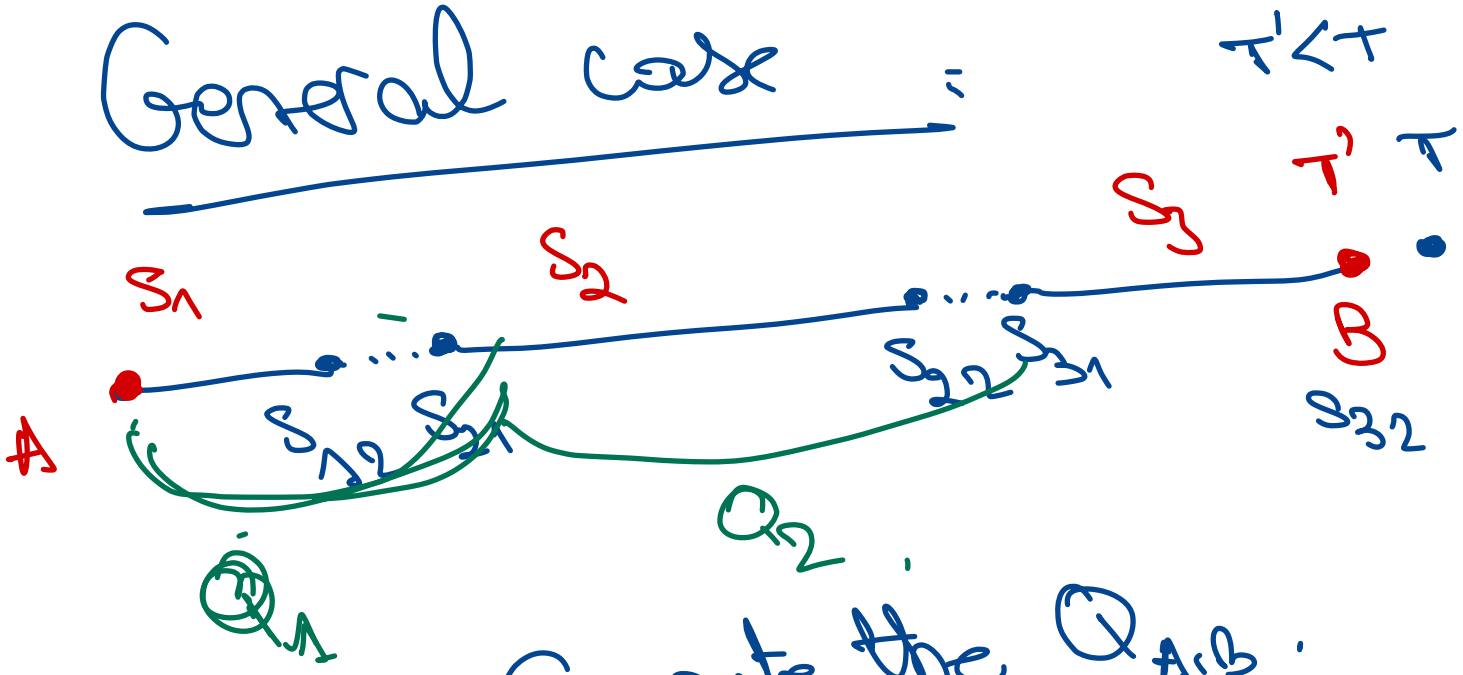
Ratio = Missing(switch Route , delay empirique (walking location)

Special case :

1 subRoute :



General case :



Obj : Compute the $Q_{A,B}$.

Route, list of delays → Formula → $\{K_i, 1 < i < 30\}$

Ratio

Real Q

EPFL → GNU

K_i = Switch - walking - expdelay

= $S_{21} - S_{12}$ - Walk - delay(i)

Q_i

stores

Multiple Qs aggregate:

Q_i = Probability of success

- Route $\{0.9, 0.9, 0.9\}$ → 0,729 → 0.9
- Route $\{0.9, 0.5, 0.6\}$ → 0,3036 → 0,67
- Route $\{0.9, 0, 0.5\}$ → 0,675 → 0
  $\{0.9, 0, 0\}$ → 0

Multiplicat°

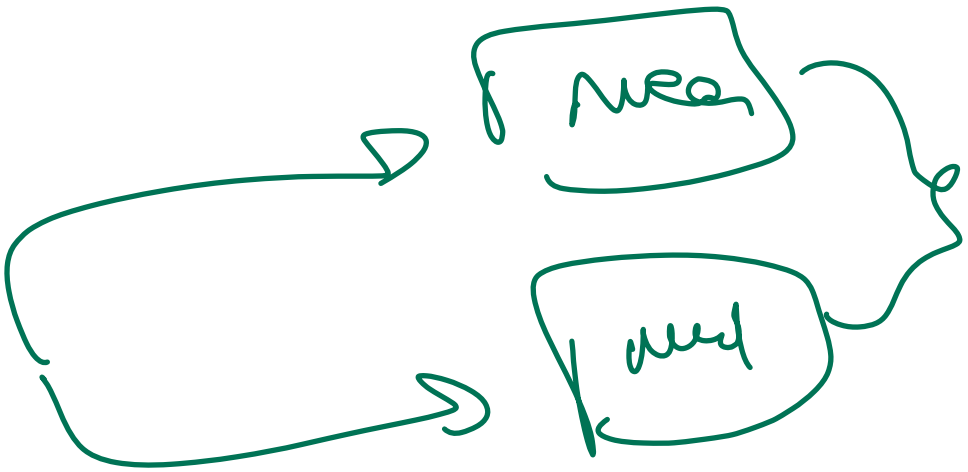
↳ Average!

Loop over the pairs

if x == 0

→ 0.

else.



trips.txt

ROUTE-ID

Lausanne

16:00

20:00

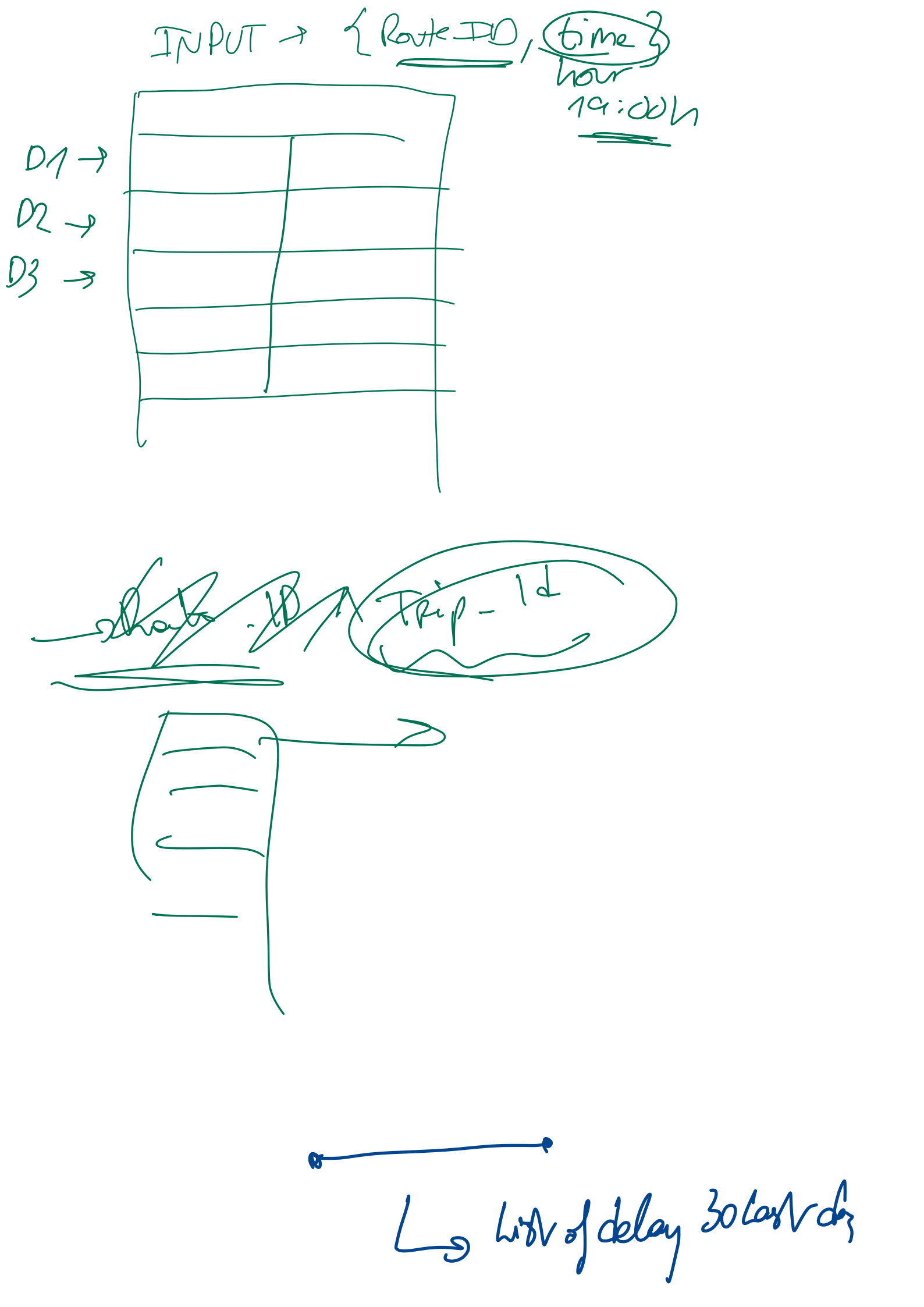
(A) ZURICH HB → LAUSANNE (UB):

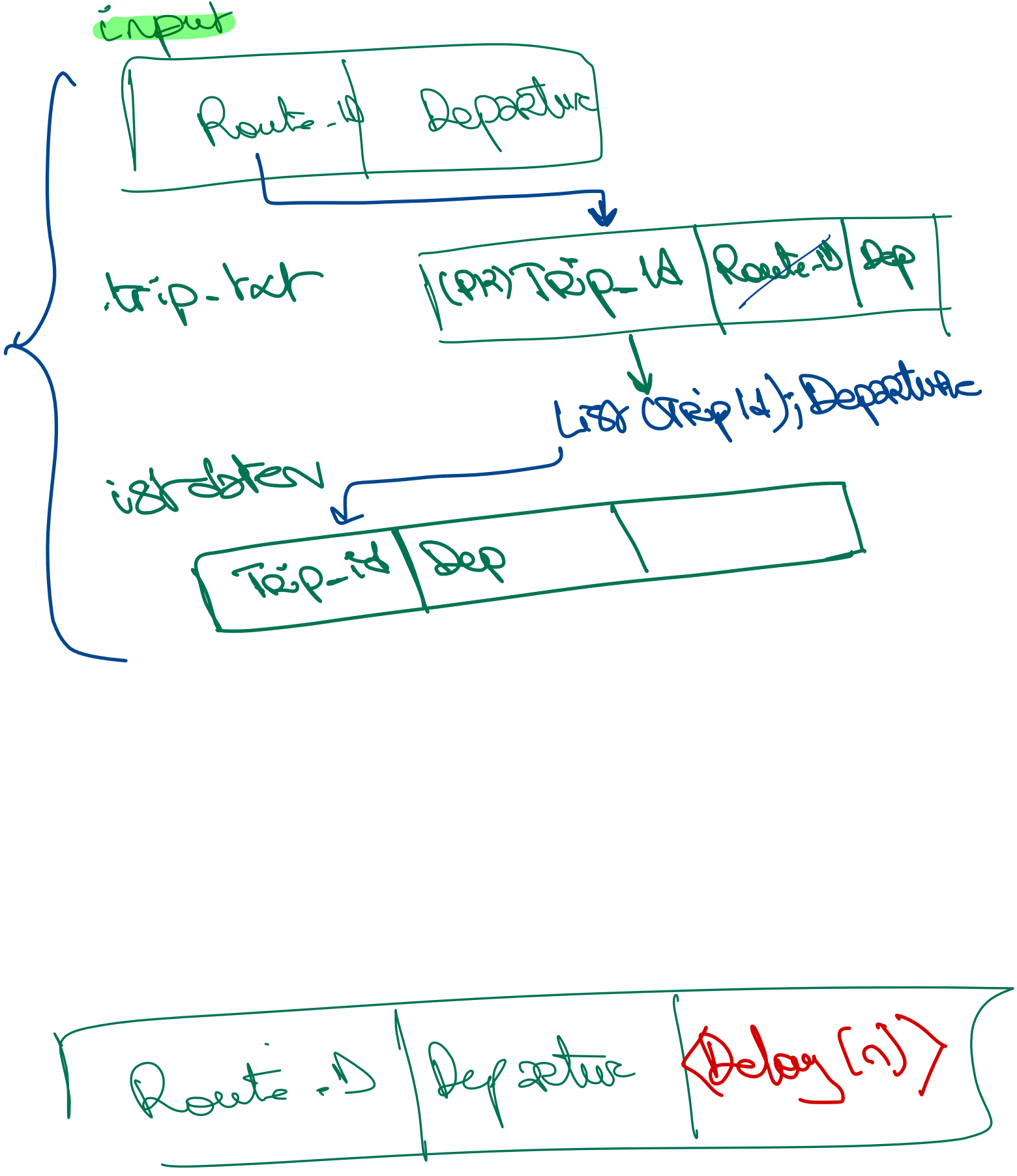
↳ 22 May Before 20:00

Fct° (nb days, Route): List Double

loop over 30 similar trips

↳ Compute the delay


INPUT → { Route ID, time }
hour
19:00h
D1 →
D2 →
D3 →
Route ID
Trip - Id
List of delay 30 last days


input
Route-ID
Departure
trip-table
(PK) Trip-ID
Route-ID
Dep
List (Trip-ID); Departure
id-table
Trip-ID
Dep
Route-ID
Departure
Delay (n)

Link(Subway) ; Link(Switch) ; Link(Walking)