Extracting Response Times from Fluid Analysis of Performance Models

Jeremy T. Bradley Richard Hayden
William J. Knottenbelt Tamas Suto

Department of Computing, Imperial College London
180 Queen's Gate, London SW7 2BZ, United Kingdom.
{jb,rh,wjk,suto}@doc.ic.ac.uk

Abstract. Recent developments in the analysis of stochastic process algebra models allow for transient measures of very large models to be extracted. By performing so-called *fluid analysis* of stochastic process algebra models, it is now feasible to analyse systems of size 10^{1000} states and beyond. This paper seeks to extend the type of measure that can be extracted from this style of fluid analysis. We present a systematic transformation of a PEPA model that will allow us to extract measures analogous to response times. We end by extracting these response-time measures from a PEPA model of a healthcare system.

1 Introduction

The ability to calculate response-time or passage-time measures in quantitative analysis is important in many industrial systems. Response-time quantiles form the basis of many service level agreements (SLAs) in the telecommunications and other industries, e.g. a broadband connection should be successfully established within 2 seconds, 95% of the time.

However such industrial-scale systems require huge state-space analysis capability. If using traditional explicit state-space performance techniques, we quickly exceed the capability of Markov chain response-time analysers [1] to be able to generate and analyse the state space.

Recently, so-called *fluid* techniques have been developed to cope with the state-space explosion. This approach, typically, approximates the state space with a sequence of time-varying real variables and describes their evolution by a set of differential equations [2]. This sounds at first to be a panacea, but these techniques typically produce transient component counts at a given time instant. What we would like to do is reproduce useful response-time measures while taking advantage of the massive state-space capability of the fluid analysis techniques.

In this paper, we present a combination of these approaches by looking at how response-time measures might be extracted from fluid analysis of a stochastic process algebra model, PEPA. We show how, by modifying the state space of the model in a systematic fashion, we can transform the problem of response time extraction from fluid models to one of component time-to-extinction measurement.

2 Stochastic Process Algebra and Fluid Modelling

2.1 PEPA

PEPA [3] as a performance modelling formalism has been used to study a wide variety of systems: multimedia applications [4], mobile phone usage [5], GRID scheduling [6], production cell efficiency [7] and web-server clusters [8] amongst others. The definitive reference for the language is [3].

As in all process algebras, systems are represented in PEPA as the composition of *components* which undertake *actions*. In PEPA the actions are assumed to have a duration, or delay. Thus the expression $(\alpha, r).P$ denotes a component which can undertake an α action at rate r to evolve into a component P. Here $\alpha \in \mathcal{A}$ where $\mathcal{A}$ is the set of action types. The rate r is interpreted as a random delay which samples from an exponential random variable with parameter, r.

PEPA has a small set of combinators, allowing system descriptions to be built up as the concurrent execution and interaction of simple sequential components. The syntax of the type of PEPA model considered in this paper may be formally specified using the following grammar:

$$\begin{aligned} S &::= (\alpha, r).S \mid S + S \mid C_S \\ P &::= P \underset{L}{\bowtie} P \mid P/L \mid C \end{aligned}$$

where S denotes a *sequential component* and P denotes a *model component* which executes in parallel. C stands for a constant which denotes either a sequential component or a model component as introduced by a definition. C_S stands for constants which denote sequential components. The effect of this syntactic separation between these types of constants is to constrain legal PEPA components to be cooperations of sequential processes.

More information and structured operational semantics on PEPA can be found in [3]. A brief discussion of the basic PEPA operators is given below:

Prefix The basic mechanism for describing the behaviour of a system with a PEPA model is to give a component a designated first action using the prefix combinator, denoted by a full stop, which was introduced above. As explained, $(\alpha, r).P$ carries out an α action with rate r, and it subsequently behaves as P.

Choice The component $P + Q$ represents a system which may behave either as P or as Q. The activities of both P and Q are enabled. The first activity to complete distinguishes one of them: the other is discarded. The system will behave as the derivative resulting from the evolution of the chosen component.

Constant It is convenient to be able to assign names to patterns of behaviour associated with components. Constants are components whose meaning is given by a defining equation. The notation for this is $X \overset{def}{=} E$. The name X is in scope in the expression on the right hand side meaning that, for example, $X \overset{def}{=} (\alpha, r).X$ performs α at rate r forever.

Hiding The possibility to abstract away some aspects of a component's behaviour is provided by the hiding operator, denoted P/L. Here, the set L identifies those activities which are to be considered internal or private to the component and which will appear as the unknown type τ.

Cooperation We write $P \underset{L}{\bowtie} Q$ to denote cooperation between P and Q over L. The set which is used as the subscript to the cooperation symbol, the *cooperation set* L, determines those activities on which the components are forced to synchronise. For action types not in L, the components proceed independently and concurrently with their enabled activities. We write $P \parallel Q$ as an abbreviation for $P \underset{L}{\bowtie} Q$ when L is empty. Further, particularly useful in fluid analysis is, $P[n]$ which is shorthand for the parallel cooperation of n P-components, $\underbrace{P \parallel \cdots \parallel P}_{n}$.

In process cooperation, if a component enables an activity whose action type is in the cooperation set it will not be able to proceed with that activity until the other component also enables an activity of that type. The two components then proceed together to complete the *shared activity*. Once enabled, the rate of a shared activity has to be altered to reflect the slower component in a cooperation.

In some cases, when a shared activity is known to be completely dependent only on one component in the cooperation, then the other component will be made *passive* with respect to that activity. This means that the rate of the activity is left unspecified (denoted $\top$) and is determined upon cooperation, by the rate of the activity in the other component. All passive actions must be synchronised in the final model.

Within the cooperation framework, PEPA respects the definition of *bounded capacity*: that is, a component cannot be made to perform an activity faster by cooperation, so the rate of a shared activity is the minimum of the apparent rates of the activity in the cooperating components.

The definition of the derivative set of a component will be needed later in the paper. The derivative set, $ds(C)$, is the set of states that can be reached from a the state C. In the case, where C is a state in a strongly connected sequential component, $ds(C)$ represents the state space of that component.

2.2 Fluid Analysis

Traditionally, stochastic process algebras such as PEPA have been analysed by expanding the model description and extracting the global state space. The underlying mathematical model of a PEPA-generated state space is a continuous-time Markov chain or CTMC. The CTMC can be analysed for steady-state measures, transient measures or response-time measures and related back to the original PEPA model. This process suffers from the state-space explosion problem.

In previous fluid modelling papers [2, 9], a PEPA model was translated into a set of ordinary differential equations which were then solved. The results gave measures that roughly equated to mean transient measures in some cases.[1] What we seek to achieve here is a type of response-time result that can be extracted from the fluid analysis of a PEPA model.

Fluid modelling of process models refers to a continuum representation of the underlying discrete state space. Deriving such a representation from a performance-annotated process model, such as PEPA, gives a description of the flow of components from one derivative state to the next over time.

The first description of fluid analysis of PEPA models was presented by Hillston [2]. This has since been expanded upon [9, 10] but in this paper we keep to the subset of PEPA originally considered by Hillston [2] for translation to a fluid model.

In brief, we will summarise how the fluid model is constructed from a PEPA model that displays a large degree of parallelism. In [2], Hillston shows how a class of PEPA models can be analysed using coupled ordinary differential equations (ODEs). In this section, we summarise the numerical vector form representation and ODE analysis of PEPA models.

Cooperating models of identical non-synchronising agents of the form, for example:

$$\underbrace{P \mid\mid P \mid\mid \cdots \mid\mid P}_{n}$$

are more succinctly represented by a vector which describes the *number* of components in a given derivative state. That is to say, suppose P has two other derivative states, P' and P'', in its component description. A triple (v_1, v_2, v_3) could be used to represent there being v_1 components in state P, v_2 in state P' and v_3 in state P'' in the cooperation above. This creates an aggregation of the original explicit state space where, for example, the states $P' \mid\mid P \mid\mid \cdots \mid\mid P$ and $P \mid\mid P' \mid\mid \cdots \mid\mid P$ are combined with other states where there is only a single P' component in cooperation with P components.

[1] The exact relationship between the deterministic solution of the fluid model and the traditional probabilistic analysis of the CTMC is the subject of current research.

Clearly $v_1 + v_2 + v_3 = n$, the total number of components in the cooperation. The ordering of the derivative states within the expression above makes no difference to the observable behaviour. Thus there is no loss of information in simply counting derivatives in this way rather than recording their relative positions. Moreover it has the effect of reducing the state-space representation to an aggregated form (described in [11]) which requires a vector representation of size $|ds(P)|$, the number of derivative states of P, rather than one of size n, in the unaggregated form.

We demonstrate this aggregation with the generic example of an n-processor/m-resource system given in [2]. We have a processor component, $Proc_0$, which can perform a $task1$ action at rate r_1 and become a $Proc_1$ component. From there the $Proc_1$ component performs a $task2$ action at rate r_2 to return to state $Proc_0$. The Res_0 component performs similarly to switch between states Res_0 and Res_1. The final system comprises n processor components in parallel cooperating over the $task_1$ action with m resource components, also in parallel. This means that in order for a single $Proc_0$ component to perform a $task_1$ action, it has to synchronise with a single Res_0 component.

$$\begin{aligned}
Proc_0 &\overset{def}{=} (task1, r_1).Proc_1 \\
Proc_1 &\overset{def}{=} (task2, r_2).Proc_0 \\
Res_0 &\overset{def}{=} (task1, r_1).Res_1 \\
Res_1 &\overset{def}{=} (reset, s).Res_0 \\
System &\overset{def}{=} Proc_0[n] \underset{\{task1\}}{\bowtie} Res_0[m] \qquad (1)
\end{aligned}$$

This model would usually be translated into an underlying CTMC according to the operational semantic rules of PEPA, given in [3]. For even small values of m and n this results in an massive CTMC state-space. An aggregate state $((n-1,1),(m,0))$ would represent a possible state where there were $n-1$ processor components in state $Proc_0$, one in state $Proc_1$, m resource components in state Res_0, and none in state Res_1.[2]

Hillston [2] further goes on to show how a set of ODEs can be constructed which can represent the discrete number of components in a given state with a continuous state-space approximation. This is particularly useful in agent-oriented models which typically have many thousands of similar components in parallel. For this type of model, this type of aggregation is essential if the resulting state-space explosion is to be avoided.

[2] For comparison, $m = n = 100$ would generate an explicit CTMC state-space of 2^{200} states, but an aggregate state-space of only 101^2 states.

2.3 Numerical Vector Form and ODE Generation

Consider a PEPA model made up of component types C_i, such that the system equation has the form:

$$C_1[n_1] \underset{L}{\bowtie} C_2[n_2] \underset{L}{\bowtie} \cdots \underset{L}{\bowtie} C_m[n_m] \tag{2}$$

where $C[n]$ is the parallel composition of n C-components. Take C_{ij} to be the jth derivative state of component C_i. The cooperation set L is made up of common actions to C_i for $1 \leq i \leq m$. Now a numerical vector form for such a model would consist of $(v_{ij} \ : \ 1 \leq i \leq m, 1 \leq j \leq |ds(C_i)|)$ where v_{ij} is the number of C_{ij} components in the system at a given time. A set of coupled differential equations can be created to describe the time-variation of v_{ij} as follows:

$$\begin{aligned}\frac{\mathrm{d}v_{ij}(t)}{\mathrm{d}t} = &-\sum_{k\,:\,C_{ij}\xrightarrow{(a,\cdot)}C_{ik}} \text{rate of } a\text{-action leaving } C_{ij}\\ &+\sum_{k\,:\,C_{ik}\xrightarrow{(b,\cdot)}C_{ij}} \text{rate of } b\text{-action leaving } C_{ik}\end{aligned} \tag{3}$$

To make this specific to PEPA models of the type in Equation (2), we need a few preliminary definitions. Let us define $Ex(C)$ to be the set of action/rate pairs or activities (a, r) that are enabled by derivative state, C. Similarly, define the set of entry activities, $En(C)$, to be the set of action/rate pairs (b, s) that lead to state C, that is, for some C', there exists a one-step evolution $C' \xrightarrow{(b,s)} C$. It is also assumed in [2] that if, for some component type C_i, a derivative state enables an a-action, then no other derivative state of C_i can enable that same action. We follow that restriction here for simplicity, but further work on extending the expressiveness of fluid translation is on-going [9, 10].

From these definitions, we can create a more precise version of Equation (3):

$$\begin{aligned}\frac{\mathrm{d}v_{ij}(t)}{\mathrm{d}t} = &-\sum_{(a,r)\in Ex(C_{ij})} r \times \min\{v_{kl} \ : \ C_{kl} \xrightarrow{(a,r)} \}\\ &+\sum_{(b,s)\in En(C_{ij})} s \times \min\{v_{kl} \ : \ C_{kl} \xrightarrow{(b,s)} \}\end{aligned} \tag{4}$$

This formulation deals with PEPA models that cooperate actively and do so with constituent components enabling shared actions with the same rate. That is:

$$P \underset{\{a\}}{\bowtie} Q \text{ where } P \xrightarrow{(a,\lambda)} P' \text{ and } Q \xrightarrow{(a,\lambda)} Q'$$

This can be generalised straightforwardly to heterogeneous rates in cooperation where:

$$P \underset{\{a\}}{\bowtie} Q \text{ where } P \xrightarrow{(a,\lambda)} P' \text{ and } Q \xrightarrow{(a,\mu)} Q'$$

by a small modification to the ODE formula to:

$$\frac{\mathrm{d}v_{ij}(t)}{\mathrm{d}t} = - \sum_{(a,r_p)\in Ex(C_{ij})} \min\{r_p v_{kl} \ : \ C_{kl} \xrightarrow{(a,r_p)} \} + \sum_{(b,s_p)\in En(C_{ij})} \min\{s_p v_{kl} \ : \ C_{kl} \xrightarrow{(b,s_p)} \} \tag{5}$$

where $\{r_p\}$ and $\{s_p\}$ represent the set of distinct rates of a-actions and b-actions as enabled by the derivatives of the component-types C_i.

For the subset of PEPA worked with in this paper, the solution of these sets of ordinary differential equations, for a particular model, represents an approximation to the mean number of components at time t.

3 Response-Time Generation

The standard definition of a response time random variable in a Markov chain is set up as below.

Consider a finite, irreducible, continuous-time Markov process, $\{X(t) \ : \ t \geq 0\}$. $X(t)$ denotes the state of the Markov process at time $t \geq 0$. $N(t)$ denotes the number of state-transitions that have occurred by time t.

The first passage-time from a source state i at time 0 into a non-empty set of target states $\boldsymbol{j}$ is:

$$P_{i\boldsymbol{j}} = \inf\{u > 0 \ : \ X(u) \in \boldsymbol{j}, N(u) > 0 \mid X(0) = i\} \tag{6}$$

for a stationary time-homogeneous Markov process.

Loosely, this can be considered as the time-to-absorption from state i to one of the states in $\boldsymbol{j}$. What we propose in this paper, is to construct a similar concept in the fluid analysis of a PEPA model of the type of Equation (2).

One of the standard techniques for extracting response-time distributions from CTMCs, is to make states in the target set, $\boldsymbol{j}$, absorbing and perform transient analysis on the resulting modified chain [12].

Our approach is to perform a similar absorbing modification, but at the PEPA abstraction level rather than at the CTMC level, and then solve the resulting fluid model. The time-to-absorption measure which represents the response time in the original CTMC calculation is translated into the component extinction-time in the new fluid model.

3.1 Constructing an Absorbing PEPA Model

First, we will set up a basic PEPA absorption operator, $\triangleright$, which can be used systematically to modify any PEPA model in preparation for extracting a response-time measure.

We will only consider response times in terms of transitions of individual component types, e.g. how long before all the voters have voted, or all the clients have received service.

This translates into finding the response time for n_i components of type C_i, to have entered one of the states $H = \{C_{ij} : j \in \boldsymbol{j}\}$, having started in state C_{i1}, where $\boldsymbol{j}$ represents the set of target states in the component type being considered. Taking a PEPA model of the form:

$$C_1[n_1] \underset{L}{\bowtie} C_2[n_2] \underset{L}{\bowtie} \cdots \underset{L}{\bowtie} C_m[n_m] \tag{7}$$

Given a set, $H = \{C_{ij} : j \in \boldsymbol{j}\}$, of component states that we wish to make absorbing, we apply the absorption operator recursively over the PEPA syntax:

$$\begin{aligned}
((a,\lambda).P) \triangleright_{(U)} H &= \begin{cases} (a,\lambda).\mathsf{Stop} & : \text{ if } P \in H \\ (a,\lambda).(P \triangleright_{(U\cup\{P\})} H) & : \text{ if } P \notin H, P \notin U \\ (a,\lambda).P & : \text{ if } P \notin H, P \in U \end{cases} \\
(P+Q) \triangleright_{(U)} H &= (P \triangleright_{(U)} H) + (Q \triangleright_{(U)} H) \\
(P\backslash L) \triangleright H &= (P \triangleright H)\backslash L \\
(P \underset{L}{\bowtie} Q) \triangleright H &= (P \triangleright H) \underset{L}{\bowtie} (Q \triangleright H)
\end{aligned}$$

$P \triangleright H$ is shorthand for $P \triangleright_{(\emptyset)} H$ where the indexed set keeps track of previously visited component states. The operator defined above, recurses across the PEPA description and on encountering a component state in the set H, it replaces it with the absorbing state Stop. We assume that derivative states are uniquely labelled across the component types to avoid multiple component types being made absorbing. If this is not the case then a simple relabelling can be applied in advance of this transformation.

In this paper, we are considering simple response times that are expressed in terms of one component type only. We will only consider derivative states in H that come from the same component type C_i for any i (as given by the definition of H).

Having absorbing states in PEPA is unusual as PEPA models usually have irreducible underlying CTMCs. The absorbing state in this instance is Stop, and a discussion of absorbing states in PEPA can be found in [13].

3.2 Processor–Resource Example

We use the earlier example of an n-processor/m-resource system from Equation (1). We require the response time of n *Proc*-components making the transition from $Proc_0$ to $Proc_0$ again. To achieve this, we apply the absorption operator to the PEPA model $System \triangleright H$ with $H = \{Proc_0\}$. This gives an absorbed model:

$$
\begin{aligned}
Proc_0 &\overset{def}{=} (task1, r_1).Proc_1 \\
Proc_1 &\overset{def}{=} (task2, r_2).\mathsf{Stop} \\
Res_0 &\overset{def}{=} (task1, r_1).Res_1 \\
Res_1 &\overset{def}{=} (reset, s).Res_0 \\
System &\overset{def}{=} Proc_0[n] \underset{\{task1\}}{\bowtie} Res_0[m]
\end{aligned}
$$

By way of comparison, solving the ODEs generated by the original model gives the plot in Figure 1. Solving the ODEs for the absorbing version gives Figure 2. In both cases $n = 100$, $m = 60$, $r_1 = 1.0$, $r_2 = 0.6$ and $s = 0.4$.

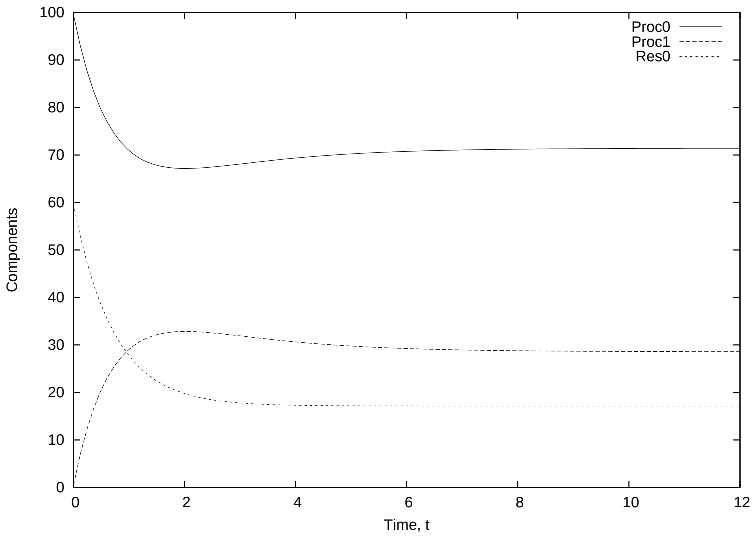

Fig. 1. ODE solution of the original Process/Resource model for number of $Proc_0$, $Proc_1$, Res_0 components

In Figure 2, we see the count of $Proc_0$ and $Proc_1$ components drop to 0 as would be expected in an absorbing model. We count the moment of absorption as the moment at which $< 0.5\%$ of the components remain in either state $Proc_0$ or $Proc_1$. This is measured at time 11.52 and represents a response-time measure

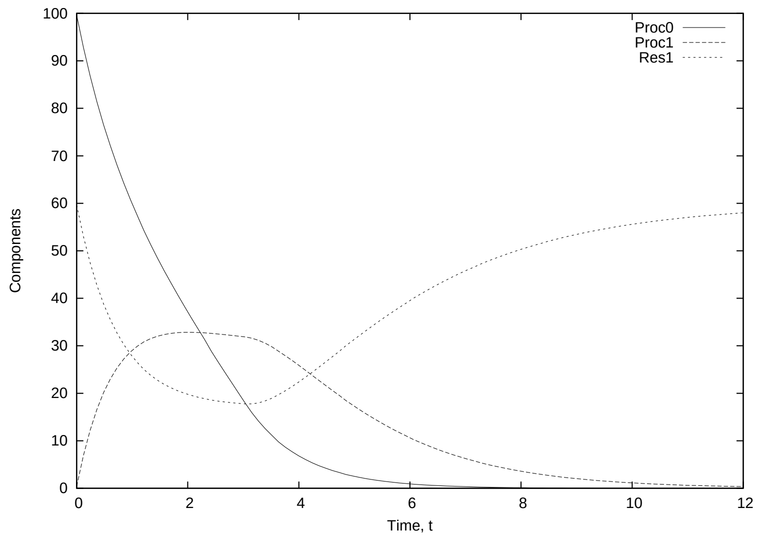

Fig. 2. ODE solution of the absorbed Process/Resource model with response time measured at 11.515s

for the time taken for $n = 100$ *Proc*-components to transit from state $Proc_0$ to $Proc_0$ while cooperating with the *Res*-components.

4 Worked Example: Healthcare System

The healthcare system in this section is a model of an accident and emergency department, first presented as a stochastic Petri net model in [14].

The system consists of patients, doctors and nurses, where patients who fall ill, are assessed by nurses before being sent to doctors for tests, treatment or surgery. The purpose of the system is to assess how fluctuations in the numbers of resources in the system, the number of nurses and doctors, affect the overall response-time for treatment.

$$System \overset{def}{=} Patient[P] \underset{L}{\bowtie} (Nurse[N] \mid\mid Doctor[D])$$

where $L = \{see\_nurse,\ complete\_assessment,\ see\_emergency\_nurse,$ $emergency\_assessment,\ see\_doctor,\ discharge\_treated\_patient,\ surgery,\ recover\}$.

The attentive reader will note that this system equation is not explicitly of the form of Equation (2), that we require for this particular style of fluid analysis. However, since the *Doctor* and *Nurse* components do not synchronise on any actions, we can effectively treat the $(Nurse[N] \mid\mid Doctor[D])$ cooperation as a single component group.

The nurses in the system can either see a standard patient or an emergency admittance. In each case an assessment is made before handing on for treatment.

$$\begin{aligned} \mathit{Nurse} &\overset{\mathit{def}}{=} (\mathit{see\_nurse}, r_4).(\mathit{complete\_assessment}, r_5).\mathit{Nurse} \\ &+ (\mathit{see\_emergency\_nurse}, r_6).(\mathit{emergency\_assessment}, r_7).\mathit{Nurse} \end{aligned}$$

The doctors in the system can either see and treat the patient or admit the patient for surgery.

$$\begin{aligned} \mathit{Doctor} &\overset{\mathit{def}}{=} (\mathit{see\_doctor}, r_8).(\mathit{discharge\_treated\_patient}, r_{11}).\mathit{Doctor} \\ &+ (\mathit{surgery}, r_9).(\mathit{recover}, r_{12}).\mathit{Doctor} \end{aligned}$$

Finally, the patients are of two types – either standard walk-in arrivals or emergency cases. They cooperate with the nurses and the doctors over the shared actions before being discharged.

$$\begin{aligned} \mathit{Patient} &\overset{\mathit{def}}{=} (\mathit{fall\_ill}, r_1).\mathit{Ill} \\ \mathit{Ill} &\overset{\mathit{def}}{=} (\mathit{walk\_in\_arrival}, r_2).\mathit{Waiting\_room} \\ &+ (\mathit{ambulance\_arrival}, r_3).\mathit{Trolley} \\ \mathit{Waiting\_room} &\overset{\mathit{def}}{=} (\mathit{see\_nurse}, r_4).\mathit{Patient\_assessment} \\ \mathit{Patient\_assessment} &\overset{\mathit{def}}{=} (\mathit{complete\_assessment}, r_5).\mathit{Waiting\_to\_be\_treated} \\ \mathit{Trolley} &\overset{\mathit{def}}{=} (\mathit{see\_emergency\_nurse}, r_6).\mathit{Ambulance\_assessment} \\ \mathit{Ambulance\_assessment} &\overset{\mathit{def}}{=} (\mathit{emergency\_assessment}, r_7).\mathit{Waiting\_to\_be\_treated} \\ \mathit{Waiting\_to\_be\_treated} &\overset{\mathit{def}}{=} (\mathit{see\_doctor}, r_8).\mathit{Treated\_by\_doctor} \\ &+ (\mathit{surgery}, r_9).\mathit{Surgery\_done} \\ &+ (\mathit{perform\_lab\_tests}, r_{10}).\mathit{Tests\_done} \\ \mathit{Treated\_by\_doctor} &\overset{\mathit{def}}{=} (\mathit{discharge\_treated\_patient}, r_{11}).\mathit{Patient} \\ \mathit{Surgery\_done} &\overset{\mathit{def}}{=} (\mathit{recover}, r_{12}).\mathit{Patient\_Recovered} \\ \mathit{Patient\_Recovered} &\overset{\mathit{def}}{=} (\mathit{discharge\_recovered\_patient}, r_{13}).\mathit{Patient} \\ \mathit{Tests\_done} &\overset{\mathit{def}}{=} (\mathit{evaluate\_results}, r_{14}).\mathit{Waiting\_to\_be\_treated} \end{aligned}$$

Using the techniques of Section 2.3, we construct a system of 17 coupled ODEs (for 16 derivative states of the original model plus the newly-introduced absorbing state) for $P = 100$ patients, $N = 30$ nurses, $D = 5$ doctors.[3] Such a system is well beyond the capability of existing explicit state-space techniques to analyse due to the size of the underlying CTMC. Without modification, we obtain solutions for patients, ill patients, nurses and doctors in Figure 3.

Now we seek the response-time measure for the time taken for $P = 100$ patients to pass through the system and go from state *Patient* back to state *Patient*. We

[3] The following rate values are used throughout this paper $r_1 = 1$, $r_2 = 4$, $r_3 = 3$, $r_4 = r_5 = r_6 = r_7 = r_{11} = r_{14} = 10$, $r_8 = 5$, $r_9 = r_{10} = r_{12} = 3$ and $r_{13} = 8$.

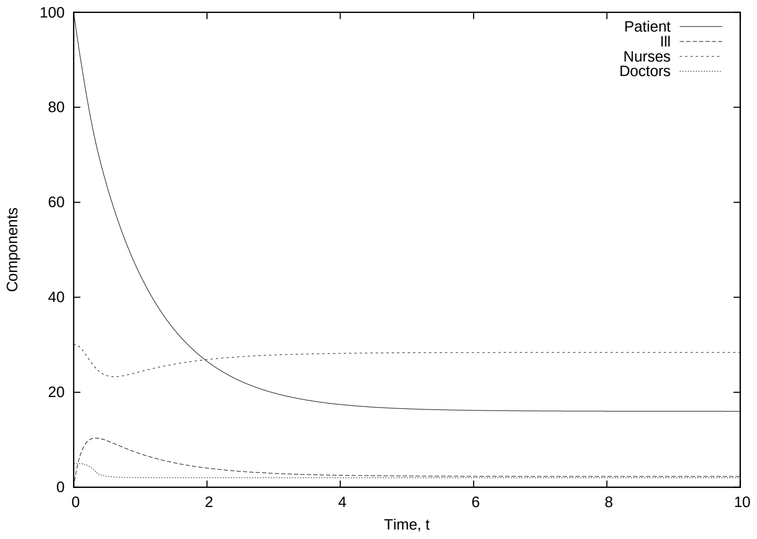

Fig. 3. ODE solution of the original hospital model for number of *Patient*, *Ill*, *Nurse* and *Doctor* components

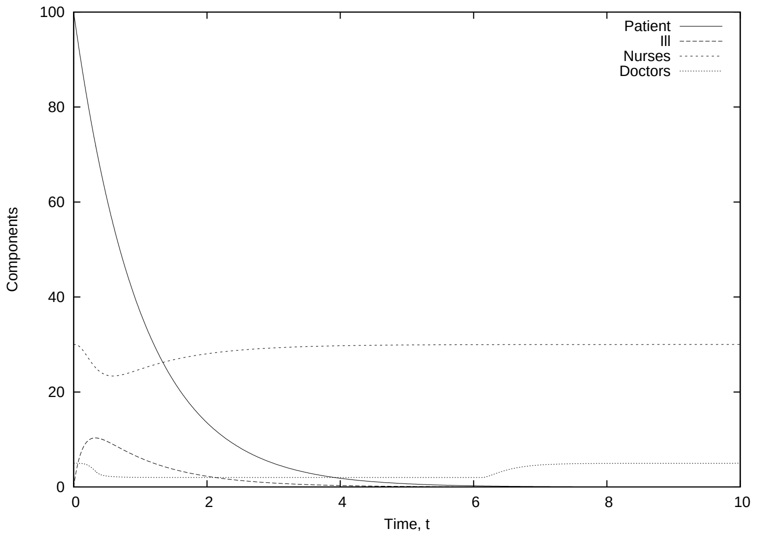

Fig. 4. ODE solution of the hospital model for patient response-time metric.

set up the absorbing PEPA model with the transformation $System \rhd \{Patient\}$. Plotting the new set of 18 resulting ODEs gives Figure 4, and on examining the data, we obtain the measure that it takes 7.17 hours for all 100 patients to progress through the system.

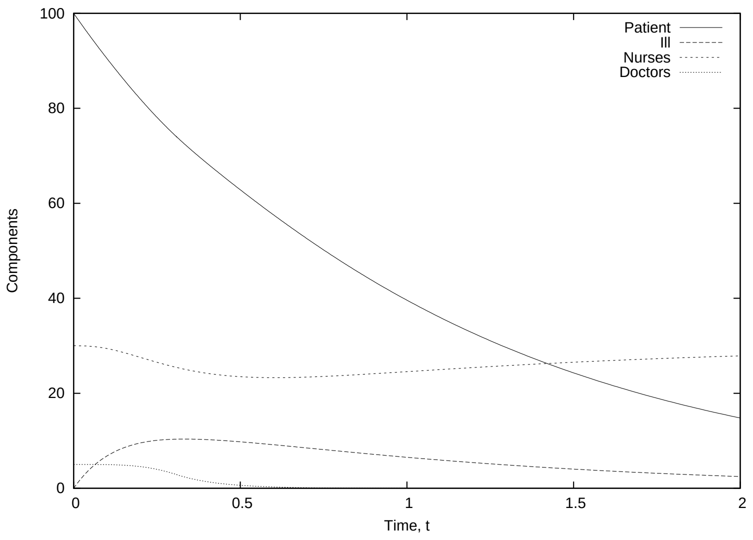

Fig. 5. ODE solution of the hospital model for patient response-time metric.

Finally, we seek a response-time measure on the progress of doctors in the model. Similarly, we calculate the response-time for $D = 5$ doctors to go from state *Doctor* to state *Doctor*. We set up the absorbing PEPA model with the transformation $\mathit{System} \rhd \{\mathit{Doctor}\}$. Plotting the new set of 18 resulting ODEs gives Figure 5, and on examining the data, we obtain the measure that it takes 1.84 hours for all 5 doctors to process at least 1 patient.

4.1 Comparison with a CTMC-Derived Passage Time

Given the motivations behind developing such techniques of fluid analysis, it is of course computationally infeasible to compute passage times in the usual manner[4] for models of the magnitude of that just presented. In this section, we consider a scaled down form of the healthcare model (5 patients, 3 nurses and 2 doctors: about 24 million states). This will allow a comparison of the quantity computed using the new technique presented here against the CTMC-derived passage time.

We work again with response-time measure of the healthcare model obtained via the transformation $\mathit{System} \rhd \{\mathit{Patient}\}$. Figure 6 shows the CDF for the CTMC-derived time of passage from the initial state to that in which all *Patient* components have reached the Stop state. Since we defined the point of absorption for the ODE-derived solution to be when the sum of the continuous variables

[4] Using the standard uniformisation technique of the underlying CTMC.

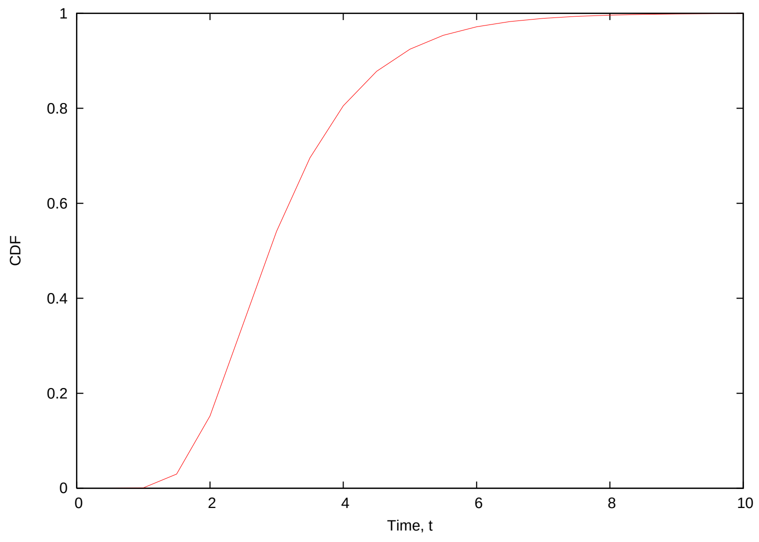

Fig. 6. CTMC passage time to patient absorption CDF for the small hospital model.

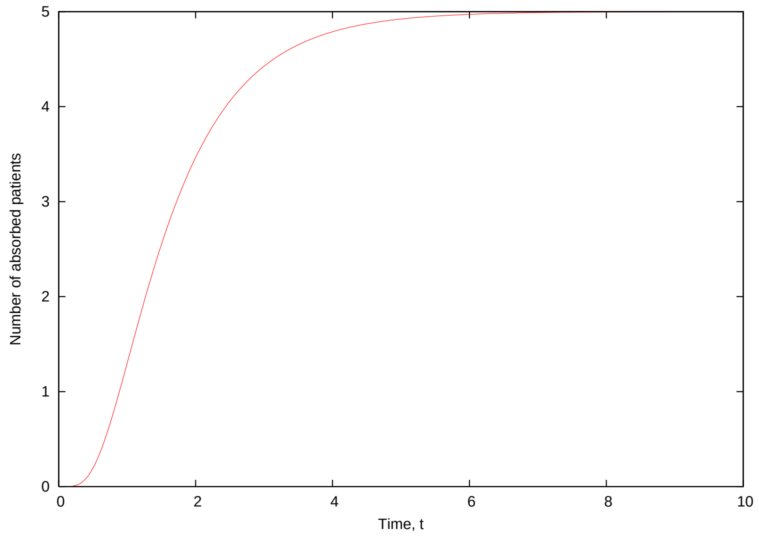

Fig. 7. ODE solution of the small hospital model for patient response-time metric, showing only the variable for the patient absorbing state (Stop).

counting the states of the patients reaches 0.5% of the original component population, it makes at least intuitive sense to look at the 99.5% quantile of the corresponding CTMC passage time. Examining the data, we see that in the direct CTMC passage-time calculation it takes approximately 7.6 hours for 99.5%

of the *Patient* components to absorb. Figure 7 shows the result of integrating the 18 ODEs for the model. In this case, it takes about 6.13 hours for 99.5% of the original patient population to be absorbed.

A general observation from this preliminary comparison is that the absorbing fluid model appears to absorb more quickly than the equivalent CTMC cumulative distibution function of the passage-time. Work is on-going to characterise this relationship more formally but it appears that it may be a justifiable result in many cases.

5 Conclusion

Fluid analysis of stochastic process algebra models is a powerful analytic tool for obtaining quantitative analysis of massive state-space models. We have summarised existing fluid techniques for a popular process algebra, PEPA, and pointed out that the type of measure that is obtainable from the standard fluid analysis [2] is restricted to a form of transient analysis. We have shown, in this paper, how it might be possible to express and extract response-time style measures from fluid analysis of stochastic process algebra models. We did this by using an analogous absorbing state technique to that used in the explicit state-space analysis of response times in CTMCs. By constructing an absorption operator for the PEPA language, we have a simple tool for allowing general PEPA models to be analysed for fluid-generated response times.

In this paper, we looked for the absorption of 99.5% of the components under consideration in the system to extract the response time. We compared that with the equivalent 99.5% CDF quantile measurement in a 24 million state CTMC version of a healthcare system. We postulate that the fluid-generated response time will tend to underestimate the CTMC response-time measure in general, but further work is required in this area to show this. We would like to establish a relationship with the mean response-time measurement of traditional CTMC analysis. We are also looking to generate variance and higher moment metrics for the response-time measure extracted in this way.

Finally, there is the potential for constructing more expressive measures on more general models. Currently, we only look for movement of an entire population of component types from one state to another. It would be a clear advantage to be able to look at response times of a combination of partial movements of populations of components types. We see no reason why this approach should not be extended directly to more general PEPA models as the fluid semantics are defined for those models.

Acknowledgements

The authors would like to thank Allan Clark for help with ipc and Jane Hillston for advice over fluid mean response time calculations. In addition, we would like to credit the anonymous referees for helpful comments which improved the paper. Jeremy Bradley, William Knottenbelt and Tamas Suto are supported in part by the EPSRC grant GRAIL, ref. EP/D505933/1.

References

1. Bradley, J.T., Dingle, N.J., Gilmore, S.T., Knottenbelt, W.J.: Extracting passage times from PEPA models with the HYDRA tool: a case study. In Jarvis, S.A., ed.: UKPEW'03, Proceedings of 19th Annual UK Performance Engineering Workshop, University of Warwick (July 2003) 79–90
2. Hillston, J.: Fluid flow approximation of PEPA models. In: QEST'05, Proceedings of the 2nd International Conference on Quantitative Evaluation of Systems, Torino, IEEE Computer Society Press (September 2005) 33–42
3. Hillston, J.: A Compositional Approach to Performance Modelling. Volume 12 of Distinguished Dissertations in Computer Science. CUP (1996)
4. Bowman, H., Bryans, J.W., Derrick, J.: Analysis of a multimedia stream using stochastic process algebras. The Computer Journal **44**(4) (2001) 230–245
5. Fourneau, J.M., Kloul, L., Valois, F.: Performance modelling of hierarchical cellular networks using PEPA. Performance Evaluation **50**(2–3) (November 2002) 83–99
6. Thomas, N., Bradley, J.T., Knottenbelt, W.J.: Stochastic analysis of scheduling strategies in a GRID-based resource model. IEE Software Engineering **151**(5) (September 2004) 232–239
7. Holton, D.R.W.: A PEPA specification of an industrial production cell. In Gilmore, S., Hillston, J., eds.: Process Algebra and Performance Modelling Workshop. Volume 38(7) of The Computer Journal., CEPIS (Edinburgh, June 1995) 542–551
8. Bradley, J.T., Dingle, N.J., Gilmore, S.T., Knottenbelt, W.J.: Derivation of passage-time densities in PEPA models using ipc: the Imperial PEPA Compiler. In Kotsis, G., ed.: MASCOTS'03, Proceedings of the 11th IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunications Systems, University of Central Florida, IEEE Computer Society Press (October 2003) 344–351
9. Bradley, J.T., Gilmore, S.T., Hillston, J.: Analysing distributed internet worm attacks using continuous state-space approximation of process algebra models. Journal of Computer and System Sciences (July 2007) (in press).
10. Hayden, R.: Addressing the state space explosion problem for PEPA models through fluid-flow approximation. Technical report, Ugrad. project report, Imperial College London (2007)
11. Gilmore, S., Hillston, J., Ribaudo, M.: An efficient algorithm for aggregating PEPA models. IEEE Transactions on Software Engineering **27**(5) (2001) 449–464
12. Dingle, N.J., Harrison, P.G., Knottenbelt, W.J.: Uniformization and hypergraph partitioning for the distributed computation of response time densities in very large Markov models. Journal of Parallel and Distributed Computing **64**(8) (August 2004) 908–920

13. Thomas, N., Bradley, J.T.: Terminating processes in PEPA. In Djemame, K., Kara, M., eds.: UKPEW'01, Proceedings of 17th Annual UK Performance Evaluation Workshop, Leeds (July 2001) 143–154
14. Suto, T., Bradley, J.T., Knottenbelt, W.J.: Performance Trees: Expressiveness and quantitative semantics. In: QEST'07, 4th International Conference on the Quantitative Evaluation of Systems, IEEE (September 2007) 41–50