25.10.2018

Traffic Simulations at Havnegata, Sandnes

DAT530 Project

Henrik Mjaaland
UNIVERSITY OF STAVANGER

Abstract

The intention of this paper is to simulate traffic in Sandnes as it is now, and to simulate the traffic after Havnegata is eventually opened for traffic during rush hours (it is currently not allowed to drive through it unless you stop) using GPensim (developed by Reggie Davidrajuh) in MATLAB in order to see how much traffic congestion in Sandnes is reduced by opening Havnegata, and thus if it is actually worth opening.

Contents

1.0 Introduction

1.1 Problem Definition

Today there are several people who drive illegaly through Havnegata to avoid traffic. The reason it's illegal to drive through Havnegata is because there are multiple apartment complexes and a lot of people living by the road there, and potential traffic congestion in Havnegata will obviously disturb the people living there.

However, if Havnegata is opened for traffic, the traffic congestion in general in Sandnes should be reduced during rush hours, because then the queue coming from Gravarsveien will split up earlier at Havnegata.

For the above reasons, the district manager of Hana, Ellen Karin Moen and a majority of Hana borough council has suggested the compromise of opening Havnegata only during rush hours[1]. Opening Havnegata would benefit a lot of people from Hana because most of the commuters drive past Havnegata.

Personally, I think Havnegata should be opened during rush hours, as long as the traffic jam is reduced significantly (simulating how much it will reduce is the scope of this project) because most of the people who live in Havnegata are commuting themselves during rush hours.

The problem that this paper addresses is whether or not Havnegata should be opened, by simulating the traffic before and after Havnegata is opened during rush hours.

1.2 Adaption

My code has been influenced by Roberto Martin Mûnoz and Tina Kristian Jensen's work, 'Simulation of Traffic at E39 Ålgård – Forus' from 2013[2]. I got the idea of using semaphores for the roundabouts from their paper.

2.0 Method and Design

2.1 Overall Design

2.1.1 System Models

I will focus my simulations on the morning rush, hence the directions of the queues in the below figures. The red lines represent the origins of the traffic, and the other lines represent the traffic jam after it splits up (the green one, goes towards Ganddal, the yellow one leads to E39, and the blue one leads towards Lura).

The below image displays the directions of today's traffic jam in Sandnes during the morning rush.

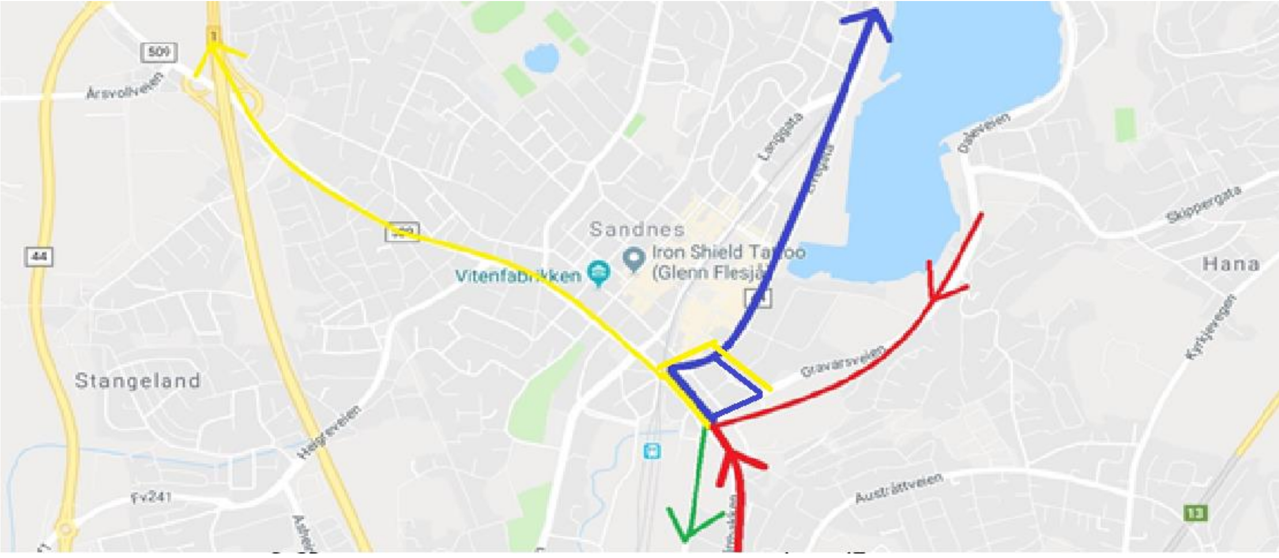

Figure 1: System model 1

Havnegata is the northernmost road along Gravarsveien where the blue and yellow lines are parallell to each other in the figure below. If Havnegata is opened, the commuters represented by the yellow line can choose to either drive through Havnegata along with the commuters represented by the blue line and re-join the other yellow commuters later on (this will cause a bottleneck in the junction where the blue line separates from the yellow line in the first figure) or follow the path as in the first image.

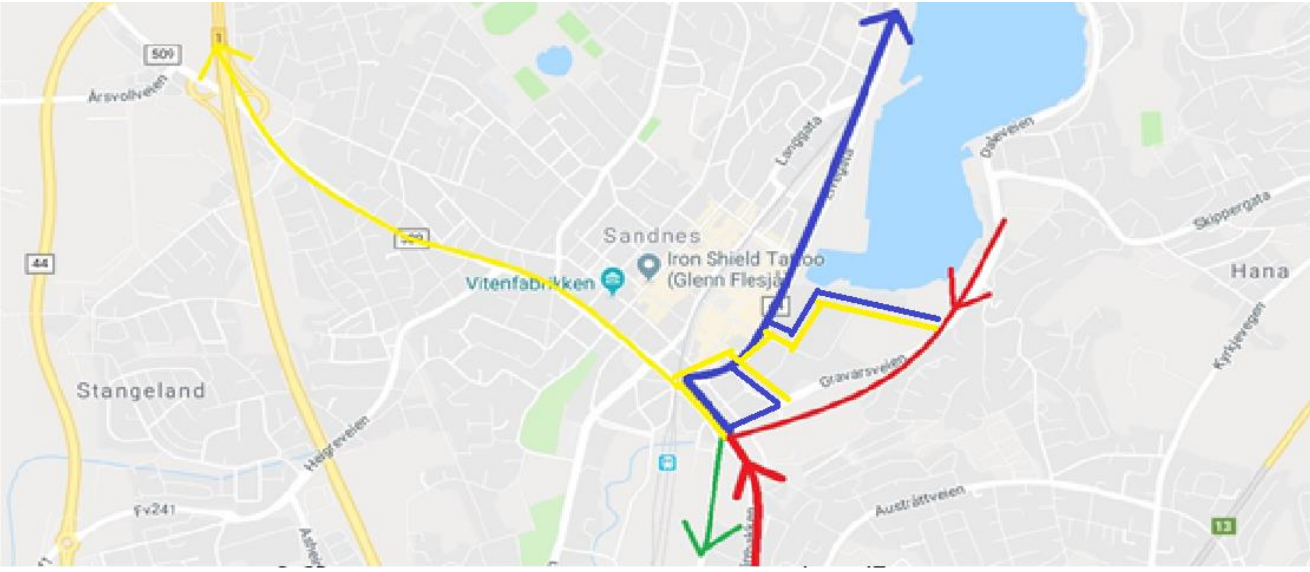

Figure 2: System Model 2

In order to find out which model is the most efficient, I compared the queues represented by the green, yellow and blue lines in the first figure (exit places), with the ones in the second figure, by creating two petri net models based on the two figures above and comparing them with each other in MATLAB using GPensim.

2.1.2 Petri Net Systems

The petri net system for the first model (the traffic jam during rush hours in Sandnes as it is today where Havnegata has not yet been opened for traffic) is in figure 3 below. The traffic starts in the places p1_Start and p2_Start. P3_Center, p4_Center, p5_Center and p6_Center are roundabouts. P7_Out, p8_Out and p9_Out are the queues exiting Sandnes.

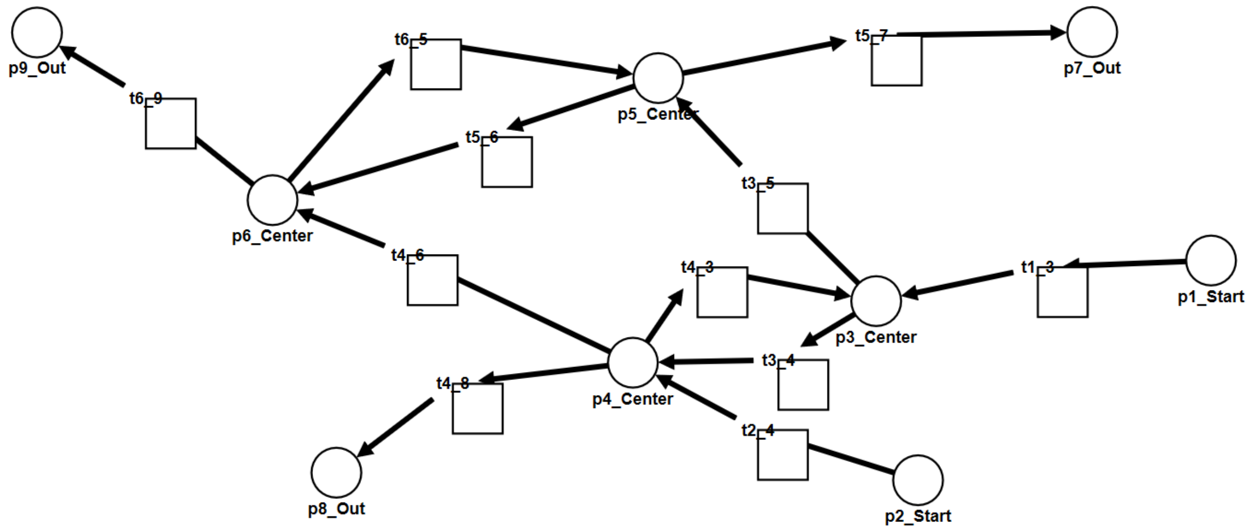

Figure 3: Petri Net Model 1

Below is the petri net system displaying the traffic congestion in Sandnes after Havnegata has been opened for traffic during rush hours. Now there is two new places/roundabouts, p3_Center and p6_Center. P3_Center now splits in two directions, one to the west continuing through Gravarsveien and a new one going north through Havnegata. Havnegata corresponds to transition t3_6.

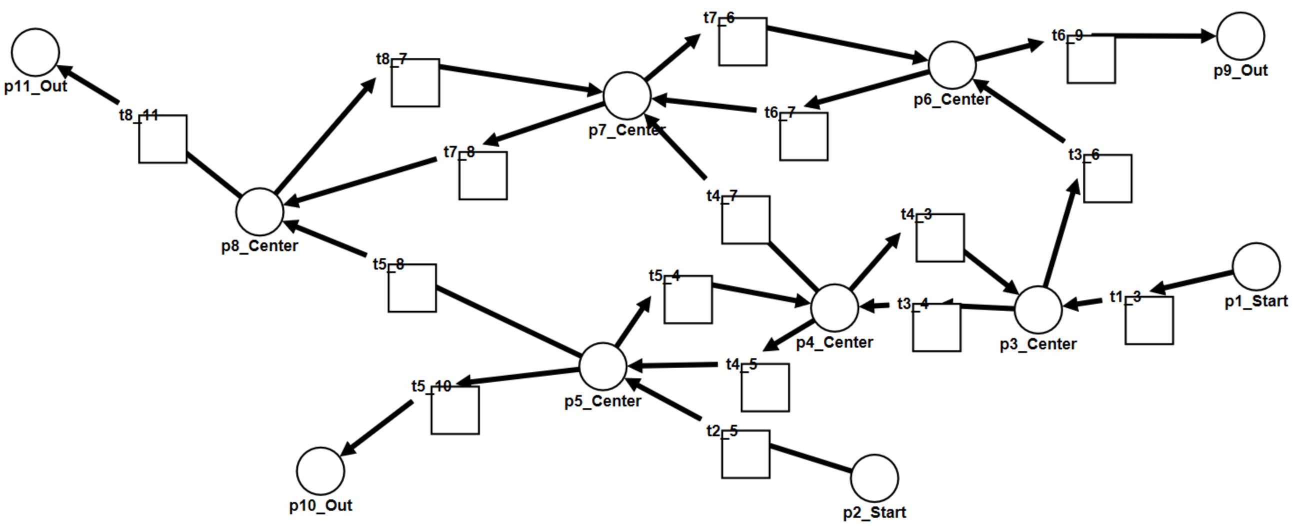

Figure 4: Petri Net Model 2

I created the above petri nets using APO[4], which is an online petri net editor.

2.2 Modular Approach

All the transitions in my petri nets are the same (roundabouts). Therefore, it is not necessary to split the petri nets into multiple modules. I simulated each transition (roundabout) using semaphores. This will be explained with a code snippet in chapter 3.4.

2.3 Techniques

I used ordinary timed petri nets for this project. I will get more into detail when it comes to gathering data for computing firing times in chapter 3.1.

2.4 Design Alternatives

Ordinary timed petri nets generate lower running times than high-level petri nets as coloured petri nets and hierarchical petri nets. For example, in coloured petri nets, all the transitions should have an individual pre-file which increases running time. If the transitions don't have individual pre-files, the COMMON_PRE file, the file will be flooded, and not very practical to read.

3.0 Implementation

3.1 Firing Times

I gathered data of firing times by driving five times through each of the transitions using a stop-watch. I got the following data for each of the transitions (System 1):

Transitions	*Finishing Times*
T1_3	65.19, 78.61, 78.98, 70.95, 66.67
T2_4	13.06, 12.26, 13.53, 11.45, 12
T3_4	15.39, 16.72, 16.25, 17.87, 15.53
T4_3	16.12, 18.07, 18.35, 18.03, 16.35
T4_8	8.32, 9.09, 8.99, 8.9, 8.14
T3_5	16.53, 14.68, 15.29, 15.35, 16.09
T4_6	21.76, 20.18, 20.98, 22.58, 20.21
T5_7	9.62, 9.21, 8.45, 8.92, 9.85
T5_6	14.11, 16.09, 16.83, 16.07, 14.45
T6_5	14.28, 14.61, 14.08, 14.02, 14.91
T6_9	8.14, 8.79, 8.34, 8.55, 9.29

Table 1: Transition Times for Petri Net System 1

Means and standard deviations for the above data:

Transitions	*Means and Standard Deviations*
T1_3	$\mu = 72.08$, $\sigma = 5.8$
T2_4	$\mu = 12.46$, $\sigma = 0.75$
T3_4	$\mu = 16.35$, $\sigma = 0.9$

T4_3	μ = 17.38, σ = 0.95
T4_8	μ = 8.68, σ = 0.38
T3_5	μ = 15.59, σ = 0.65
T4_6	μ = 21.14, σ = 0.92
T5_7	μ = 9.21, σ = 0.5
T5_6	μ = 15.51, σ = 1.05
T6_5	μ = 14.38, σ = 0.34
T6_9	μ = 8.62, σ = 0.4

Table 2: Means and Standard Deviations for Petri Net System 1

Data for System 2:

Transitions	*Finishing Times*
T1_3	20.11, 21.56, 21.06, 24.5, 22.99
T3_4	45.42, 54.32, 45.9, 47.38, 46.29
T4_5	16.21, 16.92, 17.29, 15.29, 16.87
T5_10	9.02, 9.66, 8.99, 8.38, 8.45
T4_7	14.64, 14.37, 14.7, 15.79, 14.02
T5_8	17.31, 18.21, 19.28, 17.52, 19.82
T6_7	12.4, 12.3, 11.69, 10.01, 11.18
T7_6	10.63, 12.73, 13.19, 10.75, 12.72
T7_8	16.77, 15.21, 15.71, 17.17, 16.56
T8_7	15.11, 16.76, 17.52, 14.67, 17.67
T6_9	9.36, 9.89, 9.14, 9, 8.91
T8_11	9.14, 9.42, 8.3, 8.98, 8.77
T3_6	65.84, 65.75, 65.47, 61.59, 60.48
T4_3	46.29, 51.38, 53.76, 48.97, 45.41
T5_4	13.78, 15.13, 14.13, 14.49, 14.86
T2_5	13.63, 13.33, 12.03, 11.56, 12.95

Table 3: Transition Times for Petri Net System 2

Means and standard deviations for the above data:

Transitions	*Means and Standard Deviations*
T1_3	μ = 22, σ = 1.54
T3_4	μ = 47,

	$\sigma = 3.3$
T4_5	$\mu = 16.5$, $\sigma = 0.7$
T5_10	$\mu = 8.9$, $\sigma = 0.46$
T4_7	$\mu = 14.7$, $\sigma = 0.6$
T5_8	$\mu = 18.4$, $\sigma = 0.98$
T6_7	$\mu = 11.5$, $\sigma = 0.9$
T7_6	$\mu = 12$, $\sigma = 1$
T7_8	$\mu = 16$, $\sigma = 0.72$
T8_7	$\mu = 16.3$, $\sigma = 1.2$
T6_9	$\mu = 9.3$, $\sigma = 0.3$
T8_11	$\mu = 8.9$, $\sigma = 0.4$
T3_6	$\mu = 63.8$, $\sigma = 2.3$
T4_3	$\mu = 49.2$, $\sigma = 3.1$
T5_4	$\mu = 14.5$, $\sigma = 0.49$
T2_5	$\mu = 12.7$, $\sigma = 0.78$

Table 4: Means and standard deviations for Petri Net System 2

3.2 Stochasticity

The destinations have different routes depending on the driver's gut feeling or the level of visible queue. Therefore, I assumed that the routes are chosen randomly in my implementation. The firing times will also vary depending on a number of factors, which is why I used normrnd(μ,σ) to represent firing times. Normrnd(μ,σ) is a function in MATLAB that generates a random number from the normal distribution, where μ is the mean and σ is the standard deviation.

```
dyn.ft = {'t1_3',normrnd(0.72,0.058), 't2_4',
normrnd(0.12,0.0075), 't3_4', normrnd(0.16,0.009),
't4_3',...
    normrnd(0.17,0.0095), 't4_8',
normrnd(0.087,0.0038), 't3_5', normrnd(0.16,0.0065),
't4_6', ...
    normrnd(0.21,0.0092), 't5_7',
normrnd(0.09,0.0058),'t5_6', normrnd(0.21,0.009),
't6_5', ...
    normrnd(0.14,0.0034), 't6_9',
normrnd(0.09,0.004)};
```

3.3 Priorities

I contacted Vegvesenet [3] *and they told me to use a roadmap on their web page[5]. From the map I found that the average number of cars per day for 2017 was 13350 for p1_Start and 8500 for p2_Start. Since most of the cars driving on p1_Start and p2_Start are driving between 07:00-08:00 and between 15:00-16:00 and I am simulating traffic between 07:00-08:00, I estimated the number of cars for p1_Start and p2_Start to be 13350/2 and 8500/2 in their respective order.*

I also went to Sandnes during the morning rush and manually counted cars going to the three different exits for 30 minutes by p5_Center and 30 minutes by p4_Center (using petri net system 1 as reference). Thus, I found out that the line with the most cars is the one leading to E39 (p9_Out), the one with the least cars is the one going west (p8_Out), and the line leading north (p7_Out) is in the middle. To include this in my simulations, I set the priorities for the routes leading to E39 to 2, the routes leading to the north to 1, and the routes leading to the west to 0 (default ip):

```
dyn.ip = {'t3_4', 2, 't4_3', 1, 't4_5', 2, 't5_4',
1, 't3_6', 2, 't4_7', 2, ...
    't5_8', 2, 't6_9', 1, 't6_7', 2, 't7_6', 1,
't7_8', 2, 't8_7', 1, 't8_11', 2};
```

3.4 Semaphores

I assumed that cars driving from roads leading to junctions are alternating on driving through a given roundabout, and therefore I used semaphores to represent the roundabouts[2]. Usually, the cars coming from the left has priority, so in that case one should set semaphores as cars from the left enter roundabouts and reset when they leave them. But, in real-life during rush hours, cars will alternate on driving through roundabouts (otherwise a whole queue of cars will have to wait for the queue coming from the left). Therefore, I set/reset semaphores for cars coming from both right and left. Also, since the cars are alternating and the right cars don't need to wait for cars coming from the left, there is no reason to account for the number of cars in the roundabout/waiting time.

In the pre-processor I set semaphores leading to a given place to 0 (occupied) when a transition leading to the given place is fired:

```
if (strcmp(transition.name,
't4_3')||strcmp(transition.name, 't1_3'))
    aux=(eq(p3C.tokens,0) && eq(global_info.se, 1));
    if aux
        global_info.se=0;
        fire=true;
    else
        fire=false;
    end
```

In the post-processor I reset the semaphores leading to a given place (set them back to 1) when the fired transition finishes:

```
if (strcmp(transition.name, 't4_3') ||
strcmp(transition.name, 't1_3'))
    global_info.se=1;
```

In the main (msf) file, I initially set all the semaphores to 1.

4.0 Testing, Analysis and Results

I plotted the results from the simulations in the figure 5 and 6 below. As you can see from the figures, the queue leading to p9Out and p11Out is worse after Havnegata is opened, while the queue leading to p10Out is better (using petri net system 2 as reference).

The queue leading to E39 (p11Out) is worse, because now the cars driving to p11Out will split up at Havnegata (some "cheaters" who are headed for p11Out might take the route through Havnegata if it looks like it's less queue there or just on a hunch) only to re-join later at p7_Center or p_8Center (see figure 4), thus causing bottlenecks at these two places.

There is also more queue towards p9Out now because of the "cheaters", even though there is a shorter distance between p1_Start and p9Out after Havnegata is opened.

The queue leading to p10Out (south towards Ganddal) is going faster now because as said some drivers who are headed from p1Start to p11Out might "cheat" and turn off the route leading to p10Out earlier.

Another factor that are causing bigger queues in the second model is because there are two new roundabouts, namely p3_Center and p6_Center.

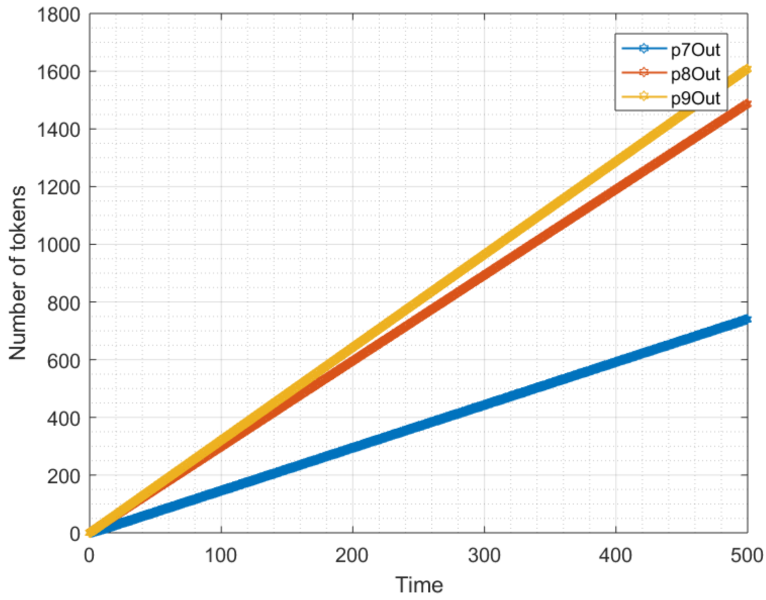

Figure 5: Plot of Simulation Results (System 1)

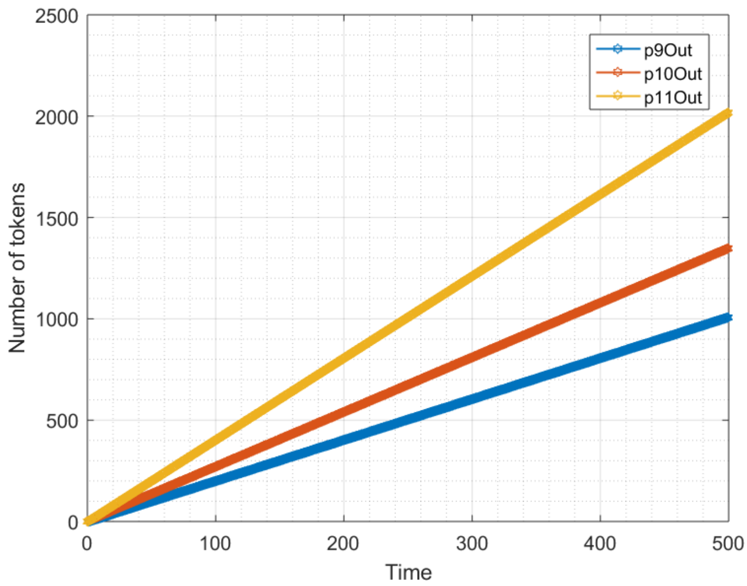

Figure 6: Plot of Simulation Results (System 2)

5 Discussion

5.1 Originality of This Work

The problem that this paper discusses is an actual ongoing problem in real life (Wants to open Havnegata for traffic during rush hours, u.d.), and there is nobody who has tried to solve this problem using petri net simulations that I know of.

5.2 Limitations of my Work and Further Work

As mentioned, I only used timed ordinary petri nets for my simulations. For more accurate results, one can implement more advanced types of petri nets like for example coloured petri nets, even though it will increase the running time. If one uses coloured petri nets, cars can for example be marked by different colours depending on which particular route they are driving along.

5.3 Learning Experience

I was struggling with finding a way to make the routes for the different cars. The best way is perhaps to use coloured petri nets like explained in the previous sub-chapter. Instead, I chose to use priorities. I gave the transitions leading along the most used route the highest priorities, the transitions leading to the second most used routes the second highest priorities, etc. There is some randomness as to which routes are picked, so with that in mind, using priorities is not bad, but using coloured petri nets would still be more accurate.

5.4 Conclusion

I revise my opinion in the introduction, the traffic jam is not reduced enough to make up for the noise that the inhabitants in Havnegata are subjected to. The performance overall is worse than before Havnegata was opened, because there are more roundabouts and "cheaters" who drive through Havnegata which leads to more bottlenecks. Ofcourse, I can't say for sure how many will "cheat" in reality, but according to my estimations it is not favourable to open Havnegata for traffic during rush hours.

References

Cover image: https://upload.wikimedia.org/wikipedia/commons/e/e2/Sandnes.JPG

[1] Sandnesposten. https://sandnesposten.no/nyheter/vil-apne-havnegata-for-trafikk-i-rushtiden/19.9102

[2] *Roberto Martin Muñoz and Tina Kristin Jensen (2013).* 2013-P09-Traffic-at-Algard-Forus.pdf.

[3] David Barka, T. H. (2014). *Simulation of Tjensvoll Junction.*

[4] Jagusch, A. (2018, 10 02). *APO A Free Online Petri Net Editor*. Retrieved from adrian-jagusch: https://adrian-jagusch.de/2017/02/apo-a-free-online-petri-net-editor/

[5] Statens Vegvesen. (2018, 10 02). *vegkart*. Retrieved from vegvesen: https://www.vegvesen.no/vegkart/vegkart/#kartlag:geodata/hva:(~(id:482,filter:(~(type_id:3

910,operator:'*3d,verdi:(~4892))),farge:'0_0),(id:540,filter:(~),farge:'2_0))/hvor:(kommune:(~1102))/@-32955,6560648,13/vegobjekt:163283130:ea2336:540

Appendix A

A.1 User Manual

Run the main file 'main_simulation.m'. The main files have the same names for both the models.

A.2 Code for Petri Net Model 1

A.2.1 'model1_pn_pdf.m'

```
function [pns] = model1_pn_pdf()

pns.PN_name = 'Petri_Net_definition';

pns.set_of_Ps = {'p1_Start', 'p2_Start', 'p3_Center', 'p4_Center',
'p5_Center', 'p6_Center', 'p7Out', ...
    'p8Out', 'p9Out'};

pns.set_of_Ts = {'t1_3','t2_4', 't3_4', 't4_3', 't4_8', 't3_5',
't4_6', 't5_6', 't6_5', 't5_7', 't6_9'};

pns.set_of_As = {'p1_Start','t1_3',1,
't1_3','p3_Center',1,'p2_Start','t2_4',1,'t2_4','p4_Center',1 ...
    ,'p3_Center', 't3_4', 1, 'p4_Center', 't4_3', 1, 't4_3',
'p3_Center', 1, 't3_4', 'p4_Center', 1, 'p4_Center', 't4_8', 1,
't4_8', 'p8Out', 1 ...
    , 'p3_Center', 't3_5', 1, 't3_5', 'p5_Center', 1, 'p5_Center',
't5_7', 1, 't5_7', 'p7Out', 1 ...
    , 'p5_Center', 't5_6', 1, 't5_6', 'p6_Center', 1, 'p6_Center',
't6_5', 1, 't6_5', 'p5_Center', 1 ...
    , 'p4_Center', 't4_6', 1, 't4_6', 'p6_Center', 1, 'p6_Center',
't6_9', 1, 't6_9', 'p9Out', 1};
```

A2.2 'main_simulation.m'

```
clear all; clc;

global global_info;

global_info.STOP_AT = 500;      %Stop simulations after 500 TU.
global_info.south = 1;
global_info.north = 1;
global_info.west = 1;
global_info.east = 1;

pns = pnstruct('model1_pn_pdf');

dyn.m0 = {'p1_Start', 13350/2, 'p2_Start',8500/2};
```

```
%variable firing times in seconds/100:
dyn.ft = {'t1_3',normrnd(0.72,0.058), 't2_4', normrnd(0.12,0.0075),
't3_4', normrnd(0.16,0.009), 't4_3',...
    normrnd(0.17,0.0095), 't4_8', normrnd(0.087,0.0038), 't3_5',
normrnd(0.16,0.0065), 't4_6', ...
    normrnd(0.21,0.0092), 't5_7', normrnd(0.09,0.0058),'t5_6',
normrnd(0.21,0.009), 't6_5', ...
    normrnd(0.14,0.0034), 't6_9', normrnd(0.09,0.004)};
%priorities
dyn.ip = {'t1_3', 2, 't3_4', 2, 't4_6', 2, 't6_9', 2, 't3_5', 1,
't6_5', 1, 't5_7', 1, 't4_3', 1};
pni = initialdynamics(pns, dyn);

sim = gpensim(pni);
%Plot of the ending places:
figure();
plotp(sim, {'p7Out','p8Out', 'p9Out'});
```

A2.3 'COMMON_PRE.m'

```
function [fire, transition] = COMMON_PRE(transition)

global global_info;

p3C = get_place('p3_Center');
p4C = get_place('p4_Center');
p5C = get_place('p5_Center');
p6C = get_place('p6_Center');

time_stamp = string_HH_MM_SS(current_time());
fprintf('transition: "%s" initiated firing at %s\n\n',
transition.name, time_stamp);

if (strcmp(transition.name, 't4_3')||strcmp(transition.name,
't1_3'))
    aux=(eq(p3C.tokens,0) && eq(global_info.east, 1));
    if aux
        global_info.east=0;
        fire=true;
    else
        fire=false;
    end
elseif (strcmp(transition.name, 't2_4')||strcmp(transition.name,
't3_4'))
    aux=(eq(p4C.tokens,0) && eq(global_info.south, 1));
    if aux
        global_info.south=0;
        fire=true;
    else
        fire=false;
    end
elseif (strcmp(transition.name, 't6_5')||strcmp(transition.name,
't3_5'))
```

```
    aux=(eq(p5C.tokens,0) && eq(global_info.north, 1));
    if aux
        global_info.north=0;
        fire=true;
    else
        fire=false;
    end
elseif (strcmp(transition.name, 't4_6')||strcmp(transition.name,
't5_6'))
    aux=(eq(p6C.tokens,0) && eq(global_info.west, 1));
    if aux
        global_info.west=0;
        fire=true;
    else
        fire=false;
    end
else
    fire = true;
end
```

A2.4 'COMMON_POST'.m

```
function [] = COMMON_POST(transition)

global global_info;

time_stamp = string_HH_MM_SS(current_time());
fprintf('transition: "%s" finished firing at %s\n\n',
transition.name, time_stamp);

if (strcmp(transition.name, 't4_3') || strcmp(transition.name,
't1_3'))
    global_info.east=1;
elseif (strcmp(transition.name, 't2_4') || strcmp(transition.name,
't3_4'))
    global_info.south=1;
elseif (strcmp(transition.name, 't6_5')||strcmp(transition.name,
't3_5'))
    global_info.north=1;
elseif (strcmp(transition.name, 't4_6') || strcmp(transition.name,
't5_6'))
    global_info.west=1;
end
```

A.3 Code for Petri Net Model 2

A3.1 'model2_pn_pdf.m'

```
function [pns] = model2_pn_pdf()

pns.PN_name = 'Petri_Net_definition';

pns.set_of_Ps = {'p1_Start', 'p2_Start', 'p3_Center', 'p4_Center',
'p5_Center', 'p6_Center', 'p7_Center' ...
    ,'p8_Center', 'p9Out', 'p10Out', 'p11Out'};
```

```
pns.set_of_Ts = {'t1_3','t3_4', 't4_3', 't4_5', 't5_4', 't2_5',
't5_10', 't3_6', 't4_7', 't5_8', 't6_9'...
    , 't6_7', 't7_6', 't7_8', 't8_7', 't8_11'};

pns.set_of_As = {'p1_Start','t1_3',1,
't1_3','p3_Center',1,'p2_Start','t2_5',1,'t2_5','p5_Center',1 ...
    , 'p3_Center', 't3_4', 1, 't3_4', 'p4_Center', 1, 'p4_Center',
't4_3', 1, 't4_3', 'p3_Center', 1, ...
    'p4_Center', 't4_5', 1, 't4_5', 'p5_Center', 1, 'p5_Center',
't5_4', 1, 't5_4', 'p4_Center', 1, ...
    'p5_Center', 't5_10', 1,'t5_10', 'p10Out', 1, 'p3_Center',
't3_6', 1, 't3_6', 'p6_Center', 1, ...
    'p4_Center', 't4_7', 1, 't4_7', 'p7_Center', 1, 'p5_Center',
't5_8', 1, 't5_8', 'p8_Center', 1, ...
    'p6_Center', 't6_9', 1, 't6_9', 'p9Out', 1, 'p6_Center', 't6_7',
1, 't6_7', 'p7_Center', 1, ...
    'p7_Center', 't7_6' ,1, 't7_6', 'p6_Center', 1, 'p7_Center',
't7_8', 1, 't7_8', 'p8_Center', 1, ...
    'p8_Center', 't8_7', 1, 't8_7', 'p7_Center', 1, 'p8_Center',
't8_11', 1, 't8_11', 'p11Out', 1};
```

A3.2 'main_simulation.m'

```
clear all; clc;

global global_info;

global_info.STOP_AT = 500; %Stop simulations after 500 TU

global_info.s = 1;
global_info.n = 1;
global_info.nw = 1;
global_info.ne = 1;
global_info.sw = 1;
global_info.se = 1;

pns = pnstruct('model2_pn_pdf');

dyn.m0 = {'p1_Start', 13350/2, 'p2_Start',8500/2};
%Stochastic firing times in seconds/100:
dyn.ft = {'t1_3',normrnd(0.22, 0.0154), 't3_4', normrnd(0.47,
0.033), 't4_5', normrnd(0.165, 0.007),...
    't5_10', normrnd(0.089, 0.0046), 't4_7', normrnd(0.147, 0.006),
't5_8', normrnd(0.184,0.0098),...
    't6_7', normrnd(0.115, 0.009), 't7_6',
normrnd(0.12,0.01),'t7_8', normrnd(0.16,0.0072), 't8_7', ...
    normrnd(0.163, 0.012), 't6_9', normrnd(0.093, 0.003), 't8_11',
normrnd(0.089, 0.004), 't3_6', ...
    normrnd(0.638, 0.023), 't4_3', normrnd(0.492, 0.031), 't5_4',
normrnd(0.145, 0.0049), 't2_5', ...
    normrnd(0.127, 0.0078)};
%Initial Priorities:
dyn.ip = {'t3_4', 2, 't4_3', 1, 't4_5', 2, 't5_4', 1, 't3_6', 2,
't4_7', 2, ...
```

```
    't5_8', 2, 't6_9', 1, 't6_7', 2, 't7_6', 1, 't7_8', 2, 't8_7',
1, 't8_11', 2};

pni = initialdynamics(pns, dyn);
sim = gpensim(pni);
%Plot of the ending places:
figure();
plotp(sim, {'p9Out','p10Out', 'p11Out'});
```

A3.3 'COMMON_PRE.m'

```
function [fire, transition] = COMMON_PRE(transition)

global global_info;

p3C = get_place('p3_Center');
p4C = get_place('p4_Center');
p5C = get_place('p5_Center');
p6C = get_place('p6_Center');
p7C = get_place('p6_Center');
p8C = get_place('p6_Center');

time_stamp = string_HH_MM_SS(current_time());
fprintf('transition: "%s" initiated firing at %s\n\n',
transition.name, time_stamp);

if (strcmp(transition.name, 't4_3')||strcmp(transition.name,
't1_3'))
    aux=(eq(p3C.tokens,0) && eq(global_info.se, 1));
    if aux
        global_info.se=0;
        fire=true;
    else
        fire=false;
    end
elseif (strcmp(transition.name, 't4_5')||strcmp(transition.name,
't2_5'))
    aux=(eq(p5C.tokens,0) && eq(global_info.sw, 1));
    if aux
        global_info.sw=0;
        fire=true;
    else
        fire=false;
    end
elseif (strcmp(transition.name, 't3_4')||strcmp(transition.name,
't5_4'))
    aux=(eq(p4C.tokens,0) && eq(global_info.s, 1));
    if aux
        global_info.s=0;
        fire=true;
    else
        fire=false;
    end
elseif (strcmp(transition.name, 't3_6')||strcmp(transition.name,
't7_6'))
```

```
    aux=(eq(p6C.tokens,0) && eq(global_info.ne, 1));
    if aux
        global_info.ne=0;
        fire=true;
    else
        fire=false;
    end
elseif (strcmp(transition.name, 't4_7')||strcmp(transition.name,
't6_7')||strcmp(transition.name, 't8_7'))
    aux=(eq(p7C.tokens,0) && eq(global_info.n, 1));
    if aux
        global_info.n=0;
        fire=true;
    else
        fire=false;
    end
elseif (strcmp(transition.name, 't5_8')||strcmp(transition.name,
't7_8'))
    aux=(eq(p8C.tokens,0) && eq(global_info.nw, 1));
    if aux
        global_info.nw=0;
        fire=true;
    else
        fire=false;
    end
else
    fire = true;
end
```

A3.4 'COMMON_POST'.m

```
function [] = COMMON_POST(transition)

global global_info;

time_stamp = string_HH_MM_SS(current_time());
fprintf('transition: "%s" finished firing at %s\n\n',
transition.name, time_stamp);

if (strcmp(transition.name, 't4_3') || strcmp(transition.name,
't1_3'))
    global_info.se=1;
elseif (strcmp(transition.name, 't4_5') || strcmp(transition.name,
't2_5'))
    global_info.sw=1;
elseif (strcmp(transition.name, 't3_4') || strcmp(transition.name,
't5_4'))
    global_info.s=1;
elseif (strcmp(transition.name, 't3_6') || strcmp(transition.name,
't7_6'))
    global_info.ne=1;
elseif (strcmp(transition.name, 't4_7') || strcmp(transition.name,
't6_7')|| ...
        strcmp(transition.name, 't8_7'))
    global_info.n=1;
```

```
elseif (strcmp(transition.name, 't5_8') || strcmp(transition.name,
't7_8'))
    global_info.nw=1;
end
```