CS 2SD3

Assignment #2. Due March 11 (Monday), 2024, 23:59 via Avenue. Do not hesitate to discuss with TA or instructor all the problems as soon as you discover them. This assignment labour consuming. Start early! Time management is your and only your responsibility.

Instructions: For all assignments, the students must submit their solution to Avenue → Assessments → Assignment #

Students can simply solve the exercises on a paper and use a smartphone app called Microsoft Lens - PDF Scanner and convert their entire solution into a single PDF file and submit it to avenue. The maximum upload file size is 2Gb in avenue for each submission. Please also attach your LTSA and PDF files separately.

Please make sure that the final PDF file is readable.

Students, who wish to use Microsoft word and do not have Microsoft Word on their computer, are suggested to use google document editor (Google Docs). This online software allows you to convert your final file into PDF file.

There will be a mark deduction for not following the submission instruction.

First please finish the assignment on your local computer and then attach your solution as a PDF file.

You will have an unlimited number of submissions until the deadline.

Students must submit their assignments to Avenue. *Any problem with Avenue, please discuss with Sepehr Bayat < bayats1@mcmaster.ca>, the lead TA for this course.*

Total: 102

1.[5] *The saving account problem.* A saving account is shared by several people. Each person may deposit or withdraw funds from the account subject to the constraint that the balance of the account must never become negative.

Develop a model for the problem using FSP processes.

2.[30] Consider machine shop with *K* identical machines. Due to heavy work each machine needs to have two parts, say *part1* and *part2* replaced from time to time by a technician. Each machine either works, or have *part1* replaced, or have *part2* replaced. The *part1* storage has capacity of *M1* parts, and *part2* storage has capacity of *M2* parts. When a storage is empty, a machine that needs this part waits until the technician refills it. If any storage is empty, the technician refills it with either *M1 parts1* or *M2 parts2*. There is only one technician. Refilling must be done as soon as possible, so they have priority over part replacing. There is no partial refilling of the storages. Assume that initially both storages are full.

a.[10] Model the behaviour of the system as FSP processes. Write a safety property that when composed with your system will check if no *part1* is replaced when *part1* storage is empty and vice versa. Compose this property with your system and verify if this it holds.

There is no standard model of priorities in Petri nets, so you have a freedom to define them to fit your solution. This is a comment for points (b), (c) and (d).

b.[5] Model the behaviour of the system as an Elementary Petri net (see Lecture Notes 3).

c.[5] Model the behaviour of the system as a Place/Transition Petri net (see Lecture Notes 9, pages 21, 22).

d.[5] Model the behaviour of the system as a Coloured Petri net (see Lecture Notes 9, pages 23-34).

e.[5] Discuss the differences between the FSP and various Petri Net solutions.

3.[17] A museum allows visitors to enter through the east entrance and leave through its west exit. Arrivals and departures are signalled to the museum controller by the turnstiles at the entrance and exit. At opening time, the museum director signals the controller that the museum is open and then the controller permits both arrivals and departures. At closing time, the director signals that the museum is closed, at which point only departures are permitted by the controller.

For Process Algebra models (as FSP), the museum system consists of processes EAST, WEST, CONTROL and DIRECTOR.

a.[5] Draw the structure diagram for the museum and

b.[5] Provide an FSP description for each of the processes and the overall composition.

c.[7] Model the above scenario with Petri nets (any kind, your choice)

4.[5] Consider the safety property:

```
property P = ( a -> P1 | b -> a -> P1)
         P1 = ( a -> b -> a -> P1 | b -> P )
```

Provide LTS generated by the property *P* and a standard process *SP* such that LTS(*P*)=LTS(*SP*)

5.[10] Provide a Petri nets solution to the Dining Philosophers with a Butler (Lecture Notes 9, pages 14-15). Any kind of Petri net is allowed, however a Coloured Petri Nets give probably the most elegant and simplest solution.

6.[10] Specify a *safety property* for the car park problem of Lecture Notes 7 or Chapter 5 of the textbook, which asserts that the car park does not overflow.
Also specify a *progress property* which asserts that cars eventually enter the car park.
If car departure is *lower priority* than car arrival, does starvation occur?

7.[25] Consider the formulation of Smokers' Problem in plain English given in Lecture Notes 10, pages 5-7. The formulation of Dining Philosophers in the same style is in Lecture Notes 9 on page 7. A straightforward FSP model of Dining Philosophers is presented in Lecture Notes 9 on page 9 ('Hungry Simple Minded Philosophers')

a.[10] Provide a straightforward FSP model of Smokers similar to that of 'Hungry Simple Minded Philosophers'. In principle add supplier to the processes described on page 6 and represent the system using FSP. Use both the compact FSP notation (as upper part of page 9 of LN 9, above the horizontal line) and it expanded version (as lower part of page 9 of LN 9, below the horizontal line). The smoker with for example tobacco could be modelled by the process (but other solutions are also possible):

```
SMOKER_T=( get_paper -> get_match->roll_cigarrette ->
smoke_cigarrette ->SMOKER_T)
```

The resource 'tobacco' could be modelled for example by the process:

```
TOBACCO = ( delivered -> picked -> TOBACCO)
```

etc. If your solution deadlock, provide the shortest trace that lead to the deadlock, if not, provide some arguments why not.

b.[5] Write (safety) *property* process (syntax `property CORRECT_PICKUP = . . .`) that verifies correct sequences of picking resources, i.e. picking up the paper

by the smoker with tobacco must be followed by picking up match by the same smoker, picking up the tobacco by the process with paper must be followed by picking up match by the same smoker, and picking up tobacco by the smoker with matches must be followed by picking the paper by the same smoker.

Then compose `CORRECT_PICKUP` with your solution to (a) above and use the system provided by the textbook to verify if this safety property is violated.

c.[5] An elegant deadlock free solution to the Smokers can be constructed by applying 'ask first, do later' paradigm. Assume that the supplier informs smokers *explicitly* about the ingredient that is *not* supplied, for example it supplies paper, matches and a sign 'no tobacco'. Each smoker reads the sign first and then start picking ingredients only if he has the ingredient that is mentioned on the sign.
Provide this solution using FSPs.

d.[5] Compose your solution from (c) with the property `CORRECT_PICKUP` from (b) and use the system provided by the textbook to verify if this safety property is *not* violated.