NOVA SCHOOL OF SCIENCE & TECHNOLOGY

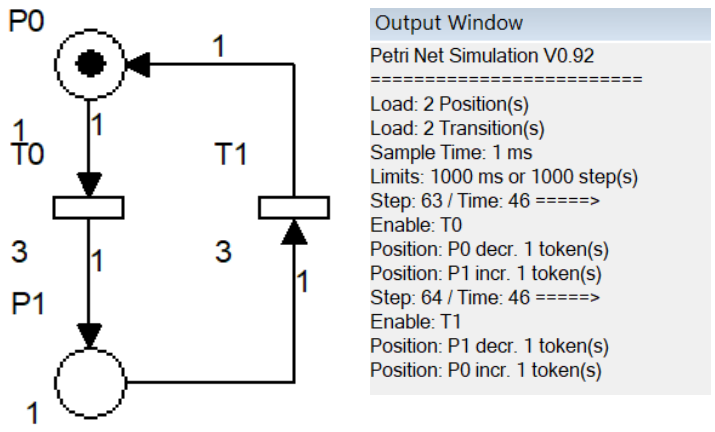
Departamento de Engenharia Eletrotécnica e de Computadores

Secção de Robótica e Manufatura Integrada

STR Lab 3 Report

PETRI NETS CONCEPTS

Secure and Conservative PN

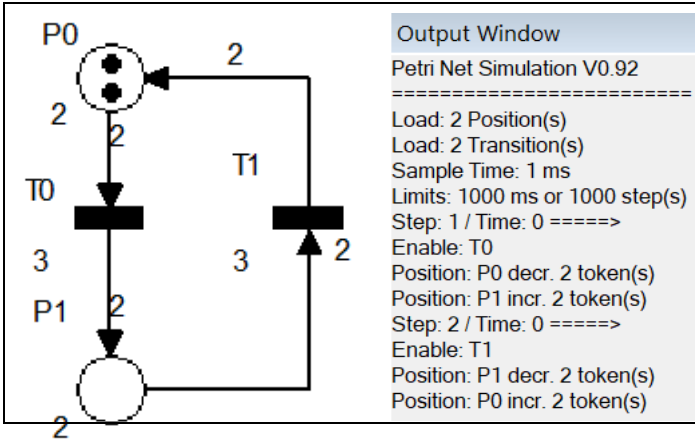


This petri net is secure and conservative.

A place in a Petri net is called **k-bounded if it does not contain more than k tokens in all reachable markings.**

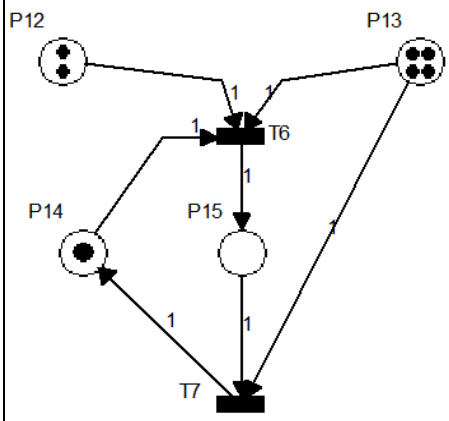
In this case the PN is 1-bounded, which means that it is secure and **conservative, because the total number of tokens is always constant.**

PN with capacity places and weighted arcs



Arcs have capacity 1 by default; if other than 1, the capacity is marked on the arc. Places have infinite capacity by default, and transitions have no capacity, and cannot store tokens at all. **In this case we have a 2 weighted PN with arc connectors with capacity 2 as well.**

PN which starts working and enters in a deadlock state after a few iterations




```
Output Window
Petri Net Simulation V0.92
========================
Load: 4 Position(s)
Load: 2 Transition(s)
Sample Time: 1 ms
Limits: 1000 ms or 1000 step(s)
Warning: 2 potentialy Conflicts found
list:
Transition: T6
Transition: T7
end list
Step: 1 / Time: 0 =====>
Enable: T6
Position: P12 decr. 1 token(s)
Position: P13 decr. 1 token(s)
Position: P14 decr. 1 token(s)
Position: P15 incr. 1 token(s)
Step: 2 / Time: 0 =====>
Enable: T7
Position: P15 decr. 1 token(s)
Position: P13 decr. 1 token(s)
Position: P14 incr. 1 token(s)
Step: 3 / Time: 0 =====>
Enable: T6
Position: P12 decr. 1 token(s)
Position: P13 decr. 1 token(s)
Position: P14 decr. 1 token(s)
Position: P15 incr. 1 token(s)
Step: 4 / Time: 0 =====>
Enable: T7
Position: P15 decr. 1 token(s)
Position: P13 decr. 1 token(s)
Position: P14 incr. 1 token(s)
Step: 5 / Time: 0 =====>
Error: Deadlock at Time: 0 ms, Step: 5
```

This is an example of a deadlock in a PN.

A transition is enabled when the number of tokens in each of its input places is at least equal to the arc weight going from the place to the transition. An enabled transition may fire at any time. When fired, the tokens in the input places are moved to output places, according to arc weights and place capacities. This results in a new marking of the net, a state description of all places.

Having this in mind, we can easily understand that **this PN transforms the first 3 accepted tokens at T6 in only 1. This decrease of tokens results in a deadlock at Step 5.**

PN from TP_9 example

Parts to be processed arrive in conveyor A. The robot picks one part at a time from that conveyor and puts it in machine M for processing (when both the robot and the machine are free). When the machine finishes, the robot picks the processed part and puts it in the exit conveyor B.

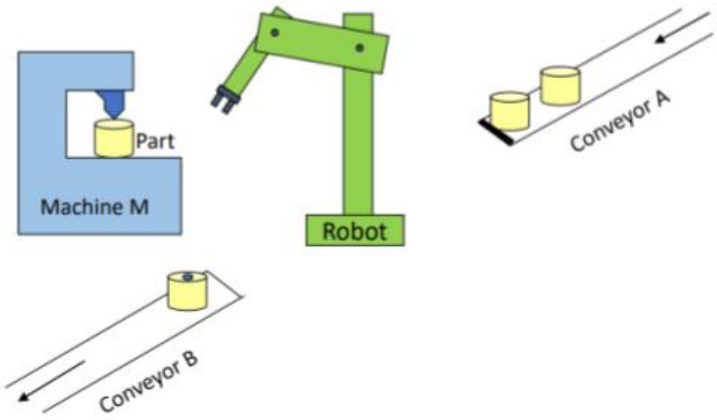
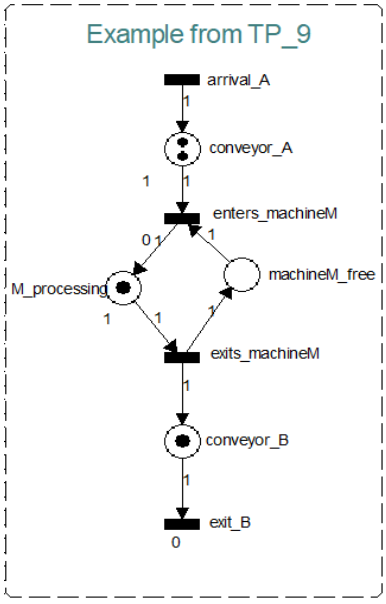


Figure 1 - Machine M and Robot example



PN TO CONTROL THE CARWASH STATION

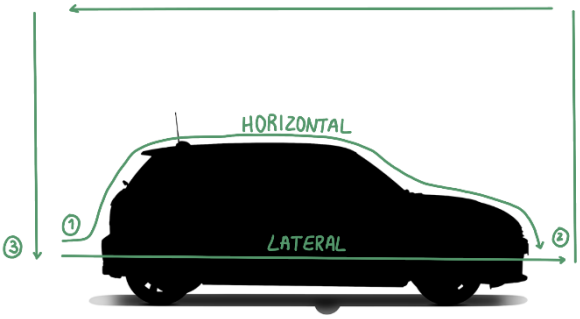
Description of the proposed implementation

Figure 2 - The car wash station kit

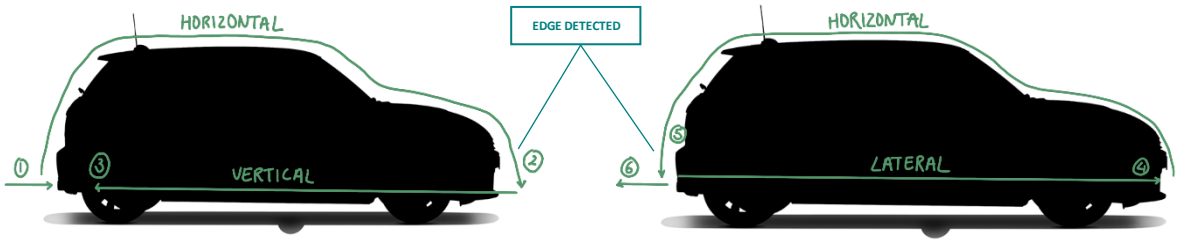
The lab work problem we are going to model consists of the design of a PN model and development of a program to control the car wash station illustrated in Figure 1.

The carwash management program must provide the following services to the customers:

• **Simple wash** - The simple wash program starts from the back of the car - moves right while both horizontal and vertical brushes are on (1) until the right-side sensor is reached (2). Then brushes stop rotating and the structure moves up, left and down in a straight line until reaching the bottom left corner – initial position (3).



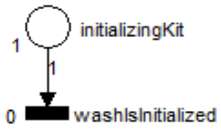
• **Professional wash** - Starting from the back of the car, the wash structure moves right, all brushes are off until we reach the car (1). The horizontal brush is turned on when the left edge of the car is reached, up to the point we reach the right edge of the vehicle (2). Afterwards the wash moves left till the left edge (3) and right till the right edge (4) with lateral brushes on. Then we move left, and the horizontal brush is turned on when we reach the car, up to when the left edge of the car is detected (5). Finally, we keep moving left, with all brushes off, until we arrive at the start position (6) – preparing for the next wash.



• **Mechanism for emergency stop and resume -** This functionality is crucial to prevent real time emergencies or errors, giving the user the option to force stop the workflow and resume it when he choses to. The movement state is saved when the stop button is pressed, and it is written back to port 2 when the resume button is pressed.

Characteristics of PN's in our lab work

Wash Initialization

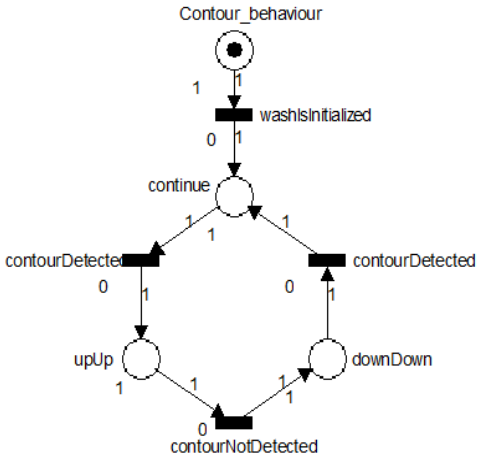


Manual Movement Control

All manual control (with buttons from the provided interface) was completed, including right, left, up and down movement and stop functionality.

All the PN's below in our lab work are 1-bounded | Strictly conservative | Safe | Secure

Vertical Calibration - Contour of the car

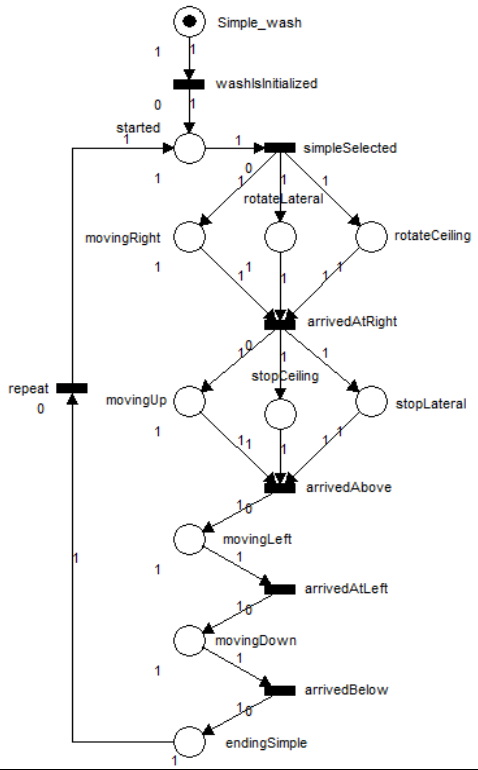


This PN is very important to our lab work, its implementation is pivotal to the flow of both the Simple and Professional wash operations.

The vertical calibration, A.K.A. the **contour of the car runs in parallel with the ongoing wash.** It is responsible for making sure that the car boundaries are respected, so that the car is properly washed on top.

The PN is initialized when the program starts and runs alternating between an Up and Down movement whenever the car is detected or not detected respectively.

Simple wash



The Simple wash follows the description given in the last chapter. This PN **waits until the user presses the Simple Wash button to start executing.**

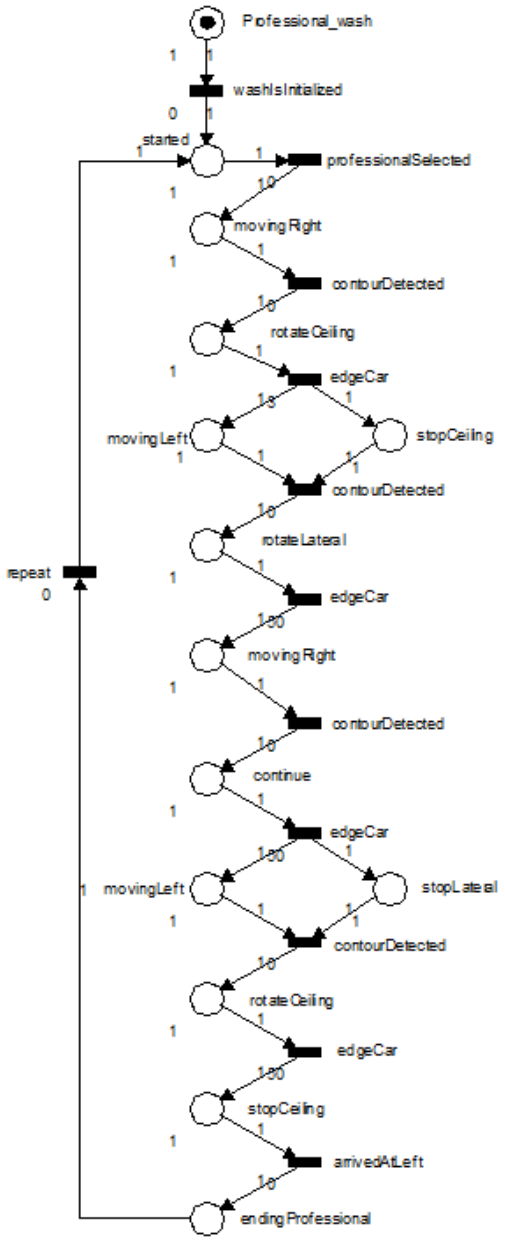
1 - applies 3 movement functions to wash the car once: move right; rotate lateral brushes; rotate top brushes. This movement state runs until the wash structure arrives at the right-side sensor.

2 - applies 3 movement functions to reset and finish the wash operation: move upwards; stop lateral brushes rotation; stop top brushes rotation.

3 - sequentially moves the wash structure until it reaches the bottom left corner, alternating between movement and confirmation of arrival at sensors.

4 - finally, it resends the token to the beginning, so that it is ready for another run.

Professional wash



Following our explanation of the Professional Wash workflow and based on the implementation of the Simple Wash, it's easy to understand this PN even though it is longer and seems more complex.

edgeCar() is a function implemented in VS that **checks if the contour sensor is detecting a car and the down sensor is active** - meaning we reached a right or left edge of the vehicle.

1 - waits until professional wash button is pressed to start

2 - move right until a car is detected - only then the rotation of the upper brush starts until the edge of the car.

3 - after reaching the right edge of the car, it immediately switches movement to the left and stops the upper brush rotation.

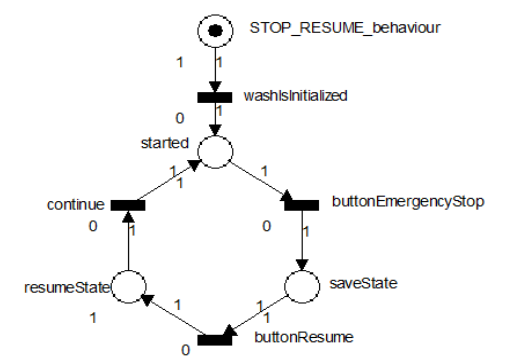
4 - when the car is detected again, lateral brushes start rotating until the left edge of the car is reached, and then return to the right edge of the car (still with the lateral brushes moving).

5 - once the right edge is reached, the car starts moving left and lateral brushes stop their rotation until the car is again detected.

6 - at that point, the upper brush starts moving while the left edge of the car is not reached and only then the brush stops, and the wash structure is reseted at the left sensor.

7 - finally the token is replaced at start so that the PN can run again once the Professional Wash button is pressed.

Emergency stop and resume



The stop and Resume PN is very simple and similar to the Contour of the car cycle. Check **saveState** function at VS.

At first, the PN waits for the emergency stop button to be pressed. At that moment, **the current movement state is saved, and all ongoing movement is suspended.**

Once the resume button is pressed, the move state is read and the previous movement resumes.