Automated Customer-Centric Performance Analysis of Generalised Stochastic Petri Nets Using Tagged Tokens

Nicholas J. Dingle William J. Knottenbelt

Department of Computing,
Imperial College London,
180 Queen's Gate,
London SW7 2BZ,
United Kingdom.

{njd200,wjk}@doc.ic.ac.uk

Abstract

Since tokens in Generalised Stochastic Petri Net (GSPN) models are indistinguishable, it is not always possible to reason about customer-centric performance measures. To remedy this, we propose "tagged tokens" – a variant of the "tagged customer" technique used in the analysis of queueing networks. Under this scheme, one token in a structurally restricted net is "tagged" and its position tracked as it moves around the net. Performance queries can then be phrased in terms of the position of the tagged token.
To date, the tagging of customers or tokens has been a time-consuming, manual and model-specific process. By contrast, we present here a completely automated methodology for the tagged token analysis of GSPNs. We first describe an intuitive graphical means of specifying the desired tagging configuration, along with the constraints on GSPN structure which must be observed for tagged tokens to be incorporated. We then present the mappings required for automatically converting a GSPN with a user-specified tagging structure into a Coloured GSPN (CGSPN), and thence into an unfolded GSPN which can be analysed for performance measures of interest by existing tools. We further show how our methodology integrates with Performance Trees, a formalism for the specification of performance queries.
We have implemented our approach in the open source PIPE Petri net tool, and use this to illustrate the extra expressibility granted by tagged tokens through the analysis of a GSPN model of a hospital's Accident and Emergency department.

1 Introduction

Performance modelling formalisms provide a convenient way to abstract and reason about the flow of customers and resources in complex concurrent systems. Amongst such formalisms, Generalised Stochastic Petri nets (GSPNs) [2] are widely used because they are conceptually easy to understand, graphical in nature and well-supported by a large body of theory as well as a large tool base. The dynamic behaviour of GSPNs centres around the creation and destruction of the tokens representing customers and resources[1] in the system. These tokens are indistinguishable from one another, which means that it is not always possible to reason

[1] Hereafter we refer solely to customers.

about the performance of a modelled system from the perspective of an individual customer. Such analysis is needed, however, to answer questions such as "Is the probability of a customer being served within t time units greater than 90%?" Customer-centric queries of this nature are important because they are increasingly used in Service Level Agreements (SLAs) in many systems, including healthcare systems, postal services and communication networks.

This paper therefore presents the use of "tagged tokens" to enable the modeller to identify and reason about the progress of an individual customer in a GSPN. This concept is a variant of the "tagged customer" technique for the analysis of queueing networks [13]. The contribution of this paper is three-fold. Firstly, we present an intuitive graphical approach based on the concept of "tagged arcs" by which a modeller can incorporate tagged tokens into existing GSPN models. We then present the automatic mapping from a GSPN with tagged arcs into a Coloured Generalised Stochastic Petri Net (CGSPN) [11]. This is preferable to the direct specification of a CGSPN as it does not require the modeller to be familiar with the more complex CGSPN formalism. Finally, we describe an efficient way of unfolding the CGSPN representation into a GSPN whose continuous time Markov chain (CTMC) can be analysed for performance measures of interest using existing tools.

Prior work on the computation of performance measures using tagged tokens in Stochastic Petri Nets (SPNs) is limited. Miner [12] describes the calculation of response times in SPN models where one entity in the system (represented by one of a number of tokens) is tagged. This has exactly the same motivation as the work presented here – namely the extraction of customer-centric performance metrics. However, unlike the methodology presented here, the tagging process is manual and model-specific.

Similarly, little prior work exists on providing tool support for tagged customer analysis in Markov models. Argent-Katwala et. al [3] have presented an automated approach for tracking individual entities in Performance Evaluation Process Algebra (PEPA) [9] models through the use of stochastic probes, while Bodrog et. al. [5] have developed the MRMSolve tool to support tagged customer analysis in Markov reward models. However, MRMSolve has two limitations in the context of our work. Firstly, the analysis is not wholly automated as the (non-trivial and model-dependent) mapping from the steady-state distribution of the embedded model to the initial probability vector of the reward model must be specified by hand. This potentially requires considerable expertise on the part of the modeller. Secondly, MRMSolve calculates the moments of the required performance measure and uses these to estimate upper and lower bounds on the actual distribution of interest, whereas the technique described here can be used with a number of existing tools to calculate distributions exactly.

The remainder of this paper is organised as follows: Section 2 briefly describes the background theory and associated notation of GSPNs and CGSPNs. Section 3 presents the intuitive graphical "tagged arc" mechanism which allows the user to incorporate tagged tokens into an existing GSPN model, along with the structural restrictions which must be observed in so doing. Section 4 describes the mapping of a GSPN with tagged arcs into a CGSPN, before Section 5 presents an efficient scheme for unfolding this CGSPN into a standard GSPN which can be analysed

using existing tools. Section 6 then shows how Performance Trees can be used to specify queries involving tagged tokens, and Section 7 demonstrates the extra expressibility conferred by tagged tokens in the analysis of quality of service metrics in a model of a hospital's Accident and Emergency department. Section 8 concludes and suggests areas for future work.

2 Background

Petri nets were originally devised as a graphical formalism for describing concurrency and synchronisation in distributed systems. In their simplest form they are also known as Place-Transition nets [4]. Generalised Stochastic Petri Nets (GSPNs) [2] extend Place-Transition nets by incorporating timing information.

Definition 2.1 A GSPN is an 8-tuple $GSPN = (P, T, T_1, T_2, W, I^-, I^+, M_0)$ where:

- $P = \{p_1, ..., p_n\}$ is a finite and non-empty set of places.
- $T = \{t_1, ..., t_m\}$ is a finite and non-empty set of transitions.
- $P \cap T = \emptyset$
- $T_1 \subseteq T$ is the set of timed transitions.
- $T_2 \subset T$ is the set of immediate transitions, where $T_1 \cap T_2 = \emptyset$ and $T = T_1 \cup T_2$.
- $W = \left(w_1, ..., w_{|T|}\right)$ is an array whose entry $w_i \in \mathbb{R}^+$ is a (possibly marking dependent)
  - · rate of a negative exponential distribution (also denoted λ_i) specifying the firing delay, when transition $t_i \in T_1$, or
  - · firing weight, when $t_i \in T_2$.
- $I^-, I^+ : P \times T \rightarrow \mathbb{N}_0$ are the backward and forward incidence functions, respectively.
- $M_0 : P \rightarrow \mathbb{N}_0$ is the initial marking.

A marking (or state) of a GSPN is a vector of integers representing the number of tokens on each place of the model. A transition can fire if the input places of the transition contain at least the number of tokens specified by the backward incidence functions. In so firing, a number of tokens are removed from the transition's input places and a number of tokens added to the transition's output places according to the backward and forward incidence functions respectively.

Denoting the number of tokens on place p in marking M by $M(p)$, the formal definition of the enabling condition for transition t is $M(p) \geq I^-(p,t), \forall p \in P$. The set of input places to transition t (also referred to as the *preset* of t), denoted $\bullet t$, and the set of output places (or *postset*) of t, $t\bullet$, are defined as:

$$\begin{aligned} \bullet t &:= \{p \in P \mid I^-(p,t) > 0\} \\ t\bullet &:= \{p \in P \mid I^+(p,t) > 0\} \end{aligned} \tag{1}$$

Timed transitions have an exponentially distributed firing rate λ_i. Immediate transitions fire in zero time. Markings that only enable timed transitions are *tangible*, while a marking that enables any immediate transition is *vanishing*. We denote the set of tangible markings by $\mathcal{T}$ and the set of vanishing markings by $\mathcal{V}$.

The stochastic process described by a GSPN's underlying reachability graph is a CTMC if $\mathcal{V} = \emptyset$ and a semi-Markov chain otherwise. It is possible, however, to reduce the reachability graph of a GSPN containing vanishing states to one which is a CTMC by using vanishing-state elimination techniques [7,10].

Coloured Generalised Stochastic Petri Nets (CGSPNs) [11] extend GSPNs by assigning colours to tokens. The marking of a place is therefore a multi-set containing varying numbers of tokens of each colour. Transitions have different firing modes which are enabled depending on the colours of the tokens on their input places and alter the multi-sets of tokens on their input and output places upon firing.

This richer behaviour is encoded in the CGSPN's backwards and forward incidence functions. For example, if $I^-(p_i, t_j)(x) = \{a\}$ and p_i is the only input place to transition t_j, then t_j is enabled in mode x if and only if one or more tokens of colour a are present on input place p_i. If t_j subsequently fires in this mode, one token of colour a will be removed from place p_i. The corresponding forward incidence function for mode x, $I^+(p_k, t)(x)$, will specify the multi-set of coloured tokens added to the output place p_k.

The set of token colours is denoted by $C(p)$ and the set of transition firing modes by $C(t)$. The enabling rule for CGSPNs is that a transition is enabled in mode c' if and only if $M(p)(c) \geq I^-(p,t)(c')(c), \forall p \in P, c \in C(p)$, where $M(p)(c)$ is the number of tokens of colour c on place p in marking M.

Definition 2.2 A CGSPN is an 9-tuple $CGSPN = (P, T, T_1, T_2, C, W, I^-, I^+, M_0)$ where [4]:

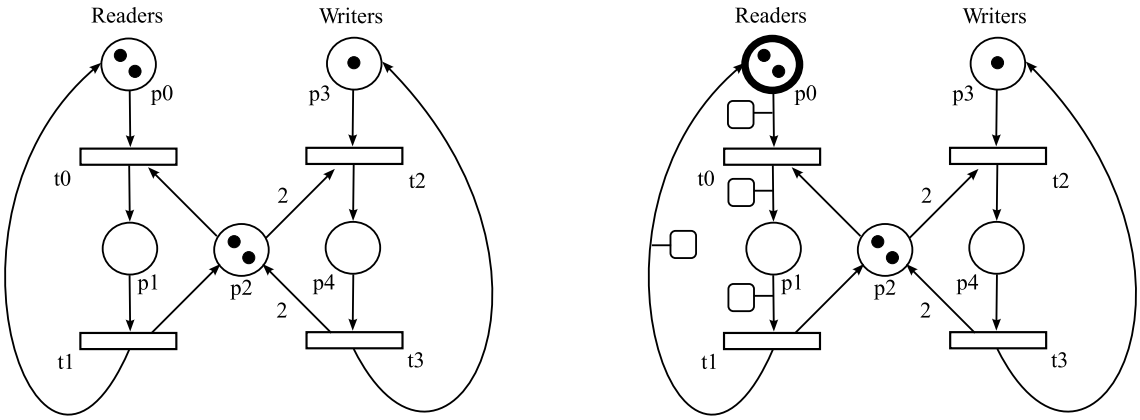
- P, T, T_1, T_2 are as defined for a GSPN.
- C is a colour function defined from $P \cup T$ into finite and non-empty sets.
- $W = \left(w_1, ..., w_{|T|}\right)$ is an array whose entry w_i is a function $[C(t_i) \rightarrow \mathbb{R}^+]$, such that $\forall c' \in C(t_i) \ : \ w_i(c') \in \mathbb{R}^+$ is the
  - rate of a (possibly marking dependent) negative exponential distribution (also denoted λ_i) specifying the firing delay in mode c', when $t_i \in T_1$, or
  - (possibly marking dependent) firing weight in mode c', when $t_i \in T_2$.
- I^-, I^+ are the backward and forward incidence functions such that:

$$I^-(p,t), I^+(p,t) \in [C(t) \rightarrow C(p)_{MS}], \forall (p,t) \in P \times T$$

  where S_{MS} denotes the set of all finite multisets over the set S.
- M_0 is a function defined on P describing the initial marking such that $M_0(p) \in C(p)_{MS}, \forall p \in P$.

Recalling Eq. 1, we define the preset of transition t in mode c', $\bullet(t, c')$, and postset, $(t, c')\bullet$, as:

$$\bullet(t, c') := \{(p, c) \mid p \in P, c \in C(p) : I^-(p,t)(c')(c) > 0\}$$
$$(t, c')\bullet := \{(p, c) \mid p \in P, c \in C(p) : I^+(p,t)(c')(c) > 0\}$$



(a) The original SPN.

(b) The user-identified tagged structure.

Fig. 1. Graphical specification of a tagged version of a simple readers-writers model.

3 Specification of Tagged Token GSPN Models

From the description in Section 2, it can be seen that tokens in a conventional Petri net are volatile and non-atomic; that is, they are created and destroyed by the firing of transitions and there is no requirement for the same number of tokens to exist on the output place(s) of a transition after it fires as there were on the input place(s). This volatility makes it problematic to track the progress of an individual token around a net where it is meaningful so to do.

We therefore introduce the concept of "tagged tokens" to permit this tracking. This further requires the concept of the "tagged arc" as the mechanism by which the modeller can specify how tagged tokens are routed around the net. Tagged arcs are distinguished from normal arcs by the addition of a small square "tag".

When checking if transitions are enabled and where tokens will be removed and placed when transitions fire, tagged tokens may only be carried along tagged arcs. Normal tokens may be transported by both tagged and normal arcs. Tagged tokens count as normal tokens towards determining whether or not a transition is enabled in a particular marking.

To reason about the presence (or otherwise) of tagged arcs, we augment Definition 2.1 with the functions $A^-, A^+ : P \times T \to \{0, 1\}$. $A^-(p, t) = 1$ if a tagged arc leads from place p to transition t and 0 otherwise, and similarly $A^+(p, t) = 1$ if a tagged arc leads from transition t to place p and 0 otherwise.

Fig. 1(a) shows a simple GSPN model of a readers-writers system with two readers and one writer. Assume that the modeller wishes to examine the performance of the system from the perspective of one of the readers. These are represented by two identical tokens and so cannot be distinguished unless tagged tokens are used. The bold border of p_0 in Fig. 1(b) signifies that one of the tokens on that place is tagged. The modeller then specifies the route through the system which the reader can take by tagging the appropriate arcs as shown. Note that the modeller's understanding of the meanings of the transitions and places is central to the tagging process and therefore this tagging of arcs must be performed manually.

3.1 Structural Restrictions

Our methodology requires that there is exactly one tagged token in a GSPN. Additional tagged tokens cannot be introduced into the net, and the unique tagged token cannot be removed from it. This leads to the following (easily-verified) structural restrictions which the modeller must obey when incorporating tagged arcs and tokens into a GSPN:

(i) There must be a single tagged token in the GSPN and the location of this token must be specified in the net's initial marking.

(ii) Any transition which has an input arc which is a tagged arc *must* have a corresponding output tagged arc.

(iii) A transition must have at most one output tagged arc, although no restriction need be placed on the number of input tagged arcs (so that the tagged token may reach the transition via different routes through the net).

Note that, although the firing of a transition with tagged input and output arcs must preserve the tagged token, no restrictions need be placed on the creation or destruction of normal tokens. We likewise do not place any restriction on the multiplicity (weight) of either tagged or untagged arcs.

4 Automatic CGSPN Conversion

After the user has specified the tagging structure for a GSPN model, it can be automatically converted into a CGSPN. This permits the tagged token to be distinguished from normal tokens through the use of different colours.

4.1 Token Colours

There is only one token of colour t, representing the tagged token; all remaining tokens are of colour ut, representing their untagged status. Hence $C(p) := \{t,\ ut\}$.

4.2 Transition Firing Modes

There are two corresponding transition firing modes, t' and ut', and hence $C(t) := \{t',\ ut'\}$. We interpret these modes as follows. A transition enabled in mode t' can fire the tagged token, and so the tagged token must be present on one of its input places. A transition firing in this mode may also consume and produce ut-coloured tokens. A transition enabled in mode ut' can only consume and produce ut-coloured tokens, although the tagged token may still be present on an input place.

4.3 Transition Firing Weights and Rates

It is possible for a single transition to be simultaneously enabled in both firing modes. For example, t_0 in Fig. 1(b) is enabled in both t' and ut' modes as $M(p_0) = \{t, ut\}$. In this case, the firing mode is selected probabilistically based on the transition's firing rate or weight (depending if it is timed or immediate) in each enabled mode. Specifically, the marking-dependent rates (weights) of transition t_j in modes t' and ut', denoted $w_j(t')$ and $w_j(ut')$ respectively, are:

$$w_j(t') := \left(\max_{p_i \in \bullet t_j} \frac{I^-(p_i, t_j) M(p_i)(t)}{M(p_i)(t) + M(p_i)(ut)} \right) w_j$$

$$w_j(ut') := w_j - w_j(t')$$

Note that this definition preserves the firing rate of transition t_j in the original GSPN when considered across both firing modes.

4.4 Incidence Functions

The backwards and forwards incidence functions of a CGSPN transition depend not only on the physical structure of the net but also on the firing modes and the colours of the tokens on its input places. With two transition modes and two token colours, the CGSPN therefore has four backwards and four forwards incidence functions.

The backwards incidence functions for a transition t_j are:

$$I^-(p_i, t_j)(ut')(ut) := \begin{cases} I^-(p_i, t_j) & \forall p_i \in \bullet t_j \\ 0 & \text{otherwise} \end{cases}$$

$$I^-(p_i, t_j)(ut')(t) := 0 \quad \forall p_i \in P$$

$$I^-(p_i, t_j)(t')(ut) := \begin{cases} I^-(p_i, t_j) - M(p_i)(t) & \forall p_i \in \bullet t_j \\ 0 & \text{otherwise} \end{cases}$$

$$I^-(p_i, t_j)(t')(t) := \begin{cases} 1 \ \forall p_i \in \bullet t_j & \text{if } \forall p_i \in \bullet t_j \ M(p_i)(t) = 0 \\ M(p_i)(t) \ \forall p_i \in \bullet t_j & \text{if } \forall p_i \in \bullet t_j \ M(p_i)(t) \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

For $I^-(p_i, t_j)(t')(t)$, it is necessary to distinguish the behaviour when the tagged token is not present on any of the transition's input places, in which case the transition cannot fire in mode t', from when it is. In the first case, we set the backwards incidence function to require the tagged token be present on all input places, which is obviously impossible, to ensure that the transition's t' firing mode is disabled.

The corresponding forward incidence functions are:

$$I^+(p_k, t_j)(ut')(ut) := \begin{cases} I^+(p_k, t_j) & \forall p_k \in t_j \bullet \\ 0 & \text{otherwise} \end{cases}$$

$$I^+(p_k, t_j)(ut')(t) := 0 \quad \forall p_k \in P$$

$$I^+(p_k, t_j)(t')(ut) := \begin{cases} I^+(p_k, t_j) - 1 & \text{if} A^+(p_k, t_j) = 1 \\ I^+(p_k, t_j) & \text{if} A^+(p_k, t_j) = 0, p_k \in t_j \bullet \\ 0 & \text{otherwise} \end{cases}$$

$$I^+(p_k, t_j)(t')(t) := \begin{cases} 1 & \text{if} A^+(p_k, t_j) = 1 \\ 0 & \text{otherwise} \end{cases}$$

5 Efficient CGSPN Analysis

We wish to use existing performance analysis tools such as DNAmaca [10] and HYDRA [8] to analyse the CGSPN for measures such as steady-state probabilities and response time distributions. Although these tools are not designed to analyse CGSPNs directly, a CGSPN can be uniquely and automatically unfolded into a (uncoloured, untagged) GSPN suitable for analysis as follows [4]:

- $\forall p \in P, c \in C(p)$ create a place (p, c) of the GSPN.
- $\forall t \in T, c' \in C(t)$ create a transition (t, c') of the GSPN with rate or weight $w_t(c')$.
- Define the incidence functions of the GSPN as:

  $$I^-((p,c)(t,c')) := I^-(p,t)(c')(c)$$
  $$I^+((p,c)(t,c')) := I^+(p,t)(c')(c)$$
- The initial marking of the GSPN is:

$$M_0(p,c) := M_0(p)(c), \forall p \in P, c \in C(p)$$

The unfolded GSPN is therefore given by:

$$\left(\bigcup_{p \in P} \bigcup_{c \in C(p)} (p,c), \bigcup_{t \in T} \bigcup_{c' \in C(t)} (t,c'), \bigcup_{t \in T} \bigcup_{c' \in C(t)} w_t(c'), I^-, I^+, M_0 \right)$$

With two colours of token and two transition firing modes, naïvely conducting this unfolding will result in the number of places and transitions doubling in the unfolded GSPN. The structural restrictions in Section 3, however, allow us to reduce the size of the unfolded GSPN and so conduct subsequent analysis more efficiently. In particular, as the CGSPN will only ever have one t-colour token we need only add a single place to the unfolded GSPN, whose marking represents the index of the place on which the tagged token currently resides. Thus we avoid doubling the number of places. Also, those transitions which have no tagged input and output arcs will only ever fire in mode *ut'* and so need only to be represented by a single transition in the unfolded GSPN.

6 Specifying Tagged Token Queries Using Performance Trees

Performance Trees [14,15] are a formalism for the representation of performance-related queries. They combine the ability to specify performance requirements – i.e. queries aiming to determine whether particular properties hold on system models – and to extract performance measures – i.e. quantifiable performance metrics.

A Performance Tree query is represented as a tree structure consisting of nodes and interconnecting arcs. Nodes can have two kinds of roles: *operation* nodes are performance-related functions, such as the calculation of a passage time density, while *value* nodes are the inputs to these functions such as a set of states, an action, or simply numerical or boolean constants. A full list of currently supported performance analysis operation nodes can be found in [15].

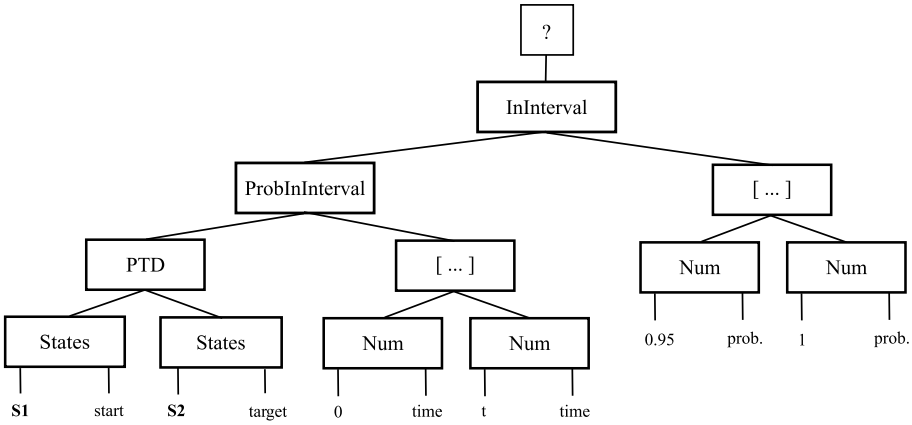


Fig. 2. An example Performance Tree query.

Performance Trees support an abstract state set specification mechanism to enable the user to specify states relevant to a performance measure of interest. For a GSPN, a set of states can be specified using conjunctions and disjunctions of constructs of the form $(M(p_i) \bowtie x)$, where $\bowtie \in \{\leq, <, =, \geq, >\}$.

Fig. 2 shows an example of a Performance Tree. It corresponds to the query "Does the model transit from a state in the set $\mathbf{S_1}$ to any of the states in set $\mathbf{S_2}$ in less than t time units at least 95% of the time?" Associated with this, we must define the sets of states $\mathbf{S_1}$ and $\mathbf{S_2}$; for example:

$$\mathbf{S_1} := (M(p_1) = 2) \wedge (M(p_2) < 3)$$

$$\mathbf{S_2} := (M(p_3) \leq 2)$$

Tagged tokens fit naturally within this state set specification mechanism, thus allowing Performance Trees to be used to specify queries incorporating tagged tokens without modification to the formalism. To reason in terms of the position of the tagged token, we use the "@" operator introduced in [15]; thus $(tag@p_i)$ specifies all states where the tagged token is on place p_i.

7 Example Results

We have implemented support for tagged tokens in the open source PIPE Petri net editor [1,6]. Our extensions allow the user to introduce a tagged token into a net, to identify certain arcs as being tagged arcs and then to verify that this structure conforms to the restrictions laid down in Section 3. The tagged GSPN is then automatically converted into a CGSPN and then unfolded into a standard GSPN which can be described in the input language shared by the DNAmaca [10] and HYDRA [8] performance analysis tools.

We illustrate our tagged token approach through the analysis of patient waiting times in the model of a hospital's Accident and Emergency department shown in Fig. 3. Corresponding rates for the timed transitions and weights for the immediate transitions are given in Table 1. Note that some transitions have functional rates which depend on the marking of places in the system. Typical passage time queries which might be asked in an untagged model of this system might be "what is the time taken to process all the patients in the system?".

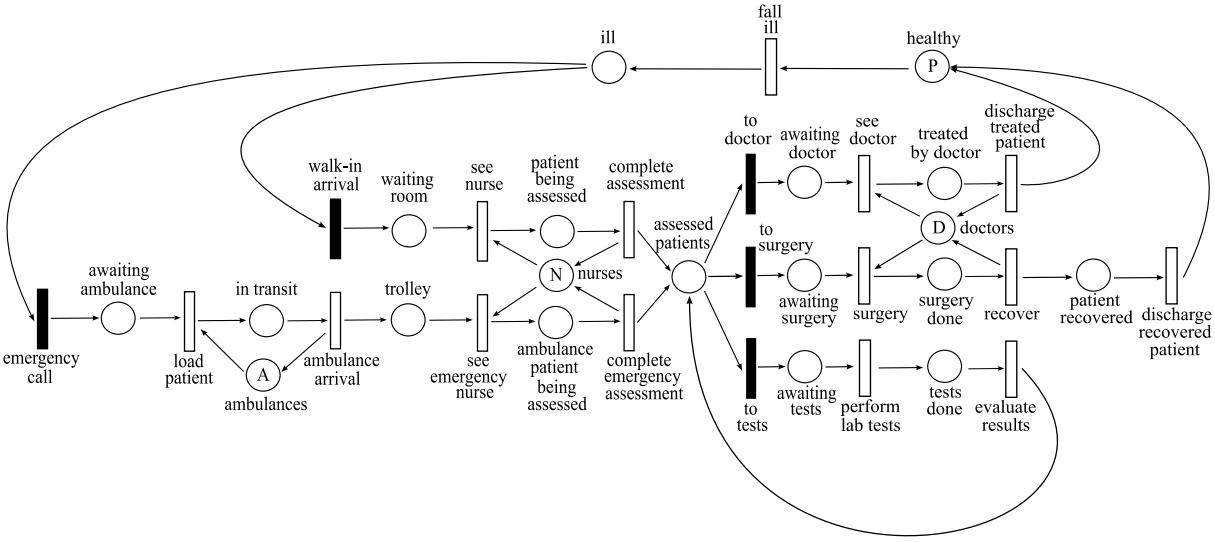


Fig. 3. GSPN model of patient flow in a hospital environment.

Table 1
Transition rates and weights for the GSPN in Fig. 3. Immediate transitions are named in italics.

Transition Name	Rate (patients/hour)/Weight
fall ill	0.1 × *M(healthy)*
walk-in arrival	3.0
emergency call	6.0
see nurse	3.0
complete assessment	3.0 × *M(patient being assessed)*
load patient	6.0
ambulance arrival	6.0 × *M(in transit)*
see emergency nurse	6.0
complete emergency assessment	6.0 × *M(ambulance patient being assessed)*
to doctor	3.0
see doctor	3.0
discharge treated patient	6.0 × *M(treated by doctor)*
to surgery	1.0
surgery	2.0
recover	2.0 × *M(surgery done)*
discharge recovered patient	6.0
to tests	2.0
perform lab tests	3.0
evaluate results	3.0

Using tagged tokens, we can ask queries which relate more naturally to an individual's experience of the modelled system. For example, we might be interested in the distribution of the time taken for one particular customer to pass through the hospital from the moment of admission to when they are finally discharged. This could not be answered without tagged tokens as there is no way to know that the token found on one place at the beginning of the passage corresponds to one found on another place at the end.

We therefore modify the model to include a tagged token. The resulting GSPN is shown in Fig. 4 with a square "tag" attached to each of the tagged arcs. The tagged token starts off on place *healthy*, as indicated by its bold border.

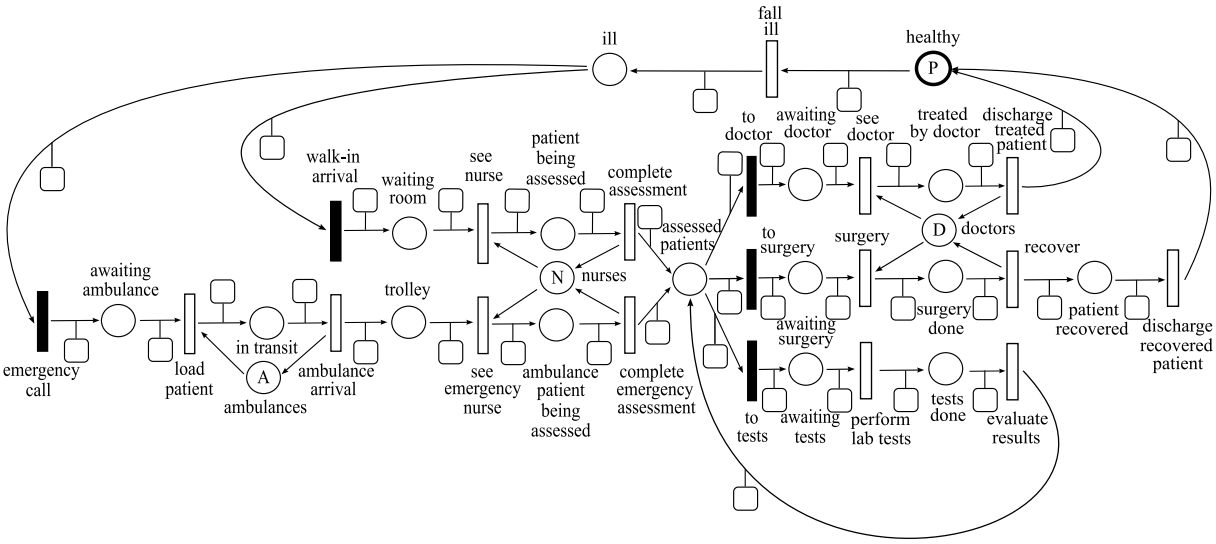


Fig. 4. The hospital model of Fig. 3 modified to support tagged tokens.

Table 2
Number of tangible states generated by the tagged and untagged hospital models in terms of the number of patients (P), nurses (N), doctors (D) and ambulances (A).

Patients (P)	Nurses (N)	Doctors (D)	Ambulances (A)	Number of States	
				Untagged Model	Tagged Model
5	2	2	1	7 260	28 995
7	2	2	1	54 228	273 894
8	2	2	1	207 996	698 922
10	2	2	1	561 704	3 499 265

Augmenting the GSPN to track the tagged token inevitably results in increasing the size of its underlying state space. Table 2 compares the number of tangible states for the tagged and untagged versions of the model for various numbers of patients (P), nurses (N), doctors (D) and ambulances (A).

As we are interested in the time taken by a specific patient to move from admission to discharge, we specify a passage time query using a Performance Tree of the form shown in Fig. 2. We set $t = 4$ hours (in accordance with UK government targets) and define the set of source and target states ($\mathbf{S_1}$ and $\mathbf{S_2}$ respectively) as:

$\mathbf{S_1} := (tag@waiting\ room) \vee (tag@trolley)$

$\mathbf{S_2} := (tag@healthy)$

Fig. 5 shows the Performance Tree for this query as specified in the PIPE tool, while Fig. 6 shows the passage time density and corresponding cumulative distribution function calculated using HYDRA. The value of the cumulative distribution function at $t = 4$ hours is 0.971123, indicating that the 95th percentile of the 4-hour target is met; thus the topmost node of the Performance Tree evaluates to "True".

8 Conclusion

We have presented tagged tokens to enable GSPN models to be analysed for customer-centric performance measures. Our contribution comprises a specification of the

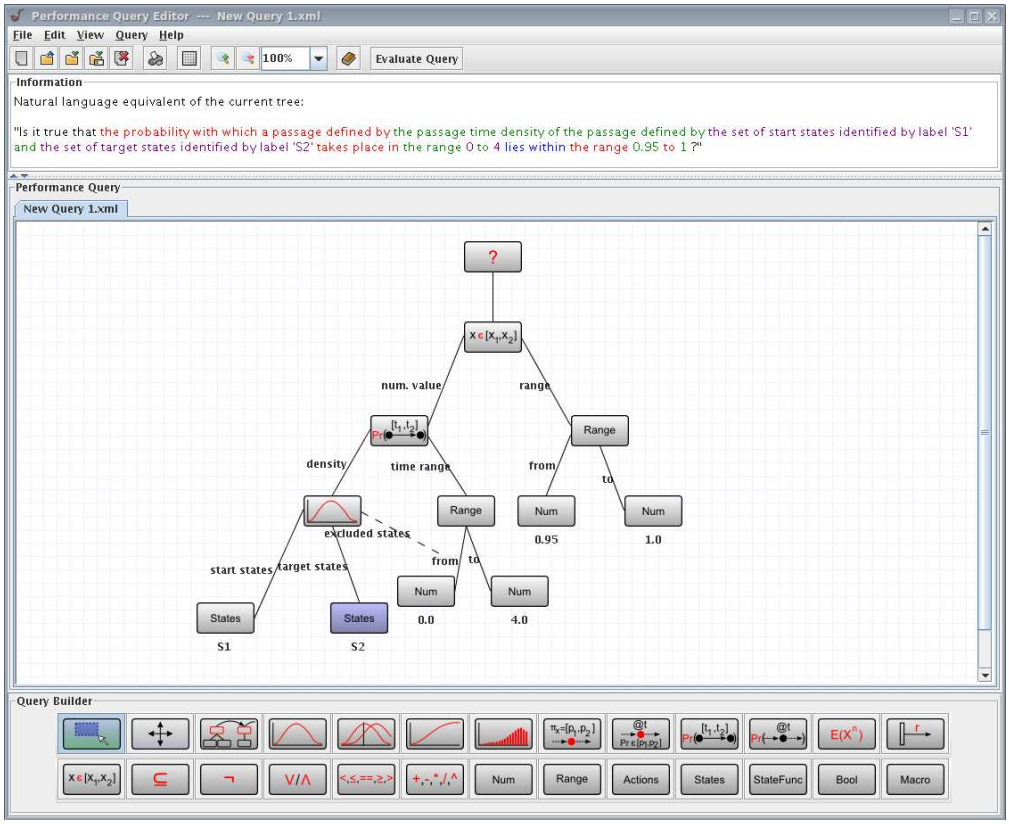


Fig. 5. Specifying the Performance Tree query using the PIPE tool.

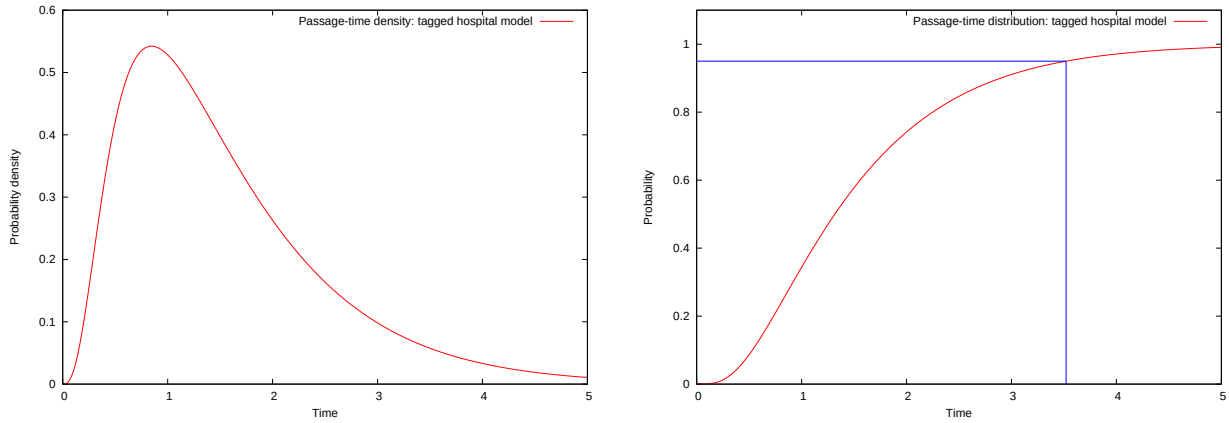


Fig. 6. Probability density (left) and cumulative distribution (right) functions of the time taken for the tagged token to move from admission to discharge in the hospital model with 698 922 states. The 95% quantile is marked on the CDF.

required model restrictions as well as an automated methodology for analysis of GSPNs with tagged tokens. We have further shown how queries involving tagged tokens can be posed using Performance Trees. We have implemented support for the specification and analysis of tagged token models in PIPE, which for the first time provides tool support for the automated analysis of such models. We have presented numerical results for a model of patient flow in a hospital's Accident and Emergency department.

For the future, we are investigating the effect of relaxing the structural restrictions described in Section 3. In particular, it would be interesting to be able to remove the tagged token and reintroduce it when required as it may not be necessary to track the tagged token in all parts of a GSPN. Such an approach would help to mitigate the impact of the growth in the number of states experienced when using tagged tokens. There may also be state lumping strategies which would achieve a similar effect. Finally, we are interested in supporting multiple tagged tokens within a single GSPN, although we are aware that this will further increase the size of the underlying state space.

References

[1] PIPE: Platform-Independent Petri net Editor – http://pipe2.sourceforge.net.

[2] M. Ajmone-Marsan, G. Conte, and G. Balbo. A class of Generalised Stochastic Petri Nets for the performance evaluation of multiprocessor systems. *ACM Transactions on Computer Systems*, 2:93–122, 1984.

[3] A. Argent-Katwala, J.T. Bradley, A. Clark, and S.T. Gilmore. Location-Aware Quality of Service Measurements for Service-Level Agreements. In *Proc. 3rd Symposium on Trustworthy Global Computing (TGC'07)*, volume 4912 of *Lecture Notes in Computer Science*, pages 222–239, Sophia-Antipolis, France, November 2007.

[4] F. Bause and P.S. Kritzinger. *Stochastic Petri Nets – An Introduction to the Theory*. Verlag Vieweg, Wiesbaden, Germany, 1995.

[5] L. Bodrog, G. Horváth, S. Rácz, and M. Telek. A tool support for automatic analysis based on the tagged customer approach. In *Proc. 3rd International Conference on the Quantitave Evaluation of Systems (QEST'06)*, pages 323–332, Riverside, CA, September 2006. IEEE Computer Society.

[6] P. Bonet, C.M. Llado, R. Puijaner, and W.J. Knottenbelt. PIPE v2.5: A Petri net tool for performance modelling. In *Proceedings of the 23rd Latin American Conference on Informatics (CLEI'07)*, San Jose, Costa Rica, October 2007.

[7] G. Ciardo, J.K. Muppula, and K.S. Trivedi. On the solution of GSPN reward models. *Performance Evaluation*, 12(4):237–253, 1991.

[8] N.J. Dingle, P.G. Harrison, and W.J. Knottenbelt. HYDRA: HYpergraph-based Distributed Response-time Analyser. In *Proc. International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'03)*, pages 215–219, Las Vegas NV, USA, June 23rd–26th 2003.

[9] J. Hillston. *A Compositional Approach to Performance Modelling*. PhD thesis, University of Edinburgh, 1994.

[10] W.J. Knottenbelt. Generalised Markovian analysis of timed transition systems. Master's thesis, University of Cape Town, Cape Town, South Africa, July 1996.

[11] M. Li and N.D. Georganas. Coloured Generalized Stochastic Petri Nets for integrated systems protocol performance modelling. *Computer Communications*, 13(7):414–424, 1990.

[12] A.S. Miner. Computing response time distributions using Stochastic Petri Nets and matrix diagrams. In *Proc. 10th International Workshop on Petri Nets and Performance Models (PNPM'03)*, pages 10–19, Urbana-Champaign, IL, September 2nd–5th 2003.

[13] I. Mitrani. *Probabilistic Modelling*. Cambridge University Press, August 1998.

[14] T. Suto, J.T. Bradley, and W.J. Knottenbelt. Performance Trees: A new approach to quantitative performance specification. In *Proc. 14th IEEE/ACM Intl. Symposium on Modeling, Analysis and Simulation of Computer and Telecommunications Systems (MASCOTS'06)*, pages 303–313, Monterey, CA, USA, September 2006.

[15] T. Suto, J.T. Bradley, and W.J. Knottenbelt. Performance Trees: Expressiveness and quantitative semantics. In *Proc. 4th International Conference on the Quantitave Evaluation of Systems (QEST'07)*, pages 41–50, Edinburgh, September 2007. IEEE Computer Society.