CS 2SD3

Assignment #3. Due April 10 (Wednesday), 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.

Due to McMaster regulations (no tests and assignments allowed after April 10), this assignment cannot be extended!

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: 151

1.[10] A lift has a maximum property of ten people. In the model of the lift control system, passengers entering a lift are signalled by an `enter` action and passengers leaving the lift are signalled by an `exit` action.

a.[5] Provide a model of the lift in FSP.

b.[5] Specify a safety property in FSP which when composed with the lift check that the system never allows the lift it controls to have more than ten occupants.

2.[20] The net model considered in Lecture Notes 12 assumed that each process can either read or write. Suppose that we have *three* kind of processes, *readers* that can only read, *writers* that can only write, and *rw-processes*, that can both read and write. Assume we have *m readers*, *k writers* and *n rw-processes*.

a.[10] Provide a Petri nets solution in the style of Lecture Notes 12 for this version of Readers and Writers problem. The solution should be for an **arbitrary** *n*, *m*, and *k*, **not** for any specific values of *n*, *m* and *k* (as for example *n*=3, *m*=2, *k*=3).

b.[10] Prove that your solution is deadlock-free by mimicking the proof of Proposition from page 27 of Lecture Notes 12.

3.[15] Consider an scenario in which a number of producers P communicate with a number of consumers C via a buffer of messages, or asynchronous channel, of capacity B. Each consumer has associated with it a particular id and it must consume only those messages addressed to it. On the other hand, producers produce messages for all of the consumers. Messages produced by producers ought not to be lost. As a particularity, there exists a process MIX that nondeterministicaly engages in either a consumption or a production cycle.

Your task is to model the above scenario in *FSP*,

Moreover, you are asked to specify some properties your model is desired to have, such as:

(a) The model is deadlock free.

(b) Producers produce infinite messages for each consumer.

(c) No producer is blocked indefinitely.

(d) No consumer is blocked indefinitely.

(e) Each consumer only consumes messages addressed to it.

and to guarantee that those properties (and others you may want) are satisfied by both the model and the implementation.

Hints

The overall structure of the system can be modelled in terms of the following processes:

(a) BUFFER: this process can be used to model the buffer of messages. You can use the characterization provided in Chapter 10 of the textbook (or Lecture Notes 14) of an asynchronous buffer of communication. Note however that in FSP the size of a queue must be of a fixed size, hence some considerations should be given to avoiding overflow (or underflow) conditions.

(b) PRODUCER: this process can be regarded as modelling the generation of the messages to every consumer and sending them to the buffer.

(c) CONSUMER: this process can be regarded modelling the receiving of messages from the buffer and consuming them.

(d) MIX: this process can be regarded as modelling the behaviour of the "mix" process.

(e) AMP: this process is just the composition of all of the aforementioned processes (together with the proper synchronization of actions) modelling the behavior of an asynchronous message passing communication.

4.[20] Two warring neighbours are separated by a field with wild berries. They agree to permit each other to enter the field to pick berries, but also need to ensure that only one of them is ever in the field at a time. After negotiation, they agree to the following protocol. When a one neighbour wants to enter the field, he raises a flag. If he sees his neighbour's flag, he does not enter but lowers his flag and tries again. If he does not see his neighbour's flag, he enters the field and picks berries. He lowers his flag after leaving the field.

a.[10] Model this algorithm for two neighbours `n1` and `n2`. Specified the required *safety* properties for the field and check that it does indeed ensure mutually exclusive access. Specify the required *progress* properties for the neighbours such that they both get to pick berries given a fair scheduling strategy. Are any adverse circumstances in which neighbours would not make progress? What if the neighbours are greedy?

b.[10] Model this algorithm for two neighbours using Petri nets (any kind)

5.[10] *Simplified Multidimensional Semaphores* are defined as follows:

(i) The extended primitives *edown* and *eup* are atomic (indivisible) and each operates on a set of semaphore variables which must be initiated with non-negative integer value.

(ii) *edown*($S_1,\ldots,S_n$):
<u>if</u> <u>for</u> <u>all</u> i, $1\le i\le n$, $S_i>0$ <u>then</u> <u>for</u> <u>all</u> i, $1\le i\le n$, $S_i := S_i - 1$
<u>else</u> block execution of calling processes

(iii) *eup*(S_1,...,S_n):
<u>if</u> processes blocked on (S_1,...,S_n) <u>then</u> awaken one of them
<u>else</u> <u>for</u> <u>all</u> i, $1 \le i \le n$, $S_i := S_i + 1$

Model the *Simplified Multidimensional Semaphores* by FSPs for n=2 and maximal value of S_1, S_2 equal to 3.

Hint. *edown*(S_1,S_2) could be interpreted as just a synchronized execution of *down*(S_1) and *down*(S_2). Similarly for *eup*(S_1,S_2

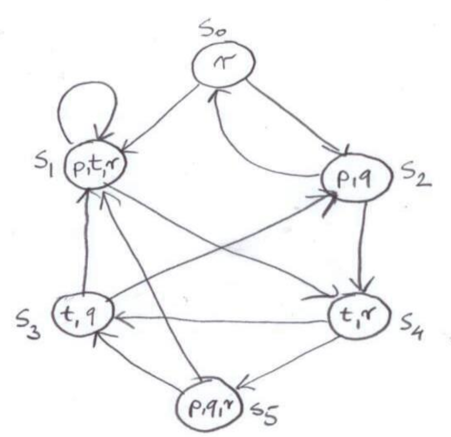
6.[20] A self-service gas station has a number of pumps for delivering gas to customers for their vehicles. Customers are expected to prepay a cashier for their gas. The cashier activates the pump to deliver gas.

a.[10] Provide a model for the gas station with *N* customers and *M* pumps. Include in the model a range for different amounts of payment and that customer is not satisfied (ERROR) if incorrect amount of gas is delivered.

b.[10] Specify and check (with *N*=2, *M*=3) a safety property FIFO (First In First Out), which ensures that customers are served in the order in which they pay.

7.[36] This question deals with Model Checking.

(a)[12] Consider the system *M* defined below:



Determine whether $M, s_0 \models \varphi$ and $M, s_2 \models \varphi$ hold and justify your answer, where φ is the LTL or CTL formula:

(i)[3] $\neg p \Rightarrow r$

(ii)[3] $\neg$ EG r

(iii)[3] E(t U q)

(iv)[3] F q

(b)[8] Express in LTL and CTL: ‘Event *p* precedes *s* and *t* on all computational paths’ (You may find it easier to code the negation of that specification first).

(c)[8] Express in LTL and CTL: ‘Between the events *q* and *r*, *p* is never true but *t* is always true’.

(d)[8] Express in CTL: ‘Φ is true infinitely often along every paths starting at s’. What about LTL for this statement?

8.[20] Consider *Readers-Writers* as described on page 14 of Lecture Notes 12 and analysed in Lecture Notes 12 after page 14. Take the case of three processes and provide a model in LTL or CTL. You have to provide a state machine that defines the model as figures on pages 30 and 33 of Lecture Notes 15 for *Mutual Exclusion*, appropriate atomic predicates as n_1, n_2, t_1, t_2, c_1, c_2 for Mutual Exclusion, and appropriate safety and liveness properties.