Lab - Petri Nets 2018-2019

Installing the Roméo software : Steps 1, 2, and 3 are useless in Room D102.

1. Download the tool **Roméo** à l'adresse : http ://romeo.rts-software.org
2. Open the dmg. Drag **Romeo** in the directory **Application** (The admin password is needed in D102).
3. Drag the directory **examples** in your workspace.
4. Launch **Roméo**. In the menu **Edit-Preferences**, choose your working directory.

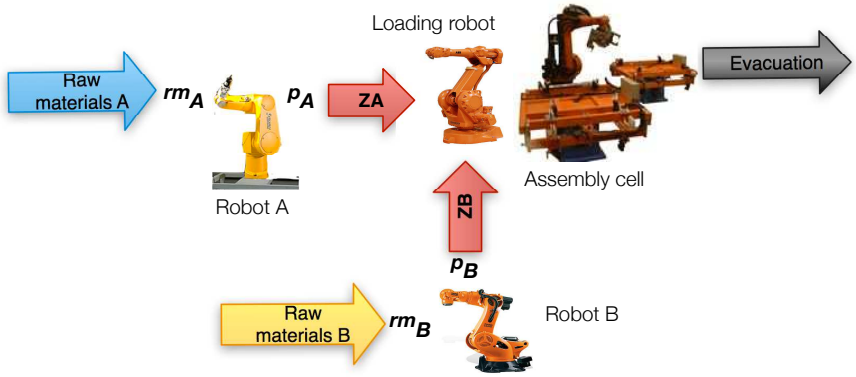
Use of the terminal : The checking of a property use the computation program **romeo-cli**. You model are not necessarily bounded and the computation may not end. In this case :

— In `Applications/Utilitaires`, launch the **Terminal**.
— In the terminal, enter the command `killall romeo-cli` that will delete the process.

Roméo Tutorial : http ://romeo.rts-software.org/doc/tutorial.html

1 Modelling and verification of a production chain.

1.1 Description of the system



Product A : Raw materials A ($\mathbf{rm}_A$) are always available in the area **Raw materials A**. The **robot A** turns raw material A into product A $\mathbf{p}_A$ and drops them in the storage zone **ZA**. The manufacturing time of robot A is between 30 and 40s. The storage capacity of product $\mathbf{p}_A$ in zone **ZA** is 3. When the zone **ZA** is full, the robot A stops until a place becomes available in the zone **ZA**.

Product B : Raw materials B ($\mathbf{rm}_B$) arrive periodically every 60s. The **robot B** turns raw material B into product B $\mathbf{p}_B$ and drops them in the storage zone **ZB** in 30s. The storage capacity of product $\mathbf{p}_B$ in zone **ZB** is only 1. If the robot B drops a product $\mathbf{p}_B$ whereas the zone **ZB** is not empty, the obstruction causes an accident.

Loading : The **loading robot** loads, when the assembly cell is ready, a product $\mathbf{p}_A$ and then a product $\mathbf{p}_B$. It requires 5s to load a product.

Assembling As soon as a product $\mathbf{p}_A$ and a product $\mathbf{p}_B$ are available in the cell, the cell assembles the products and drop the result on the conveyor belt for the evacuation. The assembly cell is then available to receive new products $\mathbf{p}_A$ and $\mathbf{p}_B$. The assembling time is between 20 and 25s.

1.2 Modelling, verification, sizing

Give a model of the system and verify that there is no obstruction in zone **ZB**.
In the following, it is recommended to use parameters in the model.

1. We want to buy a cheaper (and therefore slower) assembly cell. How much can we slow down the assembly cell without this being a problem (without obstruction in zone **ZB**) ?
2. How much can we reduce the period of arrival of raw materials $\mathbf{rm}_B$ in order to accelerate the production without causing obstruction in zone **ZB** ?
3. Using an observer, determine the min and max times between the arrival of a raw material $\mathbf{rm}_B$ and the evacuation of the corresponding products manufactured and assembled.

2 Modelling and verification of a coffee machine.

2.1 A first simple model

The machine can make coffee or tea. The cycle of the machine is i) waiting for a coin, ii) waiting for the choice between coffee or tea, iii) make the coffee or the tea corresponding to the choice.
Give the time Petri Net modelling the coffee machine that meets the following specifications :

- When a coin is inserted, if nothing happens after 10 seconds, it is given back.
- A coin is given back if the user presses the refund button.
- When a piece is present in the machine, if a second piece is inserted, it is given back (refund) immediately.
- When a coin is present in the machine, the user may request a coffee or a tea.
- A coffee is made in 30 seconds. Meanwhile, no action is possible from the user.
- A tea is made in 25 seconds. Meanwhile, no action is possible from the user.
- An action will be possible again only when the cup is actually removed by the user.

Use the Romeo tool to create and validate the model. The actions used will be :

- insertCoin when a coin is inserted,
- startCoffee when preparation of the coffee starts,
- endCoffee when the coffee is ready,

— moneyBack when the customer is refunded.
— ... you define the others

Model the system and explain all the steps (simulation, model-checking) of your validation. Check in particular :

1. functional properties such as : if the customer asks for a coffee and does not ask for the refund then he gets a coffee ...
2. safety properties : bounded, no deadlock ...

2.2 Optimal cost

The model now takes into account the payment.

1. A drink (coffee or tea) costs 40 cts and the machine accepts 5, 10 and 20 cts but does not give change. The overpayment is not memorised for the next cycle. The time between inserting 2 coins is at least $5s$.
2. Once the user has chosen his drink and the preparation of the drink begins, a new cycle can start but the new choice of drink will be possible only when the cup is actually removed by the user.
3. It takes at least 2s for the user to take the cup.

A user wants a coffee and a tea and has 4 coins of 5 cts, 4 coins of 10 cts and 1 coin of 20 cts. Using the *cost* mode of **Romeo**, generate the optimal strategy for this user to get a coffee and a tea in minimal time.