# Performance Modelling (UG4/MInf/MSc)

Practical 2

24th October, 2014

This is the second of two practicals. The coursework accounts for 25% of the marks for the course, and this practical accounts for 50% of the coursework mark. Each practical is marked out of 50. The deadline for the complete practical is 16:00 on Wednesday 19th November. Submissions should made on paper to the ITO AND electronically, as described, at the end of this document. This is an individual practical: whilst general discussion with classmates is fine, the solutions that you submit must be your own work. The School policy on academic conduct can be found at
`http://www.inf.ed.ac.uk/admin/ITO/DivisionalGuidelinesPlagiarism.html`

This practical is intended to give you some hands-on experience of developing larger models using GSPNs and PEPA. It is recommended that you use the PIPE Tool v4.3.0 (available from `http://sourceforge.net/projects/pipe2/files/PIPEv4.3.0/`) the Eclipse PEPA Plug-in Project (available from `http://www.dcs.ed.ac.uk/pepa/tools/plugin/`). Download and install these on your own computer or in your Informatics DiCE account by following the instructions on the Web sites. For the PEPA tool you will also find it useful to refer to Lecture Note 11.

## 1 Modelling airport security in GSPN

This part of the practical is intended to develop your skills in constructing a Petri net model from a system description. You should think about the events which change the state of the system and associate at least one transition with each of them. There will be two transitions associated with events which have more than one possible outcome.

You are asked to construct a GSPN model representing an airport security check-point. There is a constant supply of passengers arriving at the check-point but only a limited number of passengers are allowed in the check-point at a time. As soon as one passenger leaves the check-point another one is allowed in.

When a passenger enters the check-point, they place their belongings into a large plastic tray or container on a conveyor belt for screening. 75% of passengers can fit all their belongings into a single tray but 25% have so much handbaggage that they need to use an additional tray. Placing items into one tray on the conveyor belt takes approximately $1/d$ seconds; but this time is doubled if two trays are needed. Passengers then walk, at rate $\omega$ through the metal detector. Approximately 1 in 10 passengers are additionally searched, taking $1/s$ seconds on average. Finally the passenger must collect their belongings: picking up just a bag takes approximately $1/c$ seconds, but again this time doubles if two trays were used. The conveyor belt accepts trays and then screens them, taking $1/sc$ seconds on average.

Observation has found the following values for the parameters (all /second):

| | | | |
|---|---|---|---|
| 1/d | 40s | d | 0.025/s |
| 1/w | 10s | w | 0.1/s |
| 1/s | 125s | s | 0.008/s |
| 1/c | 40s | c | 0.025/s |
| 1/sc | 20s | sc | 0.05/s |

You are asked to do the following:

1. Draw a GSPN to represent this system. Each place and transition in your GSPN should be labelled and you should also provide a key indicating the role of each place and transition in representing the system.

2. Assuming that the system is only ever dealing with one passenger at once, give the initial marking for the GSPN and draw the reachability graph.

3. Identify the vanishing markings in the reachability graph and construct the underlying CTMC when these markings are eliminated.

4. Use your model to investigate how the throughput of the checkpoint in terms of passengers per second increases as the number of passengers allowed into the checkpoint simultaneously increases from 2 to 5 (i.e. throughput values for 2, 3, 4 and 5 simultaneous passengers). Do you think that passengers are generally waiting for their bags, or finding their bags ready to pick up? Justify your answer.

Please present your answers to this question clearly, in a printed version to the ITO.

**15 marks**

# 2 Modelling ACPI with PEPA

This part of the practical is intended to develop your skills in constructing PEPA models. Here you are asked to focus on a single component and how it interacts with the given components.

The School of Informatics is committed to achieving continued reduction of its environmental impact. To this end, it has implemented software to automatically put unused computers into a sleep state according to the Advanced Configuration and Power Interface (ACPI) specification. The ACPI specification describes six sleep states of a computer system (S0, ..., S5) in terms of a high-level view of the hardware consisting of a CPU, RAM, a hard disk, and a power supply; and the software consisting of the operating system.

Models of the CPU, power supply, RAM, Disk and operating system have been developed as shown in Figure 1. You will find these process definitions in a file `ACPI.pepa` on the course webpage. The activities *s1_sleep*, ..., *s5_sleep* put the computer into its sleep states. The activities *s1_wake*, ..., *s5_wake* cause it to wake up.

| **State** | **Description** | **Probability** |
|---|---|---|
| S0/Working | System is on. The CPU is fully up and running; power conservation is on a per-device basis. | |
| S1 Sleep | System appears off. The CPU is stopped; RAM is refreshed; the system is running in a low power mode. | From S0 the system goes to S1 30% of the time |
| S2 Sleep | System appears off. The CPU has no power; RAM is refreshed; the system is in a lower power mode than S1. | From S0 the system goes to S2 30% of the time |
| S3 Sleep (Standby) | System appears off. The CPU has no power; RAM is in slow refresh; the power supply is in a reduced power mode. This mode is also referred to as 'Save To RAM'. | From S0 the system goes to S3 20% of the time |
| S4 Hibernate | System appears off. The hardware is completely off, but system memory has been saved as a temporary file onto the hard disk. This mode is also referred to as 'Save To Disk'. | From S0 the system goes to S4 10% of the time |
| S5/Off | System is off. The hardware is completely off, the operating system has shut down; nothing has been saved. Requires a complete reboot to return to the Working state. | From S0 the system goes to S5 10% of the time |

Develop a PEPA component which can ensure that the components which make up the computer system can be put into the above sleep states and successfully awoken from the sleep states, according to the probability distribution given. Assume that the power saving action occurs after 100 minutes and on average machines are woken again after 50 minutes.

Please present your answer to this question as a printout of your .pepa file, handed in to the ITO.

**10 marks**

$$
\begin{aligned}
CPU\_running &\stackrel{def}{=} (stop\_cpu, r_{c_1}).CPU\_stopped \\
&\quad + (power\_down\_cpu, r_{c_2}).CPU\_powered\_down \\
CPU\_stopped &\stackrel{def}{=} (start\_cpu, r_{c_3}).CPU\_running \\
CPU\_powered\_down &\stackrel{def}{=} (power\_up\_cpu, r_{c_4}).CPU\_running
\end{aligned}
$$

$$
\begin{aligned}
PowerSupply\_on &\stackrel{def}{=} (low\_power\_mode, r_{p_1}).PowerSupply\_low \\
&\quad + (lower\_power\_mode, r_{p_2}).PowerSupply\_lower \\
PowerSupply\_low &\stackrel{def}{=} (full\_power\_mode, r_{p_3}).PowerSupply\_on \\
PowerSupply\_lower &\stackrel{def}{=} (full\_power\_mode, r_{p_4}).PowerSupply\_on
\end{aligned}
$$

$$
\begin{aligned}
RAM\_on &\stackrel{def}{=} (slow\_refresh\_ram, r_{r_1}).RAM\_slow \\
&\quad + (power\_down\_ram, r_{r_2}).RAM\_off \\
RAM\_slow &\stackrel{def}{=} (fast\_refresh\_ram, r_{r_3}).RAM\_on \\
RAM\_off &\stackrel{def}{=} (power\_up\_ram, r_{r_4}).RAM\_on
\end{aligned}
$$

$$
\begin{aligned}
Disk\_on &\stackrel{def}{=} (save\_to\_disk, r_{d_1}).Disk\_on \\
&\quad + (restore\_from\_disk, r_{d_2}).Disk\_on \\
&\quad + (power\_down\_disk, r_{d_3}).Disk\_off \\
Disk\_off &\stackrel{def}{=} (power\_up\_disk, r_{d_4}).Disk\_on
\end{aligned}
$$

$$
\begin{aligned}
OS\_on &\stackrel{def}{=} (shut\_down\_os, r_{o_1}).OS\_off \\
OS\_off &\stackrel{def}{=} (reboot, r_{o_2}).OS\_on
\end{aligned}
$$

Figure 1: PEPA component definitions for Question 2

# 3 Modelling a B2B application

This part of the practical involves constructing a more substantial PEPA model and carrying out some analysis to determine the performance of the system represented. It is based on a genuine case study of a bank process undertaken during a European collaborative research project. Read the system description carefully before you start to construct your model. The actions to use in your model have been identified for you.

## 3.1 Overview

A credit portal is a business-to-business (B2B) application where a bank wishes to offer its loan services to business customers. It is important that loan applications are processed efficiently (so that the bank does not lose customers to a rival bank). A high degree of automation is provided to meet this demand for efficient processing. Where human intervention is needed in the decision-making process this too must be driven by a deep sense of urgency in processing the request in a timely manner.

Some businesses which use the services of the bank are long-established customers with a lengthy and well-documented history of financial solidity and probity, together with substantial credit reserves and securities. Modest loan applications from these customers can be directly approved by a *pre-decision* process which validates the content of the loan request and the securities. In such a case it is possible that the decision to lend can be taken entirely by the application of the predefined rules stored on the server and executed by the local rule engine. This is the fastest route to approval of a loan request because it is one which completes entirely without human intervention.

Not all loan requests can be so rapidly approved. Many will require more lengthy scrutiny, evaluation and checking. In these cases a bank clerk will process the credit request. There are several possible next outcomes here. First, the clerk may approve the request but must then forward the credit request to a supervisor who must also approve the request. Second, the clerk may decline the request. Third, the clerk may enter into a negotiation with the customer with the intention of updating the request to reduce the capital requested, or change the terms of repayment. This then initiates another request from the customer which is to be processed in the manner just described.

## 3.2 Description

We begin with a description of the bank's customers and the process which they follow in order to secure a loan from the bank. This presents the customer's view of the process in terms of activities and choices along the way.

The customer's first action is to initiate a loan (*request*). To carry this through they must enter their balance data and securities (*enterData*) and send this to the credit portal with an XBRL upload (*uploadData*). (XBRL is the eXtensible Business Reporting Language.) The customer then waits to see if the request will be approved or declined (the *approve* and *decline* activities respectively, undertaken at the same rate, $r_{inform}$). If the request is approved the customer has no further business and the next waiting customer can be considered. If the request is declined the customer can try again (*reapply*) and will do so with probability $t_0$. If they do not wish to reapply they can yield to the next customer.

The service is a reactive system. It takes no action until the upload of XBRL data is complete. At this point it validates the data by using a validation web service which determines whether or not the balance data is valid (*validateData*). We are not interesting in the processing of invalid data in this scenario and so the next behaviour which we model is passing the valid data to the bank (*sendBank*). The service is then ready to receive the next request.

The relevant business functions of the bank are expressed in the *Bank* component. This documents the "predecision" phase which can have three possible outcomes. Some applications can be immediately approved (with probability $p_0$), and others immediately declined (with probability $p_1$). Some proportion need to be processed by a bank employee (with probability $p_2$) who will either decline the loan (with probability $q_1$), or approve it (with probability $q_0$). Approval requires confirmation, which may be forthcoming (with probability $s_0$) or not.

Values for rates and probabilities can be found in Tables 1 and 2.

| Customer rates | | Service rates | | Bank rates |
|---|---|---|---|---|
| $r_{request} = 3.11944$ | | $r_{validate} = 1.43141$ | | $r_{predecide} = 5.15757$ |
| $r_{enterData} = 0.04667$ | | $r_{sendBank} = 1.53785$ | | $r_{decide} = 0.01221$ |
| $r_{upload} = 0.88424$ | | | | $r_{inform} = 0.45729$ |
| $r_{reapply} = 0.02036$ | | | | |

Table 1: Table of rate values used in the model. All rates are expressed at the granularity of minutes. The reciprocal of the rate gives the mean or expected value of the duration of the activity. Thus, the average time for one bank employee to decide on a loan is about 82 minutes ($1/r_{decide} = 81.90008$).

| Predecision | | Employee decision | | Supervisor decision | | Reapplication |
|---|---|---|---|---|---|---|
| $p_0 = 0.20907$ | | $q_0 = 0.17441$ | | $s_0 = 0.56423$ | | $t_0 = 0.08970$ |
| $p_1 = 0.32075$ | | $q_1 = 0.82559$ | | $s_1 = 0.43577$ | | $t_1 = 0.91030$ |
| $p_2 = 0.47018$ | | | | | | |

Table 2: Table of probability values used in the model. Each column sums to 1.

1. Model the above system using PEPA and the PEPA Eclipse Plug-in.
2. Explain five steps you might take to verify and validate your model.
3. Use your model to estimate the average waiting time for the customer from submitting a request to receiving a response in the form of *approve* or *decline* when there are 8 customers and one bank.
4. Which one action within the bank process would you speed up to improve this waiting time? Justify your answer.

***25 marks***

## Submission instructions

Your submission for this second practical is mostly on paper but your solution to Question 3, part 1) should also be submitted electronically. Please prepare a directory somewhere in your DiCE filespace entitled `PM-Prac2` and put inside this your file `Question3.pepa`. Submit this file for assessment using the following command.

```
submit pm cw2 PM-Prac2
```

You may issue this command multiple times. Later submissions will overwrite earlier ones. The date of submission for assessment purposes will be taken as the date of the last submission.

All other answers should be presented in printed form to the ITO before the deadline.