CS 2SD3

Assignment #1. Due February 6 (Monday), 2023, 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!

Total: 166 pts

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 JAVA files separately.

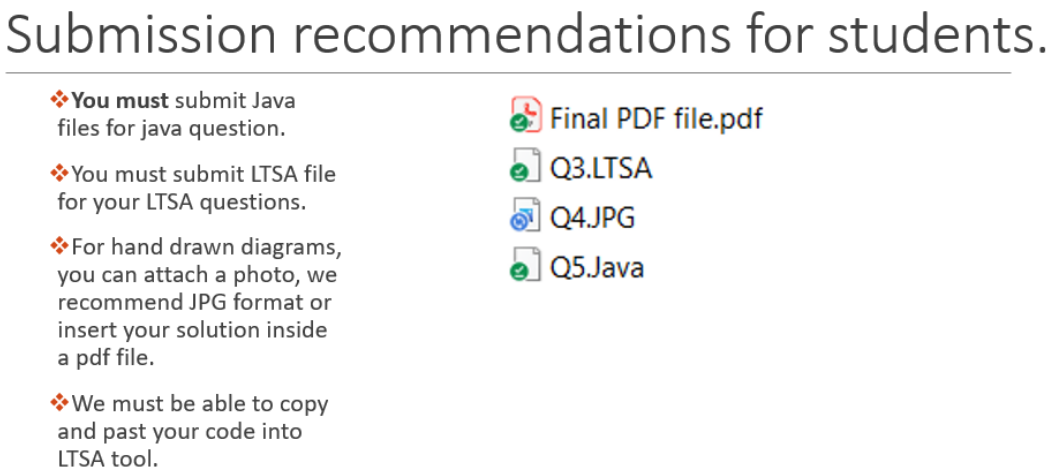
Please make sure that the final PDF file is readable.

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.

Please first finish the assignment on your local computer and at the end, **and attach your solution as a PDF file.**

You will have unlimited number of submissions until the deadline.



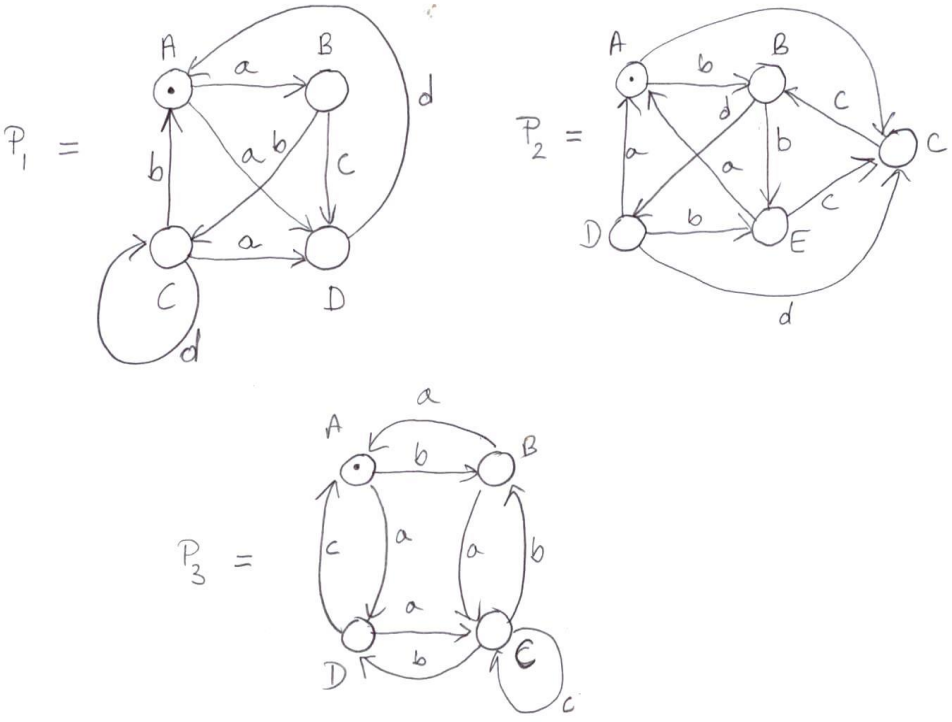
Students must submit their assignments to Avenue. *Any problem with Avenue, please discuss with Mahdee Jodayree <mahdijaf@yahoo.com>, the lead TA for this course.*

1.[10] Consider the following simple hotel reservation system. A customer makes a room request. If room is available, a confirmation is sent to the customer, otherwise the customer is put on a reservation list. If a room is confirmed, the customer may either use it, pay for the room, leave and the whole transaction is archived. However, the customer may also cancel his/her reservation. When the customer is on waiting list, a room may become available, and then a confirmation is sent to a customer. The customer may also give up waiting and cancel his/her request.

Model this reservation system as a FSP process reservation. Note that this process always stops, so you must use the process STOP. Also provide appropriate labelled transition system (use LTSA).

2.[15] a.[9] For each one of the following three processes, give the Finite State Processes (FSP) description of the labelled transition graph. Dots indicate initial states.

b.[6] Use LTSA to transform the solutions to 2.a back into labelled transition systems. Compare the results and discuss differences (if any).



3.[10] A miniature portable FM radio has three controls. An on/off switch turns the device on and off. Tuning is controlled by two buttons scan and reset which operate as follows. When the radio is turned on or reset is pressed, the radio is tuned to the top frequency of the FM band (108 MHz). When scan is pressed, the radio scans towards the bottom of the band (88 MHz). It stops scanning when it locks onto a station or it reaches the bottom (end). If the radio is currently tuned to a station and scan is pressed, then it start to scan from the frequency of that station towards the bottom. Similarly, when reset is pressed the receiver tunes to the top. Model the radio as a *FSP* process RADIO. Also provide an appropriate labelled transition system.
Hint: The alphabet of RADIO is {on, off, scan, reset, lock, end}.

4.[15] Program the radio of Question 3 in Java, complete with graphic display (if you can).

5.[15] A drinks dispending machine charges 15c for can of Sugerola, 20c for a can of SugerolaDiet and 25c for a can of SugerolaSuperDiet. The machine accepts coins with denominations 5c, 10c and 25c and gives changes. Model the machine as an *FSP* process, DRINKS.

6.[15] Consider the following set of FSPs:

A = ((a → (b → A)) | (c → (a → C | c → B)) | (c → C))
B = (b → (a → B | c → (a → A | b → B)))
C = ((a → (b → (c → B))) | (a → C))

a. Construct an equivalent Labelled Transition System using the rules from page 16 of Lecture Notes 2.

b. Use LTSA to derive appropriate LTS, and, if different then yours, analyse and explain differences.

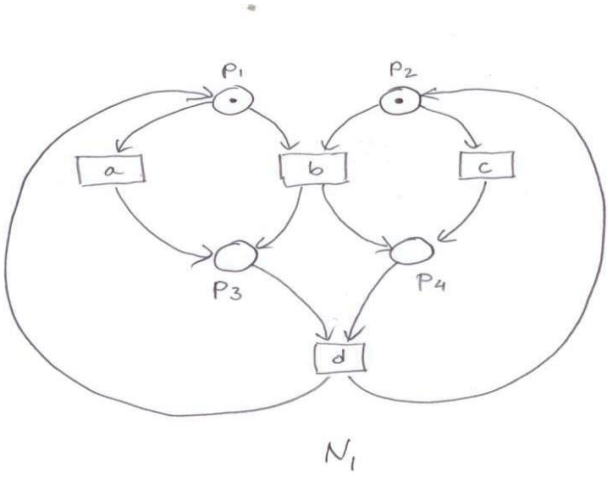
7.[18] a.[8] Show that processes ||*S*1 and *S*2 generate the same Labelled Transition Systems, i.e. LTS(||*S*1) = LTS(*S*2) (or equivalently, they generate the same behaviour)

P = (*a* -> *b* -> *d* -> *P*)
Q = (c -> *b* -> *e* -> *Q*)
|| *S*1 = (*P* || *Q*)

*S*2 = (*a* -> S2A | *c* -> S2B)
S2A = (*c* -> *b* -> *d* -> S2C | *c* -> *b* -> *e* -> S2D)
S2B = (*a* -> *b* -> *d* -> S2C | *a* -> *b* -> *e* -> S2D)
S2C = (*e* -> S2 | *a* -> *e* -> S2A)
S2D = (*d* -> S2 | *c* -> *d* -> S2B)

b.[10] Using a method presented on page 17 of Lecture Notes 3 and pages 10-11 of Lecture Notes 4, transform the processes ||*S*1 and *S*2 into appropriate Petri nets. Are these nets identical? Explain the difference. Which one allows *simultaneity*?

8.[10] Consider a Petri net below:



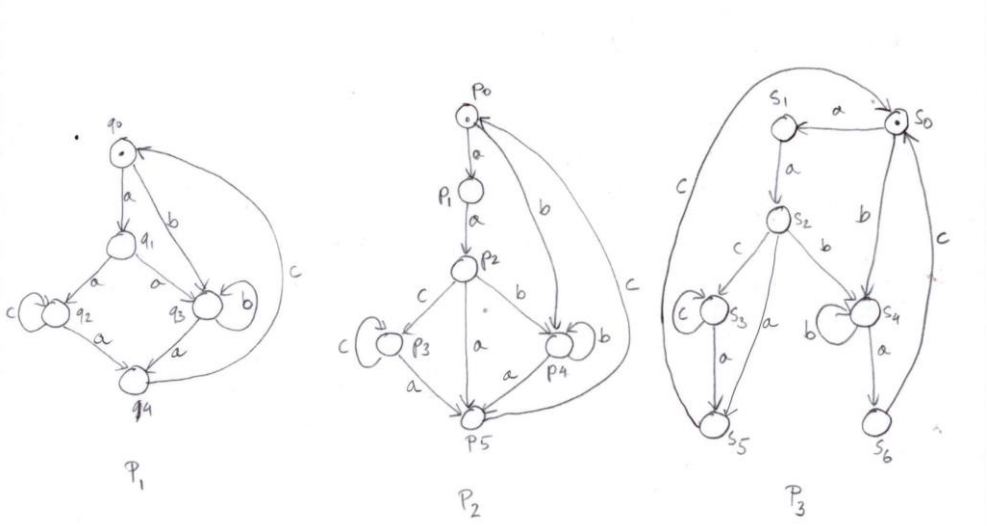
Model the net N_1 as a composition of *FSP* processes.

9.[10] Model the system from page 10 of Lecture Notes 3 as a composition of *FSP* processes. In this case, the entities that are represented by places in the Petri Nets model, must be represented by actions/transitions in *FSP* model.

10.[10] A roller-coaster control system only permits its car to depart when it is full. Passengers arriving at the departure platform are registered with the roller-coaster controller by a turnstile. The controller signals the car to depart when there are enough passengers on the platform to fill the car to its maximum capacity of *M* passengers. Ignore the synchronization detail of passengers embarking from the platform and car departure. The roller-coaster consists of three processes: *TURNSTILE*, *CONTROL* and *CAR*. *TURNSTILE* and *CONTRO*L interact by the shared action *passenger* indicating an arrival and *CONTROL* and *CAR* interact by the shared action *depart* signalling the car departure. Provide FSP description for each process and the overall composition.

11.[10] Construct *reachability graph* (defined on page 18 of Lecture Notes 3) for the Petri net from Question 8.

12.[28] Consider three Labelled Transition Systems (Finite State Machines, Finite Automata) given below: P_1, P_2 and P_3. Tokens represent initial states. Show that:



a.[8] $P_2 \approx P_3$, i.e. P_3 and P_3 are *bisimilar*,

b.[6] $P_1 \not\approx P_2$, i.e. P_1 and P_2 are *not bisimilar*,

c.[6] $P_1 \not\approx P_3$, i.e. P_1 and P_3 are *not bisimilar*,

d.[8] Traces(P_1) = Traces(P_2) = Traces(P_3) = Pref(*give a proper regular expression*).