Getting started with PEPA

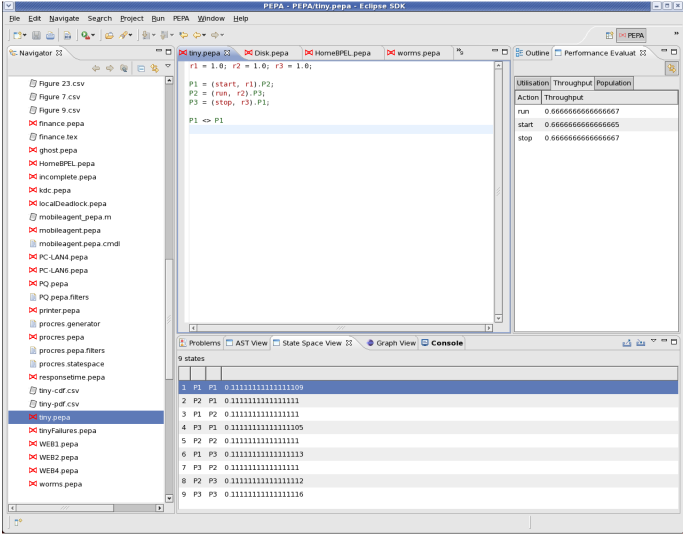
Jane Hillston

LFCS, School of Informatics
The University of Edinburgh
Scotland

9th April 2013

THE UNIVERSITY
of EDINBURGH

The PEPA Eclipse Plug-in processing the model



The PEPA website

`http://www.dcs.ed.ac.uk/pepa`

From the website the PEPA Eclipse Plug-in and some other tools are available for download.

There is also information about people involved in the PEPA project, projects undertaken and a collection of published papers.

Upgrading a PC LAN

We wish to determine the mean waiting time for data packets at a PC connected to a local area network, operating as a token ring.

Upgrading a PC LAN

We wish to determine the mean waiting time for data packets at a PC connected to a local area network, operating as a token ring.

The transmission medium supports no more than one transmission at any given time. To resolve conflicts, a token is passed round the network from one node to another in round robin order.

Upgrading a PC LAN

We wish to determine the mean waiting time for data packets at a PC connected to a local area network, operating as a token ring.

The transmission medium supports no more than one transmission at any given time. To resolve conflicts, a token is passed round the network from one node to another in round robin order.

A node can transmit, only whilst it holds the token.

Upgrading a PC LAN

There are currently four PCs (or similar devices) connected to the LAN in a small office, but the company has recently recruited two new employees, each of whom will have a PC. Our task is to find out how the delay experienced by data packets at each PC will be affected if another two PCs are added.

Modelling Assumptions

- Each PC can only store one data packet waiting for transmission at a time.

Modelling Assumptions

- Each PC can only store one data packet waiting for transmission at a time.
- When the token arrives either there is one packet waiting or no packet waiting.

Modelling Assumptions

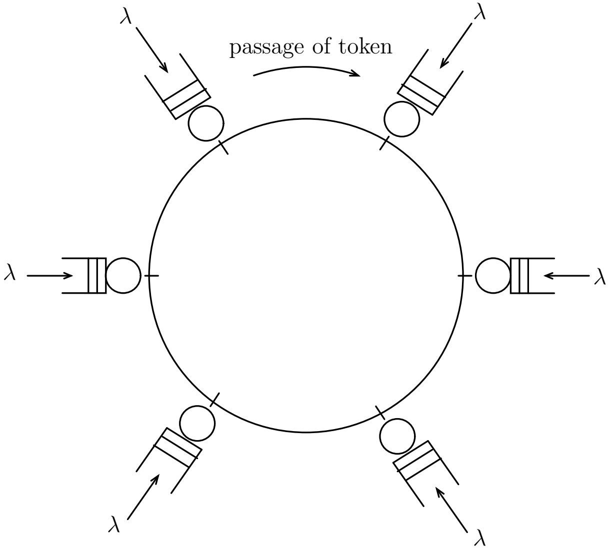
- Each PC can only store one data packet waiting for transmission at a time.
- When the token arrives either there is one packet waiting or no packet waiting.
- The average rate that a PC generates data packets is λ.

Modelling Assumptions

- Each PC can only store one data packet waiting for transmission at a time.
- When the token arrives either there is one packet waiting or no packet waiting.
- The average rate that a PC generates data packets is λ.
- The mean duration of a data packet transmission is d ($d = 1/\mu$), and the mean time for the token to pass from one PC to the next is m ($m = 1/\omega$).

Modelling Assumptions

- Each PC can only store one data packet waiting for transmission at a time.
- When the token arrives either there is one packet waiting or no packet waiting.
- The average rate that a PC generates data packets is λ.
- The mean duration of a data packet transmission is d ($d = 1/\mu$), and the mean time for the token to pass from one PC to the next is m ($m = 1/\omega$).
- Transmission is gated: each PC can transmit at most one data packet per visit of the token.


λ
λ
passage of token
λ
λ
λ
λ

Modelling the system: choosing components

The first stage in developing a model of the system in PEPA is to determine the components of the system and the actions which they can undertake.

Modelling the system: choosing components

The first stage in developing a model of the system in PEPA is to determine the components of the system and the actions which they can undertake.

It seems clear that one type of component should be used to represent the PCs. The components representing the four/six PCs with have essentially the same behaviour. But since token visits the nodes in order we will need to distinguish the components.

Modelling the system: choosing components

The first stage in developing a model of the system in PEPA is to determine the components of the system and the actions which they can undertake.

It seems clear that one type of component should be used to represent the PCs. The components representing the four/six PCs with have essentially the same behaviour. But since token visits the nodes in order we will need to distinguish the components.

We will need another component to represent the medium. As remarked previously, the medium can be represented solely by the token.

Modelling the system: choosing actvities

The description of the PC is very simple in this case. It only has two activities which it can undertake:

- generate a data packet;
- transmit a data packet.

Moreover we are told that it can only hold one data packet at a time and so these activities must be undertaken sequentially.

Modelling the system: choosing actvities

The description of the PC is very simple in this case. It only has two activities which it can undertake:

- generate a data packet;
- transmit a data packet.

Moreover we are told that it can only hold one data packet at a time and so these activities must be undertaken sequentially.

This suggests the following PEPA component for the *i*th PC:

$$
\begin{aligned}
PC_{i0} &\stackrel{\mathit{def}}{=} (\mathit{arrive}, \lambda).PC_{i1} \\
PC_{i1} &\stackrel{\mathit{def}}{=} (\mathit{transmit}_i, \mu).PC_{i0}
\end{aligned}
$$

Modelling the system: choosing actvities

The description of the PC is very simple in this case. It only has two activities which it can undertake:

- generate a data packet;
- transmit a data packet.

Moreover we are told that it can only hold one data packet at a time and so these activities must be undertaken sequentially.

This suggests the following PEPA component for the ith PC:

$$PC_{i0} \stackrel{def}{=} (arrive, \lambda).PC_{i1}$$
$$PC_{i1} \stackrel{def}{=} (transmit_i, \mu).PC_{i0}$$

This will need some refinement when we consider interaction with the token.

Modelling the system: choosing activities

For the token we can think of its current state being characterised by its current position. Thus, if there are N PCs in the network the states of the token correspond to the values $\{1, 2, \ldots N\}$.

Modelling the system: choosing activities

For the token we can think of its current state being characterised by its current position. Thus, if there are N PCs in the network the states of the token correspond to the values $\{1, 2, \ldots N\}$.

When it is at the ith PC then the token may

- transmit a data packet if there is one to transmit and then walk on; or
- walk on at once if there is no data packet waiting.

Modelling the system: choosing activities

For the token we can think of its current state being characterised by its current position. Thus, if there are N PCs in the network the states of the token correspond to the values $\{1, 2, \dots N\}$.

When it is at the ith PC then the token may

- transmit a data packet if there is one to transmit and then walk on; or
- walk on at once if there is no data packet waiting.

$$
\begin{aligned}
Token_i \stackrel{def}{=} & (walkon_{i+1}, \omega).Token_{i+1} + \\
& (transmit_i, \mu).(walk_{i+1}, \omega).Token_{i+1}
\end{aligned}
$$

Refining the components

In order to ensure that the token's choice is made dependent on the state of PC being visited, we add a walkon action to the PC when it is empty, and impose a cooperation between the PC and the Token for both walkon and transmit.

Refining the components

In order to ensure that the token's choice is made dependent on the state of PC being visited, we add a walkon action to the PC when it is empty, and impose a cooperation between the PC and the Token for both walkon and transmit.

$$
\begin{aligned}
PC_{i0} &\stackrel{def}{=} (arrive, \lambda).PC_{i1} + (walkon_2, \omega).PC_{i0} \\
PC_{i1} &\stackrel{def}{=} (transmit_i, \mu).PC_{i0}
\end{aligned}
$$

Complete model: four PC case

$$PC_{10} \stackrel{def}{=} (arrive, \lambda).PC_{11} + (walkon_2, \omega).PC_{10}$$
$$PC_{11} \stackrel{def}{=} (transmit_1, \mu).PC_{10}$$

$$PC_{20} \stackrel{def}{=} (arrive, \lambda).PC_{21} + (walkon_3, \omega).PC_{20}$$
$$PC_{21} \stackrel{def}{=} (transmit_2, \mu).PC_{20}$$

$$PC_{30} \stackrel{def}{=} (arrive, \lambda).PC_{31} + (walkon_4, \omega).PC_{30}$$
$$PC_{31} \stackrel{def}{=} (transmit_3, \mu).PC_{30}$$

$$PC_{40} \stackrel{def}{=} (arrive, \lambda).PC_{41} + (walkon_1, \omega).PC_{40}$$
$$PC_{41} \stackrel{def}{=} (transmit_4, \mu).PC_{40}$$

$$
\begin{aligned}
Token_1 &\stackrel{def}{=} (walkon_2, \omega).Token_2 + (transmit_1, \mu).(walk_2, \omega).Token_2 \\
Token_2 &\stackrel{def}{=} (walkon_3, \omega).Token_3 + (transmit_2, \mu).(walk_3, \omega).Token_3 \\
Token_3 &\stackrel{def}{=} (walkon_4, \omega).Token_4 + (transmit_3, \mu).(walk_4, \omega).Token_4 \\
Token_4 &\stackrel{def}{=} (walkon_1, \omega).Token_1 + (transmit_4, \mu).(walk_1, \omega).Token_1
\end{aligned}
$$

$$
\begin{aligned}
LAN &\stackrel{def}{=} (PC_{10} \parallel PC_{20} \parallel PC_{30} \parallel PC_{40}) \underset{L}{\bowtie} Token_1 \\
&\quad \text{where } L = \{walkon_1, walkon_2, walkon_3, walkon_4, \\
&\qquad transmit_1, transmit_2, transmit_3, transmit_4\}.
\end{aligned}
$$

Here we have arbitrarily chosen a starting state in which all the PCs are empty and the Token is at PC1.