Building a bus rapid transport system simulator with SimPy and GeoPandas

Ture Friese, Michael Gfeller, Henrik Holhjem, Gullik Vetvik Killie,
Christoph Imler, Patrick Merlot

bouvet

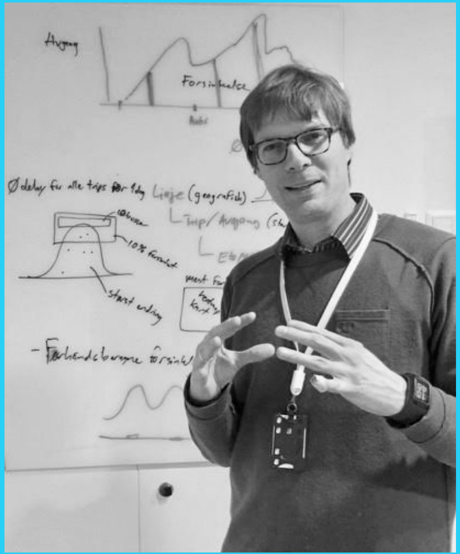
computas

About us...

Michael Gfeller
Chief Engineer

Computas Oslo

mgf@computas.com



Ture Friese
Data Scientist

Bouvet Stavanger

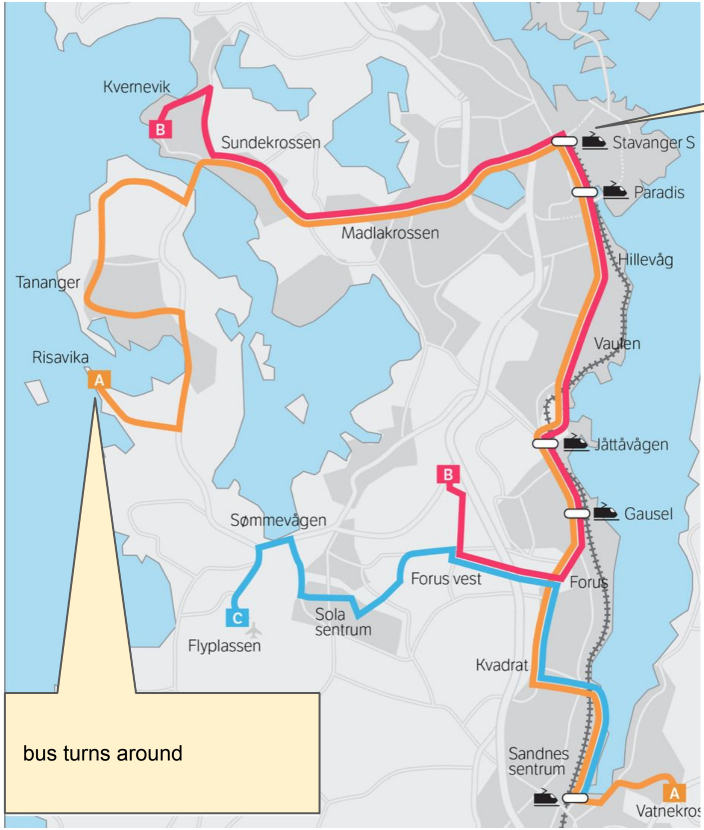
ture.friese@bouvet.no

Objective

- “Build a traffic simulator for the new BRT system”
- Client: Kolumbus AS
- Provider: Computas AS
- Time frame: mars-okt 2018
- “Don´t use microscopic traffic simulation framework e.g. SUMO”
- Our choice of technology: Python + Simpy + Geopandas

Bus rapid transport in Rogaland/Norway

- Three line:
  - A: 37 km
  - B: 25 km
  - C: 14 km
- Constructed as separate dual bus lane mid-placed between two car lanes
- Other bus lines share the same track but no other traffic (taxis, electric cars etc.)
- Currently 15% is build
- Expected date of completion: 2023
- More info:
  - http://bussveien.no
  - https://www.vegvesen.no/vegprosjekter/bussveien



Stavanger bus terminal

bus turns around

81 bus stops

104 roundabouts

bouvet

computas

Congestion prevention

Vehicle holding strategies for avoiding "bus bunching"

- First bus rules
- Simple interval regularization
- Optimal interval regularization

...other strategies

- Skipping stops
- Dwelling times

bouvet

computas

Quick demo

Find the code here:
https://github.com/Ture2019/DemoTrafficLightSimulator.git

Use of Geopandas

- We used it to calculate the distance from previous waypoints:

```
route['prev_geometry'] = route.geometry.shift(1)
route['distance'] = route.apply(lambda row: row.geometry.distance(row.prev_geometry), axis=1)
```

- Or for joining geographic data:

```
gdf = gpd.sjoin(left_df=gdf, right_df=right_df, how='left')
gdf[column] = gdf.index_right + 1
gdf.drop(columns='index_right', inplace=True)
```

- As the sentral data structure “Driving book”
  (see next slide)

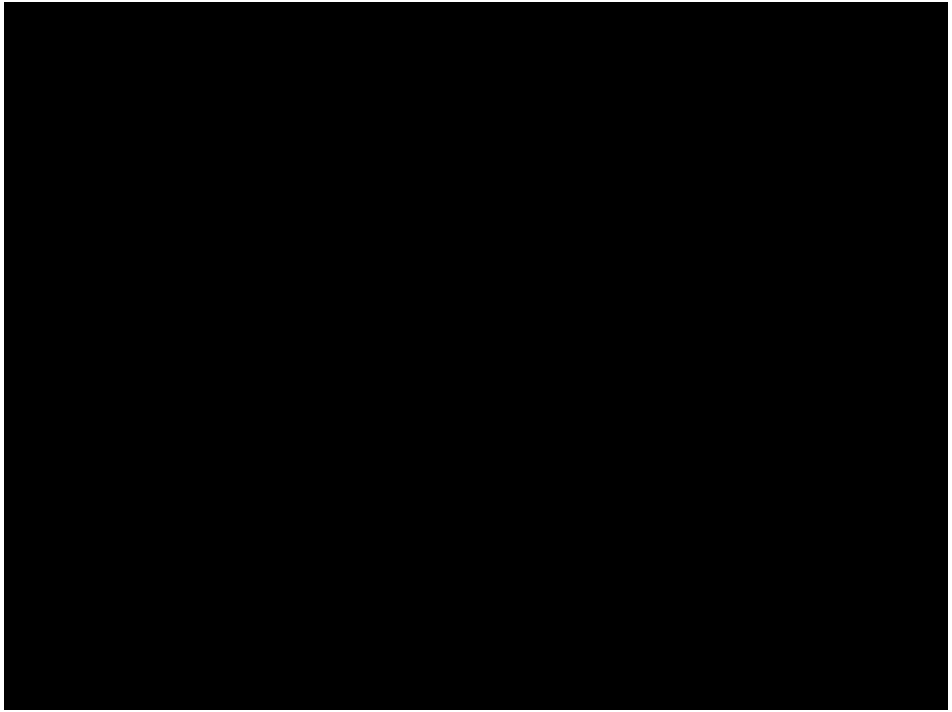
bouvet

computas

Driving book

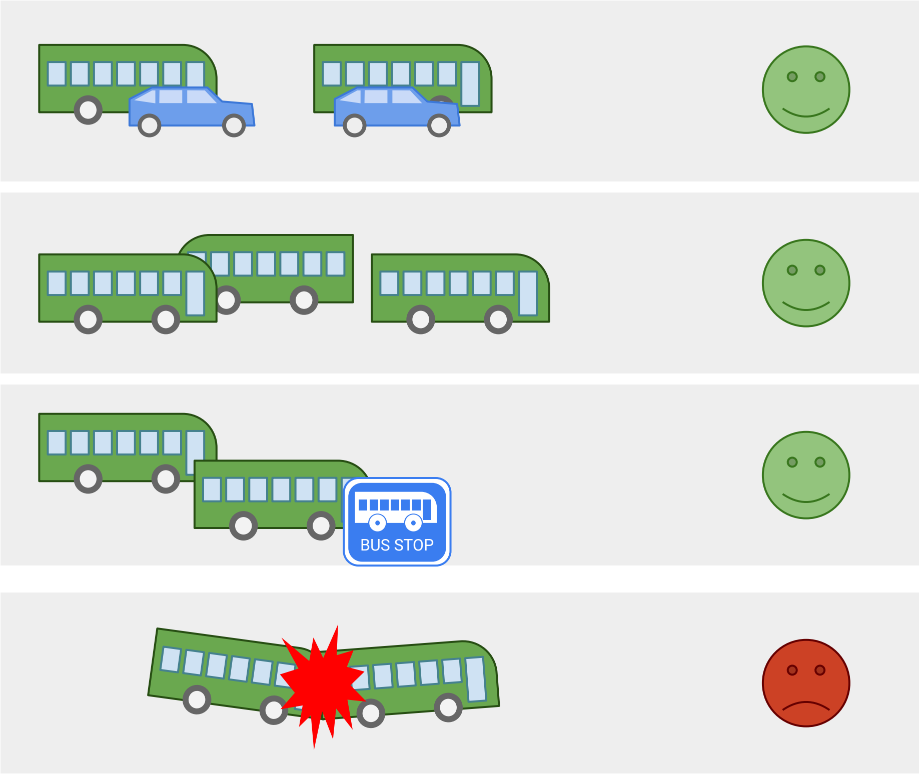
	route	A	B	C	direction	section	category	id	name	entrance	geometry	distance	azimuth	objectRef
0	A	0.0	NaN	NaN	go	a1g	stop	1.0	Risavika Havn	1.0	POINT (303172.98762263 6536164.5821019)	0.000000	322.0	stop01_RisavikaHavn
1	A	1.0	NaN	NaN	go	a1g	waypoint	0.0	waypoint_248	NaN	POINT (303417.56544243 6535834.8834327)	410.511294	323.0	NaN
2	A	2.0	NaN	NaN	go	a1g	trigger1	NaN	Kv3575 x Energivegen	1.0	POINT (303753.2779296887 6535806.445222044)	336.914835	275.0	crossing_Kv3575XEnergivegen
3	A	3.0	NaN	NaN	go	a1g	trigger2	NaN	Kv3575 x Energivegen	1.0	POINT (303857.9032163142 6535797.582412188)	105.000000	275.0	crossing_Kv3575XEnergivegen
4	A	4.0	NaN	NaN	go	a1g	trigger3	NaN	Kv3575 x Energivegen	1.0	POINT (303872.8496858321 6535796.316296494)	15.000000	275.0	crossing_Kv3575XEnergivegen
5	A	5.0	NaN	NaN	go	a1g	crossing	0.0	Kv3575 x Energivegen	1.0	POINT (303887.79615535 6535795.0501808)	15.000000	275.0	crossing_Kv3575XEnergivegen
6	A	6.0	NaN	NaN	go	a1g	trigger1	NaN	Kv3575 x Tankvegen	1.0	POINT (303994.3727563186 6535790.454311474)	106.675648	272.0	crossing_Kv3575XTankvegen
7	A	7.0	NaN	NaN	go	a1g	stop	2.0	Kontinentalvegen	1.0	POINT (304088.9769886167 6535786.374722846)	94.692153	272.0	stop02_Kontinentalvegen

Other libraries we used: matplotlib.animation

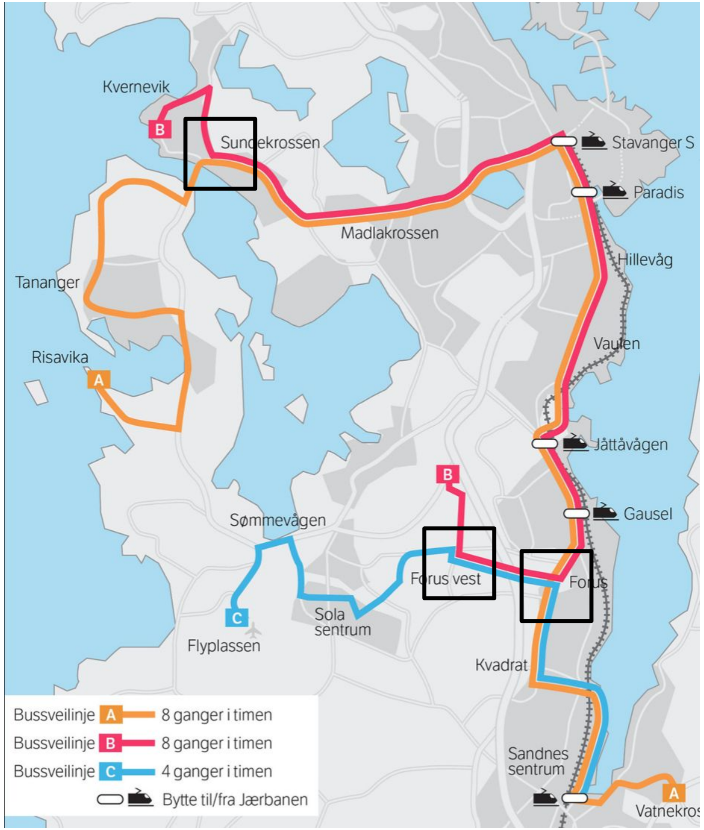


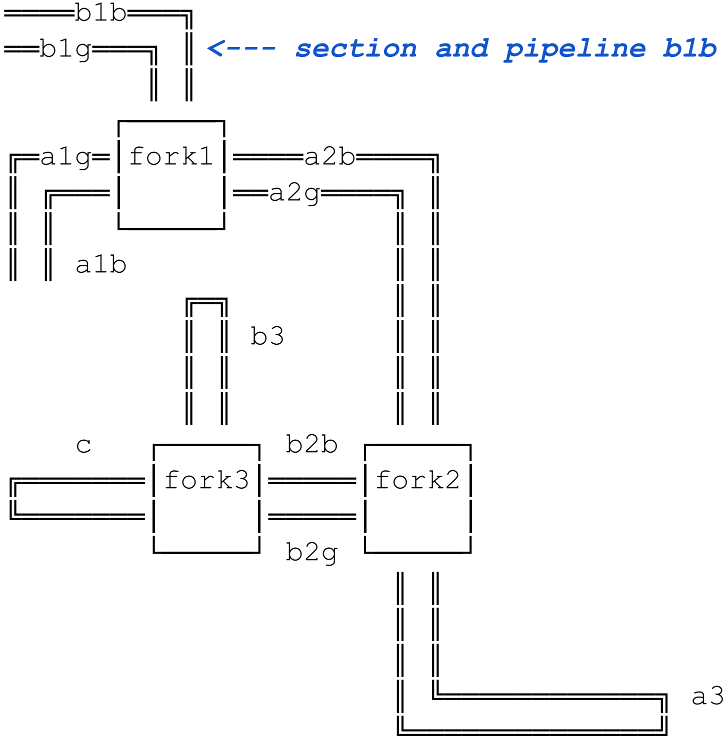
- Ref. https://matplotlib.org/3.1.0/api/animation_api.html

Collision Detection

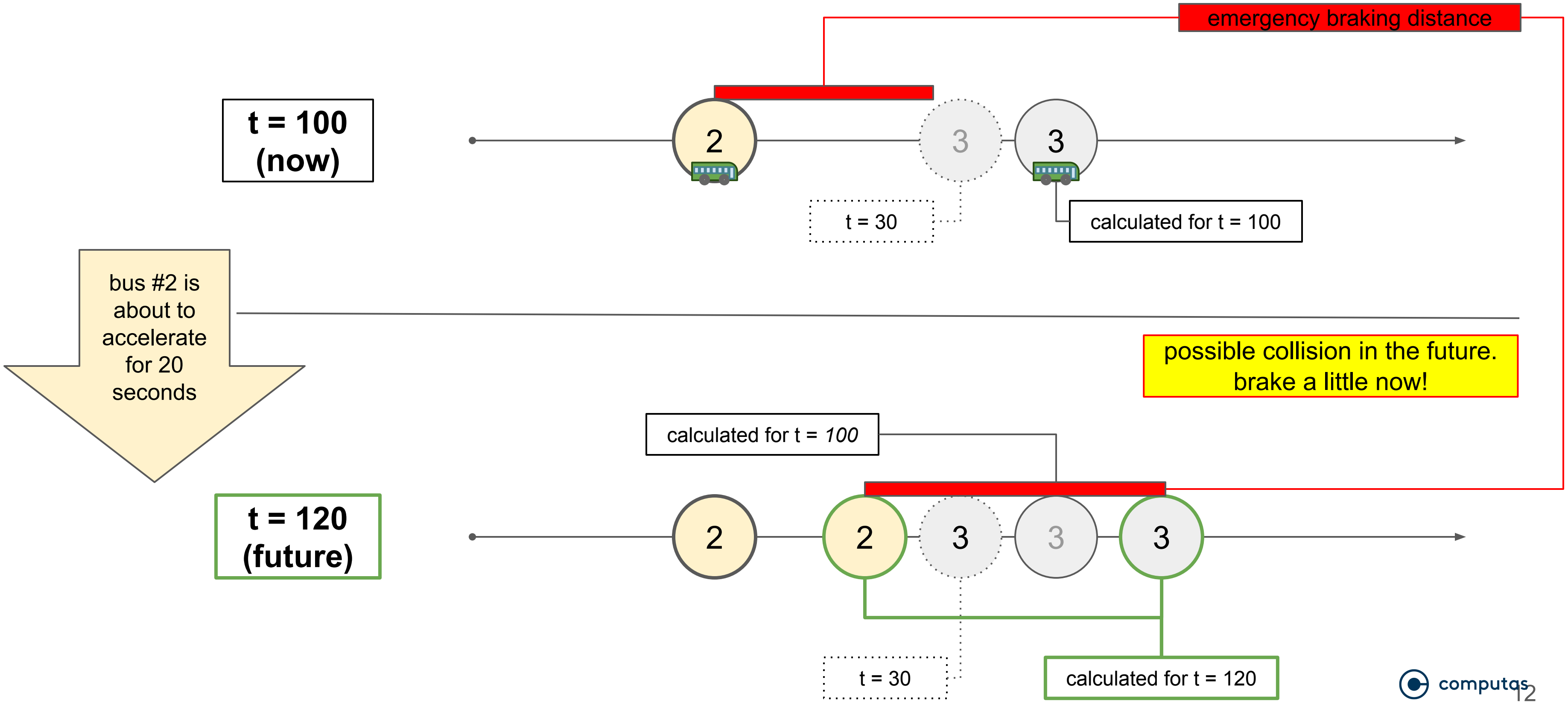


Collision Detection - Buses in Pipelines





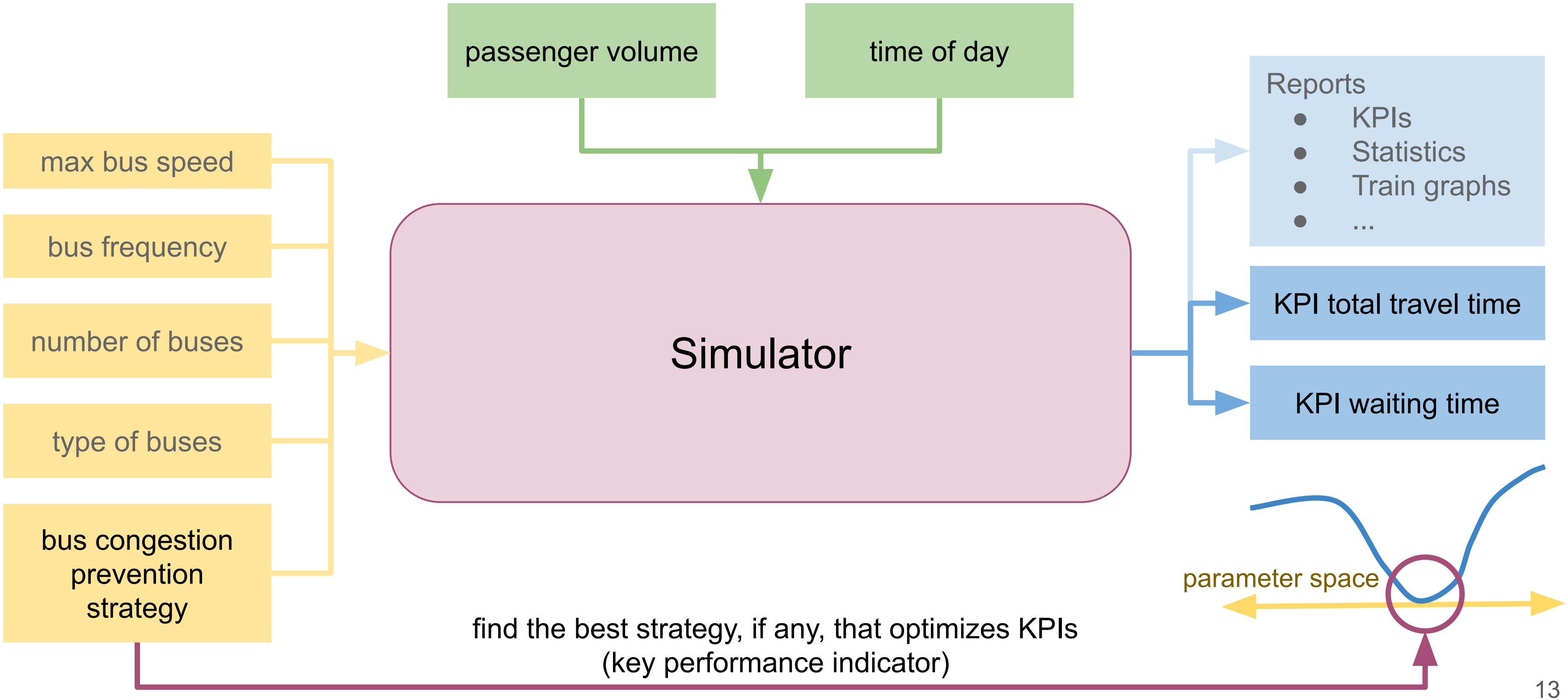
Collision Detection and Avoidance



computas

Simulator as a Decisions Support System

Find Optimal Strategy and Parametrization, Improve Understanding of the System

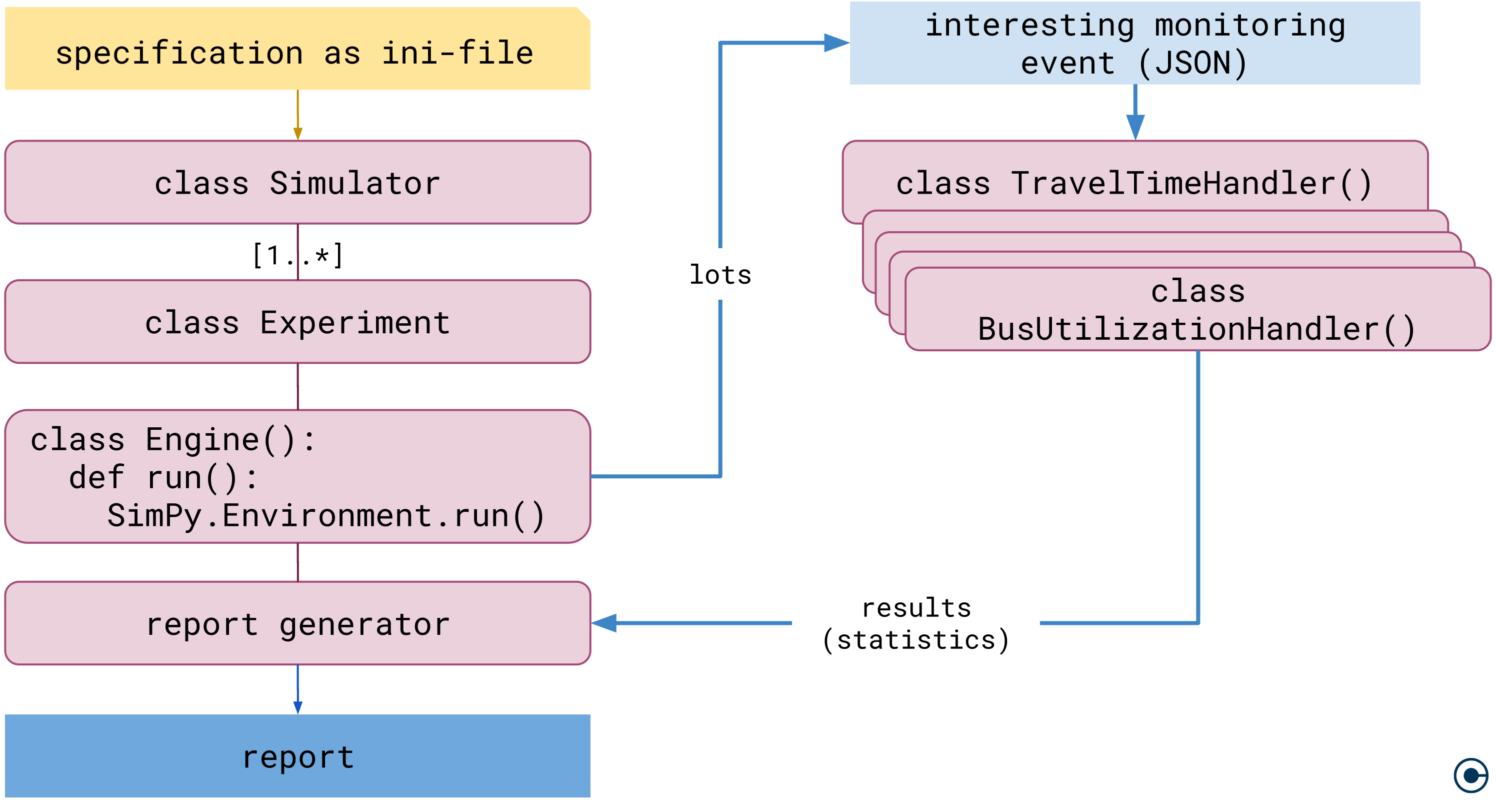


Main Simulator Application Design Requirement

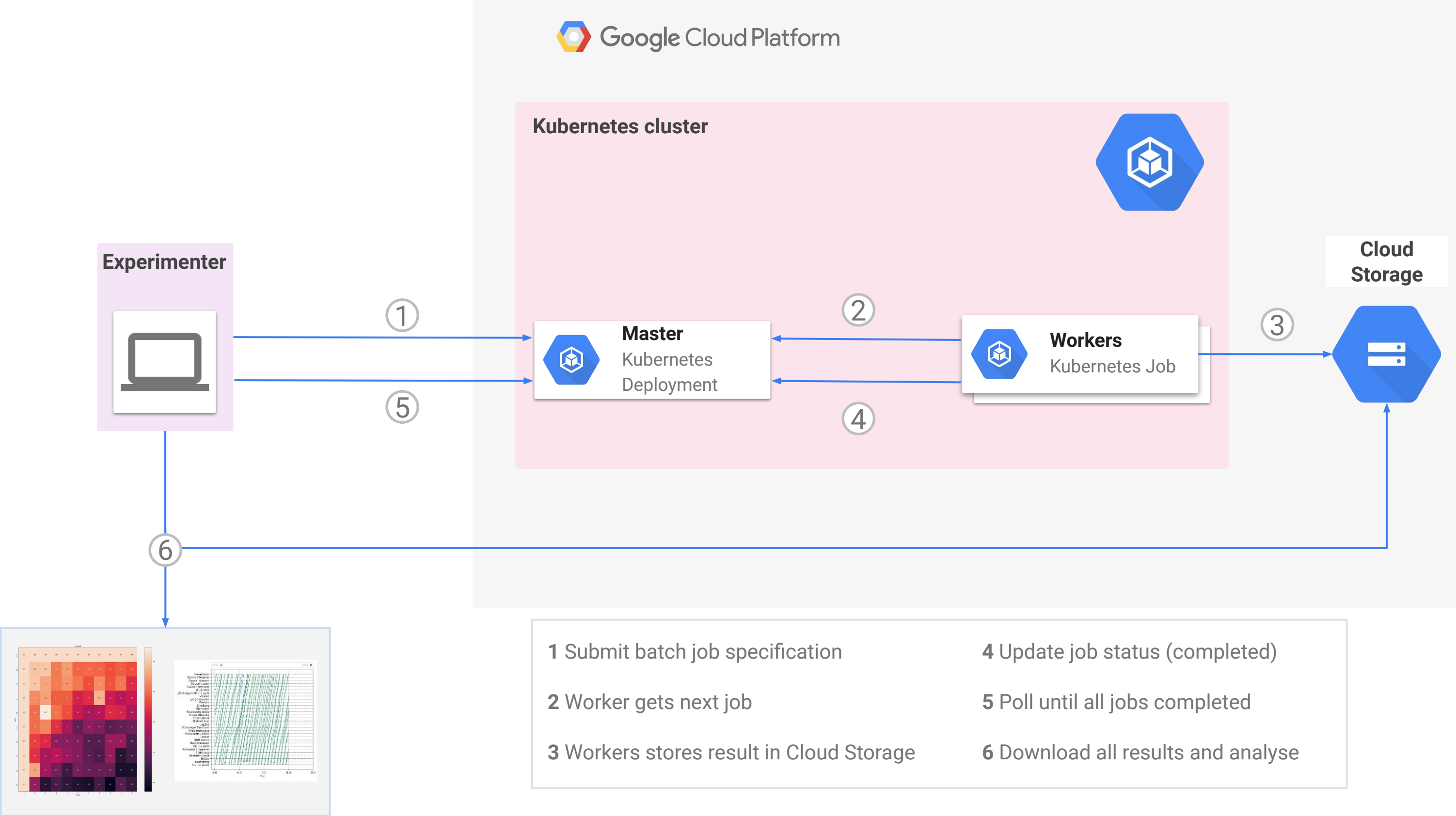
be able to generate and run
many simulations
with no user interaction required

file based input → Simulator → file based output

Application Architecture



Scaling / Parallelizing Using Kubernetes Jobs



1 Submit batch job specification

2 Worker gets next job

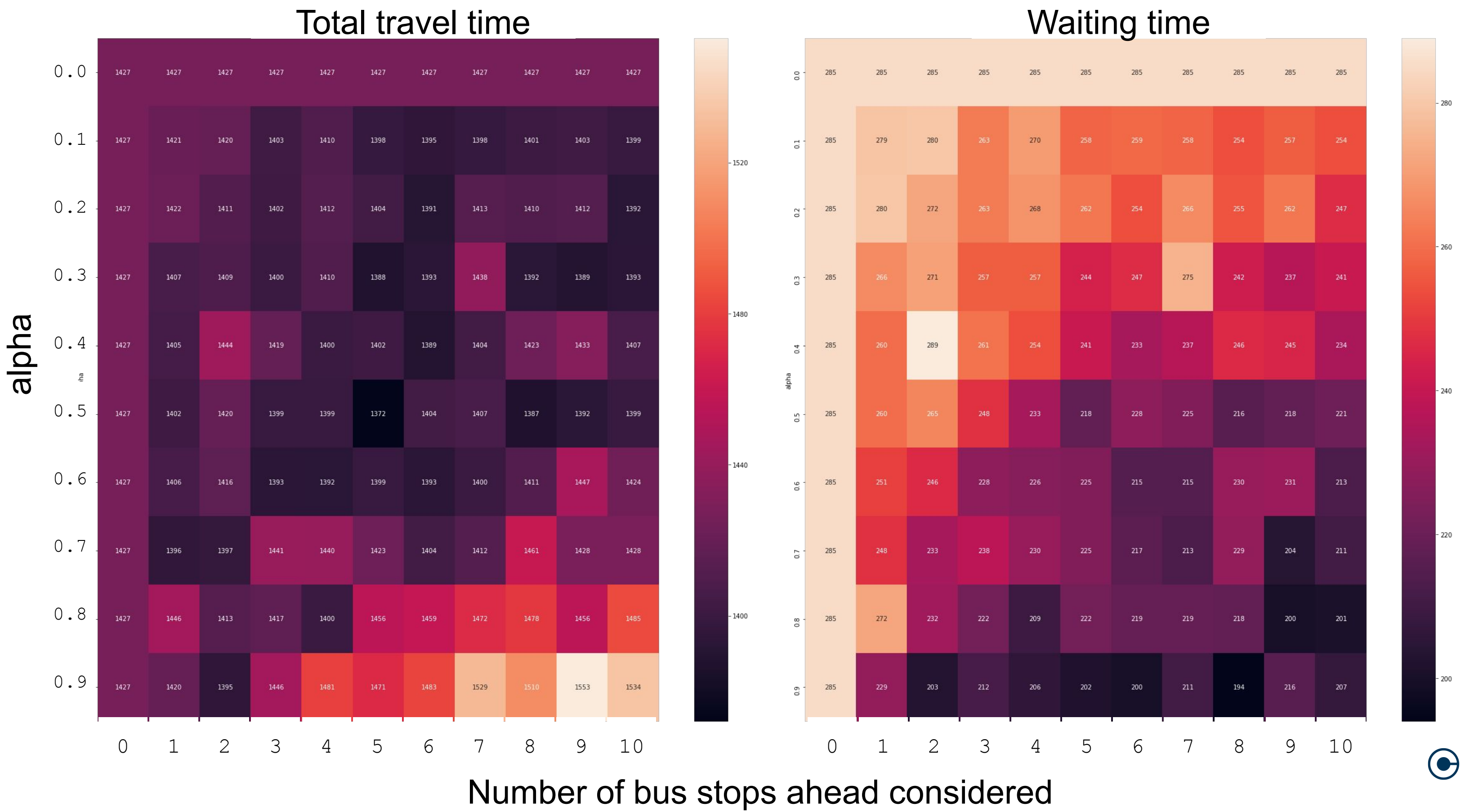
3 Workers stores result in Cloud Storage

4 Update job status (completed)

5 Poll until all jobs completed

6 Download all results and analyse

Optimizing Optimal Interval Regulation, Morning Rush 06:00-09:00



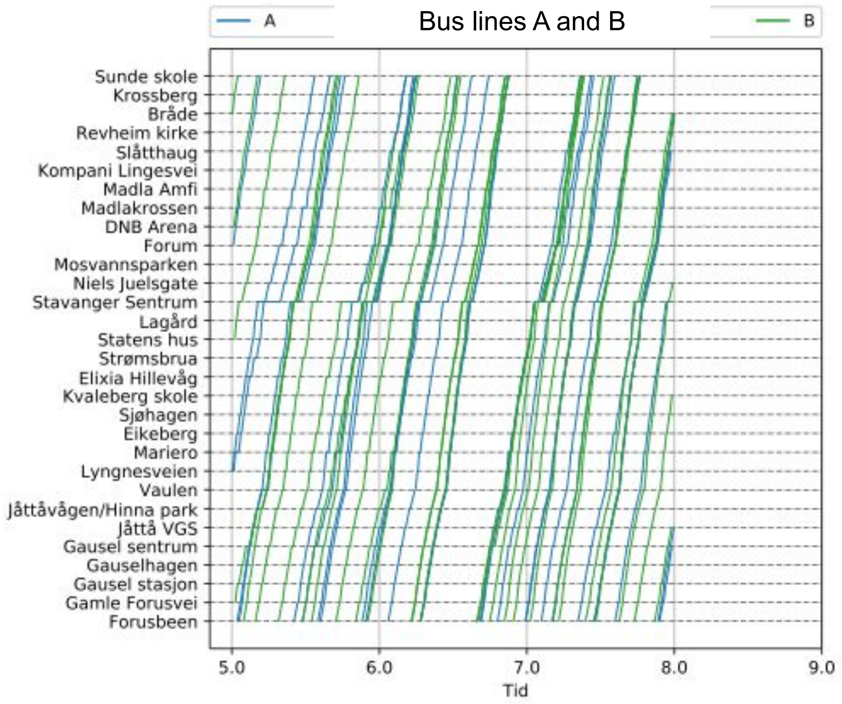
computas

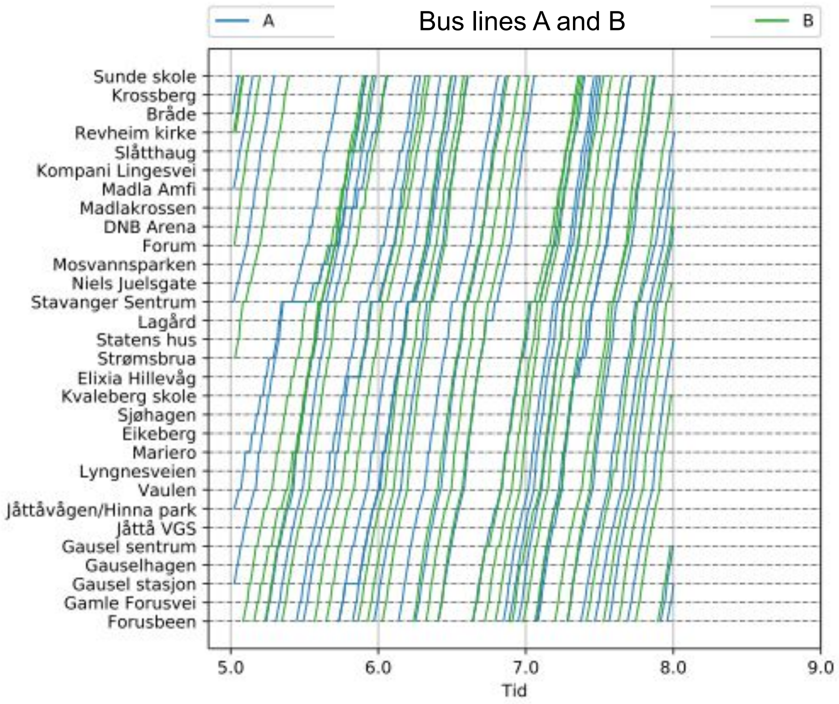
Train Graphs, Morning Rush 06:00-09:00

No interval regulation

Optimized optimal interval regulation

Bus stop





Time of day

computas

Review

- Technical limitations
  - performance
- Review design decisions
  - Discrete-Event-Simulation (DES) pattern
- Which strategy is best?
- Is the simulator suitable as a planning tool?
  - practical values of simulators and models (abstractions / simplifications of reality)
- Can it be used as a tool for real time decision support?