Cyber-Physical Ecosystems: Modelling and Verification*

Manuela L. Bujorianu[1][0000−0001−9630−1868]

University College London, Gower Street, WC1E 6EA, UK; l.bujorianu@ucl.ac.uk

Abstract. In this paper, we set up a mathematical framework for the modelling and verification of complex cyber-physical ecosystems. In our setting, cyber-physical ecosystems are cyber-physical systems of systems that are highly connected. These are networked systems that combine cyber-physical systems with an interaction mechanism with other systems and the environment (ecosystem capability). Our contribution will be on two streams: (i) modelling the constituent systems and their interfaces, and (ii) local/global verification of cyber-physical ecosystems. We introduce a concept of basic model, whose skeleton is a Markov decision process and we propose a verification based abstraction methodology.

Keywords: cyber-physical ecosystem · Markov model · reachability · abstraction.

1 Introduction

The Cyber-Physical System (CPS) paradigm was introduced by NSF in 2006 to define a new generation of systems that are built from, and rely upon, the coherent integration of computational algorithms and physical components. It is based on three technologies which are: embedded systems, sensor and actuation, and network and communication systems.

An ecosystem is a complex system, i.e. a group of interrelated things, working together to achieve a common objective. In system engineering, an ecosystem usually consists of components or subsystems, interacting via interfaces, which together satisfy a set of requirements. There exists also an external environment where the given system activates. Examples include the global financial infrastructure of banks and exchanges, transportation networks, cyber-physical systems, IoT networks and semiautomated manufacturing lines, and distributed databases.

Cyber-physical ecosystems (CPES) are ecosystems of networked CPS, meaning that they are systems of CPS (CPSoS) provided with an interaction activity between them and with their environment. Alternatively, we may call them cyber-physical infrastructures (CPI). Examples are smart grid, autonomous vehicles and maritime ships, autonomous swarm drones.

* This work has been funded by the EPSRC project EP/R006865/1: Interface reasoning for interacting systems (IRIS).

The aim of this paper is to set up a modelling framework for CPES based on the distributed system paradigm. The necessary shift to the ecosystem view means that we have to consider not only the construction of models of individual components but also the mechanism that allows them to interact with one another. The interaction mechanism is based on the concept of interface, which, in this paper, is defined in a very general way. From an engineering standpoint, the interfaces can be treated from different perspectives. It is important to note that in the architecture of an ecosystem, an interface may be itself a subsystem with its own interfaces.

In this paper, we consider that components have the possibility to connect with some free interfaces. The interfaces have separate structure and the link with specific components is realized via interface requirements.

We model the CPES components as Markov decision processes that encapsulate at a higher level of abstraction the interaction between physics and computation in the CPS model. The interaction between components will be done by means of specific interfaces, which will be modelled using again decision processes with constraints. This kind of interfaces is flexible enough to enable dynamic interactions and reconfiguration within the underlying CPES.

The novelty of our CPES modelling approach relies on the use of distributed systems paradigm combined with the dynamic behaviour of components described by suitable Markov models. In this setting, the component interaction is enabled by some independent interfaces that play the role of connectors. For this modelling framework, we propose a safety verification methodology based on abstractions.

The paper is structured as follows. In Section 2, we discuss our use of distributed systems as a metaphor for cyber-physical ecosystems of systems and show how we combine it with the dynamic system behaviours. In Section 3, we present the mathematical models for the CPES constituents using some suitable Markov models. In Section 4, we explain how to model the compositional structure of ecosystem models using a rather general notion of interface. Interfaces between models describe how they can be composed together to construct a model of an ecosystem. Conditions that ensure the soundness of the composition operation are also provided. Our notion of interface captures a wide range of the notions of interface that are in the literature on systems and modelling [6]. In Section 5, we define verification of CPES as a stochastic reachability problem. This reachability problem is specialised for models of components and interfaces. In Section 6, we introduce a specific concept of abstraction map that preserves the Markov property. Then this is used to obtain aggregations of models and interface models and to simplify the computation of the reach probabilities. The abstraction process is done in a modular way supporting the local reasoning. Modular reasoning involves breaking down a computational ecosystem into smaller, more manageable components, and to determine local properties of these components that guarantee desired properties of the global system. First, we check the local safety properties and then we combine them in order to deduce the global system safety. This modular safety approach is quite standard in the verification community [7]. We apply the philosophy of this approach in a new setting of CPES modelled as networks of Markov decision processes. The paper ends with some conclusions.

2 CPES - Conceptual Modelling

CPES are modelled as systems of systems, defined through the composition of their CPS constituents. The composition operation is done via specific subsystem interfaces. In this work, we combine the distributed system modelling approach with the behavioural approach for dynamical systems.

In a nutshell, a constituent can be viewed as a tuple

$$C = \langle Loc, X, I, Beh \rangle$$

where *Loc* represents its location, X is a finite set of variables (both computational and physical), $\mathcal{I}$ represents its interface thought of as a set of variables that can be observed by the other systems of the CPES. Finally, *Beh* denotes the set of the system behaviours that are thought as system traces (evolutions of its variables).

The interaction with other constituent systems and the environment may affect these parameters. For example, *Loc* can be modelled as a random graph, or the time evolution of X can be modelled as a stochastic dynamical system where the environment perturbation is captured as a contiguous noise (modelled as white noise in the structure of a stochastic differential equation), or as shot noise (modelled by a Poisson type process).

2.1 The Distributed Systems Metaphor

CPES are thought of as systems of systems, or systems with different (semi)-autonomous constituent systems, which interact, collaborate, inter-operate to achieve common goals. Each system may have a private activity with a specific structure, behaviour, decision mechanism or internal information encapsulated in a specific mathematical model.

We use distributed systems as a metaphor for describing CPES. Our modelling approach for ecosystems is component based, where each constituent system is modelled using a quite general basic model. A basic model is intended to capture the simplest convenient representation of a single constituent system.

There are three key ingredients upon which we draw.

- *Location.* Distributed systems naturally have a concept of distinct locations, which may be connected to one another. In the setting of computer systems, components are present at different locations and connected by a network. In the more general view, locations can be physical (e.g., a room, a container), logical (e.g., an address in computer memory), or abstract (e.g., the location where a semaphore exists).

- *Resource*. Resources exist at locations and can move between them according to the locations' connections. In general, they can represent physical objects, people, information, and more.
- *Process*. Processes manipulate resources — such as consuming, creating, and moving between locations — as they go.

These concepts can be used to build a representation of a system's structure and operation, but there is one more concept required: the environment in which the system operates.

- *Environment* — Environments capture the world outside of the system of interest and how the two interact.

Each basic model encapsulates the above primitives — locations, resources, processes, and environment.

2.2 Dynamic behaviour

Each constituent system is characterized by a specific dynamic behaviour. We can model this behaviour as a deterministic dynamical system or a stochastic process. In [4], the behaviour of CPS subsystems has been modelled as a stochastic hybrid process to encapsulate the physical part and the digital part of a component.

In this paper, for simplicity, we will use an abstraction of this behaviour modelled by a simple Markov chain, viewed as graph whose states are the locations, which have associated some resources. To capture the processes that manage the resources, we add control actions to the Markov chain, transforming it into a Markov decision process. A CPES will be modelled as a network of Markov chains, and its safety verification will be based on some coarse-graining process implemented using specific abstraction morphisms. The continuous dynamics will be abstracted into control action, in the sense that an action could enable a continuous path from a discrete state to another. This technique has been successfully applied for different models of cyber-physical systems such piecewise deterministic Markov processes or stochastic hybrid processes [1]. An interesting CPES example is the water distribution network for which a modelling framework based on Markov decision processes has been developed in [12].

3 Mathematical Modelling of CPES

In this section, we set up a mathematical framework where the basic models and their composition are formally described. Our CPES model builds on some well-known formalisms as Markov chains and interface theory. A CPES, viewed as a system of systems, will be modelled as a composition of Markov chains.

3.1 Basic model

The basic model is a representation of a single constituent system. In this paper, the basic model is defined as a discrete time Markov chain (MC) with a finite state space. The MC states represent the system locations. Resources exist in each state and their manipulation will be modelled using Markov decision processes.

A Markov Chain MC is a directed graph which consists of a set of states S as nodes, and a set of edges defined by a set of probabilistic transitions. An MC can be also specified as a (discrete-time) stochastic process (X_n) with values in S.

The relation between the states of the MC is defined by a set of transitions:

$$T = \{(s, s')|s, s' \in S\}$$

where each transition (s, s') is governed by a transition probability

$$p(s, s') = \mathbf{P}[X_{n+1} = s'|X_n = s].$$

The transition matrix $P = (p(s, s'))$ is a stochastic matrix (that means the sum of each row is equal to one).

Usually, to ensure the uniqueness of an MC, we need to have an initial state s_0 or an initial probability distribution μ_0. Sometimes, a deadlock or a cemetery state s_Δ to encounter for the case when the chain enters in a failure state or is dying (that is s_Δ is an absorbing state). Then, an MC is defined as a tuple:

$$MC = (S, T, s_0, s_\Delta).$$

The important advantage is that the infinitesimal generator has a matrix form, and the probability distributions are probability vectors. For a Markov chain, the generator is one-step increment of the transition matrix

$$\mathcal{L} = P - I \tag{1}$$

Each state of the MC is associated with some resources. We denote the set of resources by $\mathbf{R}$. We may add an algebraic structure to $\mathbf{R}$. The processes (which execute the resource management) will be modelled by using a Markov decision process (MDP). To introduce an MDP we need a set of actions $\mathcal{A}$, where each action $a \in \mathcal{A}$ will represent a resource operation decision or a resource control action. At each time step n, the corresponding decision is denoted by a_n.

Considering the decision process (a_n), the transition probabilities of the MDP is:

$$p^a(s, s') = \mathbf{P}[X_{n+1} = s'|X_n = s, a_n = a].$$

Therefore, we have a transition set T^a associated to each action $a \in \mathcal{A}$. For each $s \in S$, denote by $\mathcal{A}(s)$ the set of all actions $a \in \mathcal{A}$, which enable a transition from s.

For a $Q \subset S$, we use the notation $s \xrightarrow{a}_{\mu} Q$ whenever $a \in \mathcal{A}(s)$ and

$$\sum_{q \in Q} p^a(s,q) = \mu.$$

For an MDP, we may define also a reward function $\rho : S \times \mathcal{A} \to \mathrm{R}$, which specifies the gain and cost of being in a particular state and applying a particular action.

An MDP policy is a set of rules a controller would follow to choose the action to perform in each state. A Markov policy is a family of stochastic kernels $\pi_n : S \to \Delta(\mathcal{A})$, where $\Delta(\mathcal{A})$ is the space of probability distributions on $\mathcal{A}$:

$$\pi(s,a)(n) = \mathbf{P}[A_n = a | X_n = s]. \tag{2}$$

If a policy does not depend on time, it is called stationary. Under each policy, the MDP behaviour is described by an MC.

We define a basic model as follows:

$$\mathcal{M} = (S[\mathbf{R}], \mathcal{A}, (T^a)_{a \in \mathcal{A}}, s_0). \tag{3}$$

We can replace the initial condition s_0 with an initial probability distribution μ_0. A basic model is an MDP that models the resource dynamics. The basic model can be seen, as well, as a probabilistic automaton [11] where all the transitions are Markovian (we do not consider nondeterminism). We treat the basic model as graph with probabilistic transitions.

We can view an MDP or an MC as a dynamical system on the space of probability distributions of S, denoted by $\Delta(S)$. Let us call μ_n the probability distribution at time n; that is

$$\mu_n(s) = \mathbf{P}[X_n = s | X_0 = s_0]. \tag{4}$$

The distribution dynamics of an MC can be described the following *master equation*:

$$\mu_{n+1} = \mu_n P \tag{5}$$

where P is the associated stochastic matrix. Then (5) describes a semi-dynamical system with the initial condition equal to μ_0. For an MDP, to each action $a \in \mathcal{A}$, we have the corresponding dynamics:

$$\mu_{n+1} = \mu_n P^a, \tag{6}$$

where P^a is thought of as a matrix operator acting on the space of probability distributions. For simplicity we use the notation μ_n instead of μ_n^a.

We can adapt the master equation to capture also the resource dynamics:

$$\mu_n(s,R) = \mathbf{P}[X_n = (s,R) | X_0 = (s_0, R_0)].$$

Then the master equation describes a probabilistic modification function of the basic model graph. In the following, we consider the process dynamics contains the resource movement in an implicit way to ease the notation.

3.2 Probabilistic Modal Logic

We consider below a probabilistic modal logic (PML), which is a probabilistic version of the Henessy-Milner logic as defined in [8]

$$\phi := \neg\phi \mid \top \mid \bot \mid \phi \wedge \phi \mid \phi \vee \phi \mid \varDelta_a \mid \langle a \rangle_\mu \phi$$

where $a \in \mathcal{A}$ and $\mu \in [0,1]$. The semantics of PML is given using a probabilistic labelled transition system, which is an MDP in our setting. The satisfaction relation between states and formulas $s \models \phi$ is defined as usual for $\neg\phi$, $\top$, $\bot$, $\phi \wedge \phi$ and $\phi \vee \phi$. $s \models \varDelta_a$ holds whenever $a \notin \mathcal{A}(s)$.

The satisfaction of $s \models \langle a \rangle_\mu \phi$ holds when for some $Q \subset S$ we have $s \xrightarrow{a}_\nu Q$ for a $\nu \geq \mu$ and $q \models \phi$ for all $q \in Q$. Then

$$[a]\phi \equiv \langle a \rangle_1 \phi$$

A concept of probabilistic bisimulation can be defined on the state space S of an MDP which is characterized by the above logic (see [8] for definitions and characterizations).

4 Interfaces and Composition

To start thinking about system interaction, a concept that captures how these interactions happen is required. In this section, we introduce the notion of basic interface, and we equip our basic model with this concept.

4.1 Basic Interface

We define a basic interface for a basic model $\mathcal{M}$ as:

$$I = \langle S_I[\mathbf{R}], \mathcal{A}_I \rangle \tag{7}$$

where $\mathcal{A}_I$ is a set of actions specified by a transition function T_I. The transition function could be deterministic or stochastic. Such an interface will be connected with a basic model, and then we will define a model for a CPES subsystem.

With the above concept of interface, we define the model of a constituent system in the CPES architecture as $M = \langle \mathcal{M}, I \rangle$, where $S_I \subset S$, $\mathcal{A}_I \subset \mathcal{A}$ is a subset of observable actions and T_I coincides with T on S_I for all $a \in \mathcal{A}_I$.

For the soundness of the underlying mathematical model, we enforce the following assumption.

Assumption 1 *The state space S is partitioned into the union of a transient set and an absorbing set:*

- *$S \setminus S_I$ is a transient set for the underlying MC (that means the probability to leave this set is strictly positive);*
- *S_I contains an absorbing set for the underlying MC (it may contain also transient states).*

Moreover, the initial condition s_0 will belong to $S \setminus S_I$, otherwise the model will evolve only in S_I.

4.2 Composition of Models

The composition operation joins two models using a specified basic interface from each one. Let $M_1 = \langle \mathcal{M}_1, I_1 \rangle$ and $M_2 = \langle \mathcal{M}_2, I_2 \rangle$ be two models.

Assumption 2 *Suppose that $S_{I_1} \cap S_{I_2} \neq \emptyset$, $\mathcal{A}_{I_1} \cap \mathcal{A}_{I_2} \neq \emptyset$ and $(S_1 \setminus S_{I_1}) \cap S_2 = \emptyset$, $(S_2 \setminus S_{I_2}) \cap S_1 = \emptyset$; $(\mathcal{A}_1 \setminus \mathcal{A}_{I_1}) \cap \mathcal{A}_2 = \emptyset$, $(\mathcal{A}_2 \setminus \mathcal{A}_{I_2}) \cap \mathcal{A}_1 = \emptyset$.*

The two models need to match their transition structure on the intersection of their basic interfaces. So, the following assumption is necessary:

Assumption 3 *Let $S_{I_{12}} = S_{I_1} \cap S_{I_2}$. For any $a \in \mathcal{A}_{I_1} \cap \mathcal{A}_{I_2}$, we have:*

$$p_1^a(s, s') = p_2^a(s, s'), \ \forall s, s' \in S_{I_{12}}.$$

The composition $M_1 \circ_{I_1, I_2} M_2$ is defined as follows:

$$M = M_1 \circ_{I_1, I_2} M_2 = \langle S[\mathbf{R}], T, \mathcal{A}, s_0, I \rangle$$

where: $S = S_1 \times S_2$; $\mathbf{R} = \mathbf{R}_1 \otimes \mathbf{R}_2$; $T = T_1 \otimes T_2$; $\mathcal{A} = \mathcal{A}_1 \cup \mathcal{A}_2$; $s_0 = (s_{01}, s_{02})$, $I = I_1 \otimes I_2$.

The resource allocation for a state $s = (s_1, s_2)$ in the composed model M is defined as the union of component resources: $s(R) = s_1(R) \cup s_2(R)$.

The transition function T of model M is defined as follows:

- If $a \in \mathcal{A}_1 \setminus \mathcal{A}_2$, and $s = (s_1, s_2) \in S$ then $a \in \mathcal{A}(s)$ if and only if $a \in \mathcal{A}(s_1)$ with
$$p^a(s, s') = p_1^a(s_1, s_1') \delta_{s_2}(s_2'), \ \forall s' = (s_1', s_2') \in S.$$
- If $a \in \mathcal{A}_2 \setminus \mathcal{A}_1$, and $s = (s_1, s_2) \in S$ then $a \in \mathcal{A}(s)$ if and only if $a \in \mathcal{A}(s_2)$ with
$$p^a(s, s') = p_2^a(s_2, s_2') \delta_{s_1}(s_1'), \ \forall s' = (s_1', s_2') \in S.$$

The interface I of the model M has the following items:

- $S_I = S_{I_1} \times S_{I_2}$.

– The transition function T_I is defined similarly as T, the only difference is that we encounter an extra case:
  If $a \in \mathcal{A}_{I_1} \cap \mathcal{A}_{I_2}$, and $s = (s_1, s_2) \in S_{I_{12}} \times S_{I_{12}}$ then $a \in \mathcal{A}_I(s)$ if and only if $a \in \mathcal{A}_{I_1}(s_1) \cap \mathcal{A}_{I_2}(s_2)$ with

$$p^a(s, s') = p_1^a(s_1, s_1')p_2^a(s_2, s_2'), \forall s' = (s_1', s_2') \in S_I.$$

The synchronization of the two models is realized only on the overlapping region $S_{I_{12}}$. Outside of this region, the composed model inherits the structure (locations, resources, processes) of its components. The model composition is similar with the MDP composition, taking into account the interface separation. The following result is easy to establish.

Proposition 1 (Composition Soundness) *If M_1 and M_2 are models as above, then so is $M_1 \circ_{I_1,I_2} M_2$.*

Standard properties of composition as commutativity and associativity are straightforward.

When two models are composed using their basic interfaces it follows, by construction, that their basic structures fit together. But this is not always the case. The main observation is that for model, whenever it is composed with another model, it is necessary to specify which of the properties or functionalities must be preserved.

In this paper, we define a more general concept of interface which is defined as a model itself together with some constraints. These constraints may concern the states, the resources, the rewards or the cost functions associated to MDPs. To be more general, suppose we have given an appropriate logic for MDPs (for example, probabilistic modal logic defined in subsection 3.2).

4.3 Interface model

An interface model is a pair $I = \langle M, \psi \rangle$ where M is a model, and ψ is a set of formulae that describe properties of the model that must be preserved under composition. We refer to these formulae as interface formulae.

The concept of an interface model strictly generalizes that of a model, as we can take the interface model as $\langle M, \{\top\} \rangle$, which has no constraint on M.

Our concept of interface model can be thought of as a connector with a model structure (locations, resources, processes and basic interfaces). The interface requirements are specified as logical formulas.

4.4 Admissibility

For the interface composition, some extra conditions are required to ensure the soundness of this operation.

Let $\langle M_1, \psi_1 \rangle$ and $\langle M_2, \psi_2 \rangle$ be interface models such that M_1 and M_2 are built on MDPs as in the previous section. Let $M_1 \circ M_2$ denote a composition of

M_1 and M_2 using some choice of interfaces that satisfy the Assumption 2. Then the composition of $\langle M_1, \psi_1 \rangle$ and $\langle M_2, \psi_2 \rangle$, denoted

$$\langle M_1, \psi_1 \rangle \circ \langle M_2, \psi_2 \rangle := \langle M_1 \circ M_2, \psi_1 \wedge \psi_2 \rangle$$

is admissible if $\psi_1 \wedge \psi_2 \not\supset \perp$.

Proposition 2 (Composition Soundness) *If $\langle M_1, \psi_1 \rangle$ and $\langle M_2, \psi_2 \rangle$ are interface models, then so is their composition $\langle M_1, \psi_1 \rangle \circ \langle M_2, \psi_2 \rangle$.*

Proposition 3 (Commutativity and Associativity) *Commutativity of composition of interface model follows as for models. Associativity of composition of interface models $\langle M_1, \psi_1 \rangle$, $\langle M_2, \psi_2 \rangle$, and $\langle M_3, \psi_3 \rangle$ holds as for models provided also $\psi_1 \wedge \psi_2 \wedge \psi_3 \not\supset \perp$.*

In our setting, an ecosystem is modelled as a composition of interface models.

5 Verification

Verification of cyber-physical systems is a difficult, yet extremely important, problem. In this paper, we formulate the CPES verification as a reachability problem. Reachability analysis is a fundamental problem in verification that checks for a specific model and a set of initial states if the system will reach a specified set of unsafe states. Complementary, reachability analysis can check if the system will achieve its objective, that is if the system will reach a set of target states. For CPS, the reachability problem is challenging when we consider hybrid models that combine discrete transitions alternating with continuous dynamics. In this work, we abstract away the continuous behaviour of CPES, but the main difficulty is arising from the distributed nature of CPES.

In modular verification of distributed systems, the component verification is specified and solved independently (locally) for each module. Then the entire system verification is defined as a global property, whose solution is obtained as the composition of local solutions rather than using the global implementation of the system.

5.1 Reachability Problem for a Basic Model

Suppose that we have given a basic model $\mathcal{M}$ as described by (3).

Here, we define the state-constrained reachability, called sometimes reach avoidance problem. Let $U \subset S$ be an unsafe set, and $E \subset S$ be a target (or objective) set. Then the *reach avoidance problem* aims to compute the probability to reach the unsafe set U, before hitting the target E.

Formally, for the underlying MC, we have to compute the reach probability function, i.e.

$$q_{\mathcal{M}}(s) = q_{\mathcal{M}}(s, U, E) = \mathbf{P}\{\tau_U < \tau_E | X_0 = s\} \tag{8}$$

where we use the notation $\tau_Q = \min\{k > 0 | X_k \in Q\}$ for the first hitting time of a set $Q \subset S$. Then $q_{\mathcal{M}}$ is the solution of the following Dirichlet problem:

$$(\mathcal{L}q)(i) = 0, \forall i \in S \setminus U,$$

$$q(j) = 1, \forall j \in U,$$
$$q(l) = 0 \text{ if } l \in E,$$

which is a system of linear equations.

When the transition probabilities are triggered by the action $a \in \mathcal{A}$, we will use the notation $q^a_{\mathcal{M}}(s)$.

For an MDP, the reach probability can be computed for any policy π. Usually, the stochastic safety aims to compute the optimal policies for which the reach probabilities are bounded by an admissible threshold $p \in [0, 1]$. For the analysis and computational methods that characterize the reach avoidance problem for MDPs, we refer to [5, 13].

5.2 Reachability Problem for a Model

We adapt the reach probability function for a model that is equipped with a basic interface. In this case, we take $U \subset (S \setminus S_I)$ and $E \subset S$. Then we define:

$$q_M(s) = q_M(s, U, E) = q_{\mathcal{M}}(s, U, E \cup S_I) \tag{9}$$

i.e.

$$q_M(s) = \mathbf{P}\{\tau_{(U)} < \tau_{E \cup S_I} | X_0 = s\}.$$

In this case, our objective is to compute the probability to reach either the unsafe set U before reaching the target E, or the basic interface space S_I. Then q_M is the solution of the following Dirichlet problem:

$$(\mathcal{L}q)(i) = 0, \forall i \in S \setminus (U \cup S_I); q(j) = 1, \forall j \in U; q(l) = 0, \forall l \in E \cup S_I. \tag{10}$$

We assume that the basic interface is a safe region for the model. The problem of verification concerns only the unsafe states which are transient. As in the case of basic model, when the transition probabilities are controlled by the action $a \in \mathcal{A}$, we will use the notation $q^a_M(s)$.

5.3 Reachability for an Interface Model

Suppose now we have given an interface model $\langle M, \psi \rangle$. In order to verify safety of such a model, we need to check two conditions: (1) the logical constraints ψ are satisfied, and (2) the stochastic safety condition $q_M < p$, where $p \in [0, 1]$ is an admissible probability threshold. If the logical constraints ψ regard only the basic interface space S_I, the two conditions can be checked seprately.

5.4 Reachability for Model Composition

Let $M_1 = \langle \mathcal{M}_1, I_1 \rangle$ and $M_2 = \langle \mathcal{M}_2, I_2 \rangle$ be two models that satisfy the Assumptions 2, 3 for composition. Let $M = \langle \mathcal{M}, I \rangle$ be their composition.

Let

$$E = E_1 \times E_2 \subset S_1 \times S_2$$

be a target set, and

$$U = U_1 \times U_2 \subset (S_1 \setminus S_{I_1}) \times (S_2 \setminus S_{I_2})$$

be an unsafe set for M.

It is important to remark that the model composition does not change the behaviour on $S_1 \setminus S_{I_1}$ or $S_2 \setminus S_{I_2}$ of its constituents. Then the next result can be easily checked.

Proposition 4 *The reach probability function of M w.r.t. U and E is equal to the component reach functions as follows:*

– *If $a \in \mathcal{A}_1 \setminus \mathcal{A}_2$ then*

$$q^a_M(s, U, E) = q^a_{M_1}(s_1, U_1, E_1),$$

for all $s = (s_1, s_2) \in (S_1 \setminus S_{I_1}) \times (S_2 \setminus S_{I_2})$.

– *If $a \in \mathcal{A}_2 \setminus \mathcal{A}_1$ then*

$$q^a_M(s, U, E) = q^a_{M_2}(s_2, U_2, E_2),$$

for all $s = (s_1, s_2) \in (S_1 \setminus S_{I_1}) \times (S_2 \setminus S_{I_2})$.

The above proposition states that the computation of the reach probability for the composed model is done in a modular way, for each component. The reason is that outside of the interfaces, a control action a modifies only one component when it is enforced.

6 Abstractions

For MDPs, according to [9], there exist five types of abstraction functions. Here, we use the state abstraction function.

6.1 Abstraction of a Basic Model

Let $\mathcal{M}$ be a basic model defined by (3).

Formally, an *abstraction function* is defined as a surjective map $\varphi : S \to \overline{S}$, which maps the underlying MC into another MC. Then we define a matrix Φ of dimension $|S| \times |\overline{S}|$ by:

$$\Phi(s, \overline{s}') = \delta_{\overline{s}'}(\varphi(s)) = \mathbf{1}_{\varphi^{-1}(\overline{s}')}(s).$$

A sufficient condition for φ to be a Markovian abstraction function is the existence of a stochastic kernel $\Lambda : \overline{S} \to \Delta(S)$ such that supp$\Lambda(\overline{s}, \cdot) = \varphi^{-1}(\overline{s})$ for all $\overline{s}$ in $\overline{S}$, i.e.,

$$\Lambda(\overline{s}, \varphi^{-1}(\overline{s})) = \sum_{y \in \varphi^{-1}(\overline{s})} \Lambda(\overline{s}, s) = 1, \forall \overline{s} \in \overline{S}.$$

This assumption implies that:

$$\Lambda\Phi = \mathbf{I}_{|\overline{S}|} \tag{11}$$

where $\mathbf{I}_{|\overline{S}|}$ is the identity matrix of order $|\overline{S}|$. Note this condition is more general than the one given in [9]. The condition is inspired by the seminal paper on Markov functions of Rogers and Pitman [10].

The kernel Λ will be called *concretization kernel*. The reason is that it maps the abstract model into the concrete one. The following relationship holds:

$$\mathbf{P}\{X_n = s | \varphi(X_m), \, 0 \leq m \leq n\} = \Lambda(\varphi(X_n), s), \, \forall s \in S. \tag{12}$$

This states a very prominent thing that the estimator of the state X_n that predicts the process from the abstractions is the same as the abstracted state. We denote by $\overline{X}_n = \varphi(X_n)$ the abstraction process, which is still Markov. The relationship between the infinitesimal generator of the abstraction process and the concrete one is as follows:

$$\overline{\mathcal{L}} = \Lambda \mathcal{L} \Phi.$$

We define an equivalence relation on S by:

$$s \sim s' \iff \varphi(s) = \varphi(s').$$

Let $[s]$ be the equivalence relation of s w.r.t. $\sim$, and $S/_{\sim}$ be the quotient space. A subset F of S is closed w.r.t. $\sim$ if whenever $s \in F$ then $s' \in F$ for all $s' \in [s]$.

In fact the abstract basic model is thought of as:

$$\overline{\mathcal{M}} = (\overline{S}[\mathbf{R}], \mathcal{A}, (\overline{T}^a)_{a \in \mathcal{A}}, \overline{s}_0), \tag{13}$$

where the underlying Markov chain is the quotient process. In fact, the equivalence relation $\sim$ is thought on the hybrid space $(S, \mathbf{R})$. Due to the limited room of this paper, we keep the implicit notation.

Let $U \subset S$ and $E \subset S$ be two closed subsets (w.r.t. $\sim$) of S. Denote $\overline{U} = \varphi(U)$ and $\overline{E} = \varphi(E)$. Then $\varphi^{-1}(\overline{U}) = U$ and $\varphi^{-1}(\overline{E}) = E$. The reach probability function $\overline{q}_{\overline{\mathcal{M}}}$ for the abstraction process $\overline{X}_n$ the unsafe set $\overline{U}$ and target set $\overline{E}$ will be the solution of the Dirichlet problem associated to the generator $\overline{\mathcal{L}}$. The following result is straightforward:

Proposition 5 *The reach probability function $q_{\mathcal{M}}$ of the concrete basic model $\mathcal{M}$ w.r.t. the target set E and the unsafe set U is related with the reach probability function $\overline{q}_{\overline{\mathcal{M}}}$ of the abstraction of the basic model $\overline{\mathcal{M}}$ w.r.t. the target set $\overline{E}$ and* *the unsafe set $\overline{U}$ by the following relation:*

$$q_{\mathcal{M}} = \overline{q}_{\overline{\mathcal{M}}}\Lambda.$$

6.2 Abstraction of a Model

Let $M = \langle \mathcal{M}, I \rangle$ be a model defined as before. An abstraction function for M is defined as an abstraction function for the underlying basic model $\mathcal{M}$, which satisfies the following condition:

$$\varphi^{-1}(\overline{S}_I) = S_I, \tag{14}$$

where $\overline{S}_I = \varphi(S_I)$. Then, the interface space 'invariance' will lead to the following relations that connect the abstraction function and the concretization kernel:

$$\Lambda|_{\overline{S}\setminus\overline{S}_I}\Phi|_{S\setminus S_I} = \mathbf{I}_{|\overline{S}\setminus\overline{S}_I|}, \; \Lambda|_{\overline{S}_I}\Phi|_{S_I} = \mathbf{I}_{|\overline{S}_I|}.$$

For a model M, the reach probability function q_M w.r.t. U and E is a specialization of the reach probability function $q_{\mathcal{M}}$ of basic model $\mathcal{M}$ w.r.t. U and $E \cup S_I$. Then:

$$q_M = \overline{q}_{\overline{M}}\Lambda. \tag{15}$$

6.3 Abstraction of an Interface Model

Let $I = \langle M, \psi \rangle$ be an interface model. An abstraction function $\varphi : S \to \overline{S}$ of the model M is an abstraction function for the interface model I if it is compatible with the constraints ψ. This will be explained below.

We impose the following compatibility assumption between $\sim$ and the constraints ψ:

Assumption 4 *If there exists $s' \in [s]$ such that $s' \models \psi$ then $s'' \models \psi$ for all $s'' \in [s]$.*

Therefore, the equivalence relation $\sim$ is consistent with the interface constraints. The computation of the reach probability function remains the same, the main difficulty in this case is to find an abstraction map that preserves the interface requirements.

6.4 Abstraction Composition

Suppose that we have given two interface models $\langle M_i, \psi_i \rangle$ with $i = 1, 2$ such that the Assumptions 2, 3 hold and the admissibility condition is satisfied.

Let φ_i, $i = 1, 2$ be associated abstractions functions, with their corresponding concretization kernels Λ_i, $i = 1, 2$ such that the Assumption 4 holds for both of them.

We have the underlying models defined as $M_i = \langle \mathcal{M}_i, I_i \rangle$, $i = 1, 2$. To make the composition of their abstractions we need a further assumption as follows.

Assumption 5 *The abstraction functions coincide on the basic interface overlapping $S_{I_{12}} = S_{I_1} \cap S_{I_2}$, i.e.*

$$\varphi_1(s) = \varphi_2(s),\ \forall s \in S_{I_{12}}$$

and $S_{I_{12}}$ is 'invariant' w.r.t. both abstraction maps:

$$\varphi_1^{-1}(\varphi_1(S_{I_{12}})) = \varphi_2^{-1}(\varphi_2(S_{I_{12}})) = S_{I_{12}}$$

Then $\varphi = (\varphi_1, \varphi_2)$ will play the role of an abstraction map for the composed model M.

We denote by $\overline{M}$ the composition of the abstract interface models $\overline{M}_1$ and $\overline{M}_2$.

The reach probability function q_M corresponding to the unsafe set $U = U_1 \times U_2$ and the target set $E = E_1 \times E_2$ will be the superposition of the component reach probabilities:

$$q_M = (q_{M_1}, q_{M_2}).$$

The reach probability function $\overline{q}_{\overline{M}}$ corresponding to the unsafe set $\overline{U} = \overline{U}_1 \times \overline{U}_2$ and the target set $\overline{E} = \overline{E}_1 \times \overline{E}_2$ will be the superposition of the component reach probabilities:

$$\overline{q}_M = (\overline{q}_{\overline{M}_1}, \overline{q}_{\overline{M}_2}).$$

We introduce the notation:

$$\Lambda = (\Lambda_1, \Lambda_2)$$

where Λ_1 and Λ_2 are the concretization kernels associated to the component models M_1 and M_2. The following result holds:

Proposition 6 *The reach probability function q_M of the concrete model M and the reach probability $\overline{q}_{\overline{M}}$ of the abstraction model $\overline{M}$ are related as follows:*

$$q_M = \overline{q}_{\overline{M}} \circ \Lambda$$

where $\overline{q}_{\overline{M}} \circ \Lambda$ is the Hadamard product of matrices, i.e.

$$\overline{q}_{\overline{M}} \circ \Lambda = (\overline{q}_{\overline{M}_1} \Lambda_1, \overline{q}_{\overline{M}_2} \Lambda_2).$$

The main remark here is that the verification process is carried out in a modular way.

7 Conclusions

In this paper, we have presented an approach to modelling and verification of cyber-physical ecosystems based on concepts from distributed systems and Markov decision processes. We have developed the notion of a basic model based on the MDP skeleton, expanded it with basic interfaces to define a model. Then we have characterised an interface model as a model which has associated a set of formulae that characterize the composition requirements. We have defined the verification of CPES as a stochastic reach avoidance problem for all constituent models. Then, we have explored how to use modular abstractions to find simple computational solutions for the stochastic safety of CPES.

This paper provides the theoretical setting for CPES verification, when the system constituents are modelled as MDPs and they interact through some general interfaces. A case study of a cyber-physical ecosystem as a system of supply chains is under development. New results will be reported in a follow-up paper.

References

1. Nicole Bäuerle, Ulrich Rieder. Piecewise Deterministic Markov Decision Processes. In: Markov Decision Processes with Applications to Finance. Universitext. Springer, Berlin, Heidelberg (2011).
2. Manuela L. Bujorianu, Tristan Caulfield, Marius C. Ilau, David S. Pym. Interfaces in Ecosystems: Concepts, Form, and Implementation. Submitted to TARK.
3. Manuela L. Bujorianu. Stochastic Reachability Analysis of Hybrid Systems. In: Stochastic Reachability Analysis of Hybrid Systems. Communications and Control Engineering. Springer, London (2012).
4. Manuela L. Bujorianu, Tristan Caulfield, David Pym, Modelling and Control of Complex Cyber-Physical Ecosystems, IFAC-PapersOnLine, Volume 55, Issue 40, (2022): 253-258.
5. Manuela L. Bujorianu, Rafael Wisniewski, Evangelos Boulougouris. Stochastic Safety for Markov Chains. IEEE Control Systems Letters, **5**(2) (2021): 427-432.
6. Tristan Caulfield, Marius C. Ilau, David Pym. Engineering Ecosystem Models: Semantics and Pragmatics. In: Jiang, D., Song, H. (eds) Simulation Tools and Techniques. SIMUtools 2021. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol 424. Springer, Cham (2022).
7. Orna Kupferman, Moshe Y. Vardi. Modular Model Checking. In: de Roever, WP., Langmaack, H., Pnueli, A. (eds) Compositionality: The Significant Difference. COMPOS 1997. Lecture Notes in Computer Science, vol 1536. Springer, Berlin, Heidelberg (1998).
8. Kim G. Larsen, Arne Skou: Bisimulation through probabilistic testing. Information and Computation 94(1), (1991): 1-28.
9. Lihong Li, Thomas J. Walsh, Michael L. Littman, Michael. Towards a Unified Theory of State Abstraction for MDPs. Proceedings of the Ninth International Symposium on Artificial Intelligence and Mathematics (2006).
10. L. C. G. Rogers, J. W. Pitman: Markov Functions. The Annals of Probability, Ann. Probab. 9(4), (1981): 573-582.
11. Mariëlle Stoelinga. An Introduction to Probabilistic Automata. Bulletin of the EATCS. 78 (2004).
12. Rahul Misra, Rafał Wisniewski, Carsten S. Kallesøe. Approximating the model of a water distribution network as a Markov decision process, IFAC-PapersOnLine, Volume 55, Issue 20, (2022): 271-276.
13. Rafael Wisniewski and Manuela L. Bujorianu. Probabilistic Safety Guarantees for Markov Decision Processes. in IEEE Transactions on Automatic Control, doi: 10.1109/TAC.2023.3291952.