Cyber-Physical Computation

Last assignment

José Proença and Renato Neves

First task (managing shared resources with Uppaal)

Consider a small private airfield used by 2 planes, which can be either flying, parked, landing, or taking off. The landing field is a resource shared by the two planes. Consider the following requirements:

1. only 1 plane can use the field at a time;
2. a Controller component receives requests to *land* or to *take off*, and replies with a *wait* signal when the field is not available;
3. each plane sends requests to the Controller to *land* or to *take off*, and sends notifications when the field becomes *free*;
4. the Controller has 5 time units to notify a plane to wait;
5. after 5 time units from requesting access to the field and with no wait signal, the planes take another 5 time units to reach the field;
6. each plane takes non-deterministically between 1-3 time units to take off, and between 4-6 time units to land and park;
7. after taking off and after parking the planes notify the Controller with a *gone* signal;
8. if a plane is told to wait, we assume it will take between 5-7 time units to reach the field.

Suggest a Uppaal model for the planes and the controller. List 4 to 8 desired properties that the model should satisfy. Verify the properties via Uppaal.
Extra points: Extend your model to handle n planes at once. Can you think of other useful features that the airfield should have? If so please discuss them and describe how would you model them.

Second task (essay on using the right concepts in software development)

Write a small essay detailing the differences between modelling and verification (as you saw in the first part of the lectures) and programming (as you saw in the second part of the lectures). Discuss as well how they complement each other.
Extra points: Illustrate your explanations with concrete running examples.

Third task (unified program semantics)

A number of 'next-generation' probabilistic programming languages are currently under intensive development [1]. Let us build our own *probabilistic* language in the same spirit than the languages

[1] Take a look for example at Anglican, Gen, and Lazy ppl.

developed in the previous lectures/assignments. Start with the following grammar:

$$\texttt{Prog}(X) \ni \texttt{x} := \texttt{t} \mid \texttt{p} +_p \texttt{q} \mid \texttt{p} \,;\, \texttt{q} \mid \texttt{if b then p else q} \mid \texttt{while b do} \{ \texttt{p} \}$$

It contains a new language construct, namely $\texttt{p} +_p \texttt{q}$ which runs p with probability p and q with probability $1 - p$. Present a semantics for the extended language and implement it in HASKELL using the *monad of probabilities*. Here are some suggestions to help you get started: use the code developed in previous lectures and also the library with the probability monad (available on the website). Regarding the semantics, start with the following rule for sequential composition and then try to derive the others.

$$\frac{\langle \texttt{p}, \sigma \rangle \Downarrow \sum_i^n p_i \cdot \sigma_i \qquad \forall i \leq n. \langle \texttt{q}, \sigma_i \rangle \Downarrow \mu_i}{\langle \texttt{p} \,;\, \texttt{q}, \sigma \rangle \Downarrow \sum_i^n p_i \cdot \mu_i} \text{ (seq)}$$

Extra points: Extend the semantics to handle a selection of your favorite effects, for example delays, log messages, or exceptions. Alternatively, extend the language to handle other probabilistic effects.

What to submit: A single report in PDF for all tasks **and** all the relevant source files. Send by email (nevrenato@gmail.com) a unique zip file "`cpc2223-N1_N2.zip`", where `N1` and `N2` are your student numbers. The subject of the email should be "`cpc2223 N1 N2`".

Deadline: 26th June 2023 @ 23h59