Performance Analysis of Resource-Constrained Business Processes: a Formal Approach Based on Stochastic Petri Nets

Cesar A. L. Oliveira and Ricardo M. F. Lima
Center of Informatics, Federal University of Pernambuco
Recife, Brazil
{calo,rmfl}@cin.ufpe.br

March 28, 2009

Abstract

Performance analysis of business processes is a real necessity of middle and large-scale organizations that aim to achieve superior efficiency and competitiveness. We propose a GSPN model for performance evaluation of business processes. We show how fundamental concepts of Workflow Management can be represented through GSPN structures and how to retrieve several valuable performance information from these models. The models can also be used for correctness verification of the workflow (*soundness* property).

1 Introduction

The concept of *systems thinking* is an important instrument for understanding complex systems. In Management, systems thinking led to the conception of *Business Process Management* (BPM) [9] and [10] - a point of view through which the manager can see how the enterprise runs as a whole. It recognizes that a change in one area of a system can affect other parts of the system. BPM promotes business effectiveness and efficiency while striving for innovation, flexibility and integration with technology. A business process is a collection of related, structured activities to create a product or provide a service to meet the needs of clients.

It is a common approach to use information systems to support Business Process Management. Such systems are known as *Business Process Management Systems* (BPMS) or *Workflow Management Systems* (WfMS) [6]. They are employed for automating and monitoring the flow of activities and information along the process. A WfMS receive a Workflow as input. A Workflow is a set of rules that govern the sequence of activities to get a more complex and collaborative work done. The Workflow design is a complicated time-consuming process. Efficiency and correctness are two essential requirements for the design of complex Workflows.

In this context, Petri Nets have been successfully applied as a formal technique for verification [1][13][12][3] and performance evaluation [8][16][11][15] of business Workflows.

Generalized Stochastic Petri Nets (GSPN) [14] is a well known formalism widely used for modeling and evaluating the performance of concurrent systems with parallelism, synchronization, and resource dispute. We propose a GSPN model for performance evaluation of business processes. We show how fundamental concepts of Workflow Management can be represented through GSPN structures and how to retrieve several valuable performance information from these models. The models can also be used for correctness verification of the workflow (*soundness* property).

2 Building Blocks

This section presents the proposed models to represent Workflow systems and to analyze the performance of such systems. Generalized Stochastic Petri Nets (GSPN) are employed as modeling language. The stochastic model can be studied in more detail in order to find relations between system's variables and other properties not easily recognized in an algorithmic-like approach. We present several of these relations.

We start our discussion with the basic entities that must exist in a Workflow system (Activities, roles, worklists and instances) and define GSPN models to represent them. Furthermore, we provide a number of composition rules which allow for the construction of complex business processes containing concurrence, synchronization, loops and so on. These structures are found in several process notations, but there is a lack of uniformity in their terminology. For this reason, we adopt the terminology provided by WfMC [7]. One can refer to the WfMC glossary in order to find synonymous and related terms for a specific notation.

Regarding expressiveness, the models present the following main characteristics:

- represent simultaneous execution of multiple process instances;
- represent resources grouped in Roles that are responsible for executing several Activities (shared resources);
- distinguish Work Items and ongoing Activity Instances;
- assume that Case arrival is a Poisson process;

- assume that service times are exponentially distributed random variables.

Regardless of the assumption of exponential times, it is possible to approximate any random distribution with rational Laplace transform by combining exponential variables. Methods for modeling these distributions with GSPN are well known [5][14] and can be applied in the proposed model in order to improve its representativeness. However, this approach will not be taken into consideration in this paper.

The following information can be obtained by this model:

- soundness verification;
- minimum number of resources demanded by each Role;
- number of ongoing Activity Instances;
- number of Work Items in each Worklist;
- number of available resources in each Role;
- mean time of Case processing (response time).

2.1 Basic Blocks

Next, we describe the basic structures for modeling Workflow and formulae for calculating metrics about them.

A *pool* is a structure that groups the Roles that participate in the process. In most notations, each Role is represented by a *swimlane* in the pool. When an Activity is placed on that swimlane, it means that the respective Role is responsible for the execution of that Activity.

Definition 1 (Pool)**.** The Pool, denoted by $\mathcal{P}$, contains the Roles that participate in the Workflow. It is defined as a set $\mathcal{P} = \{R_1, R_2, ..., R_N\}$, where each R_i is a Role identifier.

Definition 2 (Process Instance)**.** Ongoing Process Instances are represented by *tokens* in the GSPN.

The fundamental structure in the Workflow model is the Activity model. This model represents the execution of an Activity by a resource. Activities are the atomic unity of work.

We used a labels notation to identify each Activity and Role, helping to recognize the elements in the net associated with these structures. Hence, the elements associated with an Activity expressed by the notation $A_i(R_k, d_i)$ are labeled by the same index i, while the Role is a place labeled by index k.

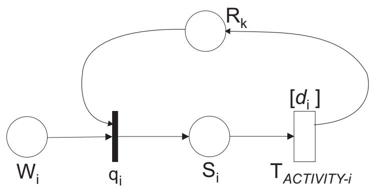

Figure 1: Activity Model in GSPN

Definition 3 (Activity Model)**.** An Activity Model is denoted by $A_i(R_k, d_i)$, where:

1. $R_k \in \mathcal{P}$ is the Role responsible for the execution of the Activity;
2. $d_i \in \mathbb{R}_+^*$ is the mean time delay for the Activity execution.

and corresponds to a GSPN, $A_i(R_k, d_i) = (P_i, T_i, \Pi_i, I_i, O_i, H_i, M_0^i, \omega_i)$, which is defined as follows:

1. $P_i = \{R_k, W_i, S_i\}$, where:
   - R_k is a place that represents the Role R_k;
   - W_i is a place for holding the Activity's Work Items, called *Worklist place*;
   - S_i is a place for containing the Activity Instances, called *Service place.*
2. $T_i = \{q_i, T_{ACTIVITY-i}\}$, where:
   - q_i is an immediate transition, with $\omega_i(q_i) = 1$ e $\Pi_i(q_i) = 1$;
   - $T_{ACTIVITY-i}$ is a timed transition with mean delay d_i and infinite server semantics.
3. and I_i, O_i, are such that:
   (a) the pre-condition ${}^\bullet q_i = \{W_i, R_k\}$ and the post-condition $q_i{}^\bullet = \{S_i\}$;
   (b) the pre-condition ${}^\bullet T_{ACTIVITY-i} = \{S_i\}$ and the post-condition $T_{ACTIVITY-i}{}^\bullet = \{R_k\}$.

We omit parameters R_k and d_i when they are not relevant for the context, using the simplified notation "A_i" for referring to an Activity "$A_i(R_k, d_i)$".

Fig. 1 presents the GSPN for the Activity Model.

This model can be used to represent both manual Activities and automated Activities with limited resources. When an Activity is executed by a software application (for example, a Web Service), it is usually possible to assume infinite resources because the limit of parallel processing is defined by the machine's memory and processor and is likely to be enough for the demand. In these cases, there are two options: 1) if the automated Activity is considered too fast, it can be ignored in the model; 2) if, despite the assumption of infinite resources, the time of execution is considered relevant, the Infinite Resource Activity Model, presented bellow, can be used.

Definition 4 (Infinite Resource Activity Model)**.** An Infinite Resource Activity Model, denoted by $A_i^a(d_i)$, where d_i is the mean execution time delay, corresponds to a GSPN containing:

1. a place S_i, called *Service place*;
2. a timed infinite-server transition $T_{ACTIVITY-i}$ with mean delay d_i, pre-condition ${}^\bullet T_{ACTIVITY-i} = \{S_i\}$ and empty post-condition.

Sometimes, resources in more than one Role can handle the same Activity. For example, a supervisor can decide that himself will execute an Activity that is usually executed by a subordinate. This situation can be though as a set of Activities sharing the same worklist. Each resource is assigned to its own Activity, but work items are shared between them. For this situation we provide the structure named *Multiple-Role Activity Model.*

Definition 5 (Multiple-Role Activity Model)**.** A Multiple-Role Activity Model is denoted by $A_i^m(\mathcal{P}_i, D_i)$, where:

1. $\mathcal{P}_i \subseteq \mathcal{P}$ is the set of Roles that can execute the Activity;
2. $D_i : \mathcal{P}_i \to \mathbb{R}_+^*$ is a function that relates each Role to a time delay that is the mean time for the Activity execution by that Role.

and corresponds to a GSPN, $A_i^m(\mathcal{P}_i, D_i) = (P_i, T_i, \Pi_i, I_i, O_i, H_i, M_0^i, \omega_i)$, which is defined as follows:

1. $P_i = \{W_i, R_k, S_i^k \quad | \quad R_k \in \mathcal{P}_i\}$, where:
    - W_i is a place for holding the Activity's Work Items, called *Worklist place*;
    - Each place R_k is a place that represents a Role R_k;
    - Each place S_i^k is a place for containing the Activity Instances that are being executed by Role R_k, called *Service places.*
2. $T_i = \{q_i^k, T_{ACTIVITY-i}^k \quad | \quad R_k \in \mathcal{P}_i\}$, where:
    - Each q_i^k is an immediate transition, with $\omega_i(q_i^k) = 1$ and $\Pi_i(q_i^k) = 1$;
    - Each $T_{ACTIVITY-i}^k$ is a timed transition with mean delay $D_i(R_k)$ and infinite server semantics.

3. and I_i, O_i, are such that:

   (a) the pre-condition ${}^{\bullet}q_i^k = \{W_i, R_k\}$ and the post-condition ${q_i^k}^{\bullet} = \{S_i^k\}$, for all $R_k \in \mathcal{P}_i$;

   (b) the pre-condition ${}^{\bullet}T^k_{ACTIVITY-i} = \{S_i^k\}$ and the post-condition ${T^k_{ACTIVITY-i}}^{\bullet} = \{R_k\}$, for all $R_k \in \mathcal{P}_i$.

Activity models are composed between them by applying composition rules, which will be defined in later sections. Through these compositions Sub-processes are created.

A Sub-process is a GSPN that attends to the restrictions presented by Def. 6. We denote by *SProc* the set of all GSPNs that form a valid Sub-process.

Definition 6 (Sub-process)**.** A Sub-process is a GSPN $U = (P_U, T_U, \Pi_U, I_U, O_U, H_U, M_0^U, \omega_U)$, such that

1. there exists a unique place $Sp \in P_U$, such that ${}^{\bullet}Sp = \emptyset$, called *Starting Place*;

2. there exists a non-empty set of transitions $Dt \subseteq T_U$, such that $\forall t \in Dt \ : \ t^{\bullet} = \emptyset$, called *Departing Transitions*;

3. for each token arriving at Starting Place Sp, exactly one token departs from the Sub-process through any one of the transitions in the set Dt.

The simplest Sub-process is that of a single Activity Model. The Starting Place of the Activity Model A_i is place W_i and the Departing Transitions is a set defined by $Dt = \{T_{ACTIVITY-i}\}$. Hence, every Activity is also a Sub-process.

Fig. 2 displays a Process Definition represented in the BPMN notation [17]. It exemplifies a process containing a single Activity. The input circle is the *start event*, which represents the beginning of the Workflow execution and the output circle is the *end event*, which represents the end of the execution.

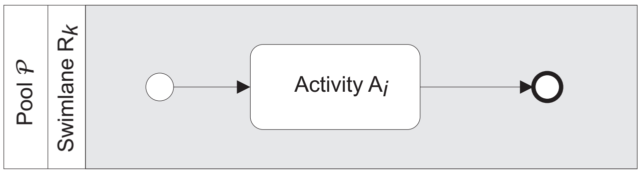

Figure 2: BPMN Workflow with a single Activity

When Sub-processes are composed, their respective GSPNs are united. Def. 7 presents a definition for GSPN union operation.

Definition 7 (GSPN Union). Let *GSPNSet* be the set of all existing GSPNs, the operation of uniting two GSPNs can be defined as follows:

$$\cup : \mathit{GSPNSet} \times \mathit{GSPNSet} \to \mathit{GSPNSet}$$

$G_3 = G_1 \cup G_2$, where:

1. $G_1 = (P_1, T_1, \Pi_1, I_1, O_1, H_1, M_0^1, \omega_1)$;
2. $G_2 = (P_2, T_2, \Pi_2, I_2, O_2, H_2, M_0^2, \omega_2)$;
3. $G_3 = (P_3, T_3, \Pi_3, I_3, O_3, H_3, M_0^3, \omega_3)$;
4. $P_3 = P_1 \cup P_2$;
5. $T_3 = T_1 \cup T_2$;
6. $$I_3(p,t) = \begin{cases} I_1(p,t) & \text{if} \quad p \in P_1 \quad \text{and} \quad t \in T_1 \\ I_2(p,t) & \text{if} \quad p \in P_2 \quad \text{and} \quad t \in T_2 \\ 0 & \text{otherwise} \end{cases}$$
7. $$O_3(p,t) = \begin{cases} O_1(p,t) & \text{if} \quad p \in P_1 \quad \text{and} \quad t \in T_1 \\ O_2(p,t) & \text{if} \quad p \in P_2 \quad \text{and} \quad t \in T_2 \\ 0 & \text{otherwise} \end{cases}$$
8. $$H_3(p,t) = \begin{cases} H_1(p,t) & \text{if} \quad p \in P_1 \quad \text{and} \quad t \in T_1 \\ H_2(p,t) & \text{if} \quad p \in P_2 \quad \text{and} \quad t \in T_2 \\ 0 & \text{otherwise} \end{cases}$$
9. $$\omega_3(t) = \begin{cases} \omega_1(t) & \text{if} \quad t \in T_1 \\ \omega_2(t) & \text{if} \quad t \in T_2 \end{cases}$$
10. $$\Pi_3(t) = \begin{cases} \Pi_1(t) & \text{if} \quad t \in T_1 \\ \Pi_2(t) & \text{if} \quad t \in T_2 \end{cases}$$
11. $$M_0^3(p) = \begin{cases} M_0^1(p) & \text{if} \quad p \in P_1 \\ M_0^2(p) & \text{if} \quad p \in P_2 \end{cases}$$

When performing the union of GSPNs, two elements of the same name present in the original GSPNs are merged in the resulting GSPN. For example, if G_1 contains a place named P and G_2 also contains a place P, the union will cause these places to be considered the same single place in the resulting GSPN. This behavior is coherent with the mathematical union operator and is desired in our model because we want Role places to be merged when they represent the same Role. Def. 8 explicit this property.

Definition 8 (Role Places Merging)**.** Let $R_k \in \mathcal{P}$ be a Role and $A_1(R_k, d_1)$, $A_2(R_k, d_2)$ any two Activities. If these Activities are composed, place R_k of each model will be the same single place in the resulting GSPN.

Following, we define some auxiliary functions to be used along this section.

Definition 9 (Starting Place Function)**.** For a Sub-process $U = (P_U, T_U, \Pi_U, I_U, O_U, H_U, M_0^U, \omega_U)$, we denote by $\mathit{Start}(U)$ the unique place $Sp \in P_U$ such that ${}^\bullet Sp = \emptyset$, called the Starting Place of U.

Definition 10 (Departing Transitions Function)**.** For a Sub-process $U = (P_U, T_U, \Pi_U, I_U, O_U, H_U, M_0^U, \omega_U)$, we denote by $\mathit{End}(U)$ the set of transitions $Dt \subseteq T_U$ such that $\forall t \in Dt \, : \, t^\bullet = \emptyset$, which correspond to the Departing Transitions set of U.

A Sub-process model represents the *Process Definition.* In order to evaluate the performance of the Workflow, one must insert this model in a system, where customers and resources are present. We define this as *Workflow System*, defined in Def. 11.

Definition 11 (Workflow System)**.** A Workflow System, defined as a tuple $\mathit{Wf} = (\lambda, \mathcal{P}, U, \mathit{Emp})$, where:

1. $\lambda \in \mathbb{R}_+^*$ is the arrival rate, which indicates the rate at which Cases are produced to the system;
2. $\mathcal{P}$ is a Pool;
3. $U \in \mathit{SProc}$ is the Sub-process model that contains the Process Definition;
4. $\mathit{Emp} : \mathcal{P} \rightarrow \mathbb{N}_+$ is a Employing function, which assigns a number of resources to each Role.

is equivalent to a GSPN which is composed by U with the following additional elements:

1. a timed transition $T_{ARRIVAL}$ with mean delay $d = 1/\lambda$, empty pre-condition and post-condition given by $T_{ARRIVAL}{}^\bullet = \{\mathit{Start}(U)\}$;
2. a place P_D with pre-condition ${}^\bullet P_D = \mathit{End}(U)$;
3. an immediate transition q_D with pre-condition ${}^\bullet q_D = \{P_D\}$ and empty post-condition;
4. a initial marking function M_0 such that $M_0(R_j) = \mathit{Emp}(r_j), \forall r_j \in \mathcal{P}$, where $R_j \in P_U$ is the place representing role $r_j \in \mathcal{P}$.

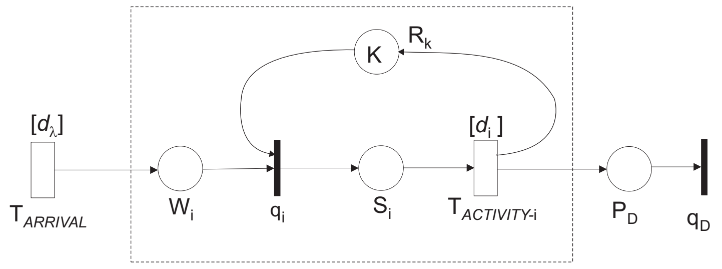

Figure 3: Simplest Workflow System

Fig. 3 presents the simplest Workflow System model. The Sub-process contains just one Activity and is highlighted by the dashed square. Notice that place R receives an initial marking K, corresponding to the value of $Emp(R)$.

Place P_D and transition q_D together can be viewed as a Sub-process, which we name *Case Departure Point*. This Sub-process can be used in compositions which have the end of the process as target.

2.2 Metrics for the Basic Blocks

The following metrics can be evaluated for these basic blocks.

Definition 12 (Minimum Number of Resources for a Role)**.** Let R_k be a Role with K resources that perform a set of activities $A_1(R_k, d_1), A_2(R_k, d_2), \ldots, A_N(R_k, d_N)$ in a Workflow System, a stationary solution for this Workflow System exists only if:

$$K > \sum_{i=1}^{N} \lambda_i d_i \ ,$$

where λ_i is the rate at which Cases arrive at Activity A_i.

Notice that arrival rate for each Activity can be different, due to the characteristics of the Case flow inside the process.

Definition 13 (Expected Number of Activity Instances)**.** For an Activity A_i with mean delay d_i and arrival rate λ_i, provided with sufficient resources, the expected number of Activity Instances during the Workflow execution is given by:

$$E(S_i) = \lambda_i d_i \ .$$

Table 1: Metrics for the basic models

Metric	Expression
Minimum Number of Resources	$K > \sum_{i=1}^{N} \lambda_i d_i$
Expected Number of Activity Instances	$E(S) = \lambda d$
Expected Number of Available Resources	$E(R) = K - \sum_{i=1}^{N} E(S_i)$
Expected Number of Work Items	$E(W) =$ expectation of place W
Expected Number of Cases	$E(n) = E(W) + E(S)$
Mean Response Time	$E(\tau) = E(n)/\lambda$

Definition 14 (Expected Number of Available Resources)**.** For a Role R_k with K resources and performing Activities $A_1, \ldots, A_N$, the mean number of available resources in the Workflow execution is given by:

$$E(R_k) = K - \sum_{i=1}^{N} E(S_i) \ ,$$

where $E(S_i)$ is the expected number of instances of A_i.

Definition 15 (Expected Number of Work Items)**.** For an Activity A_i, the mean number of *Work Items* of this Activity during the Workflow execution is equal to the expected marking of place W_i.

Definition 16 (Expected Number of Cases)**.** For an Activity A_i, the mean number of Cases being processed by this Activity during the Workflow execution is given by:

$$E(n_i) = E(W_i) + E(S_i) \ .$$

Definition 17 (Expected Activity Response Time)**.** Let A_i be an Activity with Case arrival rate λ_i and mean service time d_i, the mean Activity's response time is given by:

$$E(\tau_i) = \frac{E(n_i)}{\lambda_i} \ ,$$

where $E(n_i)$ is the mean number of Cases in A_i.

Table 1 summarizes the metrics defined for the basic models.

3 Composition Rules

The composition of basic structures allows for the creation of complex Sub-processes. For this purpose, we provide a set of composition rules (or operators). Each operator composes the Sub-processes in a different manner, creating different flow relations between them.

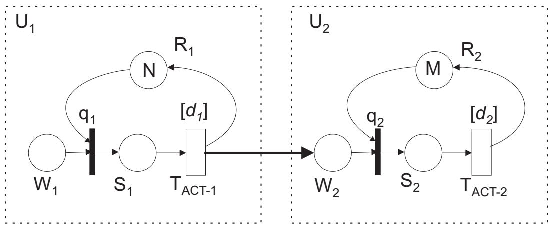

Figure 4: Sequence composition (SEQ)

Sub-processes composition is performed by union of their GSPN models and the addition of auxiliary (control) elements. For the sake of simplicity, we assign simple names to these auxiliary elements. However, one must keep in mind that such elements must receive unique labels when constructing the model, in order to avoid homonymous elements, which are merged by union operations.

3.1 Sequence

This operator combines to Sub-processes, U_1 and U_2, with a sequential relation such that U_2 is executed after U_1. The model is constructed by adding an arc connecting each departing transition of U_1 to the starting place of U_2. This is illustrated in Fig. 4.

Definition 18 (Sequence Operator – SEQ)**.**

$$SEQ : SProc \times SProc \rightarrow SProc$$

$SEQ(U_1, U_2) = U_R$, where:

$U_R = U_1 \cup U_2$, with the addition of an arc such that:

1. $Start(U_R) = Start(U_1)$;
2. $End(U_R) = End(U_2)$;
3. ${}^{\bullet}Start(U_2) = End(U_1)$.

For notation simplification, it is possible to use a more general operator $SEQ(U_1, U_2, \ldots, U_N)$ (multiple arguments), as an abbreviation to the composition $SEQ(U_1, SEQ(U_2, \ldots SEQ(U_{N-1}, U_N)))$, without differences in the resulting model.

3.2 Metrics for the Sequence

The following metrics can be obtained for a sequential composition Sub-process.

Definition 19 (Internal Arrival Rate in the Sequence)**.** Let $SEQ(U_1, U_2)$ be a Sub-process with Case arrival rate λ, the rates λ_1, λ_2 of Case arrival in each component U_1, U_2, respectively, are given by:

$$\lambda_1 = \lambda_2 = \lambda$$

Definition 20 (Expected Number of Cases in the Sequence)**.** Let $SEQ(U_1, U_2)$ be a Sub-process, the expected number of Cases inside this Sub-process during Workflow execution is given by:

$$E(n_{SEQ}) = E(n_1) + E(n_2) \ ,$$

where $E(n_i)$ is the mean number of Cases inside the Sub-process U_i.

Definition 21 (Expected Response Time of the Sequence)**.** Let $SEQ(U_1, U_2)$ be a Sub-process, the mean time spent by a Case executing the Sub-process is given by:

$$E(\tau_{SEQ}) = \frac{E(n_{SEQ})}{\lambda} \ ,$$

where λ is the arrival rate at the Sub-process.

3.3 Alternative Path (XOR)

This operator combines a set of N Sub-processes $(U_1, \ldots, U_N)$ in a way that they are alternatively executed. Each Case arriving is forwarded to one of these Sub-processes (which we call *paths*), according to a probability distribution defined by a function Pr. For each Sub-process U_i, a probability $Pr(U_i)$ for the Case be routed to that Sub-process is assigned.

This composition is modeled by the addition of a place P_{XOR}, which is the starting place of the Sub-process, a set of immediate transitions $qx_1, \ldots, qx_N$, which removes a token from P_{XOR} and put it in the starting place of Sub-process $U_1, \ldots, U_N$, respectively. Each transition receive a weight $\omega(qx_i)$ equal to the probability $Pr(U_i)$ of the Sub-process U_i be chosen.

This model is shown in Fig. 5.

Definition 22 (Alternative Path Operator – XOR)**.** Let $U_1, U_2, \ldots, U_N$ be Sub-processes and Pr a probability distribution function

$$XOR : \mathit{SProc} \times \ldots \times \mathit{SProc} \times (\mathit{SProc} \rightarrow \mathbb{R}[0;1]) \rightarrow \mathit{SProc}$$

$U_R = XOR(U_1, U_2, \ldots, U_N, Pr)$, where:

1. Let G_{XOR} be a GSPN containing a place P_{XOR} and immediate transitions
   $qx_1, \ldots, qx_N$, with $P_{XOR}{}^\bullet = \{qx_i\}, \quad i = 1, \ldots, N;$

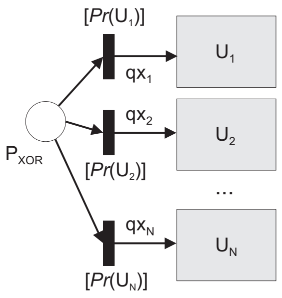

Figure 5: Alternative path composition (XOR)

2. $\omega(qx_i) = Pr(U_i), \quad i = 1, \ldots, N$;
3. $U_R = U_1 \cup \ldots \cup U_N \cup G_{XOR}$, with the addition of arcs such that:
   (a) $Start(U_R) = P_{XOR}$;
   (b) $End(U_R) = End(U_1) \cup \ldots \cup End(U_N)$;
   (c) ${}^\bullet Start(U_i) = \{qx_i\}, \quad i = 1, \ldots, N$.

3.4 Metrics for the Alternative Path

The following metrics can be measured for the Alternative Path composition.

Definition 23 (Internal Arrival Rate in the Alternative Path)**.** Let $XOR(U_1, U_2, \ldots, U_N, Pr)$ be a Sub-process with arrival rate λ, the internal arrival rates $\lambda_1, \ldots, \lambda_N$ in each of its components $U_1, \ldots, U_N$, respectively, is given by:

$$\lambda_i = \lambda Pr(U_i), \quad i = 1, \ldots, N \ .$$

Definition 24 (Expected Number of Cases in the Alternative Path)**.** Let $XOR(U_1, U_2, \ldots, U_N, Pr)$ be a Sub-process, the expected number of Cases inside it during Workflow execution is given by:

$$E(n_{XOR}) = \sum_{i=1}^{N} E(n_i) \ ,$$

where $E(n_i)$ is the mean number of Cases in Sub-process U_i.

Definition 25 (Expected Response Time of the Alternative Path)**.** Let $XOR(U_1, U_2, \ldots, U_N, Pr)$ be a Sub-process, the mean time spent by a Case in the execution of this Sub-process is:

$$E(\tau_{XOR}) = \frac{E(n_{XOR})}{\lambda} \ ,$$

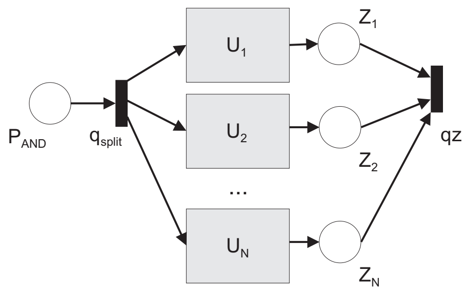

Figure 6: Parallel composition (AND)

where λ is the arrival rate at the Sub-process.

3.5 Parallel Execution (AND)

This operator creates a Sub-process that consists of the parallel execution of N other Sub-processes that compose it. Each arriving Case is sent to all of these Sub-processes simultaneously to be processed by them. Synchronization occurs before the departure of the Case, in a way that it leaves the Sub-process only after every parallel process have been done.

This composition is modeled by the addition of an initial structure, responsible for splitting the tokens that arrive and another structure in the exit, responsible for the synchronization and for merging the tokens back. This model is presented in Fig. 6.

Definition 26 (Parallel Operator – AND)**.**

$$AND : SProc \times \ldots \times SProc \times \rightarrow SProc$$

$U_R = AND(U_1, U_2, \ldots, U_N)$, where:

1. Let G_{AND} be a GSPN containing place P_{AND}, immediate transition q_{split}, with $P_{AND}{}^{\bullet} = \{q_{split}\}$, a set of places $\{Z_1, \ldots, Z_N\}$ and another immediate transition qz, such that ${}^{\bullet}qz = \{Z_1, \ldots, Z_N\}$;

2. $U_R = U_1 \cup \ldots \cup U_N \cup G_{AND}$, with the addition of arcs in such a way that:

   (a) $Start(U_R) = P_{AND}$;

   (b) $End(U_R) = \{qz\}$;

   (c) ${}^{\bullet}Start(U_i) = q_{split}, \quad i = 1, \ldots, N$;

   (d) ${}^{\bullet}Z_i = End(U_i), \quad i = 1, \ldots, N$.

3.6 Metrics for the Parallel Execution

The following metrics can be obtained for the Parallel Execution Sub-process.

Definition 27 (Internal Arrival Rate in the Parallel Execution)**.** Let $AND(U_1, U_2, \ldots, U_N)$ be a Sub-process with Case arrival rate λ, the internal arrival rates $\lambda_1, \ldots, \lambda_N$ in each component $U_1, \ldots, U_N$, respectively, is given by:

$$\lambda_i = \lambda, \quad i = 1, \ldots, N \ .$$

Definition 28 (Expected Number of Cases in Parallel Execution)**.** Let $AND(U_1, U_2, \ldots, U_N)$ be a Sub-process, the expected number of Cases inside it in the Workflow Execution is given by:

$$E(n_{PAR}) = \frac{1}{N} \sum_{i=1}^{N} \Big[E(n_i) + E(Z_i) \Big] \ ,$$

where $E(n_i)$ is the mean number of Cases inside Sub-process U_i and $E(Z_i)$ is the expected marking of place Z_i.

Definition 29 (Expected Synchronization Time of the Parallel Execution)**.** Let $AND(U_1, U_2, \ldots, U_N)$ be a Sub-process, the mean time spent by a Case waiting for synchronization in the Sub-process is given by:

$$E(\zeta_{PAR}) = \frac{1}{N\lambda} \sum_{i=1}^{N} E(Z_i) \ ,$$

where λ is the Case arrival rate at the Sub-process and $E(Z_i)$ is the expected marking of place Z_i.

Definition 30 (Expected Response Time in Parallel Execution)**.** Let $AND(U_1, U_2, \ldots, U_N)$ be a Sub-process, the mean time spent by a Case inside this Sub-process is:

$$E(\tau_{PAR}) = \frac{E(n_{PAR})}{\lambda} \ ,$$

where λ is the Case arrival rate at the Sub-process.

3.7 Iteration

An Iteration is a Sub-process executed several times for processing the same Case. In one or more points of the Sub-process' execution a decision is made about whether the Case must continue iterating or leave the structure.

The iterative structure in our model needs that a single entry point exist for the iteration, but several exit points are allowed. When there is only one exit point and no Activity exists in the return path from the exit point to the entry point, we simplify the structure and call it *Simple Iteration*, created by the LOOP operator. Otherwise, we use the more general model, called *Grid Iteration*, constructed by the GRID-LOOP operator.

Fig. 7 depicts the Simple Iteration model and Fig. 8 shows a Grid Iteration model example, with two exit points.

Note that Simple Iteration is created by a single Sub-process while Grid Iteration is a more complex composition. It is worthy of notice that the Grid Iteration contains both the iterative path – i.e. the Sub-processes that will execute repeatedly – and the exiting paths – i.e. the Sub-process executed *after* the Case leaves through each exit point (see Fig. 8). This is necessary to assure correctness of the model while maintaining uniformity in the composition rules.

A probability must be assigned to each exit point, indicating the probability of a Case to exit from the loop by that point. This is a constant value, independent of the number of iterations executed.

The LOOP operation, which creates a Simple Iteration, is created by adding a decision structure to the end of the Sub-process. Two immediate transitions are used to divide the flow. One corresponds to the departure while the other is used to return the token to the starting place. This model is presented in Fig. 7.

Definition 31 (Simple Iteration Operator – LOOP)**.** Let U_1 be a Sub-process and θ the probability of leaving the iterative loop, the Simple Iteration operator can be defined as

$$LOOP : SProc \times \mathbb{R}[0;1] \rightarrow SProc$$

$LOOP(U_1, \theta) = U_R$, where:

1. Let G_{LOOP} be a GSPN consisting of a place PL and two immediate transitions q_{back} and q_{out}, such that $PL^{\bullet} = \{q_{back}, q_{out}\}$;
2. $\omega(q_{out}) = \theta$;
3. $\omega(q_{back}) = 1 - \theta$;
4. $U_R = U_1 \cup G^{LOOP}$, with the addition of arcs in such a way that:
   (a) $\forall t \in End(U_1), \quad t^{\bullet} = \{PL\}$;
   (b) ${q^{back}}^{\bullet} = \{Start(U_1)\}$;
   (c) $Start(U_R) = Start(U_1)$;
   (d) $End(U_R) = \{q_{out}\}$.

The GRID-LOOP operator model is created by the addition of decision structures between the iterating Sub-processes, representing each exit point.

In these points, two immediate transitions divide the flow, routing tokens two the next looping Sub-process or to the exit path Sub-process, according to a probability distribution function Pr. The last Sub-process is connected to the starting place of the structure, defining the cycle. This model is outlined in Fig. 8.

Definition 32 (Grid Iteration Operator – GRID-LOOP)**.** Let k be the number of exit points $p_1, \dots, p_k$; let $UI_1, \dots, UI_{k+1}$ be Sub-processes that compose the iterative cycle, intercalated by the exit points and $UO_1, \dots, UO_k$ be Sub-processes targeted by each one of the exit points, respectively; and let Pr be a probability distribution function that assigns to each point p_i an exit probability $Pr(p_i), i = 1, \dots, k$, we define the GRID-LOOP operator as follows

$$GRID - LOOP : \mathbb{P}(SProc) \times \mathbb{P}(SProc) \times (SProc \to \mathbb{R}[0;1]) \to SProc$$

$$GRID - LOOP(\{UI_1, \dots, UI_{k+1}\}, \{UO_1, \dots, UO_k\}, Pr) = U_R, \text{ where:}$$

1. Let G_{GLOOP} be a GSPN containing k places $PL_1, \dots, PL_k$; k immediate transitions $qo_1 \dots, qo_k$; and other k immediate transitions $qb_1, \dots, qb_k$, with:
   (a) $\omega(qo_i) = Pr(p_i), \quad i = 1, \dots k;$
   (b) $\omega(qb_i) = 1 - Pr(p_i), \quad i = 1, \dots k;$
2. $U_R = UI_1 \cup \dots \cup UI_{k+1} \cup UO_1 \dots \cup UO_k \cup G_{GLOOP}$, with the addition of arcs in such a way that:
   (a) ${PL_i}^\bullet = \{qo_i, qb_i\}, \quad i = 1, \dots k;$
   (b) ${}^\bullet PL_i = End(UI_i), \quad i = 1, \dots k;$
   (c) ${qb_i}^\bullet = \{Start(UI_{i+1})\}, \quad i = 1, \dots k;$
   (d) ${qo_i}^\bullet = \{Start(UO_i)\}, \quad i = 1, \dots k.$
   (e) $\forall t \in End(UI_{k+1}), \quad t^\bullet = \{Start(UI_1)\};$
   (f) $Start(U_R) = Start(UI_1);$
   (g) $End(U_R) = End(UO_1) \cup \dots \cup End(UO_k).$

3.8 Iteration Metrics

The following metrics can be obtained from Simple Iteration Sub-process and from Grid Iteration Sub-process.

Definition 33 (Internal Arrival Rate in Simple Iteration)**.** Let $LOOP(U_1, \theta)$ be a Sub-process with Case arrival rate λ, the internal arrival rate λ_1 in Sub-process U_1 is given by:

$$\lambda_1 = \frac{\lambda}{\theta} \ .$$

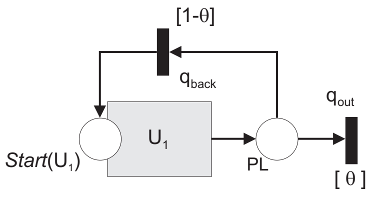

Figure 7: Iterative Composition (LOOP)

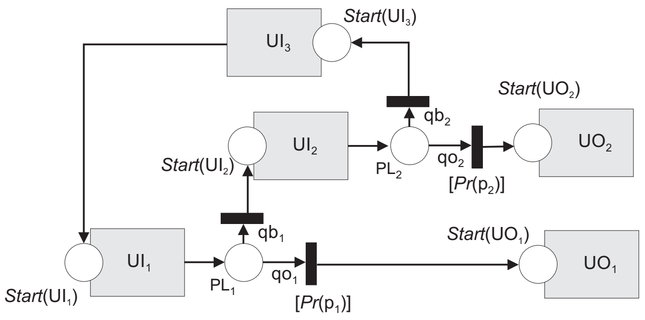

Figure 8: Grid-form Iterative Composition with two exit points (GRID-LOOP)

Definition 34 (Expected Number of Cases in Simple Iteration)**.** Let $LOOP(U_1, \theta)$ be a Sub-process, the expected number of Cases in this Sub-process in the Workflow execution is:

$$E(n_{LOOP}) = E(n_1) \ ,$$

where $E(n_1)$ is the expected number of Cases in Sub-process U_1.

Definition 35 (Expected Time in the Simple Iteration)**.** Let $LOOP(U_1, \theta)$ be a Sub-process, the mean time spent by a Case in this Sub-process is given by:

$$E(\tau_{LOOP}) = \frac{E(n_{LOOP})}{\lambda} \ ,$$

where λ is the Case arrival rate.

The equation in Def. 35 considers the time necessary for the Case to exit from the structure. Hence, it is the sum of the times spent in each cycle.

Definition 36 (Internal Arrival Rate in Grid Iteration)**.** Let $GRID - LOOP(\{UI_1, \ldots, UI_{k+1}\}, \{UO_1, \ldots, UO_k\}, Pr)$ be a Sub-process with Case arrival rate λ, the arrival rates ${\lambda^I}_i$ in each Sub-process UI_i is calculated as follows:

$$\lambda^I{}_1 = \frac{\lambda}{\sum_{j=1}^{k} Pr(p_j)} ;$$

$$\lambda^I{}_i = \lambda^I{}_1 \prod_{v=1}^{i-1} (1 - Pr(p_v)) .$$

And the arrival rate λ_i^O in each Sub-process UO_i is given by:

$$\lambda^O{}_i = \lambda^I{}_1 \prod_{v=1}^{i} Pr(p_v) .$$

Definition 37 (Expected Number of Cases in Grid Iteration)**.** Let $GRID{-}LOOP(\{UI_1, \ldots, UI_{k+1}\}, \{UO_1, \ldots, UO_k\}, Pr)$, be a Sub-process, the expected number of Cases in this Sub-process in the Workflow execution is computed by:

$$E(n_{G-LOOP}) = \sum_{i=1}^{k+1} E(n^I{}_i) + \sum_{i=1}^{k} E(n^O{}_i) ,$$

where $E(n^I{}_i)$ is the mean number of Cases in Sub-process UI_i and $E(n^O{}_i)$ is the mean number of Cases in Sub-process UO_i.

Usually, we are most interested in the number of Cases only in the iteration cycle, not the complete structure. This value corresponds to:

$$E(n^I{}_{G-LOOP}) = \sum_{i=1}^{k+1} E(n^I{}_i) .$$

Definition 38 (Expected Time in the Grid Iteration Cycle)**.** Let $GRID{-}LOOP(\{UI_1, \ldots, UI_{k+1}\}, \{UO_1, \ldots, UO_k\}, Pr)$ be a Sub-process, the expected time spent by a Case inside the cycle part of the structure is given by:

$$E(\tau^I{}_{G-LOOP}) = \frac{E(n^I{}_{G-LOOP})}{\lambda} ,$$

where λ is the Case arrival rate at the Sub-process.

The mean total time in the GRID-LOOP structure is given by the sum of times in the cycle part and in the last Sub-process, which is executed after an exit point is taken. The probability of each path being taken must be considered for obtaining the final value.

Definition 39 (Total Expected Time in Grid Loop)**.** Let $GRID{-}LOOP(\{UI_1, \ldots, UI_{k+1}\}, \{UO_1, \ldots, UO_k\}, Pr)$ be a Sub-process, the mean time spent by a Case in this Sub-process in the Workflow execution is computed by:

$$E(\tau_{G-LOOP}) = E(\tau^{I}{}_{G-LOOP}) + \sum_{v=1}^{k} \prod_{i=1}^{v} Pr(p_i) E(\tau^{O}{}_{i}) \ ,$$

where $E(\tau^{O}{}_{i})$ is the expected time spent inside Sub-process UO_i.

3.9 Multiple Paths with Synchronization (OR)

This composition correspond to patterns *Multi-Choice* (WCP6) and *Structured Synchronizing Merge* (WCP7). It represents a point in the flow where a Case can be routed to one or more paths in a non-exclusive and parallel way. In the departure side, instances that gone through a single path can leave immediately, while instances that gone through multiple paths must wait for the synchronization with its counterparts before leaving.

For the sake of simplification, we present the operator for the case we have two Sub-processes, U_1 and U_2. The Case can be routed to the first, the last or both. This model can be generalized to N paths without much difficulty.

We assign a probability α for the Case to decide for going to U_1, β for the decision for U_2 and π for the decision of going to both paths in parallel.

The model for this composition is displayed in Fig. 9. This model requires the use of seven auxiliary places. A place P_{OR} represents the decision point. Two places, C_1 and C_2, collect tokens that depart from U_1 and U_2, respectively. Other two places, H_1 and H_2, are used to count the number of tokens that are inside Sub-processes U_1 and U_2, respectively, which are instances of a replicated Case. Finally, places Z_1 and Z_2 are used to synchronize tokens which correspond to the same Case, once they depart from each Sub-process.

When a Case arrives at P_{OR}, immediate transitions are enabled. Each one represents one of the three possible decisions. If Sub-process U_1 is chosen, the token is removed from P_{OR} and sent to the starting place of U_1. Analogously, a token is put in U_2 when it is chosen. When the choice is for going to both paths in parallel, the token is removed from P_{OR} and one token is sent to each Sub-process. Also, places H_1 and H_2 receive one token each, indicating that there is one token in U_1 and another in U_2 that are representing the same Case.

Once a token departs from one of the Sub-processes, say, U_1, the token in H_1 is removed and one token is put in Z_1, where it will wait for synchronization until a token exit from U_2 and is put in Z_2.

If the Case have decided for going through a single path, no token is put in H_1 or H_2. Hence, the token that departs from the Sub-process is immediately routed to outside the structure.

Definition 40 (Multiple Path with Synchronization Operator – OR)**.** Let

U_1, U_2 be two Sub-processes, this operator is defined as

$$OR : \mathit{SProc} \times \mathit{SProc} \times \mathbb{R}_+ \times \mathbb{R}_+ \times \mathbb{R}_+ \rightarrow \mathit{SProc}$$

$OR(U_1, U_2, \alpha, \pi, \beta) = U_R$, where:

1. Let G_{OR} be a GSPN containing a place P_{OR} and immediate transitions q_1, q_2, q_s, such that

   (a) ${P_{OR}}^\bullet = \{q_1, q_2, q_s\}$;

   (b) $\omega(q_1) = \alpha$;

   (c) $\omega(q_2) = \beta$;

   (d) $\omega(q_s) = \pi$.

   and places $C_1, C_2, H_1, H_2, Z_1, Z_2$, immediate transitions $qbp_1, qsc_1, qbp_2, qsc_2, qz$, such that:

   (a) ${}^\bullet H_1 = {}^\bullet H_2 = \{q_s\}$;

   (b) ${C_i}^\bullet = \{qbp_i, qsc_i\}, \quad i = 1, 2$;

   (c) ${H_i}^\bullet = \{qsc_i\}, \quad i = 1, 2$;

   (d) ${H_i}^\circ = \{qbp_i\}, \quad i = 1, 2$;

   (e) ${qsc_i}^\bullet = \{Z_i\}, \quad i = 1, 2$;

   (f) ${}^\bullet qz = \{Z_1, Z_2\}$;

   (g) $\omega(qbp_1) = \alpha/(\alpha + \pi)$;

   (h) $\omega(qsc_1) = \pi/(\alpha + \pi)$;

   (i) $\omega(qbp_2) = \beta/(\beta + \pi)$;

   (j) $\omega(qsc_2) = \pi/(\beta + \pi)$.

2. $U_R = U_1 \cup U_2 \cup G_{XOR}$, with the addition of arcs in such a way that:

   (a) ${}^\bullet \mathit{Start}(U_i) = \{q_i, q_s\}, \quad i = 1, 2$;

   (b) ${}^\bullet C_i = \mathit{End}(U_i), \quad i = 1, 2$;

   (c) $\mathit{Start}(U_R) = P_{OR}$;

   (d) $\mathit{End}(U_R) = \{qbp_1, qbp_2, qz\}$.

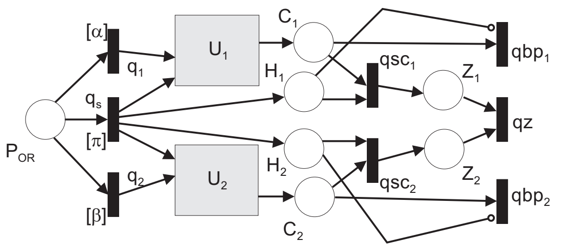

Figure 9: Multiple path with synchronization composition (OR)

3.10 Metrics for Multiple Path with Synchronization

We can obtain the following metrics for Multiple Path with Synchronization.

Definition 41 (Internal Arrival Rate at Multiple Path with Synchronization)**.** Let $OR(U_1, U_2, \alpha, \pi, \beta)$ be a Sub-process with Case arrival rate λ, the internal arrival rates λ_1, λ_2 at each component U_1, U_2, respectively, is calculated by:

$$\lambda_1 = (\alpha + \pi)\lambda \ ;$$

$$\lambda_2 = (\beta + \pi)\lambda \ .$$

Note that $\lambda_1 + \lambda_2 > \lambda$. The synchronization structure assures that the departure rate of the structure is equal to λ.

This structure contain tokens that are themselves individual Cases and also tokens that are copies of an split Case. When calculating metrics for the structure, this must be taken into consideration. For computing the number of Cases in the structure, we firstly sum the contents of places Z_1 and Z_2. These places contain tokens that have already leaved the Sub-processes. The value of this sum is divided by the half, because there are two tokens for each Case. Then, we sum to this result the number of tokens that are inside each Sub-process. The result will contain tokens that represent the same Case in different Sub-processes. To get the correct number of Cases, we subtract from this result the number of Cases that were replicated (and are still inside the Sub-processes), which is obtained from places H_1 and H_2.

Definition 42 (Expected Number of Cases in Multiple Path with Synchronization)**.** Let $OR(U_1, U_2, \alpha, \pi, \beta)$ be a Sub-process, the expected number of Cases inside this Sub-process in the Workflow execution is computed by:

$$E(n_{OR}) = E(n_1) + E(n_2) + \frac{1}{2}\Big[E(Z_1) + E(Z_2)\Big] - \frac{1}{2}\Big[E(H_1) + E(H_2)\Big] \ ,$$

where $E(n_i)$ is the expected number of Cases in U_i, $E(Z_i)$ is the expected marking of place Z_i and $E(H_i)$ is the expected marking of place H_i.

Definition 43 (Expected Time in Multiple Path with Synchronization). Let $OR(U_1, U_2, \alpha, \pi, \beta)$ be a Sub-process, the mean time spent by a Case in this Sub-process is given by:

$$E(\tau_{OR}) = \frac{E(n_{OR})}{\lambda} \ ,$$

where λ is the Case arrival rate at the Sub-process.

3.11 Interleaving

This composition represents the same situation found in the pattern *Interleaved Routing* (WCP40). A set of Activities must be executed sequentially, but an order in which they must be taken is not defined. Generalizing the pattern, this operator allows the inclusion of Sub-processes in the structure.

The model generates one token for each Sub-process, as like they could be executed in parallel, but use a *mutex* to assure that they will be executed in a mutual exclusive way. An auxiliary synchronization place is added to the final of each Sub-process to guarantee that the Case will depart from the structure only after all Sub-processes have finished processing.

When a token arrives at the starting place P_{INTER}, place P_{MUT}, which corresponds to the *mutex*, and places $H_1, \ldots, H_N$ receive one token each. Place H_i indicates that Sub-process U_i needs to be executed.

When the execution of a Sub-process U_i is finished, a token is put in place Z_i, indicating that the Sub-process has been already executed and a token is returned to place P_{MUT}. When all Sub-processes are concluded, the departing transition qz fires, removing a token from places $Z_1, \ldots, Z_N$ and P_{MUT}.

Notice that each Case adds its own token to the mutex, which allows that independent Cases are executed in parallel. There is no problem for not being able to differentiate Cases in this approach, because we are interested only in the number of executions of each Sub-process and not the data processed by them. Thus, the differentiation is not relevant for our interest.

The model is presented in Fig. 10.

Definition 44 (Interleaving Operator – INTER). Let $U_1, U_2, \ldots, U_N$ be Sub-processes

$$INTER : SProc \times \ldots \times SProc \rightarrow SProc$$

$INTER(U_1, U_2, \ldots, U_N) = U_R$, where:

1. Let G_{INTER} be a GSPN containing places P_{MUT}; P_{INTER}; $H_1, \ldots, H_N$; and $Z_1, \ldots, Z_N$; and immediate transitions qs; qz; e $qw_1, \ldots, qw_N$, such that:

   (a) ${P_{INTER}}^{\bullet} = qs$;

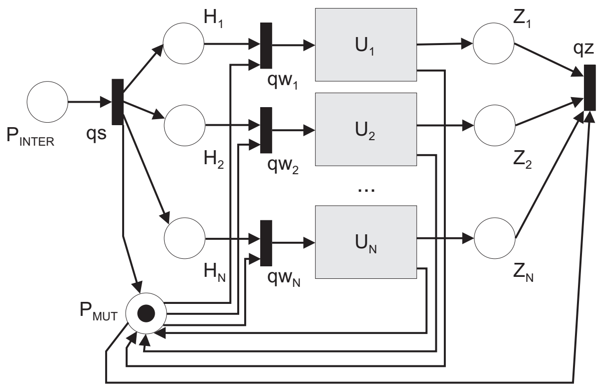

Figure 10: Interleaving composition (INTER)

(b) $qs^\bullet = \{P_{MUT}, H_1, \ldots, H_N\}$;

(c) $H_i{}^\bullet = qw_i, \quad i = 1, \ldots, N$;

(d) $P_{MUT}{}^\bullet = \{qw_1, \ldots, qw_N, qz\}$;

(e) ${}^\bullet qz = \{Z_1, \ldots, Z_N\}$.

2. $U_R = U_1 \cup \ldots \cup U_N \cup G_{INTER}$, with the addition of arcs in such a way that:

   (a) $qw_i{}^\bullet = \{Start(U_i)\}, \quad i = 1, \ldots, N$;

   (b) ${}^\bullet Z_i = End(U_i), \quad i = 1, \ldots, N$;

   (c) ${}^\bullet P_{MUT} = \{qs\} \cup End(U_1) \cup \ldots \cup End(U_N)$;

   (d) $Start(U_R) = P_{INTER}$;

   (e) $End(U_R) = \{qz\}$.

3.12 Metrics for Interleaving

The following metrics can be computed for the Interleaving model.

Definition 45 (Internal Arrival Rate in Interleaving)**.** Let $INTER(U_1, U_2, \ldots, U_N)$ be a Sub-process with Case arrival rate λ, the internal arrival rates $\lambda_1, \ldots, \lambda_N$ in each component $U_1, \ldots, U_N$, respectively, is given by:

$$\lambda_i = \lambda, \quad i = 1, \ldots, N \ .$$

Definition 46 (Expected Number of Cases in Interleaving)**.** Let $INTER(U_1, U_2, \ldots, U_N)$ be a Sub-process, the expected number of Cases inside this Sub-process in the Workflow execution is:

$$E(n_{INTER}) = \sum_{i=1}^{N} E(n_i) \ ,$$

where $E(n_i)$ is the expected number of Cases in U_i.

Definition 47 (Expected Time for Interleaving)**.** Let $INTER(U_1, U_2, \ldots, U_N)$ be a Sub-process, the mean time spent by a Case in this Sub-process is given by:

$$E(\tau_{INTER}) = \frac{E(n_{INTER})}{\lambda} \ ,$$

where λ is the arrival rate at the Sub-process.

It can be noticed from these metrics that the average performance of the Interleaving is the same of a Sequence. This is reasonable since sub-processes are actually executed in sequence, despite the absence of a pre-defined order. This also reveals that the performance of a Sequence composition in the average is the same no matter how the sub-processes inside it are ordered.

3.13 Sub-process Elimination

Sometimes, it is desired that one Sub-process in the composition is eliminated. For example, in an Alternative Path when there is a path that just routes to the departure, without executing nothing. In these situations, the Sub-process can be eliminated by the operation in Def. 48.

Definition 48 (Sub-process Elimination)**.** Let U be a Sub-process with $St = \mathit{Start}(U)$ e $Dt = \mathit{End}(U)$, it is eliminated from the Workflow by the following steps:

1. $\forall p : t \in Dt, O(t,p) > 0 \mid {}^\bullet p = {}^\bullet St$;
2. $\forall t \in {}^\bullet St \mid O(t, St) = 0$.

4 Analyzing Workflow Through the GSPN Models

This section describes techniques to evaluate Workflows using the GSPN model proposed in this work.

The same GSPN model can be employed to evaluate the correctness as well as the performance of the modeled Sub-process. Depending on the objective, the Sub-process must be inserted into the corresponding Workflow structure. Correctness is verified through the *Workflow Soundness Model* and performance is evaluated through the *Workflow System* Model.

4.1 Soundness

The minimum necessary conditions for a Workflow to be correct were defined by van der Aalst with his *Soundness* property [2], as follows:

Soundness. *A process is sound if it contains no unnecessary tasks and every Case submitted to the process is completed in full and with no references to it remaining in the process.*

For qualitative analysis of the Workflow, the Sub-process model must be set up as defined by the *Workflow Soundness Model* in Def. 49. The objective of this analysis is to verify logical properties regarding the Workflow. Some typical errors [2] that can be found through this analysis are:

- Activities without input conditions – conditions required for executing the Activity are undefined;
- Activities without output conditions – the execution of such Activities has no effect on the Case processing;
- Dead Activities – the system never reaches an state in which the Activity can be executed;
- Deadlock – Case processing is blocked and a condition that could unblock the process can never be reached;
- Livelock – the system is locked in a never-ending cycle, where the exit condition will never occur;
- Activities still to be executed after the Case has departed from the process;
- Tokens are retained in the system after the conclusion of the Case processing.

Verifying the Soundness of a GSPN Sub-process model corresponds to verify *liveness* and *boundedness* in our Workflow Soundness Model, according to Def. 50.

Definition 49 (Workflow Soundness Model)**.** The Workflow Soundness Model of a Sub-process U is a GSPN G_U such that:

1. $G_U = U$, with the following modifications:
   (a) Let $Emp : \mathcal{P} \rightarrow \mathbb{N}_+$ be a employing function, which assigns a number of resources to each Role;
   (b) ${}^\bullet Start(U) = End(U)$;
   (c) $\forall r_k \in \mathcal{P}, M_0(R_k) = Emp(r_k)$, where R_k is a place in G_U that represents the Role r_k;

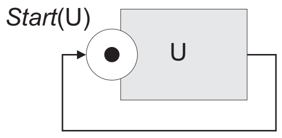

Figure 11: Workflow Soundness Model

(d) $M_0(Start(U)) = 1$.

Definition 50 (Soundness Verification)**.** Let U be a Sub-process and WS_U be the Workflow Soundness Model of U, the Sub-process U is *Sound* if and only if WS_U is *live* and *bounded*.

The composition rules presented in Section 3 ensure that the resulting GSPN model for the Workflow will be sound if the composing Sub-processes are sound.

4.2 Performance Analysis

Once the correctness of the model is verified through the Workflow Soundness Model, it is possible to evaluate the performance of the Sub-process represented through the model. For this purpose, the *Workflow System* Model, described in Def. 11, must be used.

Two approaches can be adopted:

- **Transient analysis** – evaluates the system behavior for a period of time after it starts executing;
- **Stationary analysis** – evaluates the behavior of the system when it reaches the equilibrium state.

The methods for obtaining these solutions for GSPNs are well-known [14][4].

Notice that the Workflow System model is not bound limited, i.e., place markings can theoretically grow to infinity. However, if the minimum number of resources is provided, the system reaches an equilibrium state, which is its stationary solution. We provided formulae for analytically obtaining the stationary solution of a number of metrics. Other metrics must be calculated through simulation.

5 Conclusion

In this technical report we present the results of the master's thesis of the first author. We formally defined an stochastic representation for business workflows based on Generalized Stochastic Petri Nets (GSPN). The GSPN models proposed can be used to represent a rich set of complex business processes and can be employed to evaluate both qualitative and quantitative properties of them. Quantitative results can be obtained by simulation of the models and by analytical formulae presented along this paper. Qualitative results are obtained by static analysis of the GSPN structure.

References

[1] W.M.P. van der Aalst. The Application of Petri Nets to Workflow Management. *The Journal of Circuits, Systems and Computers*, 8(1):21–66, 1998.

[2] W.M.P. van der Aalst and K.M. van Hee. *Workflow Management: Models, Methods, and Systems.* MIT press, Cambridge, MA, 2002.

[3] W.M.P. van der Aalst and A.H.M. ter Hofstede. YAWL: Yet another workflow language (revised version). QUT Technical report, FIT-TR-2003-04, Queensland University of Technology, Brisbane, 2003.

[4] Gunter Bolch, Stefan Greiner, Hermann de Meer, and Kishor S. Trivedi. *Queueing Networks and Markov Chains: Modeling and Performance Evaluation with Computer Science Applications.* John Wiley & Sons, New York, August 1998.

[5] Steven C. Bruell, Pozung Chen, and Gianfranco Balbo. Alternative methods for incorporating non-exponential distributions into stochastic timed petri nets. In *PNPM*, pages 187–197. IEEE Computer Society, 1989.

[6] Workflow Management Coalition. WfMC standards: The workflow reference model, version 1.1., 1995.

[7] Workflow Management Coalition. Workflow management coalition terminology and glossary, version 3.0 (WFMC-TC-1011). Technical report, Workflow Management Coalition, Brussels, 1999.

[8] A. Ferscha. Qualitative and quantitative analysis of business workflows using generalized stochastic petri nets, 1994.

[9] Paul Harmon and Geary A. (pról.) Rummler. *Business Process Change : A Manager's Guide to Improving, Redesigning, and Automating Processes.* Morgan Kaufmann, San Francisco, EUA, 2003.

[10] John Jeston and Johan Nelis. *Business Process Management : Practical Guidelines to Successful Implementations.* Elsevier/Butterworth-Heinemann, Amsterdam, 2006.

[11] JianQiang Li, Yushun Fan, and MengChu Zhou. Performance modeling and analysis of workflow. *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, 34(2):229–242, 2004.

[12] JianQiang Li, Yushun Fan, and MengChu Zhou. Timing contraint workflow nets for workflow analysis. *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, 33(2):179–193, 2004.

[13] Sea Ling and H. Schmidt. Time Petri Nets for workflow modelling and analysis. In *Proceedings of the 2000 IEEE International Conference on Systems, Man and Cybernetics*, volume 4, pages 3039–3044, Nashville, TN, USA, October 2000. IEE Press.

[14] M. Ajmone Marsan, G. Balbo, and G. Conte et al. *Modelling with Generalized Stochastic Petri Nets.* Wiley series in parallel computing. Wiley, New York, 1995.

[15] M. Netjes, W.M.P. van der Aalst, and H. A. Reijers. Analysis of resource-constrained processes with colored petri nets. In K. Jensen, editor, *Proceedings of the Sixth Workshop on the Practical Use of Coloured Petri Nets and CPN Tools (CPN 2005)*, volume 576 of *DAIMI*, pages 251–266, Aarhus, Denmark, October 2005. University of Aarhus.

[16] H. Reijers. *Design and Control of Workflow Processes: Business Process Management for the Service Industry.* PhD thesis, Eindhoven University of Technology, Eindhoven, The Netherlands, 2002.

[17] Stephen A. White. Introduction to bpmn. Technical report, IBM Software Group, 2006.