We are IntechOpen, the world’s leading publisher of Open Access books
Built by scientists, for scientists

3,800
Open access books available

116,000
International authors and editors

120M
Downloads

154
Countries delivered to

TOP 1%
Our authors are among the most cited scientists

12.2%
Contributors from top 500 universities

WEB OF SCIENCE™
Selection of our books indexed in the Book Citation Index in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com
ISPN: Modeling Stochastic with Input Uncertainties Using an Interval-Based Approach

Sérgio Galdino and Paulo Maciel
Polytechnic School of Pernambuco University & Federal University of Pernambuco
Brazil

1. Introduction

Analytic techniques are frequently used for performance analysis of discrete event systems. Conventional models have a set of single value input parameters (such as mean resource demands) and give single value results for each performance index of interest (such as mean system throughput). However, this single point characterization of parameters is insufficient when uncertainties and variabilities are related with system parameters. As an application domain, we may highlight software performance engineering, which accomplishes performance modelling in several phases design-cycle and throughout implementation (Girault & Valk, 2003; Smith, 1990). Even if uncertainties and variabilities may be associated with one or more parameters of the system in early stages of system design, the expert designer might have a suitable guess related to the interval of values associated with these parameters due to previous experience.

The current availability of software tools for performance evaluation allows one to hide the technicalities from the end-user. Users specify their performance model using some high-level modeling language supported by tools such as PEPSY-QNS, TimeNET 4.0, SPNP 6.0, GreatSPN 2.0, or PEPA, in which the underlying mathematical model is automatically generated and analyzed. We propose the adoption of intervals to represent the uncertainties in the parameters of ISPN (Interval Stochastic Petri Net) models (Galdino & Maciel, 2006; Galdino et al., 2007a,b). Therefore, the set of methods considered for Markov chain steady-state analysis have to be adapted for taking into account interval arithmetic. In ISPN the exponential transition rates and immediate transition weights are represented by intervals. This chapter focuses on ISPN using MATLAB with INTLAB toolbox. We briefly introduce the interval arithmetic. Afterwards, we describe the ISPN and outline the approach adopted to the respective interval steady state analysis. We present two ISPN system models and the respective results of analysis. Further possibilities of the method are also suggested.

2. Background

Prior to present the ISPN, this section introduces some basic concepts needed to understand how interval arithmetics may be used for evaluating system’s metrics. Hence, we initially introduce some concepts on interval arithmetics.
2.1 Notation
Throughout this paper, all scalar variables are denoted by ordinary lowercase letters (a). Interval variables are enclosed in square brackets ([a]). Underscores and overbars denote lower and upper bounds, respectively.

A real interval [x] is a nonempty set of real numbers

\[ [x] = [x_-, x_+] = \{ x \in \mathbb{R} : x_- \leq x \leq x_+ \} \]  

(1)

where \( x_- \) and \( x_+ \) are called the infimum (inf) and supremum (sup), respectively, and \( x \) is a point value belonging to an interval variable [x].

The set of all intervals \( \mathbb{R} \) is denoted by \( I(\mathbb{R}) \) where

\[ I(\mathbb{R}) = \{ [x, \overline{x}] : x, \overline{x} \in \mathbb{R} : x \leq \overline{x} \} \]  

(2)

2.2 Interval arithmetic

Let \(+, -, \cdot, \div\) denote the arithmetic operations. If \( \bullet \) denotes any of these arithmetic operation for real numbers \( x \) and \( y \), then the corresponding operation for arithmetic on interval numbers \([x]\) and \([y]\) is

\[ [x] \bullet [y] = \{ x \bullet y | x \in [x], y \in [y] \} \]  

Thus the interval \([x] \bullet [y]\) resulting from the operations contain every possible number that can be found as \( x \bullet y \) for each \( x \in [x] \), and each \( y \in [y] \).

A form of interval arithmetic perhaps first appeared in 1924 and 1931 in (Chen & van Emden, 1995; Young, 1931), then later in (Ullrich, 1990). Modern development of interval arithmetic began with R. E. Moore’s dissertation (Moore, 1962) as a method for determining absolute errors of an algorithm, considering all data errors and rounding, after R.E. Moore introduced interval analysis (Moore, 1966). Interval arithmetic is an arithmetic defined on sets of intervals, rather than sets of real numbers. The power of interval arithmetic lies in its implementation on computers. In particular, outwardly rounded computations allows rigorous enclosures.

An important result is the inclusion property theorem. Rall aptly calls this the fundamental theorem of interval analysis (Hansen & Walster, 2004; Rall, 1969).

**Theorem 2.1 (Fundamental Theorem).** If the function \( f([x]_1, [x]_2, [x]_3, \ldots, [x]_n) \) is an expression with a finite number of intervals \([x]_1, [x]_2, [x]_3, \ldots, [x]_n \in I(\mathbb{R})\) and interval operations \(+, -, \cdot, \div\) and if \( [w]_1 \subseteq [x]_1, [w]_2 \subseteq [x]_2, [w]_3 \subseteq [x]_3, \ldots, [w]_n \subseteq [x]_n \) then \( f([w]_1, [w]_2, [w]_3, \ldots, [w]_n) \subseteq f([x]_1, [x]_2, [x]_3, \ldots, [x]_n) \).

2.3 Enclosures for the range of function

Consider a function \( f \) from \( \mathbb{R}^n \) to \( \mathbb{R} \). The interval function \([f] \) from \( I(\mathbb{R}^n) \) to \( I(\mathbb{R}) \) is an inclusion function \([f]\) if

\[ \forall [x] \in I(\mathbb{R}^n), \quad f([x]) \subseteq [f([x])] \]  

Interval analyses provides, for a large class of functions \( f \), inclusion functions such that \([f]([x])\) is not too large.

An inclusion function for \( f : \mathbb{R}^n \rightarrow \mathbb{R} \) is obtained with two optimizations to compute the infimum and supremum of \( f \). However, these optimization problems are far from trivial in general. An more tractable approach uses the following theorem (Jaulin et al., 2001), which is a direct consequence of Theorem 2.1.

---

www.intechopen.com
Theorem 2.2 (Natural Inclusion Function Theorem). Consider a function
\[ f : \mathbb{R}^n \rightarrow \mathbb{R}, (x_1, x_2, x_3, \ldots x_n) \mapsto f(x_1, x_2, x_3, \ldots x_n), \]
expressed as a finite composition of the operators \(+, -, \times, \div\) and elementary functions (sin, cos, exp, sqr, ...). An inclusion monotonic and thin (degenerated) inclusion function \([f] : I(\mathbb{R})^n \rightarrow I(\mathbb{R})\) for \(f\) is obtained by replacing each real variable \(x_i\) by an interval variable \([x_i]\) and each operator or function by its interval counterpart. This function is called the natural inclusion function of \(f\). If \(f\) involves only continuous operators and continuous elementary functions, the \([f]\) is convergent. If, moreover, each of the variables \((x_1, x_2, x_3, \ldots x_n)\) occurs at most once in the formal expression of \(f\) then \([f]\) is minimal.

Natural inclusion functions are not minimal in general, because of the resulting problem. The accuracy of the resulting interval strongly depends on the expression of \(f\). The use of natural inclusion functions is not always the best choice. An important field of interval analysis is the use other types of inclusion functions that would provide best enclosures.

3. ISPN formalism

Petri Nets (PNs) represent a family of forms of graphical representation for description of systems whose dynamics are characterized by concurrency, synchronization, mutual exclusion, and conflict, which are typical features of distributed environments (Murata, 1989). PNs incorporate a notion of local state and a rule for state change (transition firing) that allow them to capture both the static and the dynamic characteristics of a real system being explicit about time considerations. The introduction of timing concepts into PN models were proposed later by Ramchandani (Ramchandani, 1974), Merlin (Merlin & Farber, 1976), and Sifakis (Sifakis, 1978) from distinct viewpoints. Molloy (Molloy, 1981) as well as Florin (Florin & Natkin, 1989) proposed PN models in which stochastic timing was considered. These works opened the possibility of connecting PN theory and stochastic modeling. Nowadays, these models as well as their extensions are generically named Stochastic Petri nets (SPN).

ISPN is an extension of GSPN (Generalized Stochastic Petri Nets) model in order to introduce the interval analysis (Galdino & Maciel, 2006) and has a lot of potential to practical applications. As part of the contribution in the development of this work, the authors also brought out its related practical applications in performance and dependability evaluation (Galdino & Maciel, 2006; Galdino et al., 2007a; b). GSPN is a particular timed PN (Petri Net) that incorporates both stochastic timed transitions (represented as white boxes) and immediate transitions (represented as thin black bars). Timed transitions have an exponentially distributed firing time and immediate transitions fire in zero time. GSPNs were originally defined (Marsan, Balbo & Conte, 1984) and later modified as described in (Brinksma, 2001).

A formal definition of ISPN is provided below. This definition keeps to the SPN definition presented in (German, 2000), but considers real intervals assigned to transition delays and weights instead of adopting real single values.

Let \(ISPN = (P, T, I, O, I_i, G, M_0, Atts)\) be an interval stochastic Petri net, where

- \(P = \{p_1, p_2, \ldots p_n\}\) is the set of places,
- \(T = \{t_1, t_2, \ldots t_m\}\) is the set of transitions,
- \(I \in (\mathbb{N}^n \rightarrow \mathbb{N})^{n \times m}\) is a matrix of marking-dependent multiplicities of input arcs, where the \(i_k\) entry of \(I\) gives the possibly marking-dependent arc multiplicity of input arcs from place \(p_j\) to transition \(t_k\) \((A \subset (P \times T) \cup (T \times P) - set of arcs)\),
\( O \in (\mathbb{N}^n \rightarrow \mathbb{N})^{n \times m} \) is a matrix of marking-dependent multiplicities of output arcs, where \( o_{jk} \) entry of \( O \) gives the possibly marking-dependent arc multiplicity of output arcs from transition \( t_j \) to place \( p_k \),

\( H \in (\mathbb{N}^m \rightarrow \mathbb{N})^{n \times m} \) is a matrix of marking-dependent multiplicities of inhibitor arcs, where \( h_{jk} \) entry of \( H \) gives the possibly marking-dependent arc multiplicity of inhibitor arcs from place \( p_j \) to transition \( t_k \),

\( G \in (\mathbb{N}^m \rightarrow \{\text{true}, \text{false}\})^m \) is a vector that assigns a guard condition related to place markings to each transition,

\( M_0 \in \mathbb{N}^n \) is a vector that assigns the initial marking of each place (initial state),

\( \text{Atts} = (\text{Dist}, W, \text{Markdep}, \text{Police}, \text{Concurrency})^m \) comprises the set of attributes for transitions, where

- \([\text{Dist}] \in \mathbb{N}^m \rightarrow [\mathcal{F}] \) is a possible marking-dependent firing interval distribution function (the domain of \([\mathcal{F}] \) is \([0, \infty)\)),
- \([W] \in \mathbb{N}^m \rightarrow I(\mathbb{R}^+) \) is a possible marking-dependent interval weight,
- Markdep \( \in \{\text{constant}, \text{enabdep}\} \) where the transition interval firing timing distribution could be marking independent (constant) or enabling dependent (enabdep - the distribution depends on the actual enabling condition),
- Police \( \in \{\text{prd}, \text{prs}\} \) is the preemption policy (prd - preemptive repeat different means that when a preempted transition becomes enabled again the previous elapsed firing time is lost; prs- preemptive resume, in which the firing time related to a preempted transition is resumed when the transition becomes enabled again),
- Concurrency \( \in \{\text{ss}, \text{is}\} \) is the degree of concurrency of transitions, where ss represents single server semantics and is depicts infinite-server semantics.

It is worth noting that if only point intervals are assigned to delays and immediate transition weights the model is reduced to a GSPN. This case the ISPN analysis provides GSPN verified results.

### 3.1 ISPN steady state analysis

ISPN is considered to be a high-level formalism for ICTMC (Interval Continuous Time Markov Chain) generation Galdino et al. (2007b). The classical algorithms found in literature (Bolch et al., 2006) are adapted to take into account the interval coefficients of the ISPN model. This is a contribution to analysis of models ISPN that will be presented in sequel. The analysis of models using ISPN is accomplished in four subtasks:

- generation of the IERG (Interval Extended Reachability Graph),
- elimination of vanishing markings and the corresponding state transitions,
- interval steady-state analysis,
- computation of measures. Standard measures such as the average number of tokens in places and the throughput of timed transitions are computed using interval arithmetic.
3.1.1 Generation of the IERG

The first of four steps of ISPN analysis is the IERG generation (interval extended reachability graph). From the IERG the set of markings $\mathcal{M} = \mathcal{T} \cup \mathcal{V}$ is divided into set of tangible markings $\mathcal{T}$ and vanishing $\mathcal{V}$. Through the elimination of vanishing markings discussed below, using methods of interval analysis, we obtain the infinitesimal generator matrix $[\mathcal{Q}]$ of ICTMC underlying an ISPN model.

From a given ISPN, an interval extended reachability graph (IERG) is generated containing markings as nodes and interval stochastic information attached to arcs so as to relate markings to each other. The ISPN reachability graph is a directed graph $\text{RG}(\text{ISPN}) = (V, E)$, where $V = \text{RS}(\text{ISPN})$ and $E = \{ (m, t, m') \mid m, m' \in \text{RS}(\text{ISPN}) \text{ and } m \xrightarrow{t} m' \}$ are the set of nodes and edges, respectively. If an ISPN model is bounded, the $\text{RG}(\text{ISPN})$ is finite and it can be constructed, for example, based on Algorithm 5.1: Computation of the Reachability Graph p. 61 from (Girault & Valk, 2003).

The $\text{RG}(\text{ISPN})$ is constructed, in this work, using the Algorithm 1 below. The activity defined in Step 2.1 ensures that no marking is visited more than once. Each visited marking is labeled (Step 2.1), and Step 2.2.3 ensures that only unique added markings defined in Step 2.1 ensures that no marking is visited more than once. Each visited marking is labeled (Step 2.1), and Step 2.2.3 ensures that only unique added markings to $\mathcal{V}$ are those that were not previously added. When the marking is visited, only those edges that represents the firing of an enabled transition are added to the set $E$ (Step 2.2.4).

Algorithm 1

(∗∗ IERG generation ∗∗)

Input - A ISPN model.
Output - A directed graph $\text{RG}(\text{ISPN}) = (V, E)$ of a limited network system.

1. Initialize $\text{RG}(\text{ISPN}) = \{ \{m_0\}, \varnothing \}; m_0$ is unlabelled.
2. while there are an unlabeled node $m$ in $V$ do
   2.1 Select an unlabeled node $m \in V$ label it
   2.2 for each enabled transition $t$ in $m$ do
      2.2.1 Calculate $m'$ such that $m \xrightarrow{t} m'$;
      2.2.2 if there are $m'' \in V$ such that $m'' \xrightarrow{t} m'$ and $m'' \leq m'$
         then the algorithm fails and ends;
         (no limitation condition was detected).
      2.2.3 if there is no $m'' \in V$ such that $m'' = m'$
         then $V := V \cup \{m''\}$; ($m''$ is a node not labeled).
      2.2.4 $E := E \cup \{ (m, t, m') \}$
   3. The algorithm is successful and $\text{RG}(\text{ISPN})$ is the interval extended reachability graph.

3.1.2 Elimination of vanishing markings

The second of four steps of ISPN analysis is the elimination of vanishing markings, which is the step for generating the ICTMC from a given ISPN. Once the IERG has been generated, it is transformed into an ICTMC by the use of matrix algorithms Bolch et al. (2006).

The markings set $\mathcal{M} = \mathcal{V} \cup \mathcal{T}$ in the reachability set of an ISPN is partitioned into two sets, the vanishing markings $\mathcal{V}$ and the tangible markings $\mathcal{T}$. Let:

$$[\mathcal{P}]^V = [\mathcal{P}]^{\mathcal{V}\mathcal{V}} \mid [\mathcal{P}]^{\mathcal{V}\mathcal{T}}$$

(3)
denote an interval matrix, where

- \([P]^{VV}\) denotes the interval transition probabilities between vanishing markings,
- \([P]^{VT}\) denotes the interval transition probabilities from vanishing markings to the tangible markings.

Furthermore, let
\[
[U]^T = [U]^{TV} | [U]^{TT}
\]
denote an interval matrix, where

- \([U]^{TV}\) represents interval transition rates from tangible to vanishing markings;
- \([U]^{TT}\) represents interval transition rates between tangible markings.

Now, we obtain the interval rate matrix \([U]\). This matrix has dimensions \(|T| \times |T|\), where \(T\) denotes the set of tangible markings.

\[
[U] = [U]^{TT} + [U]^{TV}(1 - [P]^{VV})^{-1}[P]^{VT}
\]

The interval matrix of the infinitesimal generator is \([Q] = [q]_{ij}\), where its entries are given by:
\[
[q]_{ij} = \begin{cases} 
[u]_{ij} & \text{if } i \neq j \\
- \sum_{k \in T} [u]_{ik} & \text{if } i = j 
\end{cases}
\]

where \(T\) denotes the set of tangible markings.

### 3.1.3 Steady-state probability vector evaluation

Now we describe the third of four steps of ISPN analysis. The steady-state solution of the ICTMC model underlying the ISPN is obtained by solving the interval linear system of equations with as many equations as the number of tangible markings.

\[
\{ \pi \cdot [Q] = 0 \} \quad \sum_{M \in T} \pi(M) = 1
\]

\([\pi]\) is the interval vector for the equilibrium pmf (probability mass function) over the reachable tangible markings, and we write \([\pi](M)\) for the interval steady-state probability of a given tangible marking \(M\).

Once the interval generator matrix \([Q]\) of the ICTMC associated with a ISPN model has been derived, the steady state probability is calculated so that other respective metrics might be subsequently computed.

ISPN models deal with system uncertainties by considering intervals for representing time as well as weights assigned to transition models. The proposed model and the respective methods, adapted to take interval arithmetic into account, allow the influence of simultaneous parameters and variabilities on the computation of metrics to be considered, thereby providing rigorously bounded metric ranges. It is also important to stress that even when only taking into account thin intervals, one may make use of the proposed model, since rounding and truncation errors are naturally dealt with in interval arithmetic, so that the metrics results obtained are certain to belong to the intervals computed.
3.1.4 Interval performance indices
The computation of performance indices (metrics) of interest is the fourth and final step in the analysis ISPN. In the case of ISPN steady state analysis, where interval p.m.f. has already been obtained, indices are calculated by interval function evaluation. Interval performance indices are interval functions extended on classical indices (Marsan, Bobbio, Conte & Cumani, 1984).

4. Examples of ISPN models
The purpose of this section is to present clearly all steps of ISPN analysis. Two examples are used. One is very simple and can be followed up and have calculations performed without using a computer. The second case, however, you must use a software with an interval arithmetic library as a tool to carry out all his calculations. Example 1 has only two tangible markings and two vanishing markings. Example 2 has sixteen tangible markings and twelve vanishing markings. The performance evaluations are carried out in MATLAB with the INTLAB toolbox (MATLAB toolbox INTLAB framework). The ISPN model analysis considering only degenerated intervals (points) leads to the classic model GSPN, with verified computations (self-validating).

4.1 Example 1: ISPN analysis of a single machine
The model depicted in Figure 1 represents a failure prone machine and finite capacity buffer (Desrochers & Al-Jaar, 1994). Table 1 presents (degenerated) interval rates of timed transition firing per unit time, where $[\nu]$ represents the production rate interval, $[\lambda]$ represents the failure rate interval, and $[\mu]$ represents the repair rate interval. Here we have a model equivalent to the GSPN model, because there are only degenerate interval parameters.

![Fig. 1. The Single Machine module.](image)

<table>
<thead>
<tr>
<th>Transition</th>
<th>Value</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>$[t_2]$</td>
<td>$[10, 10]$</td>
<td>$\nu$</td>
</tr>
<tr>
<td>$[t_4]$</td>
<td>$[3, 3]$</td>
<td>$\mu$</td>
</tr>
<tr>
<td>$[t_5]$</td>
<td>$[5, 5]$</td>
<td>$\lambda$</td>
</tr>
</tbody>
</table>

Table 1. Transition Firing Rates (degenerated intervals) for the Single Machine One-Buffer Transfer Line.

As a result of the first step of ISPN analysis we obtain the reachability set (Table 2), and the reachability graph (Figure 2).
Table 2. Reachability set and distribution markings from ISPN of Figure 1.

<table>
<thead>
<tr>
<th>State</th>
<th>Marking ((m_1, m_2, m_3, m_4))</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(M_0 = (1, 0, 0, 0))</td>
</tr>
<tr>
<td>2</td>
<td>(M_1 = (0, 1, 0, 0))</td>
</tr>
<tr>
<td>3</td>
<td>(M_2 = (0, 0, 1, 0))</td>
</tr>
<tr>
<td>4</td>
<td>(M_3 = (0, 0, 0, 1))</td>
</tr>
</tbody>
</table>

Finally, we obtain the matrices \([P]^{VV}, [P]^{VT}, [U]^{TV}\) and \([U]^{TT}\):

\[
[P]^{VV} = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix}, \quad [P]^{VT} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \\
[U]^{TV} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 10 & 10 \end{pmatrix}, \quad [U]^{TT} = \begin{pmatrix} 0 & 0 & 10 & 10 \\ 0 & 0 & 0 & 0 \end{pmatrix}.
\]

Afterwards, carry out the elimination of vanishing markings (Equation 5) to obtain the matrix of rate intervals \([U]\). The matrix of rate intervals represents an IREMC (Interval Reduced Embedded Markov Chain on Figure 3):

\[
[U] = \begin{pmatrix} 10 & 10 & 5 & 5 \\ 3 & 3 & 0 & 0 \end{pmatrix}.
\]

Finally, using Equation 6, we find the infinitesimal generator interval matrix:

\[
[Q] = \begin{pmatrix} -5 & -5 & 5 & 5 \\ 3 & 3 & -3 & -3 \end{pmatrix}.
\]

The third step of ISPN analysis solves the system of interval linear equations described by Equation (7). The interval linear equations solution is carried out by the verifylss function of the MATLAB toolbox INTLAB. Substituting the last equation of system \((\pi_1, \pi_2) \cdot [Q] = 0\) by the normalization condition \(\pi_1 + \pi_2 = 1\), the linear system \((\pi_1, \pi_2) \cdot [A] = [b]\) is obtained. The solution of this system directly provides the steady state probabilities of tangible states. Considering

\[
[A] = \begin{pmatrix} -3 & 5 \\ 1 & 1 \end{pmatrix} \quad \text{and} \quad [b] = \begin{pmatrix} 0 \\ 1 \end{pmatrix}
\]
the M-file MATLAB toolbox INTLAB case1v.m, used for calculating verified probabilities and machine production rate, is given bellow:

1. % INPUT: A coeffifiente matrix  
2. % b right hand side vector  
3. % OUTPUT: x interval probabilities vector solution  
4. % P machine production rate  
5. format long  
6. intvalinit(‘displayinfsup’)  
7. A=intval([-3,5;1,1])  
8. b=[0;1]  
9. x=verifylss(A,b)  
10. P=10*x(1)

Executing case1v.m yields:

```
>> case1v
===> Default display of intervals by infimum/supremum (e.g. [ 3.14 , 3.15 ])
intval A = [ -3.00000000000000, -3.00000000000000] [ 5.00000000000000, 5.00000000000000]
[ 1.00000000000000, 1.00000000000000] [ 1.00000000000000, 1.00000000000000]
intval b = 0
intval x = 0 0.62499999999998, 0.62500000000001
[ 0.37499999999999, 0.37500000000001]
intval P = 6.24999999999998, 6.25000000000001
>>
```

The verified interval bounds of each state probabilities on tangible states are:

\[ [\pi_1]_1 = [0.62499999999998, 6.25000000000001] \]

and

\[ [\pi_2]_2 = [0.37499999999999, 0.37500000000001] \]

Finally we can make the fourth and final step of analysis ISPN, computation of metrics. The machine production rate is

\[ [P] = [6.249999999999998, 6.250000000000001] \]

(calculated with the formula \( [P] = [\pi_1]_1 \cdot [t_2] \)). This results exhibit the enclosure of exact value obtained by GSPN analysis. The ISPN analysis results give us verified results, ensuring that the exact value is certain to belong to the intervals computed. One can, for example, to compare this result with interval \( P = 6.25 \) exact value in this simple case.

### Introducing parameters with input uncertainties

Now we calculate a solution in which the parameters are not known exactly, but it is known that they are within certain intervals. Lets consider that rates are \([\mu] = 3 \pm 0.01 = [2.99, 3.01] \) and \([\lambda] = 5 \pm 0.01 = [4.99, 5.01] \) intervals.
As a result from the first step of analysis (by-product of the reachability set), we obtain the matrices \([P]^{VV}, [P]^{VT}, [U]^{TV}\) and \([U]^{TT}\):

\[
[P]^{VV} = \begin{pmatrix}
0 & 0 \\
1 & 1
\end{pmatrix},
\quad
[P]^{VT} = \begin{pmatrix}
1 & 1 \\
0 & 0
\end{pmatrix},
\quad
[U]^{TV} = \begin{pmatrix}
0 & 0 \\
10 & 10
\end{pmatrix},
\quad
[U]^{TT} = \begin{pmatrix}
0 & 0 \\
0 & 0
\end{pmatrix}.
\]

Afterwards, carry out the elimination of vanishing markings (Equation 5), to obtain the matrix of rate intervals \([U]\):

\[
[U] = \begin{pmatrix}
10 & 10 \\
2.99 & 3.01
\end{pmatrix}.
\]

Finally, using Equation 6, we find the infinitesimal generator interval matrix:

\[
[Q] = \begin{pmatrix}
-5.01 & -4.99 \\
2.99 & 3.01
\end{pmatrix}.
\]

Considering

\[
[A] = \begin{pmatrix}
-3 & 5 \\
1 & 1
\end{pmatrix}
\quad
[b] = \begin{pmatrix}
0 \\
1
\end{pmatrix}
\]

the M-file MATLAB toolbox INTLAB case1i.m, used for calculating verified probabilities and machine production rate, is given below:

```matlab
1. % INPUT: A coefficiente matrix
2. % b right hand side vector
3. % OUTPUT: x interval probabilities vector solution
4. % P machine production rate
5. format long
6. intvalinit('displayinfsup')
7. A=infput([-3.01,4.99;1,1],[-2.99,5.01;1,1])
8. b=[0;1]
9. x=verifylss(A,b)
10. P=10*x(1)

Executing case1i.m yields:

```
```

The interval bounds of each state probabilities on tangible states are:

\[
[\pi](1) = [0.62374656249999, 0.62625343750001]
\]

www.intechopen.com
and

\[ [\pi]_{\{2\}} = [0.37374656249998, 0.37625343750001]. \]

Finally we can make the computation of machine production rate:

\[ [P] = [6.23746562499999, 6.26253437500001] \]

(calculated with the formula \[P = [\pi]_{\{1\}} \cdot [t_2]\]). This result represents the variabilities when the rates are in \([\mu] = [2.99, 3.01]\) and \([\lambda] = [4.99, 5.01]\) intervals.

### 4.2 Example 2: ISPN analysis of Two-Machine One-Buffer Transfer Line Model

Consider the Two-Machine One-Buffer Transfer Line Model in Figure 4 (Desrochers & Al-Jaar, 1994). Table 3 presents (degenerated) interval rates of timed transition firing per unit time, where \([\nu_i]\) represents the production rate intervals, \([\lambda_i]\) represents the failure rate intervals, and \([\mu_i]\) represents the repair rate intervals. Here we have a model equivalent to the GSPN model, because there are only degenerate interval parameters.

![Fig. 4. Two-Machine One-Buffer Transfer Line Model (k = 3)](image)

<table>
<thead>
<tr>
<th>Transition</th>
<th>Value ([t_i^{-1}])</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>(t_2)</td>
<td>([1, 1])</td>
<td>(\nu_1)</td>
</tr>
<tr>
<td>(t_3)</td>
<td>([3, 3])</td>
<td>(\lambda_1)</td>
</tr>
<tr>
<td>(t_4)</td>
<td>([5, 5])</td>
<td>(\mu_1)</td>
</tr>
<tr>
<td>(t_6)</td>
<td>([2, 2])</td>
<td>(\nu_2)</td>
</tr>
<tr>
<td>(t_7)</td>
<td>([4, 4])</td>
<td>(\lambda_2)</td>
</tr>
<tr>
<td>(t_8)</td>
<td>([6, 6])</td>
<td>(\mu_2)</td>
</tr>
</tbody>
</table>

Table 3. Interval transition firing rates for the Two-Machine One-Buffer Transfer Line model.

As a result of the first step of ISPN analysis we obtain the reachability set (Table 4) and the reachability graph (Table 5).

Markings enabling the transitions \(t_1\) and \(t_5\) are vanishing, because enabled transitions are immediate (state changes that take negligible amounts of time to occur). Can be identified twelve vanishing markings \(M_0, M_2, M_4, M_5, M_7, M_{12}, M_{13}, M_{17}, M_{19}, M_{22}, M_{24}, M_{26}\) (firing of immediate transitions \(t_1\) and \(t_5\)) and other markings are tangibles.
Table 4. Reachability set and distribution markings from ISPN of Figure 4.

<table>
<thead>
<tr>
<th>State</th>
<th>Marking$^1$</th>
<th>State</th>
<th>Marking$^1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$M_0 = [1, 0, 0, 0, 0, 0, 0, 0, 0, 0]$</td>
<td>15</td>
<td>$M_{14} = [0, 0, 1, 0, 1, 0, 1, 0, 1, 0]$</td>
</tr>
<tr>
<td>2</td>
<td>$M_1 = [0, 1, 0, 0, 0, 1, 0, 0, 0, 0]$</td>
<td>16</td>
<td>$M_{15} = [0, 1, 0, 0, 1, 0, 1, 1, 0, 0]$</td>
</tr>
<tr>
<td>3</td>
<td>$M_2 = [1, 0, 0, 0, 1, 0, 0, 0, 0, 0]$</td>
<td>17</td>
<td>$M_{16} = [0, 1, 0, 2, 0, 1, 0, 0, 0, 0]$</td>
</tr>
<tr>
<td>4</td>
<td>$M_3 = [0, 0, 1, 0, 0, 1, 0, 0, 0, 0]$</td>
<td>18</td>
<td>$M_{17} = [0, 0, 1, 1, 0, 1, 0, 0, 0, 0]$</td>
</tr>
<tr>
<td>5</td>
<td>$M_4 = [0, 1, 0, 0, 1, 0, 0, 0, 0, 0]$</td>
<td>19</td>
<td>$M_{18} = [0, 0, 1, 1, 0, 0, 1, 1, 0, 0]$</td>
</tr>
<tr>
<td>6</td>
<td>$M_5 = [1, 0, 0, 0, 0, 0, 1, 0, 0, 0]$</td>
<td>20</td>
<td>$M_{19} = [1, 0, 0, 0, 0, 1, 0, 0, 0, 0]$</td>
</tr>
<tr>
<td>7</td>
<td>$M_6 = [0, 1, 0, 0, 1, 0, 0, 0, 0, 0]$</td>
<td>21</td>
<td>$M_{20} = [1, 0, 0, 0, 0, 1, 0, 0, 0, 0]$</td>
</tr>
<tr>
<td>8</td>
<td>$M_7 = [1, 0, 0, 0, 1, 0, 0, 0, 0, 0]$</td>
<td>22</td>
<td>$M_{21} = [0, 1, 0, 2, 0, 1, 0, 0, 0, 0]$</td>
</tr>
<tr>
<td>9</td>
<td>$M_8 = [0, 0, 1, 0, 0, 0, 0, 0, 0, 0]$</td>
<td>23</td>
<td>$M_{22} = [0, 1, 0, 2, 0, 1, 0, 0, 0, 0]$</td>
</tr>
<tr>
<td>10</td>
<td>$M_9 = [0, 1, 0, 0, 0, 1, 0, 0, 0, 0]$</td>
<td>24</td>
<td>$M_{23} = [0, 1, 0, 2, 0, 0, 0, 0, 0, 0]$</td>
</tr>
<tr>
<td>11</td>
<td>$M_{10} = [0, 1, 0, 0, 1, 0, 0, 0, 0, 0]$</td>
<td>25</td>
<td>$M_{24} = [1, 0, 0, 3, 0, 1, 0, 0, 0, 0]$</td>
</tr>
<tr>
<td>12</td>
<td>$M_{11} = [0, 0, 1, 0, 0, 0, 0, 0, 0, 0]$</td>
<td>26</td>
<td>$M_{25} = [1, 0, 0, 3, 1, 0, 0, 0, 0, 0]$</td>
</tr>
<tr>
<td>13</td>
<td>$M_{12} = [1, 0, 0, 1, 0, 0, 1, 0, 0, 0]$</td>
<td>27</td>
<td>$M_{26} = [0, 0, 1, 2, 0, 0, 0, 0, 0, 0]$</td>
</tr>
<tr>
<td>14</td>
<td>$M_{13} = [1, 0, 0, 2, 0, 1, 0, 0, 1, 0]$</td>
<td>28</td>
<td>$M_{27} = [0, 0, 1, 2, 0, 0, 1, 0, 0, 0]$</td>
</tr>
</tbody>
</table>

$^1$ Marking $= [m_1, m_2, m_3, m_4, m_5, m_6, m_7, m_8]$

Table 5. Literal description of reachability graph from ISPN of Figure 4.

Finally, we obtain the matrices $[P]^{VV}$, $[P]^{VT}$, $[U]^{TT}$ and $[U]^{TT}$:

$$[P]^{VV} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

www.intechopen.com
\[
[p]^{TT} = \begin{bmatrix}
[1, 1] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] \\
[0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] \\
[0, 0] & [0, 0] & [1, 1] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] \\
[0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [1, 1] & [0, 0] \\
[0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] \\
[0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] \\
[0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] \\
[0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] \\
[0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] \\
[0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] \\
\end{bmatrix}
\]

\[
[U]^{TT} = \begin{bmatrix}
[0, 0] & [3, 3] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] \\
[5, 5] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] \\
[2, 2] & [0, 0] & [0, 0] & [3, 3] & [4, 4] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] \\
[0, 0] & [2, 2] & [5, 5] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] \\
[0, 0] & [0, 0] & [6, 6] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [3, 3] & [0, 0] \\
[0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] \\
[0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [4, 4] \\
[0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] \\
[0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] \\
[0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] \\
[0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] \\
\end{bmatrix}
\]
Afterwards, carry out the elimination of vanishing markings (Equation 5), to obtain the matrix of rate intervals \([U]\) representing the IREMC:
Finally, using Equation 6, we find the infinitesimal generator interval matrix: 

\[
\mathbf{Q} = 
\begin{pmatrix}
[-4, -4] & [3, 3] & [1, 1] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] \\
[5, 5] & [-5, -5] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] \\
[2, 2] & [0, 0] & [-10, -10] & [3, 3] & [4, 4] & [1, 1] & [0, 0] & [0, 0] & [0, 0] \\
[0, 0] & [2, 2] & [5, 5] & [-11, -11] & [0, 0] & [0, 0] & [4, 4] & [0, 0] & [0, 0] \\
[0, 0] & [0, 0] & [6, 6] & [-10, -10] & [0, 0] & [5, 5] & [0, 0] & [0, 0] & [1, 1] \\
[0, 0] & [0, 0] & [0, 0] & [2, 2] & [0, 0] & [-10, -10] & [0, 0] & [3, 3] & [4, 4] \\
[0, 0] & [0, 0] & [0, 0] & [6, 6] & [5, 5] & [0, 0] & [-11, -11] & [0, 0] & [0, 0] \\
[0, 0] & [0, 0] & [0, 0] & [6, 6] & [0, 0] & [6, 6] & [5, 5] & [0, 0] & [0, 0] \\
[0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] \\
[0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] \\
[0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] \\
[0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] \\
[0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] & [0, 0] \\
\end{pmatrix}
\]

The third step of ISPN analysis solves the system of interval linear equations described by Equation (7). The interval linear equations solution is carried out by the verifylss function of the MATLAB toolbox INTLAB. Substituting the last equation of system \([\mathbf{\pi}] \cdot [\mathbf{Q}] = 0\) by the normalization condition \(\sum_{i=1}^{16} |\pi_i| = 1\), the linear system \(([\pi_1], [\pi_2]) \cdot [A] = [b]\) is obtained. The solution of this system directly provides the steady state probabilities of tangible states:
Finally we can make the fourth (final) step of analysis ISP, i.e. computation of metrics. The average utilization of machines, i.e., the probability that a machine is processing a part are: 

\[ \text{UM}_1 = \text{prob}(m(p_2) = 1) \quad \text{and} \quad \text{UM}_2 = \text{prob}(m(p_6) = 1). \]

The evaluation result provides the following values:

\[ \text{UM}_1 = [0.59650101272372, 0.59650101272374] \quad \text{and} \quad \text{UM}_2 = [0.29825050636186, 0.29825050636187]. \]

These results gives interval bounds to exact value and can be used to verify conventional analysis of GSPN results.

Experiment for Two-Machine One-Buffer Transfer Line Model

Table 6 shows the average machine utilization, UM_1 and UM_2, for three \( \mu_1 \) rate intervals (degenerated intervals). ISP analysis results, provided by ISP MATLAB toolbox INTLAB, are GSPN ordinary results with verified interval bounds.

<table>
<thead>
<tr>
<th>Interval rate [( \mu_1 )]</th>
<th>Machine utilization</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>UM_1</td>
</tr>
<tr>
<td>0.1E2, 0.1E2</td>
<td>0.11946700722573, 0.11946700722574</td>
</tr>
<tr>
<td>0.1E1, 0.1E1</td>
<td>0.59650101272374, 0.59650101272374</td>
</tr>
<tr>
<td>0.2E0, 0.2E0</td>
<td>0.6249010470775, 0.6249010470775,</td>
</tr>
</tbody>
</table>

Table 6. Experiment for Two-Machine One-Buffer Transfer Line Model for three MR (Machining Rate) = \( \mu_1 \) (degenerated interval). Results obtained with ISP MATLAB toolbox INTLAB Prototype Tool.
Table 7. The average machine utilization results obtained with ISPN MATLAB toolbox.

<table>
<thead>
<tr>
<th>Interval rate $[\mu_1]$</th>
<th>Machine utilization</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$UM_1$</td>
</tr>
<tr>
<td>$[0.099E2, 0.101E2]$</td>
<td>$0.1139969921745, 0.12493721523401$</td>
</tr>
<tr>
<td>$= 0.100E2 \pm 0.001E2$</td>
<td>$0.11946700722573 \pm 0.00547020800828$</td>
</tr>
<tr>
<td>$[0.099E1, 0.101E1]$</td>
<td>$0.49611631459760, 0.69888751084986$</td>
</tr>
<tr>
<td>$= 0.100E1 \pm 0.001E1$</td>
<td>$0.59650101272237 \pm 0.10038469812613$</td>
</tr>
<tr>
<td>$[0, 199E0, 0.201E0]$</td>
<td>$0.5444910375658809, 0.70531171756699$</td>
</tr>
<tr>
<td>$= 0.200E0 \pm 0.001E0$</td>
<td>$0.62490104707754 \pm 0.08041067048945$</td>
</tr>
</tbody>
</table>

Introducing parameters with input uncertainties:

In sequel, the variations in the rates of exponential transitions are considered. To avoid redundancy, will not be displayed detailing of ISPN analysis as in previous examples. Table 7 shows the average machine utilization, $UM_1$ and $UM_2$ for three $[\mu_1]$ rate intervals. All exponential rate variabilities have $\pm 1$ as errors in the 3rd significant digits:

**ISPN.m Line 59** modification for each experiment:

- AT(3,1)= [infsup(0.099E2,0.101E2),infsup(2.99,3.01),infsup(4.99,5.01),infsup(1.99,2.01), infsup(3.99,4.01),infsup(5.99,6.01)];
- AT(3,1)= [infsup(0.099E1,0.101E1),infsup(2.99,3.01),infsup(4.99,5.01),infsup(1.99,2.01), infsup(3.99,4.01),infsup(5.99,6.01)];
- AT(3,1)= [infsup(0.199E0,0.201E0),infsup(2.99,3.01),infsup(4.99,5.01),infsup(1.99,2.01), infsup(3.99,4.01),infsup(5.99,6.01)];

5. ISPN MATLAB toolbox INTLAB prototype tool

ISPN M-file MATLAB toolbox INTLAB is a prototype for the modeling and evaluation of ISPNs in which exponential transition rates and immediate transition weights may be represented by intervals. Models are specified by matrix input/output arc multiplicity of transitions as a direct mapping of usual graphical Petri Nets representation description of systems. The stationary analysis is based on Markov theory. An interval embedded Markov chain (IEMC), constructed and solved by interval methods, allow us computation metrics. The current prototype is still being used but **ISPN.n** will allow you to write your own features and to tailor ISPNs to your own needs.
The ISPN.m used for calculating verified probabilities and the machine utilization rate from ISPN model of Figure 4, is given below:

Uncomment specified lines to display:

- Line 191: Reachability set and distribution markings from ISPN model (Table 4)
- Line 192: Literal description of reachability graph from ISPN model (Table 4)
- Line 213: \([P]^{VV}\)
- Line 218: \([P]^{VT}\)
- Line 223: \([U]^{TV}\)
- Line 229: \([U]^{TT}\)
- Line 237: \([Q]\)

1. %
2. % ISPN prototype tool
3. %
4. % INPUT MODEL : Two-Machine One-Buffer Transfer Line Model (Fig. 4)
5. %
6. % Obs: Using matrix notation of ISPN model
7. datenow=now;
8. format long
9. clear At % Clear variable At
10. % input arc multiplicity of immediate transitions, (-) minus means input
11. At(1,1)= [-1, 0; 0, 0; 0, 0; 0, 0; 0, -1; 0, -1; 0, 0; 0, 0; 0, 0; 0, 0]; % P1
12. 0, 0; % P2
13. 0, -1; % P4
14. 0, 0; % P6
15. 0, 0; % P7
16. 0, 0; % P8
17. % labels of immediate transition
18. At(2,1)= [‘t1’;’t5’];
19. % weight of immediate transition
20. At(3,1)= [1, 1];
21. % celldisp(At) % uncomment display cell array contents
22. clear AtO % Clear variable AtO
23. % output arc multiplicity of immediate transitions
24. AtO(1,1)= [0, 0; 1, 0; 0, 0; 0, 0; 0, 0; 0, 0; 0, 1; 0, 1]; % P1
25. 0, 0; % P2
26. 0, 0; % P3
27. 0, 0; % P4
28. 0, 0; % P5
29. 0, 0; % P6
30. 0, 0; % P7
31. 0, 0; % P8
32. % celldisp(AtO) % uncomment display cell array contents
33. clear Ai % Clear variable Ai
34. % arc multiplicity of inhibitor arcs (associated to immediate transitions)
35. Ai(1,1)= [0, 0; 0, 0; 0, 0; 0, 0; 0, 0; 0, 0; 0, 0; 0, 0]; % P1
36. 0, 0; % P2
37. 0, 0; % P3
38. 0, 0; % P4
39. 0, 0; % P5
40. 0, 0; % P6
41. 0, 0; % P7
42. 0, 0; % P8
43. % celldisp(Ai) % uncomment display cell array contents
44. clear AT % Clear variable AT
45. % input arc multiplicity of timed transitions, ( - ) minus means input
48. AT(1,1)= {
    [0, 0, 0, 0, 0, 0;
    -1,-1, 0, 0, 0, 0;
    0, 0, 0, 0, 0, 0;
    0, 0, 0, 0, 0, 0;
    0, 0, 0, 0, 0, 0;
    0, 0, 0, 0, 0, 0];
}
56. % labels of timed transitions
57. AT(2,1)= { ['T2';'T3';'T4';'T6';'T7';'T8']};
60. % interval rate of timed transitions
61. AT(3,1)= { [1,3,5,2,4,6]};
61. % server semantics of timed transitions
62. AT(4,1)= { ['SS';'SS';'SS';'SS';'SS';'SS']};
63. clear ATO % Clear variable ATO
64. ATO(1,1)= {
    [1, 0, 0, 0, 0, 0;
    0, 0, 1, 0, 0, 0;
    0, 1, 0, 0, 0, 0;
    1, 0, 0, 0, 0, 0;
    0, 0, 0, 1, 0, 0;
    0, 0, 0, 0, 0, 1;
    0, 0, 0, 0, 0, 1;
    0, 0, 0, 0, 0, 0];
}
72. % celldisp(AT) % uncomment display cell array contents
73. clear M % Clear variable M
74. clear d % Clear variable d
75. % Initial marking of each place (initial state)
76. M(1,1)={ [1;0;0;0;1;0;0;3] };
116. end
117. end
118. if xt=='new'
119. strx=strcat('M',num2str(c));
120. d{1-1,2}=strx;
121. M{1,c+1}=md;
122. M{2,c+1}=strx;
123. Q{im+1,c+1}=At{3,1}(i);
124. end
125. end
126. % If there is no firing of immediate transitions so we try
127. % to firing timed transitions
128. if t==0
129. for i=1:nT
130. mt=At{1,1}(1:end,i);
131. md=mx+mt;
132. min(md);
133. if min(md)>=0
134. d{1,1}=strcat(M{2,1+im}(1:end),'|',At{2}(1,1:end),'|');
135. md=md+At{1,1}(1:end,i);
136. ma=md;
137. ga=0;
138. while min(ma)>=0
139. ma=ma+At{1,1}(1:end,i);
140. ga=ga+1;
141. end
142. t=1;
143. id=id+1;
144. c=size(M,2);
145. xt='new';
146. for ix=1:c
147. if md==M{1,ix}
148. d{1,2}=x;
149. xt='old';
150. if AT{4}(1,1:en)=SS'
151. Q{im+1,ix}=At{3,1}(i);
152. end
153. if AT{4}(1,1:en)=IS'
154. Q{im+1,ix}=ga*At{3,1}(i);
155. end
156. end
157. break
158. end
159. end
160. end
161. strx=strcat('M',num2str(c));
162. d{1-1,2}=strx;
163. M{1,c+1}=md;
164. M{2,c+1}=strx;
165. if AT{4}(1,1:end)'SS'
166. Q{im+1,c+1}=At{3,1}(i);
167. end
168. if AT{4}(1,1:end)'IS'
169. Q{im+1,c+1}=ga*At{3,1}(i);
170. end
171. end
172. end
173. end
174. end
175. im=im+1; % provoca termino do loop while
176. if tvm==0
177. vm{ivm}=im-1;
178. ivm=ivm+1;
179. else
180. tm{itm}=im-1;
181. itm=itm+1;
182. end
183. end
184. end
185. clear At % Clear variable At
186. clear Ao % Clear variable Ao
187. clear AT % Clear variable AT
188. clear ATO % Clear variable ATO
189. clear Ai % Clear variable Ai
190. %===============================================
191. % celldisp(M); % uncomment display reachability set and distribution
192. % markings from ISPN = ( Table 4 )
193. % d % uncomment display literal description of reachability graph from
194. % ISPN = ( Table 5 )
195. %==============================================
196. % 'number of vanishing markings'
197. ivm=ivm-1;
198. % 'number of tangible markings'
199. itm=itm-1;
200. n=ivm+itm;
201. clear PVV % Clear variable PVV
202. PVV=intval(zeros(ivm,ivm));
203. i = (1:ivm);
204. j=(1:ivm);
205. PVV(i,j)=Q(vm(i)+1,vm(j)+1);
206. % weight uniformization of immediate transitions
207. for i= 1:ivm
208. s=sum(PVV(i,1:ivm));
209. if s>1
210. PVV(i,1:ivm)=PVV(i,1:ivm)/s;
211. end
212. % PVV % uncomment display PVV
213. clear PVT % Clear variable PVT
214. i = (1:ivm);
215. j=(1:itm);
216. PVT(i,j)=Q(vm(i)+1,tm(j)+1);
217. % PVT % uncomment display PVT
218. clear UTV % Clear variable UTV
219. i = (1:itm);
220. j=(1:ivm);
221. UTV(i,j)=Q(tm(i)+1,vm(j)+1);
222. % UTV % uncomment display UTV
223. clear UTT % Clear variable UTT
224. i = (1:itm);
225. j=(1: itm);
226. UTT(i,j)=Q(tm(i)+1,tm(j)+1);
227. clear Q % Clear variable Q
228. % 'Calculating X=eye(size(PVV,1))-PVV'
229. X=eye(size(PVV,1))-PVV;
230. % 'Calculating X=inv(X)' %
231. X=inv(X);
232. % 'Calculating U=UTT+UTV*X*PVT'
233. U=UTT+UTV*X*PVT;
234. % U % uncomment display U
235. clear PVT % Clear variable PVT
236. clear UTV % Clear variable UTV
237. clear UTT % Clear variable UTT
238. clear X % Clear variable X
239. n=size(U,1);
240. t=sum(U);
241. for i=1:n
242. U(i,i)=t(i);
243. end
244. Q=U;
245. % Q % uncomment display Q
246. clear U % Clear variable U
247. clear UTV % Clear variable UTV
248. clear UTT % Clear variable UTT
249. clear PVT % Clear variable PVT
250. clear X % Clear variable X
251. n=size(U,1);
252. t=sum(U);
253. for i=1:n
254. U(i,i)=t(i);
255. end
256. Q=U;
257. % Q % uncomment display Q
258. clear U % Clear variable U
259. clear UTV % Clear variable UTV
260. clear UTT % Clear variable UTT
261. clear PVT % Clear variable PVT
262. clear X % Clear variable X
263. n=size(U,1);
264. t=sum(U);
265. for i=1:n
266. U(i,i)=t(i);
267. end
268. Q=U;
269. % Q % uncomment display Q
270. clear U % Clear variable U
271. clear UTV % Clear variable UTV
272. clear UTT % Clear variable UTT
273. clear PVT % Clear variable PVT
274. clear X % Clear variable X
275. n=size(U,1);
276. t=sum(U);
277. for i=1:n
278. U(i,i)=t(i);
279. end
280. Q=U;
281. % Q % uncomment display Q
282. clear U % Clear variable U
283. clear UTV % Clear variable UTV
284. clear UTT % Clear variable UTT
285. clear PVT % Clear variable PVT
286. clear X % Clear variable X
287. n=size(U,1);
288. t=sum(U);
289. for i=1:n
290. U(i,i)=t(i);
291. end
292. Q=U;
293. % Q % uncomment display Q
294. clear U % Clear variable U
250. QZ=Q.^
251. clear Q % Clear variable Q
252. m=size(QZ,1);
253. j= 1:m;
254. QZ(m,j)=1;
255. QZ;
256. Z(m)=1;
257. Z=Z.^
258. % 'Calculating x=verifylss(QZ,Z)' 
259. x=verifylss(QZ,Z)
260. % [UM1] = [prob] (m(p_{2}) = 1)
261. 'UM1'
262. s1=0;
263. for i= 1:n 
264. if M{1,tm(i)+1} (2) >0 
265. s1=s1+x(i);
266. end
267. end 
268. intvalinit('displaymidrad')
269. s1 
270. intvalinit('displayinfsup')
271. s1 
272. % [UM2] = [prob] (m(p_{6}) = 1)
273. 'UM2'
274. s1=0;
275. for i= 1:n 
276. if M{1,tm(i)+1} (6) >0 
277. s1=s1+x(i);
278. end 
279. end 
280. intvalinit('displaymidrad')
281. s1 
282. intvalinit('displayinfsup')
283. s1 
284. datestr(now,0) 

Executing ISPN.m yields:

>> ISPN
ans =
01-Apr-2011 23:59:14
intval x =
[ 0.30162341059172, 0.30162341059173]
[ 0.20129241213850, 0.20129241213851]
[ 0.10001579083719, 0.10001579083720]
[ 0.05079591445866, 0.05079591445867]
[ 0.05122652318522, 0.05122652318523]
[ 0.03402132703571, 0.03402132703572]
[ 0.02728986215974, 0.02728986215975]
[ 0.03607316886026, 0.03607316886027]
[ 0.02968055852738, 0.02968055852739]
[ 0.01976172320179, 0.01976172320180]
[ 0.02527256969829, 0.02527256969830]
[ 0.01396928749535, 0.01396928749536]
[ 0.02086458086920, 0.02086458086922]
[ 0.02032580994373, 0.02032580994374]
[ 0.01077008114445, 0.01077008114446]
ans =
UM1
===> Default display of intervals by midpoint/radius (e.g. < 3.14 , 0.01 >)
intval s1 =
< 0.59650101272373, 0.59650101272374>
===> Default display of intervals by infimum/supremum (e.g. [ 3.14 , 3.15 ])
intval s1 =
[ 0.59650101272372, 0.59650101272374]
ans =
6. Concluding remarks

In this chapter, ISPN is used as an approach to ISPN performance analysis in which the exponential rates fall within pre-assumed intervals. ISPN is mainly applied in modeling, where input data are known within definite interval of accuracy. Such uncertainties include the errors involved with experimental data obtained from measurements. This framework provides a way to formalize and study problems related to the presence of uncertainties. Such uncertainties include data errors occurring during data measurements and rounding errors generated during calculations. The model proposed and the related method of analysis, involves the case of simultaneous variability in values of parameters. As an immediate consequence, the ISPN analysis, designed for evaluation of results obtained from measurements, may appear to be useful for engineers and technicians as a tool for decision making. As future works, methods for interval transient analysis and simulation should considered. Furthermore, other case studies should also be take into account. ISPN MATLAB toolbox INTLAB Prototype Tool will allow you to specify your own ISPNs. We plan to post future developments of ISPN MATLAB toolbox INTLAB Prototype Tool.

7. References


www.intechopen.com


The book consists of 24 chapters illustrating a wide range of areas where MATLAB tools are applied. These areas include mathematics, physics, chemistry and chemical engineering, mechanical engineering, biological (molecular biology) and medical sciences, communication and control systems, digital signal, image and video processing, system modeling and simulation. Many interesting problems have been included throughout the book, and its contents will be beneficial for students and professionals in wide areas of interest.

How to reference
In order to correctly reference this scholarly work, feel free to copy and paste the following:
