02246 Mandatory Assignment
Assignment 04 - Probabilistic Model Checking [*]

To be submitted on DTU Learn - see deadline on DTU Learn

You are encouraged to work in groups, but you must clearly identify the contributions of each group member, and you will be jointly responsible for the finished report. Register your group on DTU Learn before submitting as group submission.

Answers to all parts should be typed up using LaTeX and submitted electronically as a PDF report using the provided template. Drawings and formulae may be handwritten and scanned. More detailed instructions as to the style of answer we expect for each part are included below.

Some tasks require to upload files.

[*]Thanks to Michael Smith (the original author), and Lijun Zhang, Kebin Zeng, Flemming Nielson, Alberto Lluch Lafuente and Andrea Vandin (contributors).

Assignment 04 - Probabilistic Model Checking

A04P: Practical Problems

A04P.1 *Lottery scheduling* is a scheduling discipline where each task is assigned a number of tickets. To decide which task to execute, the scheduler randomly selects a ticket (under a uniform distribution), and the task owning the ticket is allowed to execute. A lottery scheduler can be either *pre-emptive*, in which a task can only execute for a certain amount of time before being interrupted, or *non pre-emptive* if a task always executes to completion once selected.

a) Construct a PRISM model of a pre-emptive lottery scheduler with three clients, where each task is given a single ticket, and the quantum (the length of time before a task is interrupted) is one time unit. You will have to decide on an appropriate way to model the tickets, which means deciding on an appropriate level of abstraction.

Provide your answer here. Leave the special color (blue). Figures, tables, code snippets can be placed somewhere else but they need to be referred here.

UPLOAD REQUIRED: a prism model file `A04P.1.a.prism`.

b) We are now interested in the time taken for the first task from $\texttt{Client}_1$ to complete. To do so, let's introduce a new module called `Monitor`, which has a single Boolean variable called `finished`. The module should only perform $\texttt{finish}_1$ actions — the first time it does a $\texttt{finish}_1$ it changes its state so that `finished` = `true`, and for subsequent actions it remains in this state.

Provide your answer here. Leave the special color (blue). Figures, tables, code snippets can be placed somewhere else but they need to be referred here.

UPLOAD REQUIRED: a prism model file `A04P.1.b.prism`.

c) Write down a PCTL formula (using the '`P=?`' syntax), expressing the probability that the first task from $\texttt{Client}_1$ completes within t time units — you should define t as an integer constant, without a value. Use the experimentation feature of PRISM (as described in the tutorial), to plot a graph of this probability, for $0 \leq t \leq 20$.

Provide your answer here. Leave the special color (blue). Figures, tables, code snippets can be placed somewhere else but they need to be referred here.

UPLOAD REQUIRED: a prism property file `A04P.1.c.props`.

d) Is it possible for your lottery scheduler to suffer from starvation?

Provide your answer here. Leave the special color (blue). Figures, tables, code snippets can be placed somewhere else but they need to be referred here.

e) Modify your model so that the quantum is two time units. How does this affect the time taken to complete a task from $\texttt{Client}_1$?

Provide your answer here. Leave the special color (blue). Figures, tables, code snippets can be placed somewhere else but they need to be referred here.

UPLOAD REQUIRED: a prism model file `A04P.1.e.prism`.

A04P.2 Rather than assigning the same number of tickets to each task, we can give varying priority to different tasks by assigning them *different* numbers of tickets.

a) By assigning different numbers of tickets to different tasks, approximate an SRT scheduler using lottery scheduling.

   Provide your answer here. Leave the special color (blue). Figures, tables, code snippets can be placed somewhere else but they need to be referred here.

   UPLOAD REQUIRED: a prism model file `A04P.2.a.prism`.

b) How does this affect the time taken to complete a task from $\texttt{Client}_1$? Is starvation possible?

   Provide your answer here. Leave the special color (blue). Figures, tables, code snippets can be placed somewhere else but they need to be referred here.

c) Increase the number of clients in your model until you reach the maximum for which PRISM can still build the underlying DTMC. Compare the time to build the DTMC to the time to run some analysis on it (e.g. a PCTL property).

   Provide your answer here. Leave the special color (blue). Figures, tables, code snippets can be placed somewhere else but they need to be referred here.

d) Investigate what happens if one client tries to perform long jobs, but the others all generate very short jobs. Is the scheduler vulnerable to a denial of service attack?

   Provide your answer here. Leave the special color (blue). Figures, tables, code snippets can be placed somewhere else but they need to be referred here.

e) Can you express denial of service as a PCTL property? Explain your answer.

   Provide your answer here. Leave the special color (blue). Figures, tables, code snippets can be placed somewhere else but they need to be referred here.

A04P.3 In A04P.2 we reached the limits of model checking—here we explore the obscure abyss beyond them by resorting to simulation.

a) Replicate the model from A04P.2 but defining 10 clients, each with tasks of size six (i.e. of type `[0..5]`), and each also having six tickets defined in the same manner. Define a monitor that synchronises with these clients, and lets us observe when *all clients* have finished their tasks. Check how, in its default configuration, PRISM mems-out when trying to build this model. Report the *(i)* runtime and *(ii)* max memory used to reach this failure.

   Provide your answer here. Leave the special color (blue). Figures, tables, code snippets can be placed somewhere else but they need to be referred here.

   UPLOAD REQUIRED: a prism model file `A04P.3.a.prism`.

b) Define a PCTL property using `P=?`, that queries the probability with which all clients may finish their tasks within t time units (the same integer constant defined for A04P.1). Estimate this probability for $t \in \{50, 56, 58, 60, 70, 80, 90, 100\}$ using the SMC capabilities of PRISM in their default configuration, as shown in class. Besides the estimated probabilities, report the width of the confidence interval achieved, and the runtime used to compute this estimate.

   Provide your answer here. Leave the special color (blue). Figures, tables, code snippets can be placed somewhere else but they need to be referred here.

c) Repeat the previous exercise with 10^4 simulations—in the modal dialog that opens when we simulate a property, in the field "Number of samples"—and also 10^5.

Provide your answer here. Leave the special color (blue). Figures, tables, code snippets can be placed somewhere else but they need to be referred here.

d) Using simulation as above, find the probability with which all clients may finish their tasks within $t = 56$ time units (hint: it ain't zero), producing a confidence interval that does not contain zero.

   Provide your answer here. Leave the special color (blue). Figures, tables, code snippets can be placed somewhere else but they need to be referred here.

A04T: Theoretical Problems

A04T.1 Consider the DTMC in Figure 1(a), where the initial state is s_0, and the labels are shown on the states.

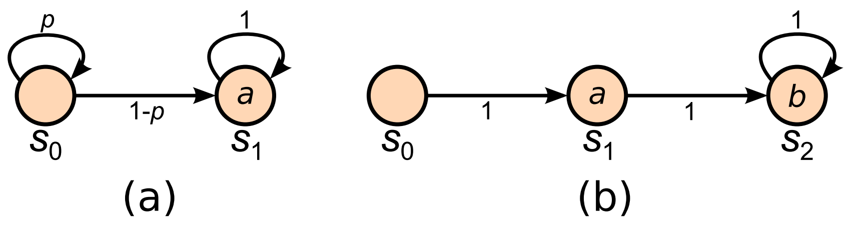

Figure 1: More DTMCs

a) For what values of p is it the case that $s_0 \models \mathbb{P}_{\geq \frac{1}{2}}(F^{\leq 2}\ a)$?

Provide your answer here. Leave the special color (blue). Figures, tables, code snippets can be placed somewhere else but they need to be referred here.

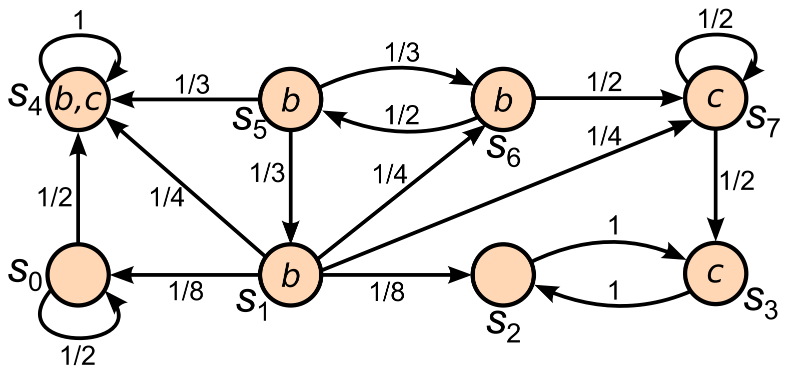

Figure 2: Another DTMC

A04T.2 Consider the DTMC in Figure 2, which we will call $\mathcal{M} = (S, \boldsymbol{P}, s_1, AP, L)$, where $AP = \{ b, c \}$. The initial state is s_1 and the labels are shown on the states. Determine which states of the DTMC satisfy the following PCTL formulae, by following the PCTL model checking algorithm seen in the course:

a) $\mathbb{P}_{\geq \frac{17}{19}}(b\ U\ c)$

Provide your answer here. Leave the special color (blue). Figures, tables, code snippets can be placed somewhere else but they need to be referred here.

b) $\mathbb{P}_{\geq \frac{1}{2}}(X\ \mathbb{P}_{> \frac{1}{3}}((b \vee c)\ U^{\leq 2}\ (b \wedge c)))$

Provide your answer here. Leave the special color (blue). Figures, tables, code snippets can be placed somewhere else but they need to be referred here.

A04T.3 As we have seen in the course, a step-bounded until formula $\Phi_1\ U^{\leq n}\ \Phi_2$ in PCTL is satisfied by a path if Φ_2 holds after at most n time steps, and until then Φ_1 always holds. We would like to generalise this to the form $\Phi_1\ U^{I}\ \Phi_2$, where I is an interval over the natural numbers.

a) Define the semantics $\mathcal{M}, \sigma \models \Phi_1 \; U^I \; \Phi_2$ of this new path formula, where σ is a path in the DTMC $\mathcal{M}$. You have some freedom to define the semantics, and there are several meaningful options.

   Provide your answer here. Leave the special color (blue). Figures, tables, code snippets can be placed somewhere else but they need to be referred here.

b) Consider four cases on the form of the interval I, for $0 < n \leq n'$: $[0, \infty)$, $[0, n]$, $[n, \infty)$ and $[n, n']$. For each case, can you encode the formula $\mathbb{P}_J(\Phi_1 \; U^I \; \Phi_2)$ in terms of the existing PCTL operators? If so, show how, and if not, explain why.

   Provide your answer here. Leave the special color (blue). Figures, tables, code snippets can be placed somewhere else but they need to be referred here.

c) For the cases where you cannot encode the formula, could you encode them in PTCL*?

   Provide your answer here. Leave the special color (blue). Figures, tables, code snippets can be placed somewhere else but they need to be referred here.

d) Design a model checking algorithm for $\mathbb{P}_J(\Phi_1 \; U^{\geq n} \; \Phi_2)$, i.e. $\mathbb{P}_J(\Phi_1 \; U^{[n,\infty)} \; \Phi_2)$. Get inspiration from the algorithms seen in the course for determining $\mathbb{P}_J(\Phi_1 \; U \; \Phi_2)$. Explain the correctness of your proposed algorithm.

   Provide your answer here. Leave the special color (blue). Figures, tables, code snippets can be placed somewhere else but they need to be referred here.

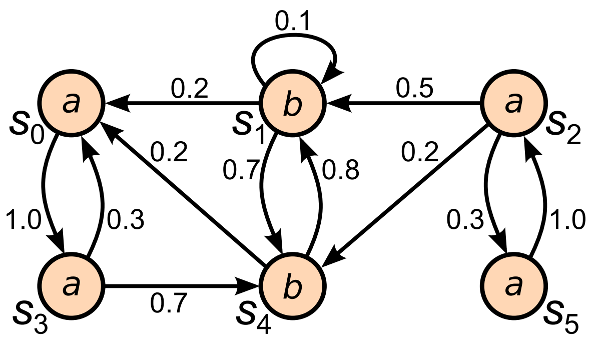

Figure 3: One more DTMC