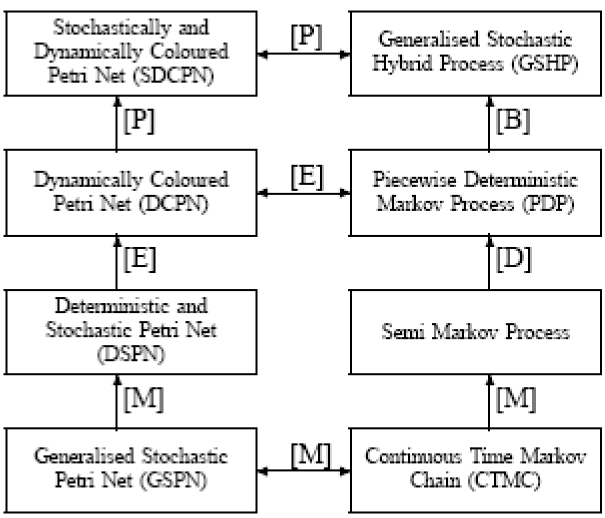
NLR-TP-2008-402

NLR Air Transport Safety Institute

Research & Consultancy

Executive summary

Enhancing hybrid state Petri nets with the analysis power of stochastic hybrid processes



Problem area
Petri nets have shown to be useful for developing models for various practical applications. Numerous extensions to the basic formalism have been developed, which combine different modelling features in an integrated way. Of particular interest are hybrid state Petri net extensions, which combine discrete and continuous system aspects and offer a framework for the modelling of a hybrid system. Their weakness, however, is that the resulting hybrid systems lack powerful stochastic analysis tools such as those existing for Markov processes.

Description of work
This paper presents a power hierarchy of Petri Nets and Stochastic Hybrid Processes. At the left-hand-side of this power hierarchy are stochastic Petri net models at the bottom, and high level Petri nets at the top. At the right-hand-side of this power hierarchy are Markov chains at the bottom, and generalized hybrid state Markov processes at the top. The Petri net side of the power hierarchy makes it possible to specify a stochastic system model in a compositional way. The Markov process side of the power hierarchy exploits the available stochastic analysis tools. Between the Petri nets on

Report no.
NLR-TP-2008-402

Author(s)
M.H.C. Everdij
H.A.P. Blom

Report classification
UNCLASSIFIED

Date
May 2008

Knowledge area(s)
Safety & Security

Descriptor(s)
Petri net
Stochastic process
Safety
Air Traffic

This report is based on a presentation held at the International Workshop on Discrete Event Systems (WODES 2008), Göteborg, Sweden, 28-30 May, 2008.

NLR Air Transport Safety Institute
Research & Consultancy

the left and the Markov processes on the right are mathematical one-to-one mappings, which enable taking advantage of both the modelling power of hybrid state Petri nets and the analysis capability of stochastic hybrid Markov processes.

Results and conclusions
This paper reveals the existence of one-to-one mappings between Petri net classes and Markov process classes, and explains that both formalisms benefit from each other's strengths.

Applicability
The hybrid state Petri net formalism developed in this paper has been used within the framework of NLR's Traffic Organization and Perturbation AnalyZer (TOPAZ) for more than a decade. As such, it has been applied to numerous safety assessments of air traffic operations, both at NLR and beyond.

Nationaal Lucht- en Ruimtevaartlaboratorium, National Aerospace Laboratory NLR

Anthony Fokkerweg 2, 1059 CM Amsterdam,
P.O. Box 90502, 1006 BM Amsterdam, The Netherlands
Telephone +31 20 511 31 13, Fax +31 20 511 32 10, Web site: www.nlr.nl

NLR Air Transport Safety Institute
Research & Consultancy

NLR-TP-2008-402

ENHANCING HYBRID STATE PETRI NETS WITH THE ANALYSIS POWER OF STOCHASTIC HYBRID PROCESSES

M.H.C. Everdij
H.A.P. Blom

This report is based on a presentation held at the International Workshop on Discrete Event Systems (WODES 2008), Göteborg, Sweden, 28-30 May, 2008.

Customer	NLR
Contract number	----
Owner	NLR
Division	Air Transport
Distribution	Unlimited
Classification of title	Unclassified
	May 2008

Approved by:

Author	Reviewer	Managing department
ME 5/12/2008	Anonymous peer reviewers	9-12-2008

Contents

Enhancing hybrid state Petri nets with the analysis power of stochastic hybrid processes

Mariken H.C. Everdij and Henk A.P. Blom

***Abstract*— This paper presents a power hierarchy of Petri Nets and Stochastic Hybrid Processes. At the left-hand-side of this power hierarchy are stochastic Petri net models at the bottom, and high level Petri nets at the top. At the right-hand-side of this power hierarchy are Markov chains at the bottom, and generalized hybrid state Markov processes at the top. The Petri net side of the power hierarchy makes it possible to specify a stochastic system model in a compositional way. The Markov process side of the power hierarchy exploits the available stochastic analysis tools. Between the Petri nets on the left and the Markov processes on the right are mathematical one-to-one mappings, which enable taking advantage of both the modelling power of hybrid state Petri nets and the analysis capability of stochastic hybrid Markov processes.**

I. Introduction

Petri nets, e.g. David and Alla (1994), have shown to be useful for developing models for various practical applications. Typical features are graphical and equational representation, natural expression of causal dependencies, concurrency and synchronisation mechanism, and hierarchical and modular construction. Numerous extensions to the basic formalism have been developed, which combine different modelling features in an integrated way. Of particular interest are hybrid state Petri net extensions, see e.g. the collection at (Giua, 1999), which combine discrete and continuous system aspects and offer a powerful framework for the modelling of a hybrid system. Their weakness, however, is that the resulting hybrid systems lack powerful stochastic analysis tools such as those existing for Markov processes. This paper reveals the existence of one-to-one mappings between Petri net classes and Markov process classes, and explains that both formalisms benefit from each other's strengths.

Malhotra and Trivedi (1994) and Muppala *et al.* (2000) started the development of a hierarchy of various dependability models based on their modelling power. At the left-hand-side of this power hierarchy are Petri net models, with Generalised Stochastic Petri Nets (GSPN) at the bottom, and Deterministic and Stochastic Petri Nets (DSPN) at the top. At the right-hand-side of this power hierarchy are Markov chains at the bottom and Semi-Markov Processes at the top. Bobbio *et al.* (1998) extended this hierarchy to include Fluid Stochastic Petri Nets (FSPN) on the left, and Fluid Flow models on the right. Arrows between different formalisms indicate that one-to-one mappings exist, i.e. that the elements of one formalism can be represented in terms of the elements of the other formalism, such that the executions,

Both authors are with National Aerospace Laboratory NLR, P.O. Box 90502, 1006 BM Amsterdam, The Netherlands, Fax: +31 20 511 3210, Tel: +31 20 511 3522, Email: everdij@nlr.nl, blom@nlr.nl

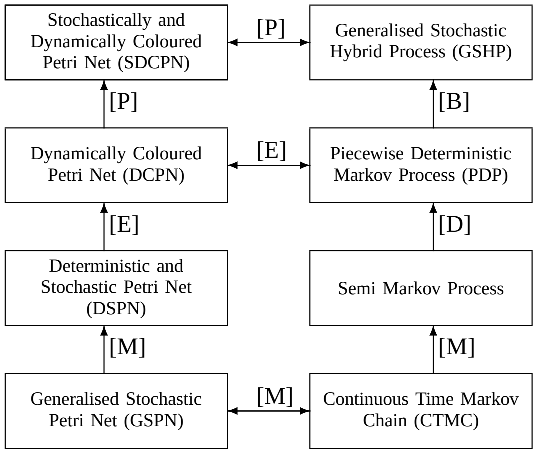


Fig. 1. Power hierarchy among various model types. An arrow from a model to another model indicates that the second model has more modelling power than the first model. The [M] arrows have been established in (Malhotra and Trivedi, 1994; Muppala *et al.*, 2000). The [D] arrow is established in (Davis, 1984). The [B] arrow is established in (Bujorianu and Lygeros, 2006). The [E] arrows are established in (Everdij and Blom, 2003, 2005). The [P] arrows are established in (Everdij and Blom, 2006).

i.e. their solutions as a stochastic process, are equivalent. Such external equivalence between systems is also referred to as a bisimulation, and the executions are said to be bisimilar, see e.g. (Van der Schaft, 2004; Bujorianu *et al.*, 2005; Bujorianu and Bujorianu, 2006).

Everdij and Blom (2003) extended the power hierarchy to include one-to-one mappings between Dynamically Coloured Petri Nets (DCPN) at the left-hand-side and Piecewise Deterministic Markov Processes (PDP) at the right-hand-side. PDPs have been introduced by Davis (1984, 1993) as the most general class of continuous-time Markov processes which include both discrete and continuous processes, except diffusion. Branicky (1995) identified a close relation between PDPs and hybrid automata, which have shown to be useful for application in problems of decidability, formal verification and control synthesis (Alur *et al.*, 1993; Lygeros *et al.*, 1998; Van Schuppen, 1998; Sipser, 1997; Tomlin *et al.*, 1998; Weinberg *et al.*, 1996). Stochastic hybrid processes have proven to be supported by powerful analysis tools (Davis, 1993; Elliott, 1982; Elliott *et al.*, 1995; Ethier and Kurtz, 1986). Due to the existence of one-to-one mappings between DCPN and PDP, the class of DCPN developed in Everdij and Blom (2003, 2005) supports the modelling of hybrid automata and PDPs for complex practical problems, similarly as Stochastic Petri Nets support the development of a Markov chain for complex discrete valued problems. The current paper presents the further extension of the power hierarchy, see Fig. 1, by including Brownian motion both in PDP and in DCPN, and by extending the one-to-one mappings.

Including Brownian motion in PDP at the top-right-hand-side of Fig. 1 yields Generalised Stochastic Hybrid Processes (GSHP) (Bujorianu and Lygeros, 2006) as an execution of a stochastic hybrid automaton which is referred to as Generalised Stochastic Hybrid System (GSHS). Bujorianu and Lygeros (2006) showed that GSHP is a strong Markov process. At the top-left-hand-side of the resulting power-hierarchy (Fig. 1) are Stochastically and Dynamically Coloured Petri Nets (SDCPN), which is an extension of DCPN by incorporating Brownian motion (Everdij and Blom, 2006). SDCPN can be mapped one-to-one to the elements of a GSHS, in such way that the executions of both are bisimilar, i.e. they are probabilistically equivalent. This relation between systems, processes and mappings is depicted in Fig. 2. The SDCPN formalism supports compositional specification of complex hybrid systems very well (e.g. Everdij *et al.*, 2006). GSHS supports exploitation of automaton-based formal methods. GSHP supports exploitation of powerful stochastic analysis tools. Due to the one-to-one mappings, tools available for one formalism become available to the other formalisms as well.

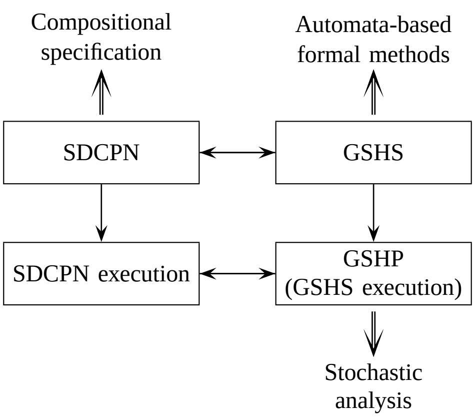


Fig. 2. Relationship between SDCPN, GSHS and GSHP, and their main capability support

The organisation of this paper is as follows. Section II defines GSHS and GSHP. Section III defines SDCPN and the related SDCPN process. Section IV shows that the SDCPN elements can be mapped one-to-one to the GSHS elements and vice versa, and that the processes generated by SDCPN and by GSHS are equivalent. Section V gives an example SDCPN. Section VI presents this SDCPN example by a GSHS and GSHP. Section VII gives conclusions.

II. GSHS Automaton

A GSHS (Bujorianu *et al.*, 2005; Bujorianu and Lygeros, 2006) is presented as an automaton ($\mathbf{K}$, d, $\mathcal{X}$, f, g, *Init*, λ, Q), where

- $\mathbf{K}$ is a countable set.
- $d : \mathbf{K} \to \mathbb{N}$ maps each $\theta \in \mathbf{K}$ to a natural number.
- $\mathcal{X} : \mathbf{K} \to \{E_\theta \mid \theta \in \mathbf{K}\}$ maps each $\theta \in \mathbf{K}$ to an open subset E_θ of $\mathbb{R}^{d(\theta)}$. With this, the hybrid state space is given by $E \triangleq \{(\theta, x) \mid \theta \in \mathbf{K}, x \in E_\theta\}$.
- $f : E \to \{\mathbb{R}^{d(\theta)} \mid \theta \in \mathbf{K}\}$ is a vector field.
- $g : E \to \{\mathbb{R}^{d(\theta)\times b} \mid \theta \in \mathbf{K}\}$ is a matrix field, with $b \in \mathbb{N}$.
- *Init*: $\mathcal{B}(E) \to [0, 1]$ is an initial probability measure on $(E, \mathcal{B}(E))$, where $\mathcal{B}(E)$ is the Borel σ-algebra on E.
- $\lambda : E \to \mathbb{R}^+$ is a jump rate function.
- $Q : \mathcal{B}(E) \times (E \cup \partial E) \to [0, 1]$ is a GSHS transition measure, where $\partial E \triangleq \{(\theta, x) \mid \theta \in \mathbf{K}, x \in \partial E_\theta\}$, in which ∂E_θ is the boundary of E_θ. For any $(\theta, x) \in E \cup \partial E$, $Q(\cdot; \theta, x)$ is a probability measure.

The execution of a GSHS is defined as a stochastic process, i.e. a GSHP, which takes values in the hybrid state space E and consists of two components: a discrete valued component $\{\theta_t\}$, and a continuous valued component $\{x_t\}$. From an initial state value onwards, which is generated by *Init*, the GSHP follows a trajectory which is generated by a stochastic differential equation defined by drift coefficient f and diffusion coefficient g. A jump in the GSHP state occurs either when a doubly stochastic Poisson point process generates a point with rate λ or when the GSHP state hits the boundary ∂E. The GSHS transition measure Q generates the value of the GSHP state after the jump.

Formally, this is defined as follows (Bujorianu *et al.*, 2005): A stochastic process $\{\xi_t\} = \{\theta_t, x_t\}$ is called a *GSHS execution* if there exists a sequence of stopping times $0 = \tau_0 < \tau_1 < \tau_2 \cdots$ such that for each $k \in \mathbb{N}$: $\xi_0 = (\theta_0, x_0)$ is an E-valued random variable extracted according to the probability measure *Init*; For $t \in [\tau_k, \tau_{k+1})$, $\theta_t = \theta_{\tau_k}$ and $x_t = x_t^k$, where for $t \geq \tau_k$, x_t^k is a solution of the stochastic differential equation $dx_t^k = f(\theta_{\tau_k}, x_t^k)dt + g(\theta_{\tau_k}, x_t^k)dw_t$ with initial condition $x_{\tau_k}^k = x_{\tau_k}$, and where $\{w_t\}$ is b-dimensional standard Brownian motion; $\tau_{k+1} = \tau_k + \sigma_k$, where σ_k is chosen according to a survivor function which is given by $F(t) = \mathbf{I}_{(t<\tau^*)} \exp(-\int_0^t \lambda(\theta, x_s^k)ds)$. Here, $\mathbf{I}$ is the indicator function and τ^* is the first time $> \tau_k$ when $\{x_t^k\}$ hits the boundary $\partial E_{\theta_{\tau_k}}$; The probability distribution of $\xi_{\tau_{k+1}}$, i.e. the hybrid state right after the jump, is governed by the law $Q(\cdot; (\theta_{\tau_k}, x_{\tau_{k+1}-}))$. Under standard assumptions (on the diffusion coefficients, non-Zeno executions, transition measure, etc., Bujorianu and Lygeros (2006)), any GSHS is a strong Markov Process and it has the cadlag property (i.e. it is right continuous with left hand limits).

III. SDCPN

Everdij and Blom (2006) defined a Stochastically and Dynamically Coloured Petri Net (SDCPN) by a collection ($\mathcal{P}$, $\mathcal{T}$, $\mathcal{A}$, $\mathcal{N}$, $\mathcal{S}$, $\mathcal{C}$, $\mathcal{I}$, $\mathcal{V}$, $\mathcal{W}$, $\mathcal{G}$, $\mathcal{D}$, $\mathcal{F}$), together will some SDCPN transition firing rules, where:

$\mathcal{P}$ is a set of places.

$\mathcal{T}$ is a set of transitions which consists of a set $\mathcal{T}_G$ of guard transitions, a set $\mathcal{T}_D$ of delay transitions, and a set $\mathcal{T}_I$ of immediate transitions.

$\mathcal{A}$ is a finite set of arcs, which consists of a set $\mathcal{A}_O$ of ordinary arcs, a set $\mathcal{A}_E$ of enabling arcs, and a set $\mathcal{A}_I$

of inhibitor arcs.

$\mathcal{N}$ is a node function which maps each arc to an ordered pair of one transition and one place.

$\mathcal{S}$ is a set of colour types for the tokens occurring in the net (a colour is the value of an object or process in Petri net terminology).

$\mathcal{C}$ is a colour type function which maps each place to a colour type in S.

$\mathcal{I}$ is an initial marking which defines the set of tokens initially present, i.e., it specifies in which places they initially reside, and the colours they initially have.

$\mathcal{V}$ is a set of place specific token colour functions which describe the drift coefficient of a stochastic differential equation for the colour of a token while it resides in a specific place. This drift coefficient is locally Lipschitz continuous.

$\mathcal{W}$ is a set of place specific token colour matrix functions which describe the diffusion coefficient of a stochastic differential equation for the colour of a token while it resides in a specific place. This diffusion coefficient is locally Lipschitz continuous.

$\mathcal{G}$ is a set of boolean-valued transition guards associating each transition in $\mathcal{T}_G$ with a guard function which is evaluated when the transition has a token in each of its input places. The guard function must evaluate to True before the transition is allowed to fire (i.e. remove and produce tokens). Its evaluation depends on the colours of the input tokens of the transition.

$\mathcal{D}$ is a set of transition delays associating each transition in $\mathcal{T}_D$ with a delay function which is evaluated when the transition has a token in each of its input places. The delay function determines for how long the transition must wait before it is allowed to fire (i.e. remove and produce tokens). The firing rate depends on the colours of the input tokens of the transition.

$\mathcal{F}$ is a set of firing measures describing the quantity and colours of the tokens produced by the transitions at their firing. Their evaluation depends on the colours of the input tokens of the transition.

Below, the graphical representation of the elements in $\mathcal{P}$, $\mathcal{T}$ and $\mathcal{A}$ are given. The node function $\mathcal{N}$ describes how these components are connected, so that together they define a Petri net graph.

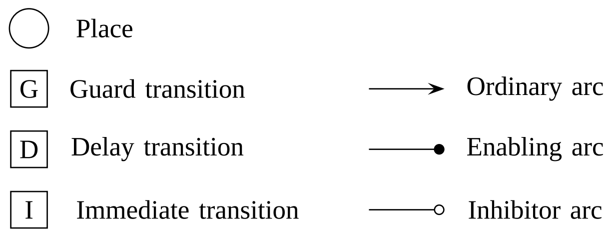


Fig. 3. Graphical notation for places, transitions and arcs in an SDCPN

Each token in an SDCPN place has a colour which takes values in a domain specified by $\mathcal{C}$. The colour value of a token evolves through time according to a stochastic differential equation that is governed by the token colour function and by the token colour matrix function of the specific place where the token resides.

The SDCPN transition firing rules are as follows: A transition can fire if it is *enabled*, which is if two conditions are both satisfied. The first condition is that the transition must be *pre-enabled*, i.e. have at least one token per ordinary arc and one token per enabling arc in each of its input places and have no token in the input places to which it is connected by an inhibitor arc. The second condition differs per type of transition. For immediate transitions the second condition is automatically satisfied if the transition is pre-enabled. For guard transitions the second condition is specified by the set of transition guards $\mathcal{G}$ and for delay transitions it is specified by the set of transition delays $\mathcal{D}$. An enabled transition fires, i.e. removes tokens from the input places by which it is connected through an *ordinary arc*; subsequently, the transition produces a token for some or all of its output places. The firing function $\mathcal{F}$ specifies the colours of the produced tokens and the places for which they are produced. The evaluation of $\mathcal{G}$, $\mathcal{D}$ and $\mathcal{F}$ may be dependent on the colours of the input tokens of the corresponding transition. The following additional rules (and their applicable combinations) apply in case of simultaneous enablings:

R_0 The firing of an immediate transition has priority over the firing of a guard or a delay transition.

R_1 If one transition becomes enabled by two or more sets of input tokens at exactly the same time, and the firing of any one set will not disable one or more other sets, then it will fire these sets of tokens independently, at the same time.

R_2 If one transition becomes enabled by two or more sets of input tokens at exactly the same time, and the firing of any one set disables the other sets, then the set that is fired is selected randomly, with the same probability for each set.

R_3 If two or more transitions become enabled at exactly the same time and the firing of any one transition will not disable the other transitions, then they will fire at the same time.

R_4 If two or more transitions become enabled at exactly the same time and the firing of any one transition disables some other transitions, then each combination of transitions that can fire independently without leaving enabled transitions gets the same probability if firing.

The marking of the SDCPN is given by the numbers of tokens in the places and the associated colour values of these tokens. The marking is unique except possibly when one or more transitions fire (particularly, immediate transitions fire without delay hence a sequence of immediate transitions firing will generate a sequence of markings at the same time instant). The SDCPN marking at each time instant can be mapped to a unique SDCPN-generated stochastic process, as follows: at times when no transitions fire, the SDCPN-generated process state is equal to the SDCPN marking. If at time t one or more transitions fire, then the set of applicable markings is collected in $M_t = \{M_t^i \mid M_t^i$ is a marking at time $t\}$, and the SDCPN-generated process state at time t is defined by $\{M_t^i \mid M_t^i \in M_t$ and no transitions are enabled in $M_t^i\}$. In other words, the process state is defined to be the marking that occurs after all transitions that fire at time t have been fired. With this, the SDCPN-generated stochastic process is unique, and it is continuous from the right with left-hand limits (cadlag).

IV. EXECUTIONS OF SDCPN AND GSHS ARE BISIMILAR

An important property of SDCPN is that there exists a Generalised Stochastic Hybrid System (GSHS) which is a bisimulation. This is shown in (Everdij and Blom, 2006) as formulated in the two theorems below.

Theorem 1:
For an execution of an arbitrary Generalised Stochastic Hybrid System with a finite domain $\mathbf{K}$ there exists an equivalent process which is the execution of a Stochastically and Dynamically Coloured Petri Net ($\mathcal{P}$, $\mathcal{T}$, $\mathcal{A}$, $\mathcal{N}$, $\mathcal{S}$, $\mathcal{C}$, $\mathcal{I}$, $\mathcal{V}$, $\mathcal{W}$, $\mathcal{G}$, $\mathcal{D}$, $\mathcal{F}$) satisfying R_0 through R_4.

Theorem 2:
For an execution of a Stochastically and Dynamically Coloured Petri Net ($\mathcal{P}$, $\mathcal{T}$, $\mathcal{A}$, $\mathcal{N}$, $\mathcal{S}$, $\mathcal{C}$, $\mathcal{I}$, $\mathcal{V}$, $\mathcal{W}$, $\mathcal{G}$, $\mathcal{D}$, $\mathcal{F}$) satisfying R_0 through R_4 there exists an equivalent process which is the execution of a Generalised Stochastic Hybrid System if the following conditions are satisfied:

- D_1 There are no explosions, i.e. the time at which a token colour equals $+\infty$ or $-\infty$ approaches infinity whenever the time until the first guard transition enabling moment approaches infinity.
- D_2 After a transition firing (or after a sequence of firings that occur at the same time instant) at least one place must contain a different number of tokens, or the colour of at least one token must have jumped.
- D_3 In a finite time interval, each transition is expected to fire a finite number of times and for $t \to \infty$, the number of tokens remains finite.
- D_4 In the initial marking, no immediate transition is enabled.

Theorems 1 and 2 imply that the executions of SDCPN and GSHS are bisimilar (Bujorianu and Bujorianu, 2006). The implications of this bisimilarity result are great: On the one side, analysis tools designed for GSHP and properties of GSHP become available for Petri nets. Examples of these properties are convergence in discretisation, existence of limits, existence of event probabilities, strong Markov properties, reachability analysis. See e.g. (Davis, 1993; Elliott, 1982; Elliott *et al.*, 1995; Ethier and Kurtz, 1986). On the other side, numerous Petri net features such as those listed in the introduction to this paper, become available when modelling GSHP. This is illustrated in the following sections.

V. SDCPN EXAMPLE

To illustrate the advantages of SDCPN when modelling a complex system, consider a simplified model of the evolution of an aircraft in one sector of airspace.

Assume the deviation of this aircraft from its intended path depends on the operationality of two of its aircraft systems: the engine system, and the navigation system. Each of these aircraft systems can be in one of two modes: *Working* (functioning properly) or *Not working* (operating in some failure mode). Both systems switch between their modes independently and with exponentially distributed sojourn times, with rates δ_3 (engine repaired), δ_4 (engine fails), δ_5 (navigation repaired) and δ_6 (navigation fails), respectively. The operationality of these systems has the following effect on the aircraft path: if both systems are *Working*, the aircraft evolves in *Nominal* mode and the position y_t and velocity s_t of the aircraft are determined by the vector solution of $dx_t = \mathcal{V}_1(x_t)dt + \mathcal{W}_1 dw_t$, where $x_t = (y_t, s_t)'$. If either one, or both, of the systems is *Not working*, the aircraft evolves in *Non-nominal* mode and the position and velocity of the aircraft are determined by the vector solution of $dx_t = \mathcal{V}_2(x_t)dt + \mathcal{W}_2 dw_t$. The factors $\mathcal{W}_1$ and $\mathcal{W}_2$ are determined by wind fluctuations. Initially, the aircraft has position y_0 and velocity s_0, while both its systems are *Working*. The evaluation of this process may be stopped when the aircraft has *Landed*, i.e. its vertical position and velocity are equal to zero.

Fig. 4 shows the SDCPN graph for this example, where,

- P_1 denotes aircraft evolution *Nominal*, i.e. evolution is according to $\mathcal{V}_1$ and $\mathcal{W}_1$.
- P_2 denotes aircraft evolution *Non-nominal*, i.e. evolution is according to $\mathcal{V}_2$ and $\mathcal{W}_2$.
- P_3 and P_4 denote engine system *Not working* and *Working*, respectively.
- P_5 and P_6 denote navigation system *Not working* and *Working*, respectively.
- P_7 denotes the aircraft has landed.
- T_{1a} and T_{1b} denote a transition of aircraft evolution from *Nominal* to *Non-nominal*, due to engine system or navigation system *Not working*, respectively.
- T_2 denotes a transition of aircraft evolution from *Non-nominal* to *Nominal*, due to engine system and navigation system both *Working* again.
- T_3 through T_6 denote transitions between *Working* and *Not working* of the engine and navigation systems.
- T_7 and T_8 denote transitions of the aircraft landing.

The graph in Fig. 4 completely defines SDCPN elements $\mathcal{P}$, $\mathcal{T}$, $\mathcal{A}$ and $\mathcal{N}$, where $\mathcal{T}_G = \{T_7, T_8\}$, $\mathcal{T}_D = \{T_3, T_4, T_5, T_6\}$ and $\mathcal{T}_I = \{T_{1a}, T_{1b}, T_2\}$. The other SDCPN elements are specified below:

- $\mathcal{S}$: Two colour types are defined; $\mathcal{S} = \{\mathbb{R}^0, \mathbb{R}^6\}$.
- $\mathcal{C}$: $\mathcal{C}(P_1) = \mathcal{C}(P_2) = \mathcal{C}(P_7) = \mathbb{R}^6$, i.e. tokens in P_1, P_2 and P_7 have colours in $\mathbb{R}^6$; the colour components model the 3-dimensional position and 3-dimensional velocity of the aircraft. $\mathcal{C}(P_3) = \mathcal{C}(P_4) = \mathcal{C}(P_5) = \mathcal{C}(P_6) = \mathbb{R}^0 \triangleq 0$.

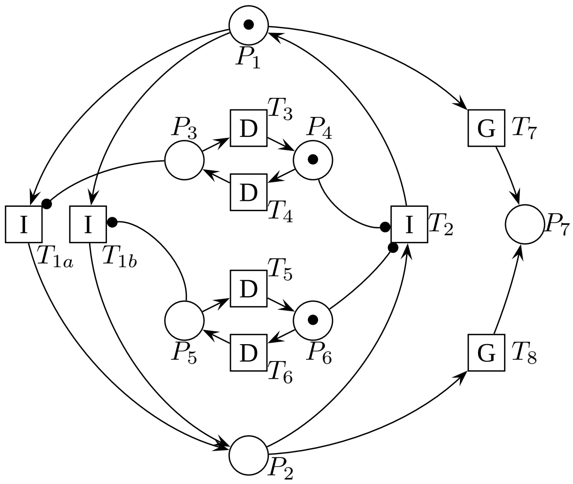


Fig. 4. SDCPN graph for the air traffic example

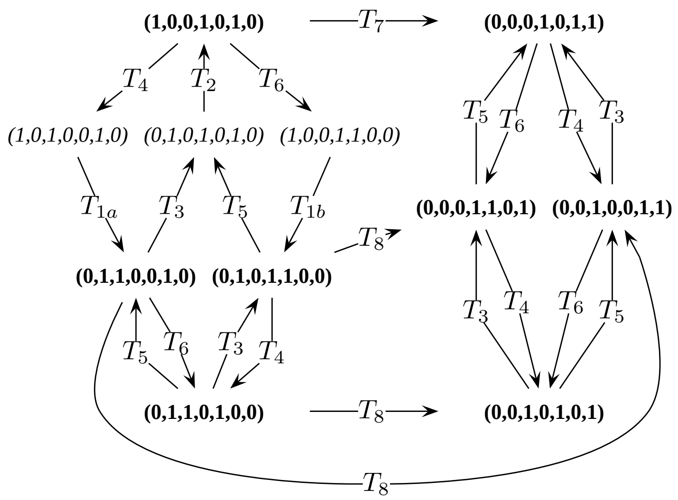


Fig. 5. Reachability graph for the SDCPN of Fig. 4. The nodes in bold type face correspond with the elements of the GSHP discrete state space.

$\mathcal{I}$: Place P_1 initially has a token with colour $x_0 = (y_0, s_0)'$, with $y_0 \in \mathbb{R}^2 \times (0, \infty)$ and $s_0 \in \mathbb{R}^3 \backslash \text{Col}\{0, 0, 0\}$. Places P_4 and P_6 initially each have a token without colour.

$\mathcal{V}$, $\mathcal{W}$: The token colour functions for places P_1, P_2 and P_7 are determined by $(\mathcal{V}_1, \mathcal{W}_1)$, $(\mathcal{V}_2, \mathcal{W}_2)$, and $(\mathcal{V}_7, \mathcal{W}_7)$, respectively, where $(\mathcal{V}_7, \mathcal{W}_7) = (0, 0)$. For places P_3 – P_6 there is no token colour function.

$\mathcal{G}$: Transitions T_7 and T_8 have a guard that is defined by $\partial G_{T_7} = \partial G_{T_8} = \mathbb{R}^2 \times \{0\} \times \mathbb{R}^2 \times \{0\}$.

$\mathcal{D}$: The jump rates for transitions T_3, T_4, T_5 and T_6 are $\delta_{T_3}(\cdot) = \delta_3$, $\delta_{T_4}(\cdot) = \delta_4$, $\delta_{T_5}(\cdot) = \delta_5$ and $\delta_{T_6}(\cdot) = \delta_6$.

$\mathcal{F}$: Each transition has a unique output place, to which it fires a token with a colour (if applicable) equal to the colour of the token removed.

VI. Mapping of SDCPN example to GSHP

In this section, the SDCPN for the aircraft evolution example is transformed into a GSHP, the existence of which is stated in Theorem 2.

The first step is to construct the state space for the GSHP discrete process $\{\theta_t\}$. This is done by identifying the SDCPN *reachability graph*. Nodes in the reachability graph provide the number of tokens in each of the SDCPN places. Arrows connect these nodes as they represent tokens moving by transitions firing. The SDCPN of Fig. 4 has seven places hence the reachability graph for this example has elements that are vectors of length 7. These nodes, excluding the nodes that enable immediate transitions, form the GSHP discrete state space. The reachability graph is shown in Fig. 5, with nodes that form the GSHP discrete state space in Bold typeface, i.e. $\mathbf{K} = \{V_1, \ldots, V_8\}$, with $V_1 = (1,0,0,1,0,1,0)$, $V_2 = (0,1,1,0,0,1,0)$, $V_3 = (0,1,1,0,1,0,0)$, $V_4 = (0,1,0,1,1,0,0)$, $V_5 = (0,0,0,1,0,1,1)$, $V_6 = (0,0,1,0,0,1,1)$, $V_7 = (0,0,1,0,1,0,1)$, $V_8 = (0,0,0,1,1,0,1)$.

Since initially there is a token in places P_1, P_4 and P_6, the initial mode θ_0 equals $\theta_0 = (1,0,0,1,0,1,0)$. The GSHP initial continuous state value equals the vector containing the initial colours of all initial tokens. Since the initial colour of the token in Place P_1 equals x_0, and the tokens in places P_4 and P_6 have no colour, the GSHP initial continuous state value equals x_0.

The GSHP drift coefficient f is given by $f(\theta, \cdot) = \mathcal{V}_1(\cdot)$ for $\theta = V_1$, $f(\theta, \cdot) = \mathcal{V}_2(\cdot)$ for $\theta \in \{V_2, V_3, V_4\}$, and $f(\theta, \cdot) = 0$ otherwise. For the diffusion coefficient, $g(\theta, \cdot) = \mathcal{W}_1$ for $\theta = V_1$, $g(\theta, \cdot) = \mathcal{W}_2$ for $\theta \in \{V_2, V_3, V_4\}$, and $g(\theta, \cdot) = 0$ otherwise.

In GSHP, if the continuous process x_t hits the boundary of its state space, a forced jump occurs. In the SDCPN equivalent, a forced jump occurs if the colour of a token enters the colour boundary of a guard transition that has this token in its input place. Therefore, the state space for the GSHP continuous state process is determined from the transitions guards that, under token distribution θ, are enabled.

The rate of GSHP jumps generated by the Poisson point process is determined from the enabling rates corresponding with the set of delay transitions in $\mathcal{T}_D$ that, under token distribution θ, are pre-enabled. At each time in this example, always two delay transitions are pre-enabled: either T_3 or T_4 and either T_5 or T_6. Hence the GSHP jump rate is equal to $\sum_{i=j,k} \delta_{T_i}(\cdot)$ if T_j and T_k are pre-enabled.

The GSHS transition measure that determines the size of the jumps is determined by the reachability graph, the sets $\mathcal{D}$, $\mathcal{G}$ and $\mathcal{F}$ and the rules R_0–R_4. In Table I, $Q(\theta', x'; \theta, x) = p$ denotes that if (θ, x) is the value of the GSHP before the hybrid jump, then, with probability p, (θ', x') is the value of the GSHP immediately after the jump.

VII. Conclusions

In order to combine the compositional specification power of Petri nets with the analysis power of Markov processes, Malhotra and Trivedi (1994) and Muppala *et al.* (2000) developed a power hierarchy of dependability models. In (Everdij and Blom, 2003, 2005), the power hierarchy was extended with Dynamically Coloured Petri Nets (DCPN)

TABLE I
EXAMPLE GSHS COMPONENT THAT DETERMINES SIZE OF JUMP

For $x \notin \partial E_{V_1}$:	$Q(V_2, x; V_1, x) = \frac{\delta_4}{\delta_4+\delta_6}$, $Q(V_4, x; V_1, x) = \frac{\delta_6}{\delta_4+\delta_6}$
For $x \in \partial E_{V_1}$:	$Q(V_5, x; V_1, x) = 1$
For $x \notin \partial E_{V_2}$:	$Q(V_3, x; V_2, x) = \frac{\delta_6}{\delta_3+\delta_6}$, $Q(V_1, x; V_2, x) = \frac{\delta_3}{\delta_3+\delta_6}$
For $x \in \partial E_{V_2}$:	$Q(V_6, x; V_2, x) = 1$
For $x \notin \partial E_{V_3}$:	$Q(V_4, x; V_3, x) = \frac{\delta_3}{\delta_3+\delta_5}$, $Q(V_2, x; V_3, x) = \frac{\delta_5}{\delta_3+\delta_5}$
For $x \in \partial E_{V_3}$:	$Q(V_7, x; V_3, x) = 1$
For $x \notin \partial E_{V_4}$:	$Q(V_3, x; V_4, x) = \frac{\delta_4}{\delta_4+\delta_5}$, $Q(V_1, x; V_4, x) = \frac{\delta_5}{\delta_4+\delta_5}$
For $x \in \partial E_{V_4}$:	$Q(V_8, x; V_4, x) = 1$
For all x:	$Q(V_6, x; V_5, x) = \frac{\delta_4}{\delta_4+\delta_6}$, $Q(V_8, x; V_5, x) = \frac{\delta_6}{\delta_4+\delta_6}$
For all x:	$Q(V_7, x; V_6, x) = \frac{\delta_6}{\delta_3+\delta_6}$, $Q(V_5, x; V_6, x) = \frac{\delta_3}{\delta_3+\delta_6}$
For all x:	$Q(V_8, x; V_7, x) = \frac{\delta_3}{\delta_3+\delta_5}$, $Q(V_6, x; V_7, x) = \frac{\delta_5}{\delta_3+\delta_5}$
For all x:	$Q(V_7, x; V_8, x) = \frac{\delta_4}{\delta_4+\delta_5}$, $Q(V_5, x; V_8, x) = \frac{\delta_5}{\delta_4+\delta_5}$

and Piecewise Deterministic Markov Processes (PDP). This paper explained further extensions of this power hierarchy: (1) Incorporating Brownian motion in the PDP definition, yielding Generalised Stochastic Hybrid Processes (GSHP); (2) Incorporating Brownian motion in the DCPN definition, yielding Stochastically and Dynamically Coloured Petri Nets (SDCPN); and (3) Showing one-to-one mappings between these formalisms, yielding bisimilar executions. The paper also explained that the bisimilar executions mean that the strengths of the hybrid state Petri nets and stochastic hybrid processes are inherited by each formalism, as has been depicted in Fig. 2 and explained in Section III. As may be clear from the example in Sections V and VI, the GSHP model does not show the structure of the SDCPN. Because of this, the SDCPN model of Section V is simpler to comprehend and to verify against its aircraft evolution operational description than the GSHP example of Section VI. However, by a bisimulation the SDCPN model yields a GSHP model which has full support in stochastic process theory.

It is noted that in (Everdij *et al.*, 2006), the SDCPN specification power is further enhanced for large scale systems through the development of high-level interconnection arcs. The advantages of these enhancements are that a hierarchical modelling approach is available that separates local modelling issues from global modelling issues, and in addition, the number of arcs and transitions necessary to interconnect low-level Petri nets is considerably reduced. The complementary advantages of SDCPN, GSHS and GSHP perspectives tend to increase with the complexity of the considered operation. Due to the bisimulation mappings, the complementary advantages can be exploited: first specify a model in terms of SDCPN, using the compositional specification features, then map this to GSHS (which is supported by formal verification tools) and to GSHP, and then use the stochastic analysis tools of GSHP.

REFERENCES

[1] Alur, R., Courcoubetis, C., Henzinger, T., Ho, P-H. (1993). Hybrid Automata: An algorithmic approach to the specification and verification of hybrid systems, *Hybrid Systems I*, LNCS 736, pp. 209–229, Springer-Verlag.

[2] Bobbio, A., Puliafito, A., Telek, M., Trivedi, K.S. (1998). Recent developments in non-Markovian stochastic Petri nets. *Journal of Systems Circuits & Computers*, Vol. 8, No. 1. pp. 119–158.

[3] Branicky, M.S. (1995). Studies in Hybrid Systems: Modelling, analysis and control, *PhD thesis, M.I.T., Cambridge, MA*.

[4] Bujorianu, M.L., Lygeros, J., Bujorianu, M.C. (2005). Bisimulation for general stochastic hybrid systems. In: M. Morari & L. Thiele (Eds), *Proc. 8th Int. Workshop Hybrid Systems: Computation & Control (HSCC), Zürich, Switzerland*, LNCS 3414, pp. 198–214.

[5] Bujorianu, M.L., Lygeros, J. (2006). *Toward a general theory of stochastic hybrid systems*. In: H.A.P. Blom & J. Lygeros (Eds), *Stochastic hybrid systems: theory and safety critical applications*, LNCIS 337, pp. 33–30. Springer.

[6] Bujorianu, M.L., Bujorianu, M.C. (2006). Model checking for a class of performance properties of fluid stochastic models. In: A. Horváth & M. Telek (Eds), *Proc. 3rd European Performance Evaluation Workshop (EPEW), Budapest, Hungary*, LNCS 4054, pp. 93–107.

[7] David, R., Alla, H. (1994). Petri Nets for the modeling of dynamic systems — A survey, *Automatica*, Vol. 30, No. 2, pp. 175–202.

[8] Davis, M.H.A. (1984). Piecewise Deterministic Markov Processes: a general class of non-diffusion stochastic models, *Journal Royal Statistical Soc. (B)*, Vol. 46, No. 3, pp. 353–388.

[9] Davis, M.H.A. (1993). Markov models and optimization. *Monographs on statistics and applied probability*, Vol. 49, Chapman & Hall.

[10] Elliott, R.J., (1982). Stochastic calculus and applications, Vol. 18 of *Applications of mathematics: Stochastic modelling and applied probability*. A.V. Balakrishnan (Ed), Springer-Verlag.

[11] Elliott, R.J., Aggoun, L., Moore, J.B., (1995). Hidden Markov Models: Estimation and control, Vol. 29 of *Applications of mathematics: Stochastic modelling and applied probability*. I. Karatzas & M. Yor (Eds), Springer-Verlag.

[12] Ethier, S.N., Kurtz, T.G. (1986). Markov processes: Characterization and convergence, *Wiley series probability & mathematical statistics*.

[13] Everdij, M.H.C., Blom, H.A.P. (2003). Petri nets and hybrid state Markov processes in a power-hierarchy of dependability models. In *Proc. IFAC Conf. Analysis & Design of Hybrid Systems (ADHS), Saint-Malo, France*, pp. 355–360.

[14] Everdij, M.H.C., Blom, H.A.P. (2005). Piecewise deterministic Markov processes represented by dynamically coloured Petri nets. *Stochastics*, Vol. 77, No. 1, pp. 1–29.

[15] Everdij, M.H.C., Blom. H.A.P. (2006). Hybrid Petri nets with diffusion that have into-mappings with generalised stochastic hybrid processes. In H.A.P. Blom & J. Lygeros (Eds), *Stochastic hybrid systems: theory and safety critical applications*, LNCIS 337, pp. 31–63. Springer.

[16] Everdij, M.H.C., Klompstra, M.B., Blom, H.A.P. B. Klein Obbink, B. (2006). Compositional specification of a multi-agent system by stochastically and dynamically coloured Petri nets. In H.A.P. Blom & J. Lygeros (Eds), *Stochastic hybrid systems: theory and safety critical applications*, LNCIS 337, pp. 325–350. Springer.

[17] Giua, A. (Editor) Bibliography on hybrid Petri nets, Maintained from 1999. http://bode.diee.unica.it/ hpn/.

[18] Lygeros, J., Pappas, G.J., Sastry, S. (1998). An approach to the verification of the Center-TRACON automation system, *Proc. 1st Int. Workshop Hybrid Systems: Computation & Control (HSCC), Berkeley, CA*, LNCS 1386, pp. 289–304.

[19] Malhotra, M., Trivedi, K.S. (1994). Power-hierarchy of dependability-model types, *IEEE Trans. Reliability*, Vol. R-43, No. 3, pp. 493–502.

[20] Muppala, J.K., Fricks, R.M., Trivedi, K.S. (2000). Techniques for system dependability evaluation, In W. Grasman (Ed), *Computational probability*, pp. 445–480, Kluwer.

[21] Sipser, M. (1997). Introduction to the theory of computation, PWS publishing company, Boston.

[22] Tomlin, C., Lygeros, J., Sastry, S (1998). Synthesising controllers for nonlinear hybrid systems, *Proc. 1st Int. Workshop Hybrid Systems: Computation & Control (HSCC), Berkeley, CA*, LNCS 1386, pp. 360–373.

[23] Van der Schaft, A.J. (2004). Equivalence of dynamical systems by bisimulation. *IEEE Trans. Automatic Control*, Vol. 49, No. 12.

[24] Van Schuppen, J.H. (1998). A sufficient condition for controllability of a class of hybrid systems, *Proc. 1st Int. Workshop Hybrid Systems: Computation & Control (HSCC), Berkeley, CA*, LNCS 1386, pp. 374–383.

[25] Weinberg, H.B., Lynch, N., Delisle, N. (1996). Verification of automated vehicle protection systems, *Hybrid Systems III, Verification & Control*, R. Alur *et al.* (Eds) pp. 101–113, Springer.