CPN Assignment BIS 2IIC0/2IHI10 (2017/2018 Q1)

version from 26-09-2017

1 Overview

This assignment belongs to the course Business Information Systems (BIS). This is a 5 ECTS course consisting of a combination of lectures (4 hours per week), instructions (4 hours per week), and self-study (about 10 hours a week). The CPN assignment is part of the self-studies and instructions.

Assignment Structure and Deadlines

On the following pages you will find the description of a business information system with various requirements. The basic environment and definitions of the system as well as the general assignment are explained in Section 2; on OASE, you can find a file **`baseModel_ID<YOURSTUDENTID>.cpn`** that already contains this environment and definitions. In this assignment, you are asked to extend this provided model to satisfy a number of different requirements to simulate the model in order to understand its performance characteristics, and to describe your model and your simulation result in a short report. The assignment is split into two parts.

- **Part I** consists of Tasks 1, 2, and 3 described in Section 3 that gradually build up all requirements for the system. Your solution for **Task 3** has to be handed in before Sunday, **8th October 2017, 22:00** via Canvas.
- **Part II** consists of Tasks 4 described in Section 4. Your solution for **Task 4** has to be handed in before Friday, **23rd October 2016, 22:00** via Canvas.

Section 5 gives hints on creating good models, running simulations, and writing your reports, and gives instructions on how to hand in your solutions. The deadlines are firm and you cannot resubmit solutions to your tasks after the respective deadline.

Evaluation

The CPN Assignment **accounts for 20% of the final grade** of the course (10% of the grade are obtained in the pre-exam and 70% of the grade are obtained in the final exam). The points obtained in the CPN assignment remain valid for the rest of the academic year, i.e., a resit of the exam cannot be used to improve points obtained in the CPN assignment.

Your submission for a solution of either part (i.e., Task 3 and Task 4) consists of the model made using CPN Tools and a short report describing your model (use the report template provided).

Models are evaluated based on:

1) Technical requirements, i.e. does the model correctly implement the specification? Are there no deadlocks? Are there no tokens left behind? Etc. All these aspects will be addressed in the lectures.

2) Readability and understandability by a human. An essential aspect of modeling is to capture a complex situation in a readable, easy to understand model.

To evaluate these two criteria, we use a mixture of automatic checks and human grading. The automatic checks will evaluate syntactic correctness, the ability to run a complete simulation, and whether the model satisfies the technical requirements described in this report. It is therefore essential to provide the files in the right format as specified in this document.

Reports are evaluated based on:

1) Does the report explain how the model works? Does the report explain how the technical solutions created in the model relate to the assignment?
2) Does the report contain a readable model?
3) Does the report contain and discuss simulation results?

This assignment is **to be solved individually** and solutions have to be **original**. Nevertheless, we encourage you to discuss the assignment and your ideas during the instructions.

To evaluate readability, the instructors will look at each model and report individually. In order to get intermediate feedback on the readability of your model, make use of the instructions. Questions with respect to readability or technical requirements will not be answered by e-mail.

Lecturers:

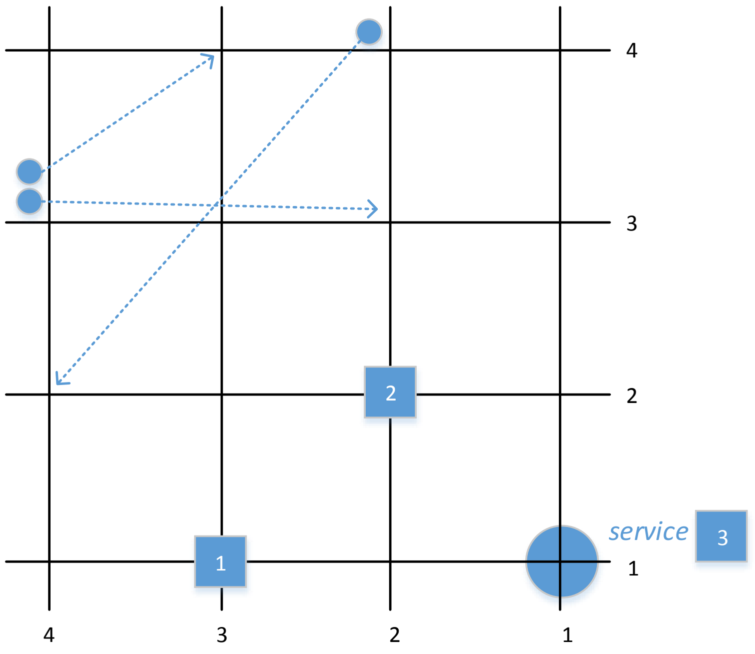
- Dr. Massimiliano (lectures)
- Dr. Dirk Fahland (instructions & assignment)
- Dr. Boudewijn van Dongen (instructions & assignment),
- Ms. Alifah Syamsiyah (instructions)
- Mr. Mitchel Brunings (instructions)

For questions please contact the lecturers during, in-between, or directly after lectures and instructions. Use Canvas for technical questions related to CPN Tools.

2 Assignment Description

The setting of the assignment is a ride share service using autonomous, electric cars, and its processes for handling trip requests by passengers and recharging and maintaining the vehicle fleet.

The ride share service operates in a city with a grid layout of 4x4 points as shown below.



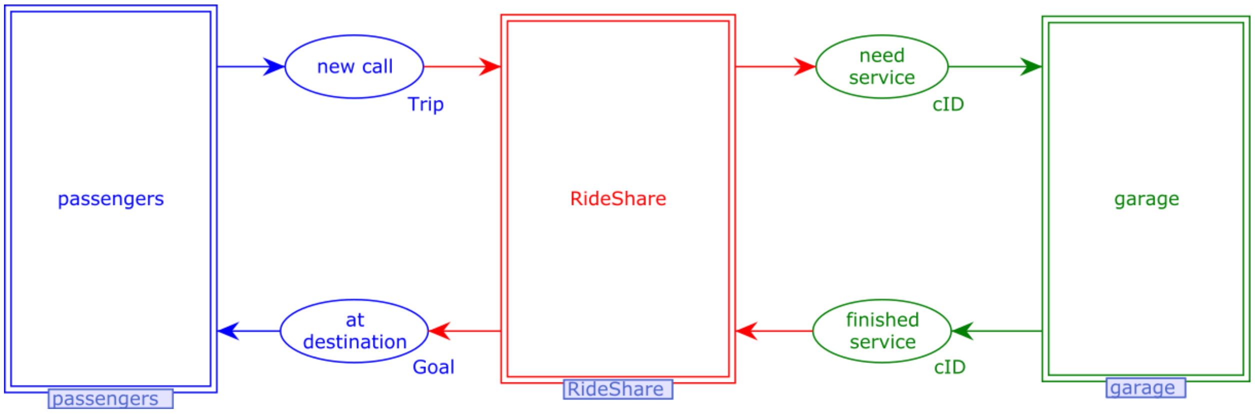
Passengers can call for single-hop rides from one point in the grid to another point in the grid. In the image above, there are three ride calls: from (4,3) to (3,4), from (4,3) to (2,3), and from (2,4) to (4,2). The distance between two points is measured in Manhattan distance, assuming a 1km distance between the x and the y coordinates, respectively. For example, the trip from (2,4) to (4,2) has a distance of 4km, the longest possible trip is 6km.

A calls is answered by a ride share vehicle who drives to a passenger's call location and then bring them to their desired destination (along the roads in the grid). Cars regularly have to return to the service location of the ride share service being located at point (1,1) for servicing.

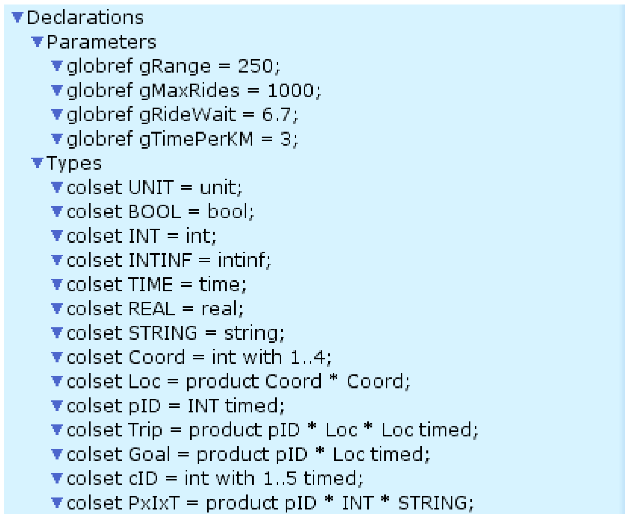
Next, we give an overview on the system, basic data types and definitions, then describe the passenger arrival process, and the servicing process in the garage, and finally give the requirements for the ride share service.

System overview

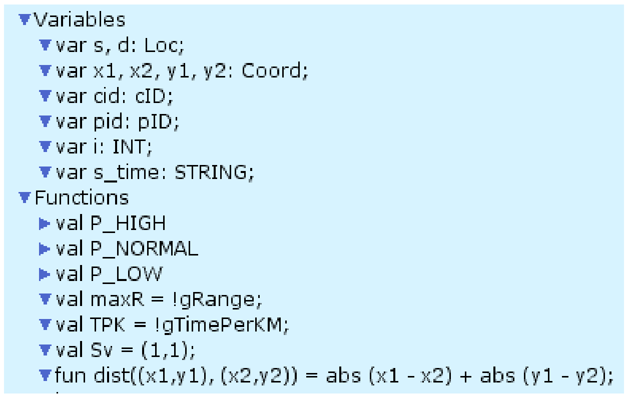
Part of the model is already given in CPN Tools. The top level model of the system is shown below.



The subnets corresponding to `passengers` and the `garage` (for servicing cars) are given. The subnet corresponding to the ride share service (`RideShare`) is empty. As the above figure shows the three subnets are connected through places of type `Trip`, `Goal`, and `cID`. The blue places correspond to the placement of a new ride call and the passenger reaching their destination, the green places correspond to a car needing or finishing service. The following color sets, variables, and basic functions are given under `Declarations`:



```
▼Declarations
  ▼Parameters
    ▼globref gRange = 250;
    ▼globref gMaxRides = 1000;
    ▼globref gRideWait = 6.7;
    ▼globref gTimePerKM = 3;
  ▼Types
    ▼colset UNIT = unit;
    ▼colset BOOL = bool;
    ▼colset INT = int;
    ▼colset INTINF = intinf;
    ▼colset TIME = time;
    ▼colset REAL = real;
    ▼colset STRING = string;
    ▼colset Coord = int with 1..4;
    ▼colset Loc = product Coord * Coord;
    ▼colset pID = INT timed;
    ▼colset Trip = product pID * Loc * Loc timed;
    ▼colset Goal = product pID * Loc timed;
    ▼colset cID = int with 1..5 timed;
    ▼colset PxIxT = product pID * INT * STRING;
```



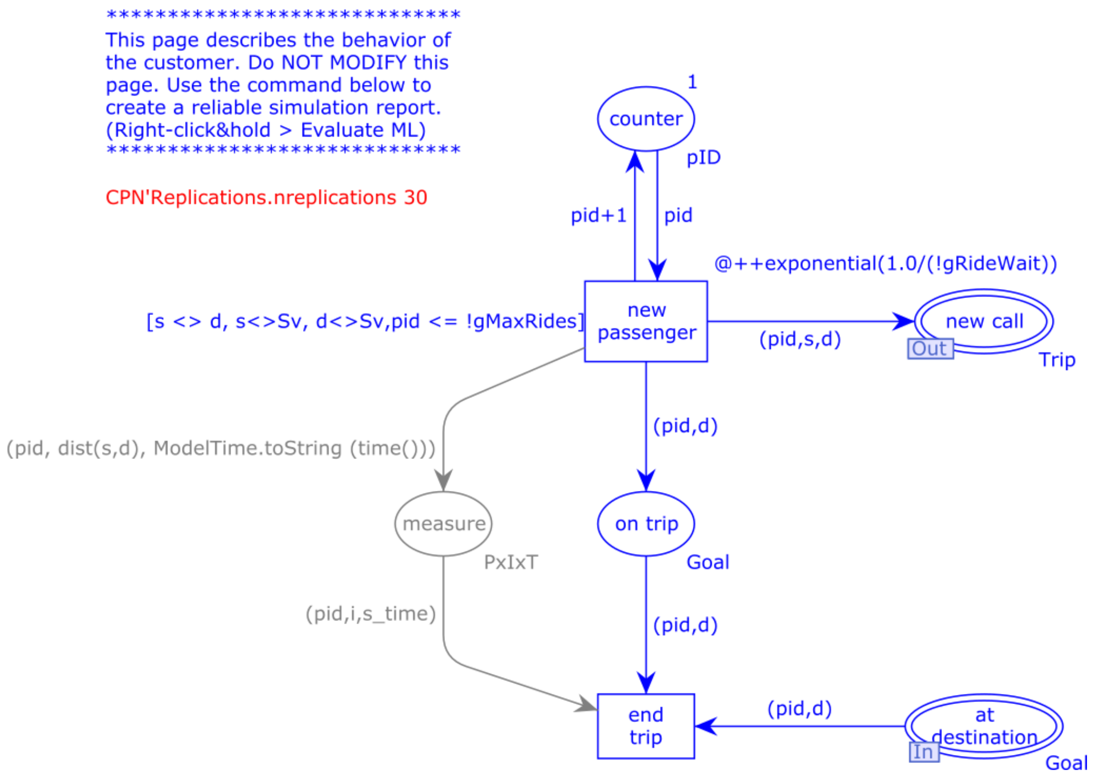
```
▼Variables
  ▼var s, d: Loc;
  ▼var x1, x2, y1, y2: Coord;
  ▼var cid: cID;
  ▼var pid: pID;
  ▼var i: INT;
  ▼var s_time: STRING;
▼Functions
  ▶val P_HIGH
  ▶val P_NORMAL
  ▶val P_LOW
  ▼val maxR = !gRange;
  ▼val TPK = !gTimePerKM;
  ▼val Sv = (1,1);
  ▼fun dist((x1,y1), (x2,y2)) = abs (x1 - x2) + abs (y1 - y2);
```

Note that a `Trip` consists of a unique passenger ID, a source location (an x-Coordinate and a y-Coordinate in the 4x4 grid), and a destination location, and a timestamp. The timestamp indicates the time the call was placed. A `Goal` consists of a unique passenger ID, the destination location of the trip, and the timestamp of reaching the destination.

A `cID` is a unique identifier for a car, with a timestamp – the ride share service has 5 cars. Cars can ride at an average of 20km/h through the city (3min per km, value TPK), a car has a maximum range of 250km (value `maxR`), and always needs service after completing 120 trips.

Passenger Arrival Process

Passengers place a new call for a ride (consisting of a source and a destination location) according to a Poisson arrival process. The subnet `passengers` is shown below:



****
This page describes the behavior of the customer. Do NOT MODIFY this page. Use the command below to create a reliable simulation report. (Right-click&hold > Evaluate ML)
****
CPN'Replications.nreplications 30
counter
1
pID
pid+1
pid
@++exponential(1.0/(!gRideWait))
[s <> d, s<>Sv, d<>Sv,pid <= !gMaxRides]
new passenger
(pid,s,d)
new call
Out
Trip
(pid, dist(s,d), ModelTime.toString (time()))
(pid,d)
measure
PxIxT
on trip
Goal
(pid,i,s_time)
(pid,d)
end trip
(pid,d)
at destination
In
Goal

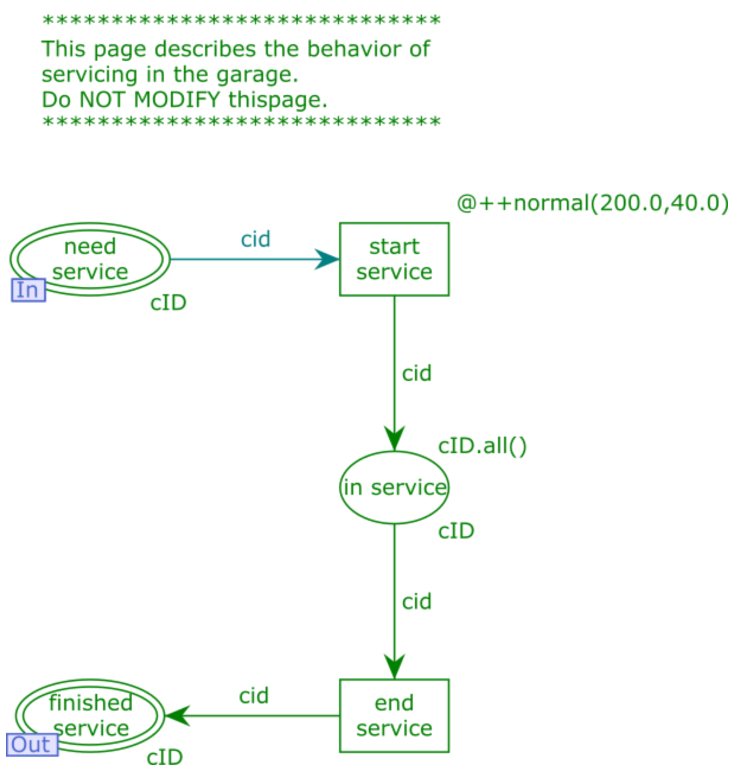
The subnet shows that the average time between two calls is 6.7 minutes (Poisson arrival process, see global constant `gRideWait`). Source and destination of a called trip are uniformly distributed over the city, though source and destination have to be different and a trip may not start or end in the service location (value `Sv = (1,1)`) of the ride share service; these requirements are implemented by the guard of transition `[new passenger]`.

The lower part of the subnet uses so-called monitors to measure statistics. These statistics are stored in the subdirectory "Output" if a single run is simulated. If multiple subruns are generated, then these are stored in subdirectory "Reps_X". For the assignment it is necessary to compute confidence intervals, so multiple subruns are needed (use the command `CPN'Replications.nreplications 30` as described at the end of this assignment document). The statistics measured in the simulation are

- Number of passengers "on trip" at the same time (after placing a new call and before arriving at their destination).
- Travel time for trips of 1-3km length and for trips of 4-6km length.
- Trips with a too long travel time. A trip is considered as taking too long if the passenger would have reached their destination faster by walking (i.e., if it took 5 times longer than an immediate ride plus times for entering/leaving the car, which corresponds to an average 4km/h walking speed).
- Cars in service in the garage (see next page).

Garage Process

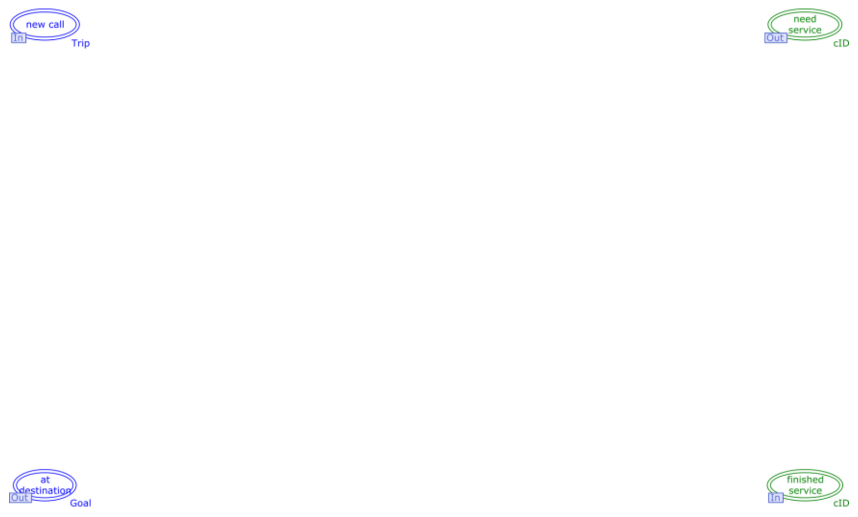
We assume the following rather simple process for servicing a car in the garage:



Cars need to be serviced regularly (cleaning etc.) and are brought to the garage via place `(need service)`. Service times are normally distributed (taking 200 minutes on average with a standard deviation of 40 minutes). Initially, all 5 cars of the ride share service are in the garage and just completed service, i.e., are in place `(in service)` and available to `[end service]`. Cars can only start/end service in the garage when they are at the service location Sv = (1,1).

General Assignment: RideShare service (to be modeled by you)

You are given the task to model the complete car handling of the ride share service by completing the corresponding service template of `RideShare` given below.



The ride share service has to satisfy the requirements on trips, street network layout, and cars as described above and handle

- allocating incoming ride calls from passengers to cars,
- routing cars to received calls, route cars to their destination, and drop off passengers,
- managing energy consumption and recharging of cars, and
- sending cars to servicing when needed.

In the following, you are given 4 more specific tasks that detail different parameters or assumptions on the ride share service. You have to solve each task of this assignment using CPN Tools (version 4.0.1, see http://cpntools.org/download)[1] by modifying the **`baseModel_ID<YOURSTUDENTID>.cpn`** provided to you on Canvas.

The file with your student id in the name **must be used** to model the **individual** tasks. Do not use a base model designated for another student; otherwise, we will not be able to assign you points for your solution.

Unless specified otherwise, your solution should be *generic*, i.e., it should be easy to support other street network layouts, different numbers of cars with other characteristics. In other words: the solution should be configurable and not depend on the particular definitions of `Coord, cID, maxR, TPK, Sv, dist(s,d)`.

Important note: The above process description and also the subsequent tasks are partly – and on purpose – **underspecified**. In other words, there is not a step-by-step description of the process, but rather a list of abstract requirements. The goal and purpose of this assignment is that **you resolve this underspecification** and come up with a concrete model satisfying all requirements. The purpose of the instructions is that **you ask the instructors** about your ideas and assumptions as you come up with a solution.

Please see Section 5 for guidelines to submitting your models and reports.

[1] This version of CPN Tools has been tested to work on Windows 7, 8, 8.1., and 10, Ubuntu 16.04 and Mac OS X using Java 7 and Java 8.

3 Part I

Part I consists of Task 1, Task 2 (extends Task 1), and Task 3 (extends Task 2). Only Task 3 shall be submitted.

3.1 Task 1 – Basic Control Flow

Provide a CPN model that describes the rideshare service (page `RideShare`). For this task *abstract from energy consumption of cars*, i.e., cars travel according to the grid layout and their speed, but traveling does not consume any energy, and energy does not have to be recharged. The solution should be *generic* (see Section 2).

The following requirements should be satisfied:

- All cars coming from the garage start at location Sv = (1,1) and are free to service Trips; cars can only enter the garage for servicing when they are at location Sv.
- A car can serve 1 passenger at a time.
- A new Trip can be answered by any free car (there is no priority or ordering of calls).
- Once a Trip is answered by a car, the car drives from its current location directly to the source location of the trip, picks up the passenger, drives to the destination location, drops off the passenger, and then is free again.
- Driving takes 3 minutes per km, picking up a passenger or dropping off a passenger needs 1 minute.
- Trips are not necessarily answered in order of arrival, i.e., one call can overtake another call.
- Once a car completed 120 trips, it is in need of service in the garage and cannot answer another Trip.

Create the model in CPN tools and simulate it.

Note: In case your simulation report is empty or shows strange statistics, please consult Section 6 on how to configure your model to generate a simulation report.

Hints: *When modeled correctly, the process should show that 1000 rides can be completed at a relatively good, but not perfect performance (about 2.5% of all rides are too slow).*

3.2 Task 2 - Resources and Charging

Extend and modify your solution for Task 1 to describe the service now taking into account the energy resources it takes for driving and the recharging of vehicles.

The following requirements should be satisfied:

- All requirements of Task 1.
- Every car has a range of 250km and is fully charged when leaving the garage; each km driven reduces the available charge in the car accordingly.
- A car can recharge at a charger at the service location Sv = (1,1). Recharging takes 2 minutes per km of range that has to be recharged. Cars are always recharged to their maximum range before leaving the charger. At most three cars can be charged at the same time.
- Cars must not run out of charge, i.e., they must not reach a charge of <= 0.
- Cars may only go to the garage when fully charged.
- Cars still have to get service after completing 120 trips, but they are allowed to postpone service and serve more trips until their current charge is used up, before returning to the service location.

Create the model in CPN tools and simulate it.

Hints: *Validate your model by comparing the performance measures you get now to the measures for the model of Task 1. Consider the following questions: Because of the above changes, should the performance of the service improve or worsen? Which changes introduced above impact which performance measures of the service? Is the service now operating below capacity or has it still capacity left?*

3.3 Task 3 – Trip Batching (7 points[2])

Extend and modify your solution for Task 2 to describe the service now taking into account that a car may answer multiple calls from the same source at the same time.

The following requirements should be satisfied:

- All requirements of Task 2 and Task 1 unless specified otherwise below.
- A car can now serve up to 4 passengers at a time.
- Trips from different passengers starting at the same source location can be grouped together to groups of 4 Trips (or less); passengers in a group may have different destinations. Whenever a new trip can still join a group at its source location, it does so. Otherwise, a new group is formed by the service.
- Any group of trips can be answered by a free car (there is no priority or ordering of groups).
- Once a group of trips is answered by a car, the car drives from its current location directly to the source location of the group. While the car is driving, further Trips can still be added to the group. When the car arrives, it picks up all passengers (from this point on no more Trips are added to the group), and then drives to each destination location dropping off passengers until it is free again.
- Passengers are dropped off in the order in which they joined the group of trips (first come, first serve).

Create the model in CPN tools and simulate it. Describe your model and list your simulation results in a **short report** (use the template provided) and **submit model and report**; see Section 5 for hints and requirements on the model format, the report contents, and submission instructions.

Hints: *Validate your model by comparing the performance measures you get now to the measures for the model of Task 2. Consider the following questions: Because of the above changes, should the performance of the service improve or worsen? Which changes introduced above impact which performance measures of the service? Is the service now operating below capacity or has it still capacity left?*

[2] The requirements for Task 1 correspond to about 2 points, for Task 2 to about 2 points, and for Task 3 to about 3 points.

4 Part II

Part II is an *extension* of Part I. As you receive feedback on Part I before the deadline for Part II, you are expected to **fix any mistakes and issues** in your model related to the requirements stated in Part I. Solutions for Part II that do not satisfy the requirements for Part I will not be given full points.

4.1 Task 4 – Sensitivity Analysis through Simulation (3 points)

The ride share service has the following cost and income structure:

- One car has a daily fixed ownership costs of 20 EUR (yearly ownership costs of about 7300 EUR).
- Operating a car costs 1.5 EUR per km driven.
- The ride share service has an income of 2 EUR per km in a Trip, but rides that take "too long" are free.

The simulation analysis of Tasks 2 and 3 shows that the ride share service cannot serve all calls fast enough, leading to a net loss. The objective of this task is to identify how to make the car service profitable.

- The ride share service has a number of parameters that can be changed without changing the underlying processes, in particular, *number of passengers in a car*, and *number of cars*. Other parameters can be considered after consulting instructors during an instruction.
- Use simulation to identify for which parameter settings, the ride share service becomes profitable.
- For your calculations, assume that each trip has a length of 2.6km (which is the average length of a trip in the given street layout), and that the total operating time is 7000mins (to complete all trips).

Create the model in CPN tools, run simulations for several different parameter settings for *number of passengers in a car*, and *number of cars*, and identify a parameter setting where the ride share service becomes profitable. In a short report, show your fixed model (describe any changes made to the previous solution), **present simulation results for at least 5 different parameter settings** (including the situation in Task 3 and your "profitable" setting), and clearly explain which parameter settings should be chosen. **Submit model and report**; see Section 5 for hints and requirements on the model format, the report contents, and submission instructions.

5 Models, Reports, and Submission

Please use the following hints and guidelines when preparing your solutions for the tasks above.

Models

For every task, a part of the model is already given as a CPN. This model is provided as a file "**`baseModel_IDxxxxxxx.cpn`**" via OASE. The file with your student id in the name **must be used** to model the **individual** tasks. Do not use a base model designated for another student; otherwise, we will not be able to assign you points for your solution.

- ☐ Always use the base file specific for you, do not use base files specific to other students.
- ☐ Only modify the subpage "`RideShare`" of your base model
  - ☐ You may add further subpages to "`RideShare`", but experience tells that the solution fits nicely within the provided subpage.
  - ☐ Do not change any of the provided declarations or other subpages. Models in which the given environment (`passengers` and `garage`) or the given declarations have been changed will not be judged. We recommend that you use an extra declaration block, for instance the provided block called "`Own Declarations`", for the declarations you add.
- ☐ Models will be judged on correctness (i.e., conformance to specification) and readability.
  - ☐ Discuss requirements and scenarios with your fellow students and ensure that your model can always complete 1000 Trips, cars remain operational, and no tokens are left behind (except for tokens present in the initial marking).
  - ☐ Use any feedback you receive to improve your solutions; in particular, use the feedback on your solutions for Part I to prevent carrying over any mistakes or flaws to your solution for Part II.
  - ☐ Give your net structure a nice, easy to follow layout (i.e., avoid a spaghetti-like net structure); you can make use of colors to separate different functional elements of your model.
  - ☐ Use self-explanatory, understandable labels for places, transitions, and functions.
  - ☐ Keep the model simple (i.e., go for simple data structures); it should be possible for a fellow student to understand your model within a few minutes.
  - ☐ Models that cannot be simulated automatically or models of which the simulation does not terminate are evaluated with 0 points.

Make use of the instructions to get an indication of whether your models are readable or not.

Reports

The purpose of the report is to let another person *understand* your model, i.e., its central functionality. Think of a fellow student of the BIS course as a reader who takes the report to understand how you designed your model, which declarations you made, how it works, and how it realizes the individual requirements. Keep the following points in mind.

- ☐ The report can be short (about 2-3 pages of text + pictures will suffice in most cases) and should have a proper title indicating the task being solved and your student id.

- ☐ **Use the template provided to structure your report.**
- ☐ Include a **readable** diagram (picture) of your solution, i.e., subpage `RideShare` and every extra subpage you created.
- ☐ Explain how your model works:
    - ☐ Explain the general flow of the process. That is, explain how a car responds to a call and how it is being serviced. Name each place and transition on the way. For example, *“Transition [new passenger] in subpage ‘passengers’ creates a call for a Trip on place (new call) with a fresh passenger pid, a source location s, and a destination location d.”*
    - ☐ Explain your design decisions, that is, explain every *new* colorset, guard, function, and non-trivial arc inscription (i.e., everything more complex than a tuple) in your solution. For example, *“In the subpage ‘passengers’, the guard ‘pid <= !maxRides’ ensures that exactly 1000 Trips are created.”*
    - ☐ When describing your model for Part II, only describe the changes you made to the previous model (do not describe the entire model again, but include changes made to fix mistakes in your solution for Part I).
- ☐ Analyze the flow times and present the results of your analysis.
    - ☐ Include a copy of the entire analysis report produced by CPN Tools at the end of your report; in the text, select relevant numbers to support your analysis.
    - ☐ The analysis of flow times should be based on running at least 30 subruns each for handling 1000 customer orders (this is the default setting, see also the very last page of this document).
    - ☐ Try to explain the numbers obtained in the simulation.

Submission Format

For each part, the following results should be submitted:

- Save the solution for Task X as a CPN file named “**`ID[yourID]-taskX.cpn`**".
- Save the report, explaining the model of Task X, as a pdf file “**`ID[yourID]-taskX.pdf`**".
- Submit both files **`ID[yourID]-taskX.cpn`** and **`ID[yourID]-taskX.pdf`** via Canvas

The assignments are, in part, checked automatically by software. Any submission that does not conform to the submission formats will therefore be rejected and receive no points.

Submit

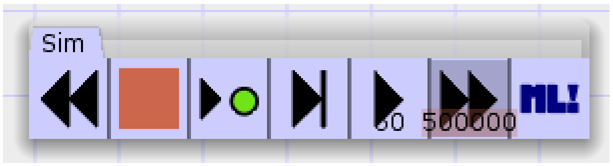
Submit your solution for each part individually via Canvas (select the corresponding assignment part to upload your solution for that part).

In case you have to change an uploaded version, resubmit **all** required files. We will evaluate the submission with the *latest* timestamp.

Hints

- Check the CPN Tools Web page (http://cpntools.org/) if you have questions about the CPN syntax and how to model with CPN Tools.

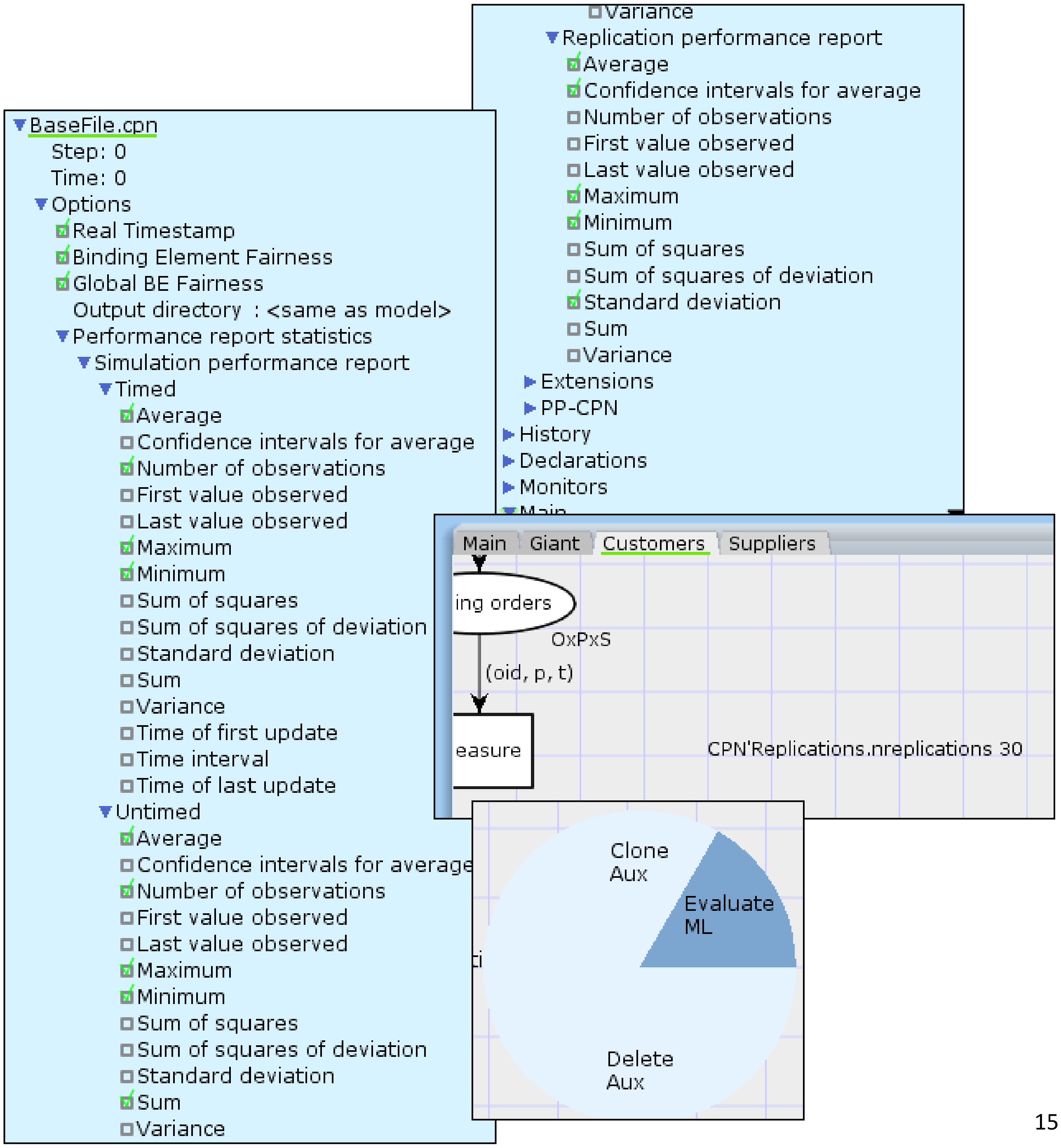
- Running a single simulation of a CPN model on a modern computer takes only few seconds (for 30 replications it takes a bit over one minute). If the simulation takes several minutes, then this is a hint that your model is too complex (e.g., you have places containing thousands of tokens) or the simulation does not even stop (i.e., in the `replication_report.txt` file in `output/rep_x/` no simulation is reported as completed). In this case, simulate your model manually to figure out the problem.
- If you experience that your simulation of 30 sub-runs is incomplete, i.e., the model is not simulated until the end and less than 1000 trips are completed in each run, then there can be two reasons: your model has an error that prevents completion of 1000 trips, or the simulator is set to stop after a specific number of steps before reaching 1000 trips. To check which one is the case, open the “Simulation tool box” and click on the “fast-forward” simulation. The “maximum number of steps” to simulate is highlighted. Set it to a large number, e.g., 500000, and run the simulation once.

  This will set the simulator to stop after 500000 steps (or when there is no transition enabled anymore). If you now run your simulation of 30 sub-runs you should get complete simulation results. If not, your model contains an error. Use manual simulation/fast-forward simulation of 50 steps to find the issue.
- Use the instructions to ask questions about the assignment, but don’t forget to work on the regular exercises.

6 Analyzing Flow Times

CPN Tools allows to automatically simulate your model and to collect statistics about the simulation in a report. In order to properly collect the statistics, your CPN model should have the following options selected (see left and top right in the screenshots below. In case you obtain empty simulation reports, please check these options first); **in particular also check the "Fairness" options**. To start the simulation go to the "Customer" page of your model, right-click on "`CPN'Replications.nreplications 30`" and select "Evaluate ML". See http://cpntools.org for details.



```
▼BaseFile.cpn
    Step: 0
    Time: 0
  ▼Options
    ☑Real Timestamp
    ☑Binding Element Fairness
    ☑Global BE Fairness
     Output directory  : <same as model>
    ▼Performance report statistics
      ▼Simulation performance report
        ▼Timed
          ☑Average
          ☐Confidence intervals for average
          ☑Number of observations
          ☐First value observed
          ☐Last value observed
          ☑Maximum
          ☑Minimum
          ☐Sum of squares
          ☐Sum of squares of deviation
          ☐Standard deviation
          ☐Sum
          ☐Variance
          ☐Time of first update
          ☐Time interval
          ☐Time of last update
        ▼Untimed
          ☑Average
          ☐Confidence intervals for average
          ☑Number of observations
          ☐First value observed
          ☐Last value observed
          ☑Maximum
          ☑Minimum
          ☐Sum of squares
          ☐Sum of squares of deviation
          ☐Standard deviation
          ☑Sum
          ☐Variance
```

```
          ☐Variance
      ▼Replication performance report
          ☑Average
          ☑Confidence intervals for average
          ☐Number of observations
          ☐First value observed
          ☐Last value observed
          ☑Maximum
          ☑Minimum
          ☐Sum of squares
          ☐Sum of squares of deviation
          ☑Standard deviation
          ☐Sum
          ☐Variance
    ▶Extensions
    ▶PP-CPN
  ▶History
  ▶Declarations
  ▶Monitors
```