Formal performance modelling: from protocols to people

Nigel Thomas, Michael Harrison, Yishi Zhao, and Xiao Chen
{Nigel.Thomas}@ncl.ac.uk

School of Computing Science, Newcastle University, UK

Abstract. In this paper we consider two very different case studies explored using scalable analysis techniques and stochastic process algebra. The first case study is a classical computer science problem: determining the efficiency of two non-repudiation protocols. We use PEPA to specify the model derived from the protocol specification and mean value analysis and fluid approximation to derive the desired metrics. In the second case study we model a human-centric system, concerning patient flow through a hospital clinic. The model is derived from the clinic practice and observed takt times are used to populate the model. We use PEPA and fluid approximations to derive measures. The two case studies demonstrate the power and versatility of the modelling and analysis approaches used.

1 Introduction

Stochastic process algebra, such as PEPA [1], have been used for around twenty years to formally model and analyse a wide range of computer science applications. The compositional approach to modelling has been demonstrated as being extremely efficient at specifying large models with interactions between many concurrent components. However, with that efficiency in specification has come the need for efficient and scalable analysis techniques.

Much initial work in this area was concerned with decomposing the model for solution [2], for example to derive a product form solution [3, 4]. Work on product form solutions in stochastic process algebra is still taking place, most notably using reversed processes [5, 6], but the class of model amenable to such techniques will always be limited. As a result other techniques were needed and a significant breakthrough came with the application of fluid approximations to biochemical models specified in PEPA [7]. This form of approximation uses ordinary differential equations to solve the model deterministically. Subsequent results have shown that the approximation tends to the exact solution in the limit where the number of components of each type becomes infinite [8]. This limit does not hold in the models in this paper since we have a fixed small number of service centres, however the approximation has been shown to give useful results in a number of previous studies [9–11].

In this paper we present two case studies to illustrate the applicability of the fluid approximation with PEPA. The first case study is a traditional computer

science problem in analysing the performance of two security protocols. This main feature of this problem concerns the scalability of a server faced with requests from a potentially large number of pairs of client processes. The second case study is conceptually quite different, but is equally amenable to the same analysis techniques. In this case we seek to find the throughput of patients through a hospital clinic under various appointment regimes. The aim here is to minimise waiting times for patients whilst maintaining efficient working practices for consultants and other hospital staff.

The rest of the paper is organised as follows. In the next section we give a brief overview of PEPA, followed by the two case studies. In the final section we draw some conclusions and highlight some directions for future work.

2 PEPA

A formal presentation of PEPA is given in [1], in this section a brief informal summary is presented. PEPA, being a Markovian Process Algebra, only supports actions that occur with rates that are negative exponentially distributed. Specifications written in PEPA represent Markov processes and can be mapped to a continuous time Markov chain (CTMC). Systems are specified in PEPA in terms of *activities* and *components*. An activity (α, r) is described by the type of the activity, α, and the rate of the associated negative exponential distribution, r. This rate may be any positive real number, or given as unspecified using the symbol $\top$.

The syntax for describing components is given as:

$$P ::= (\alpha, r).P | P + Q | P/L | P \underset{\mathcal{L}}{\bowtie} Q | A$$

The component $(\alpha, r).P$ performs the activity of type α at rate r and then behaves like P. The component $P + Q$ behaves either like P or like Q, the resultant behaviour being given by the first activity to complete.

The component P/L behaves exactly like P except that the activities in the set L are concealed, their type is not visible and instead appears as the unknown type τ.

Concurrent components can be synchronised, $P \underset{\mathcal{L}}{\bowtie} Q$, such that activities in the cooperation set L involve the participation of both components. In PEPA the shared activity occurs at the slowest of the rates of the participants and if a rate is unspecified in a component, the component is passive with respect to the activities of that type. $A \stackrel{def}{=} P$ gives the constant A the behaviour of the component P. The shorthand $P||Q$ is used to denote synchronisation over no actions, i.e. $P \underset{\emptyset}{\bowtie} Q$. We employ some further shorthand that has been commonly used in the study of large parallel systems. We denote $A[N]$ to mean that there are N instances of A in parallel, i.e. $A|| \ldots ||A$.

In the first case study we only consider models which are cyclic, that is, every derivative of components P and Q are reachable in the model description $P \underset{\mathcal{L}}{\bowtie} Q$. Necessary conditions for a cyclic model may be defined on the component and

model definitions without recourse to the entire state space of the model. In the second case study we employ the notion of terminating components [12]. In this case components enter a *Stop* behaviour

3 Case study: Non-repudiation protocols

A *Key Distribution Centre* (key exchange protocol) has been studied in our previous work, which shows the possibility of modelling by a stochastic process algebra PEPA and analysis by several alternative techniques [9–11]. In this paper, we focus on that how we can apply the modelling and analysis techniques which we developed in the *Key Distribution Centre* study, to two non-repudiation protocols. Firstly, *partial evaluation* [13] has been adopted for model simplification, then we use fluid flow approximations to solve models with large populations.

A non-repudiation service will prevent either of the principals involved from denying the contract after the agreement. The two protocols depicted here were first proposed by Zhou and Gollmann [14, 15] and use a non-repudiation server, known as a *Trusted Third Party* (TTP). We denote these two protocols by ZG1 and ZG3, respectively.

3.1 ZG1 Specification

- A: originator of the non-repudiation exchange
- B: recipient of the non-repudiation exchange
- TTP: on-line trusted third party provide network services accessible to the public
- M: message sent from A to B
- C: ciphertext for message M
- K: message key defined by A
- $NRO = sS_A(f_{NRO}, B, L, C)$: Non-repudiation of origin for M
- $NRR = sS_B(f_{NRR}, A, L, C)$: Non-repudiation of receipt of M
- $sub\_K = sS_A(f_{SUB}, B, L, K)$: proof of submission of K
- $con\_K = sS_T(f_{CON}, A, B, L, K)$: confirmation of K issued by TTP

First, A sends the ciphertext (C) and a non-repudiation origin (NRO) for message M to B, and then B replies back with a non-repudiation receipt (NRR) to A. Now B possesses the ciphertext, but cannot read it as he still hasn't got the key to decrypt M. According to the non-repudiation requirement, B is not a trusted agency to A for sending the key directly to B, they only can resort to a trusted third party (TTP). After receiving the key and proof of submission (sub_K), the TTP will generate a confirmation of K (con_K) and publish in a read only public area. Finally, B can get the key from this public area to decrypt ciphertext (C) and A fetches the confirmation of submission as non-repudiation evidence.

3.2 ZG3 specification

- L: a unique label chosen by TTP to identify the message M
- T_s : the time that TTP received A's submission
- T_d : the time that TTP delivered and available to B
- $NRO = sS_A(f_{NRO}, TTP, B, M)$: non-repudiation of origin for M
- $NRS = sS_D(f_{NRS}, A, B, T_s, L, NRO)$: non-repudiation of submission of M
- $NRR = sS_B(f_{NRR}, TTP, A, L, NRO)$: non-repudiation of receiving a message labelled L
- $NRD = sS_D(f_{NRD}, A, B, T_d, L, NRR)$: non-repudiation of delivery of M

ZG1 describes a non-repudiation protocol with minimized involvement of a trusted third party, acting as a "low weight notary". However, timing evidence of sending and receiving is required in some applications; hence ZG3 can be adopted in this situation. A sends the plaintext (M) and a non-repudiation origin (NRO) to the trusted third part (TTP), and then fetches the time of receiving (T_s) and non-repudiation of submission (NRS) from a public area, after TTP has published this information. The TTP tells B it received M from A by sending the NRO. B generates a non-repudiation of receiving for TTP following. Finally, B and A can fetch M and the time of delivery (T_d), with other non-repudiation evidence, from the public area, after the TTP has published.

$$
\begin{aligned}
(request)\ & 1.A \rightarrow TTP :\ f_{NRO}, TTP, B, M, NRO \\
(response\& & \\
getByA1)\ & 2.A \leftrightarrow TTP :\ f_{NRS}, A, B, T_s, L, NRS \\
(response)\ & 3.TTP \rightarrow B :\ A, L, NRO \\
(sendTTP)\ & 4.B \rightarrow TTP :\ f_{NRR}, L, NRR \\
(response\& & \\
getByB)\ & 5.B \leftrightarrow TTP :\ L, M \\
(response\& & \\
getByA2)\ & 6.A \leftrightarrow TTP :\ f_{NRD}, T_d, L, NRR, NRD
\end{aligned}
$$

3.3 ZG1 PEPA Model

We begin by forming components of a pair of principals A and B.

$$TTP \stackrel{def}{=} (publish, r_p).TTP$$

$$
\begin{aligned}
A0 &\stackrel{def}{=} (sendB, r_b).A1 \\
A1 &\stackrel{def}{=} (sendA, r_a).A2 \\
A2 &\stackrel{def}{=} (sendTTP, r_t).A3 \\
A3 &\stackrel{def}{=} (publish, r_p).A4 \\
A4 &\stackrel{def}{=} (geyByA, r_{ga}).A5 \\
A5 &\stackrel{def}{=} (work, r_w).A0
\end{aligned}
$$

$$B0 \stackrel{def}{=} (sendB, r_b).B1$$
$$B1 \stackrel{def}{=} (sendA, r_a).B2$$
$$B2 \stackrel{def}{=} (publish, r_p).B3$$
$$B3 \stackrel{def}{=} (getByB, r_{gb}).B4$$
$$B4 \stackrel{def}{=} (work, r_w).B0$$

$$SystemZG1 \stackrel{def}{=} TTP[K] \underset{publish}{\bowtie} (A0 \underset{\mathcal{L}}{\bowtie} B0)[N]$$

Where, $\mathcal{L} = \{sendB, sendA, work\}$.

In order to simplify the model specification and analysis, we combine A and B into a new component called AB, using a process referred to as *partial evaluation* [13]. This gives rise to the following description for the complete system when there are N pairs of principals.

$$TTP \stackrel{def}{=} (publish, r_p).TTP$$
$$AB_0 \stackrel{def}{=} (sendB, r_b).AB_1$$
$$AB_1 \stackrel{def}{=} (sendA, r_a).AB_2$$
$$AB_2 \stackrel{def}{=} (sendTTP, r_t).AB_3$$
$$AB_3 \stackrel{def}{=} (publish, r_p).AB_4$$
$$AB_4 \stackrel{def}{=} (getByA, r_{ga}).AB_5 + (getByB, r_{gb}).AB_6$$
$$AB_5 \stackrel{def}{=} (getByB, r_{gb}).AB_7$$
$$AB_6 \stackrel{def}{=} (getByA, r_{ga}).AB_7$$
$$AB_7 \stackrel{def}{=} (work, r_w).AB_0$$
$$SystemZG1 \stackrel{def}{=} TTP[K] \underset{publish}{\bowtie} AB_0[N]$$

AB_0 to AB_7 in the above ZG1 PEPA model denote the different behaviours of the AB component, and its evolution along the sequence of prescribed actions in the protocol. The choice from AB_4 to AB_5 and AB_6 means step 4 and step 5 in ZG1 can happen in any order. The *work* action is used to define that B can do something with the key and ciphertext after he has obtained these, before returning to the state AB_0 to make a new request again, which forms a working cycle to investigate the steady state.

3.4 ZG3 PEPA Model

Once again we begin by defining the behaviour of a pair of principals.

$$TTP \stackrel{def}{=} (response, r_p).TTP$$

$$A0 \stackrel{def}{=} (request, r_{t1}).A1$$
$$A1 \stackrel{def}{=} (response, r_p).A2$$
$$A2 \stackrel{def}{=} (getByA1, r_{ga1}).A3$$
$$A3 \stackrel{def}{=} (response, r_p).A4$$
$$A4 \stackrel{def}{=} (sendTTP, r_{t2}).A5$$
$$A5 \stackrel{def}{=} (response, r_p).A6$$
$$A6 \stackrel{def}{=} (getByA2, r_{ga2}).A7$$
$$A7 \stackrel{def}{=} (work, r_w).A0$$

$$B0 \stackrel{def}{=} (response, r_p).B1$$
$$B1 \stackrel{def}{=} (getByA1, r_{ga1}).B2$$
$$B2 \stackrel{def}{=} (response, r_p).B3$$
$$B3 \stackrel{def}{=} (sendTTP, r_{t2}).B4$$
$$B4 \stackrel{def}{=} (response, r_p).B5$$
$$B5 \stackrel{def}{=} (getByB, r_{gb}).B6$$
$$B6 \stackrel{def}{=} (work, r_w).B0$$

$$SystemZG3 \stackrel{def}{=} TTP[K] \underset{response}{\bowtie} (A0 \underset{\mathcal{L}}{\bowtie} B0)[N]$$

Where, $\mathcal{L} = \{getByA1, sendTTP, work\}$.

As before these are combined to form the merged component AB in the description of the complete system.

$$TTP \stackrel{def}{=} (response, r_p).TTP$$
$$AB_0 \stackrel{def}{=} (request, r_{t1}).AB_1$$
$$AB_1 \stackrel{def}{=} (response, r_p).AB_2$$
$$AB_2 \stackrel{def}{=} (getByA1, r_{ga1}).AB_3$$
$$AB_3 \stackrel{def}{=} (response, r_p).AB_4$$
$$AB_4 \stackrel{def}{=} (sendTTP, r_{t2}).AB_5$$
$$AB_5 \stackrel{def}{=} (response, r_p).AB_6$$
$$AB_6 \stackrel{def}{=} (getByB, r_{gb}).AB_7$$
$$+(getByA2, r_{ga2}).AB_8$$
$$AB_7 \stackrel{def}{=} (getByA2, r_{ga2}).AB_9$$
$$AB_8 \stackrel{def}{=} (getByB, r_{gb}).AB_9$$
$$AB_9 \stackrel{def}{=} (work, r_w).AB_0$$

$$SystemZG3 \stackrel{def}{=} TTP[K] \underset{response}{\bowtie} AB_0[N]$$

The PEPA model of ZG3 has a similar structure to that for ZG1. The main difference is the TTP component in ZG3 should respond three times for different requests in one cycle, which increases the difficulty of modelling and analysis.

3.5 ODE analysis

ODE analysis is an approximate analysis technique based on the solution of coupled ordinary differential equations (ODEs), first applied to stochastic process algebra by Hillston [7]. In this style of model analysis, the model is expressed as a finite number of replicated components and ODEs which represent the flow between behaviours of the components. Thus, by solving the ODEs, it is possible to count the number of components behaving as a given derivative at any given time, t. In the absence of oscillations, the limit, $t \longrightarrow \infty$, then tends to a steady state value.

It is important to note that the ODE approach transforms the original stochastic discrete event system to a deterministic continuous system. In doing so, we consider fractions of any component behaving in some way at any given time, which may be difficult to interpret in a physical system. Furthermore, ODE analysis is only applicable to certain classes of model. Despite these restrictions, the technique is extremely useful when considering very large numbers of components.

In experiments we have performed with different models, we have observed that the ODEs give good predictions of the steady state behaviour only when there is at most one active minimum function [16]. This condition holds for the models considered here as there is only one type of Trusted Third Party.

The results we obtain are not exact, but converge on the true value as the number of customers increases. There is a point of maximum error, the location of which we can predict by deriving the point at which the two sides of the minimum function coincide.

The ODEs for ZG1 and ZG3 can be derived following the approach of Hillston [7].

ODEs of ZG1:

$$\begin{aligned}
\frac{d}{dt}AB_0 &= r_w AB_7(t) - r_b AB_0(t) \\
\frac{d}{dt}AB_1 &= r_b AB_0(t) - r_a AB_1(t) \\
\frac{d}{dt}AB_2 &= r_a AB_1(t) - r_t AB_2(t) \\
\frac{d}{dt}AB_3 &= r_t AB_2(t) - r_p min(AB_3(t), TTP(t)) \\
\frac{d}{dt}AB_4 &= r_p min(AB_3(t), TTP(t))
\end{aligned}$$

$$
\begin{aligned}
&-r_{ga}AB_4(t) - r_{gb}AB_4(t) \\
\frac{d}{dt}AB_5 &= r_{ga}AB_4(t) - r_{gb}AB_5(t) \\
\frac{d}{dt}AB_6 &= r_{gb}AB_4(t) - r_{ga}AB_6(t) \\
\frac{d}{dt}AB_7 &= r_{gb}AB_5(t) + r_{ga}AB_6(t) - r_w AB_7(t) \\
\frac{d}{dt}TTP &= 0
\end{aligned}
$$

ODEs of ZG3:

$$
\begin{aligned}
\frac{d}{dt}AB_0 &= r_w AB_9(t) - r_{t1}AB_0(t) \\
\frac{d}{dt}AB_1 &= r_{t1}AB_0(t) - [r_p \frac{AB_1(t)}{AB_1(t) + AB_3 t + AB_5(t)} \\
&\quad \times min(AB_1(t) + AB_3(t) + AB_5(t), TTP(t))] \\
\frac{d}{dt}AB_2 &= -r_{ga1}AB_2(t) + [r_p \frac{AB_1(t)}{AB_1(t) + AB_3 t + AB_5(t)} \\
&\quad \times min(AB_1(t) + AB_3(t) + AB_5(t), TTP(t))] \\
\frac{d}{dt}AB_3 &= r_{ga1}AB_2(t) - [r_p \frac{AB_3(t)}{AB_1(t) + AB_3 t + AB_5(t)} \\
&\quad \times min(AB_1(t) + AB_3(t) + AB_5(t), TTP(t))] \\
\frac{d}{dt}AB_4 &= -r_{t2}AB_4(t) + [r_p \frac{AB_3(t)}{AB_1(t) + AB_3 t + AB_5(t)} \\
&\quad \times min(AB_1(t) + AB_3(t) + AB_5(t), TTP(t))] \\
\frac{d}{dt}AB_5 &= r_{t2}AB_4(t) - [r_p \frac{AB_5(t)}{AB_1(t) + AB_3 t + AB_5(t)} \\
&\quad \times min(AB_1(t) + AB_3(t) + AB_5(t), TTP(t))] \\
\frac{d}{dt}AB_6 &= -r_{gb}AB_6(t) \\
&\quad -r_{ga2}AB_6(t) + [r_p \frac{AB_5(t)}{AB_1(t) + AB_3 t + AB_5(t)} \\
&\quad \times min(AB_1(t) + AB_3(t) + AB_5(t), TTP(t))] \\
\frac{d}{dt}AB_7 &= r_{gb}AB_6(t) - r_{ga2}AB_7(t) \\
\frac{d}{dt}AB_8 &= r_{ga2}AB_6(t) - r_{gb}AB_8(t) \\
\frac{d}{dt}AB_9 &= r_{ga2}AB_7(t) + r_{gb}AB_8(t) - r_w AB_9(t) \\
\frac{d}{dt}TTP &= 0
\end{aligned}
$$

These ODEs can be solved in a number of ways. Most commonly they are simulated with a suitably small time step over a long period. This gives rise to a trace of component numbers over time. Alternatively, if we assume that a steady state exists, the ODEs can be solved analytically at the limit by taking $\frac{d}{dt}AB_i = 0$, $\forall i$, and solving the resultant set of simple simultaneous equations. Either approach gives an efficient numerical computation, even when N is extremely large.

Our analysis is interested primarily in the number of clients waiting for a *publish* (or *response* in ZG3) action from the TTP, as the clients can then fetch what they need from the public area or obtain a service results. This is represented in the model by the number of AB_3 in ZG1, AB_1, AB_3 and AB_5 in ZG3. The average queuing length $L(N)$ is the number of requests awaiting a response from the TTP. It is the number of the AB_3 (in ZG1), or AB_1, AB_3 and AB_5 (in ZG3), derivatives when $t \longrightarrow \infty$ when there are N customers in the population.

The average response time is another interesting metric for us. To obtain this we apply the arrival theorem. If an arriving request sees a free server, then the average response time will be the average service time. However, if the random observer sees all the servers busy, then the average response time will be the average service time plus the time it takes for one server to become available (including scheduling the other jobs waiting ahead of the random observer). This gives rise to the following equations.

$$
\begin{aligned}
W(N) &= \frac{1}{r_p}\ ,\ L(N-1)+1 \leq K \\
W(N) &= \frac{1}{r_p} + \frac{L(N-1)+1-K}{Kr_p} \\
&= \frac{L(N-1)+1}{Kr_p}\ ,\ L(N-1)+1 > K
\end{aligned}
$$

Obviously, as the TTP in ZG1 is designed as a "low weight notary", the number of waiting requests at TTP of ZG1 should always be smaller than that in ZG3 with the same parameters. However, a system engineer should clearly be very careful to choose either of these two protocols based on the trade off between performance and the need for added security functionality.

4 Case study 2: A rheumatology clinic

Healthcare is subject to many performance targets and metrics used to assess the success of all clinical environments. In addition the notion of patient experience remains at the centre of all clinical operation. As such the provisioning of healthcare resources is constrained not only in providing an efficient service, but also one which meets the needs of the patients, not just clinically but also personally and socially. Important aspects in patient experience include the minimisation

of waiting time and the availability of appropriate information concerning future interactions.

In this study we look at one aspect of clinical performance, namely the throughput of patients and their associate waiting times. We model a rheumatology clinic in a major NHS hospital in the UK using data captured from observations. The clinic is relatively small, consisting of a central waiting area with registration, a number of consulting rooms and treatment rooms where nurses my take blood samples or administer injections. In addition patients may be sent to a separate x-ray service outside the clinic (but still within the hospital). Patients are classified as either *new*, meaning that they have just been referred to the clinic and this is their first appointments, or *follow-up*, which denotes that the patients is attending a repeat appointment having previously attended the clinic in the past. For brevity in this presentation we will not distinguish these two classes, although in practice it adds only a little additional complexity to the specification and analysis.

A typical PEPA model of this system can be specified as follows:

$$
\begin{aligned}
Patient &\stackrel{def}{=} (arrive, rA).Register \\
Register &\stackrel{def}{=} (register, rR).Test \\
Test &\stackrel{def}{=} (test, rT).Consultation \\
Consultation &\stackrel{def}{=} (consult, rC).BloodTest \\
BloodTest &\stackrel{def}{=} (blood, rB).Xray \\
XRay &\stackrel{def}{=} (xray, rX).Depart \\
Depart &\stackrel{def}{=} (depart, rD).Stop
\end{aligned}
$$

$$
Registration \stackrel{def}{=} (register, rR).Registration
$$

$$
Nurse_1 \stackrel{def}{=} (test, rT).Nurse_1
$$

$$
Consultant \stackrel{def}{=} (consult, rC).Consultant
$$

$$
Nurse_2 \stackrel{def}{=} (blood, rB).Nurse_2
$$

$$
XRay \stackrel{def}{=} (xray, rX).XRay
$$

The complete system can be described as

$$
Patient[N_p] \underset{\mathcal{L}}{\bowtie} (Registration||Nurse_1[N_1]||Consultant[N_c]||Nurse_2[N_2]||XRay)
$$

where $\mathcal{L} = \{register, test, consult, blood, xray\}$.

The components $Registration$, $Nurse_1$, $Consultant$, $Nurse_2$ and $XRay$ represent the various resources in the system. Patients can only use those resources if they are available, i.e. not already in use by another patient. A key consideration therefore is in provisioning sufficient resource (denoted by N_1, N_c and N_2) such that patients do not experience excessive waiting given a particular volume, N_p. Since the $Patient$ component is terminating (indicating that the patient has left the clinic), it makes no sense to derive steady state metrics. We could, if it was desired, alter the behaviour so that patients return to the $Patient$ behaviour after a suitable delay following the *depart* action (as in the previous case study).

Using this model we can investigate a number of scenarios of resource availability and appointment scheduling. To do this we derive the ODEs (as in the previous case) and simulate them to derive transient metrics of interest. The particular metrics we are interested in are typically the number of patients completing their appointment within a given time, the time taken for all patients to complete, the maximum number of waiting patients at any given point and the maximum end to end response time for any patient.

As the model is specified here all the patients will begin to arrive at the same time. It turns out that this is the optimal solution for minimising the time taken for all patients to complete, hence form a process-centric view this might be thought of as a good solution. However, in this configuration some patients would be present in the clinic for the entire time, experiencing long waiting times at each stage. Other patients (those at the front of the arrival queue as dictated by the race condition on the concurrent *arrive* actions) would find each resource relatively unused as they come to it and so would experience a very fast response time.

This disparity in patient experience would clearly lead to dissatisfaction amongst those patients with very long waits. To counter this problem we have experimented with various functional rates to stagger the arrival of patients and simulate an appointment schedule. The simplest such function is to spread the arrivals over a longer period, allowing the first patients to progress through the system before other patients arrive. This has the desired effect of reducing maximum wait times, but increases the overall time to completion and reduces the average utilisation of the resources. In practice this is particularly problematic for consultants, who may then experience wait times between patients leading to an inefficient (and more costly) provision. The optimal function to reduce wait times and maintain a high utilisation of key resources is to employ a non-linear function which creates an initial load (high burst of arrivals), and then a spread of patients arriving throughout the remaining period. When the resource provision and rates in the arrival function are optimised this results in small average waits for patients at the $Consultation$ phase, but otherwise a fast response through the system. The small numbers of waiting patients at the $Consultation$ phase ensures that there is nearly always a patient to be seen by the consultant when they are available, so utilisation of consultants is maintained until all patients have been seen.

Using the fluid approximation we are able to introduce further functions to fluctuate the available resource (for example, to introduce scheduled breaks or unforeseen complications) or to reallocate resource between different areas. For example, with a functional rate approach to arrivals it is advantageous to process the initial burst of patients through the *test* action as quickly as possible so that the system reaches a sustainable state. This means provisioning a higher number of $Nurse_1$ components (N_1) in the initial phase, but this number can be reduced after the initial burst has gone through. At the same time during the initial phase there is less demand for later actions in the sequence, hence the number of $Nurse_2$ components can be reduced initially, but increased once patients start to complete the *consult* action. This means we can maintain a steady total of nurses ($N_1 + N_2$) but reallocate staff between these two roles.

5 Conclusions and future work

The examples here show the use of PEPA and its fluid approximation in two very different scenarios. As stated in the introduction, this form of analysis was originally applied to PEPA for biochemical models, we have demonstrated here that the approach can also be applied to traditional computer science problems and to problems of a more techno-social nature involving the movement of people. The fact that the analysis can be used to derive some very useful information in both scenarios illustrates the power and flexibility of the approach.

The work described in this paper is ongoing. In the first example we aim to validate the models against real implementations of these protocols. There are clearly many more protocols of a similar nature that can also be studied using these techniques. In the case of the second example we aim to extend our work to consider other more complicated treatment pathways in different areas of healthcare. In addition we intend to develop some interface tools to enable healthcare practitioners and managers to access the analysis tools without needing to understand PEPA.

The models presented in this paper have also been solved using mean value analysis [17] which has validated the fluid approximation results. There is still some work to be done to better understand the accuracy of the fluid approximation in different scenarios. Some empirical study in this regard has been undertaken [16], but additional results on sensitivity to different distributions and functional rates (as used in the second example) remains to be undertaken.

References

1. Hillston, J.: A Compositional Approach to Performance Modelling, Cambridge University Press (1996)
2. Hillston, J.: Exploiting Structure in Solution: Decomposing Compositional Models, in: E. Brinksma *et al*, Lectures on Formal Methods and Performance Analysis, LNCS 2090, Springer-Verlag, (2003)
3. Hillston, J., Thomas, N.: Product form solution for a class of PEPA models, Performance Evaluation, **35**(3-4), pp. 171-192. (1999)

4. Hillston, J., and Thomas, N.: A syntactic analysis of reversible PEPA processes, in: Proceedings of the 6th International Workshop on Process Algebra and Performance Modelling, Nice (1998)
5. Harrison, P.G.: Turning back time in Markovian process algebra, Theoretical Computer Science, **290**, pp. 19471986. (2003)
6. Harrison, P.G., and Thomas, N.: Product form solution in PEPA via the reversed process, in: Next Generation Internet: Performance Evaluation and Applications, LNCS 5233, Springer-Verlag (2010)
7. Hillston, J.: Fluid flow approximation of PEPA models, in: Proceedings of QEST'05, pp. 33-43. IEEE Computer Society (2005)
8. Hayden, R., Bradley, J.: A fluid analysis framework for a Markovian process algebra, Theoretical Computer Science, **411**(22-24), pp. 2260-2297. (2010)
9. Zhao, Y., Thomas, N.: Approximate solution of a PEPA model of a key distribution centre, in: Performance Evaluation - Metrics, Models and Benchmarks: SPEC International Performance Evaluation Workshop, pp. 44-57. LNCS 5119, Springer-Verlag (2008)
10. Thomas, N., Zhao, Y.: Fluid flow analysis of a model of a secure key distribution centre, in: Proceedings of the 24th UK Performance Engineering Workshop, Imperial College London (2008)
11. Zhao, Y., Thomas, N.: Efficient solution of a PEPA model of a key distribution centre, Performance Evaluation, **67**(8), pp. 740-756. (2010)
12. Thomas, N., and Bradley, J.: Terminating processes in PEPA, in: Proceeding of the 17th UK Performance Engineering Workshop, University of Leeds (2001)
13. A. Clark, A. Duguid, S. Gilmore and M. Tribastone, Partial Evaluation of PEPA Models for Fluid-Flow Analysis, in: Computer Performance Evaluation: 5th European Performance Engineering Workshop, LNCS 5261, Springer-Verlag (2008)
14. Zhou, J., Gollmann, D.: A Fair Non-repudiation Protocol, in: Proceedings of IEEE Symposium on Security and Privacy (SP'96), IEEE Computer Society (1996)
15. Zhou, J., Gollmann, D.: Observation on Non-repudiation, in: Advances in Cryptology-ASIACRYPT'96, pp. 133-144. LNCS 1163/1996, Springer-Verlag (1996)
16. Thomas, N.: Using ODEs from PEPA models to derive asymptotic solutions for a class of closed queueing networks, in 8th Worshop on Process Algebra and Stochastically Timed Activities, University of Edinburgh (2009)
17. Thomas, N., Zhao, Y.: Mean value analysis for a class of PEPA models, The Computer Journal, **54**(5), pp. 643-652. (2011)