POWVER

Technical Report 2017-13

Title: **Models of Connected Things: On Priced Probabilistic Timed Reo**

Authors: Kangli He, Holger Hermanns, Yixiang Chen

Report Number: 2017-13

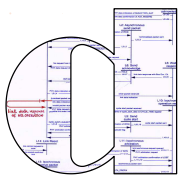
ERC Project: Power to the People. Verified.

ERC Project ID: 695614

Funded Under: H2020-EU.1.1. – EXCELLENT SCIENCE

Host Institution: Universität des Saarlandes, Dependable Systems and Software
Saarland Informatics Campus

Published In: COMPSAC 2017



POWER TO THE PEOPLE.
VERIFIED.
erc

Models of Connected *Things*: On Priced Probabilistic Timed Reo

Kangli He*, Holger Hermanns† and Yixiang Chen*
*School of Computer Science and Software Engineering, East China Normal University, Shanghai, China
Email: klhe@ecnu.cn, yxchen@sei.ecnu.edu.cn
†Dependable Systems & Software, Saarland University, Saarbücken, Germany
Email: hermanns@cs.uni-saarland.de

***Abstract*—The Internet of Things (IoT) is announced to swamp the world. In order to understand the emergent behaviour of connected *things*, effective support for the modelling of connection and failure probabilities, execution and waiting times, as well as resource consumptions of various kinds is needed. At the heart of IoT are flexible and adaptive communication and interaction patterns between *things*, meant to enable advanced as well as radically new emerging functionalities. Since these interaction patterns are determined by topological characteristics, they can naturally be modelled by channel-based exogenous coordination primitives.**

In this paper, we tackle the IoT modelling challenge. Our modelling approach is based on a conservative extension of Reo circuits. On a technical level, we work with a model called *Priced Probabilistic Timed Constraint Automaton*, which combines existing models of probabilistic and timed aspects, and is equipped with pricing information. The latter enables us to reason about resource consumption, especially important in light of severely limited power, memory and computation budgets in *things*. The approach is set up in such a way that the original constituent models can be retrieved without changes in syntax and semantics. A small but illustrative IoT case is modelled and evaluated, demonstrating the principal benefits of the proposed approach.

***Keywords*-IoT; Reo; cost; time; probability; modelling; automata**

I Introduction

As one of the most prominent emerging techniques, the Internet of Things (IoT) is expected to change the world as we know it. The IoT connects not only traditional end devices in the internet, but also more general physical *things*, such as a robot that can sense the outer world and give an independent interaction. Any system composed of such *things* and the associated coordination and communication infrastructure such as the internet, real-time systems or local networks can be considered as fractions of IoT. Despite their diverse, often intricate and topologically varying phenotypes, we find the need for flexible and adaptive communication and interaction patterns between *things* at the heart of IoT. These are meant to enable advanced as well as radically new emerging functionalities and components. To ease their development, effective support for the modelling of IoT systems is forthwith needed.

Furthermore, the diversity on research areas and professional abilities of stakeholders (which for instance consist of researchers, project leaders, company managers etc.) in IoT systems makes an important motivation for a lightweight formal notation that, on the one hand, provides an easy interface for design engineers (which can be any stakeholder) and, on the other hand, supports the description of quantitative IoT system aspects.

The coordination language Reo [1] offers a powerful glue language for implementation of coordination mechanisms via connectors, based on a calculus of mobile channels. This naturally yields a friendly and easy-understanding user interface. IoT systems consist of distributed *components* (i.e., physical things) coordinated with each other under different distributed *communication mechanisms*. One of the most important aspects in IoT is the data-acquisition-driven interaction among such components which can naturally be viewed as exogenous interactions. In this paper, we focus on that exogenous interactions in IoT systems, leaving the interesting interior mechanisms of each component for later research. The Reo language appears as a convenient lightweight formalism, where *components* are represented as *nodes* and *communication mechanisms* give rise to *channels or connectors*.

When exploring this approach, it soon becomes apparent that extensions to the original Reo syntax and underlying semantics are needed to better fit relevant aspects of the IoT domain. In IoT systems, resource consumption is pivotal for mission accomplishment and self-sustainability. This asks for matching support in the language to specify such aspects, in the form of a notation for reward or cost decorations of the model, together with appropriate extensions of the semantic model. This extension enables analyses of several other features including, for instance, the amount of tasks finished within a given duration. The present paper develops such an extension, and places it into the system context of IoT.

On the semantic level, Reo has a basic semantics defined in terms of so-called Constraint Automata (CA) [2] with several variants [3]. It is understood how to include nondeterministic aspects in CA, as well as timed aspects – forming Timed Constraint Automata (TCA) [4] – and probabilistic aspects – forming so-called Probabilistic Constraint Automata (PCA) [5]. Since these aspects exist naturally and are of crucial relevance for interesting properties in the IoT domain, our approach is based on these extensions.

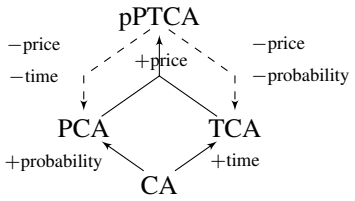


Figure 1: Pedigree for pPTCA.

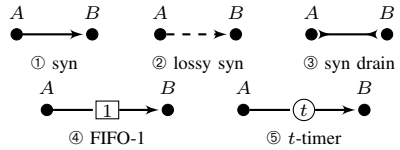


Figure 2: Some basic Reo channels.

Our contribution: In order to enable a faithful modelling of IoT systems, we propose a model called *Priced Probabilistic Timed Constraint Automata* (pPTCA). pPTCA combines features to represent nondeterministic, probabilistic, and timed aspects with aspects of energy consumption (or any kind of resource consumption considered important by the user). The model is constructed on the basis of PCA and TCA, and it combines these two models together with dedicated support for cost aspects. The extension is strictly conservative in the sense that we can retrieve the original base models (PCA and TCA) and their semantics by restricting to the respective fragment of pPTCA. We can, for instance, do this for some components not involving probability and cost, and can analyse timing features of these models through theories and tools [6] for the TCA fragment of pPTCA. These analyses can then be used within the more general setting of pPTCA. The relationship between our model and the base models (i.e., CA, TCA and PCA) is shown in Figure 1.

Organisation of the paper: We first present a primer of Reo circuits in Section II. Section III reviews the semantic model of constraint automata together with the variants adding timed and probabilistic aspects to it. Section IV presents the details of Reo with priced, probabilistic and timed features and its semantic constraint automata model. A case study of an IoT scenario is presented in Section V, demonstrating the expressiveness and principal benefits of our model in such domains. Section VI presents the related work. Section VII concludes this paper.

II A PRIMER OF REO

Reo [1] is a channel-based exogenous coordination model where complex coordinators for component instances, called *connectors*, are compositionally built out of simpler ones. Reo is entirely exogenous in that only the communication and coordination among components are taken into account without considering the inner activities or communications of each component. The basic connectors are a set of *channels* with well-defined behaviour. Each channel has two *channel ends*, which can be seen as ports through which data items can enter or leave the channel. The channel ends are classified as *source* ends, providing input data items into the channel through `write` operations, and *sink* ends, taking data items from the channel by `read` operations. Reo generalizes this channel notion by allowing arbitrary channel ends according to different channel types with user-defined semantics, as shown partly in Figure 2. Synchronous channels require that `write` operations at source ends synchronize with matching `read` operations at sink ends, whilst asynchronous channels (e.g., FIFO-1, t-timer) do not. A classical channel type is a *synchronous channel* (i.e., Figure 2.①), which has a source end and a sink end. The `write` operations at its source end and the matching `read` operations at its sink end are restricted to succeed only simultaneously. Other two variants of synchronous channels are introduced. A *lossy synchronous channel* (i.e., Figure 2.②) allows `write` operations on the source end are always enabled. If no matching `read` operations on the sink node, then the input data items are lost. Otherwise, it behaves like a standard synchronous channel. A *synchronous drain* (i.e., Figure 2.③) has two source ends, so no `read` operations can be taken to obtain the input data items. The `write` operations on two source ends are required to be synchronized, and both written values are lost. FIFO-1 *channel* (i.e., Figure 2.④) is a typical type of asynchronous channel with one buffer cell. It has a source end and a sink end, and the box in the middle stands for the buffer cell. The buffer can be initiated empty or with a data item defined by users. The `write` operations on source end can only succeed if the buffer is empty, while the read operations on sink end can only fire when some data item is stored in the buffer. A t-timer (i.e., Figure 2.⑤) is triggered by some `write` operation on the source end, and outputs time-out after exact t time units.

A complex connector is built out of basic channels of arbitrary types by so-called *topological operations*, namely `join` and `hide`. The result is a graphical representation, called a Reo *circuit*. The compositional framework provides features of composability and dynamic reconfigurability in Reo. The nodes of a Reo circuit are considered as pairwise disjoint and non-empty sets of channel ends. The edges represent the connecting channels. For a node A, $Src(A)$ denotes the set of source ends coinciding on A, and $Snk(A)$ denotes the set of coincident sink ends. A node A in Reo circuit is classified as a *source node* (where $Src(A) \neq \emptyset$ and $Snk(A) = \emptyset$), a *sink node* (where $Src(A) = \emptyset$ and $Snk(A) \neq \emptyset$), or a *mixed node* (where $Src(A) \neq \emptyset$ and $Snk(A) \neq \emptyset$). A `write` operation on one node succeeds only if *all* the coincident source ends accept the data item. A `read` operation on one node succeeds if and only if at

least *one* of the coincident sink ends offers suitable data items, and only one is selected nondeterministically. The `join` operation on two nodes with the same name creates a new node (with the same name) and combines all channel ends coincident on original ones. We use `hide` operations to encapsulate mixed nodes inside a circuit, making them invisible and unaccessible to the environment.

Since flexible and adaptive communication and interaction patterns in IoT are determined by topological characteristics, they can naturally be modelled by channel-based Reo circuits.

III Constraint Automata

The semantics of Reo can be defined in terms of relations on *timed data streams* (TDSs) [7]. In [2], Baier et al. introduce *Constraint Automata* (CA) as an operational semantics for describing the behaviour of Reo circuits. They relate the languages of these automata with TDSs, where CA accept TDS-tuples rather than strings, as for ordinary automata. CA are variants of labelled transition systems where transitions are labelled with pairs $\langle N, g\rangle$ instead of action labels, where $N \subseteq \mathcal{N}$ stands for a finite set of nodes and g is a boolean condition on the observed data items. The nodes play the role of input and output ports of components and connectors to model several channels gluing together. The locations of a CA refer to the network configurations, and transitions out of a location represent the possible data-flow according to the current configuration and the effect on it.

Notation 1 (Data assignments and data constraints)**.** In the sequel, we assume a finite and non-empty set $Data$ consisting of data items that can be transmitted through channels, and a finite set of nodes $\mathcal{N}$. A *data assignment* (DA) denotes a function $\delta : N \to Data$ where $\emptyset \neq N \subseteq \mathcal{N}$. We use $\delta.A \in Data$ to denote the data item assigned to every node $A \in N$ under δ and $DA(N)$ for the set of all data assignments of node set N. If $M \subseteq N \subseteq \mathcal{N}$ and $\delta \in DA(N)$ then $\delta|_M$ stands for the data assignment for M that assigns data item $\delta.A$ to each $A \in M$, and $\delta|_\emptyset = \emptyset$. Given $\delta_1 \in DA(N_1)$ and $\delta_2 \in DA(N_2)$, if $\delta_1.A = \delta_2.A$ for all $A \in N_1 \cap N_2$, then $\delta_1 \uplus \delta_2$ stands for the data assignment for $N_1 \cup N_2$ that assigns data item $\delta_1.B_1$ to each $B_1 \in N_1$ and data item $\delta_2.B_2$ to each $B_2 \in N_2$.

Data constraints can be viewed as a symbolic representation of data assignments. Formally, *data constraints* (DC) are propositional formulas built from the atoms $d_A = d_B$, $d_A = d$, and $d_A \in D$ (plus the standard boolean connectors $\wedge, \vee, \neg$, etc.) where $A, B \in \mathcal{N}$, d_A and d_B are symbols for the observed data item at node A and B respectively, $d \in Data$ and $D \subseteq Data$. For a node set $N \in \mathcal{N}$, $DC(N)$ denotes the set of data constraints that refer to the terms d_A for $A \in N$. We write DA for $\bigcup_{\emptyset \neq N \subseteq \mathcal{N}} DA(N)$ and DC for $\bigcup_{\emptyset \neq N \subseteq \mathcal{N}} DC(N)$. Given a data constraint $g \in DC(N)$, the semantics for each g is a data assignment $\delta \in DA(N)$,

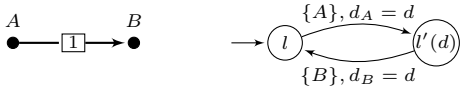


Figure 3: Reo circuit and CA for a FIFO-1 channel.

where $\delta \models g$. Here $\models$ stands for the general satisfaction relation which results from interpreting data constraints over data assignments. $g = \mathrm{true}$ is equivalent with $\delta = \emptyset$ which stands intuitively for no constraints on the set of nodes N.

Definition 1 (CA)**.** *A* constraint automaton (CA) *is a tuple* $\mathcal{A} = (L, \mathcal{N}, \longrightarrow, L_0)$ *where* L *is a finite set of locations,* $\mathcal{N}$ *is a finite set of nodes, disjointly partitioned into* $\mathcal{N}^{src}$*,* $\mathcal{N}^{snk}$*, and* $\mathcal{N}^{mix}$*. The transition relation* $\longrightarrow$ *is a subset of* $L \times 2^{\mathcal{N}} \times DC \times L$*, and* $L_0 \subseteq L$ *the set of initial locations.*

We write $l \xrightarrow{N,g} l'$ instead of $(l, N, g, l') \in \longrightarrow$ and refer to N as the node-set and g the guard of the transition, where $N \neq \emptyset$ and $g \in DC(N)$. The semantics for an instance of $l \xrightarrow{N,g} l'$ is a transition of the form $l \xrightarrow{N,\delta} l'$ where $\delta \models g$. If the current location is l then an instance of the outgoing transitions from l is chosen nondeterministically, and the corresponding I/O-operation (i.e., $\langle N, g\rangle$) is taken leading to the next location l'.

Example 1. A FIFO-1 (first-in first-out with one buffer place) channel and its CA are shown in Figure 3. Location l represents the initial configuration with the buffer being empty, while $l'(d)$ stands for the configuration where data item d is stored in the buffer. Note that this is a simplified parametric model where we use parameter d ranging over all data items. A corresponding non-parametric CA has for each $d \in Data$ a location $l'(d)$ and transitions $l \xrightarrow{\{A\}, d_A = d} l'(d)$ and $l'(d) \xrightarrow{\{B\}, d_B = d} l$. △

Reo supports a `hide` operation, realized on CA by a hiding structure, and a `join` operation, realized by two constructions. For the join of a source node with another (sink, source or mixed) node, we can use their *product*, while joining sink or mixed nodes can be specified by a *merger* CA. The reader interested in more details with respect to basic Reo channels and the corresponding CA is referred to [1], [2].

III-A Timed Constraint Automata

Arbab et al. introduce *timed constraint automata* (TCA) in [4], extended from the original CA by adding real-time aspects to describe the behaviour specification of channels and component interfaces involving timing constraints. As in classical timed automata with location invariants [8], [9], TCA have two kinds of transitions: (i) Internal changes of the locations caused by some time constraints and (ii) transitions that represent the synchronized execution of I/O-operations at some of the ports.

Notation 2 (Clock assignments and clock constraints)**.** Let $\mathcal{C}$

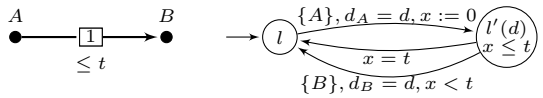


Figure 4: Timed Reo circuit and TCA for an expiring FIFO-1 channel.

be a finite set of clocks. A function $\nu : \mathcal{C} \to \mathbb{R}_{\geq 0}$ is referred to be as a *clock assignment* (CA). For a clock assignment ν and a time $t \in \mathbb{R}_{\geq 0}$, $\nu + t$ denotes the clock assignment that assigns the value $\nu(x) + t$ to every clock $x \in \mathcal{C}$. If $C \subseteq \mathcal{C}$ then $\nu[C := 0]$ stands for the clock assignment that returns the value 0 for every clock $x \in C$ and the value $\nu(x)$ for every clock $x \in \mathcal{C} \setminus C$. A *clock constraint* ($CC$) for $\mathcal{C}$ is defined as $cc ::= true | x \odot n | cc \wedge cc$, where $x \in \mathcal{C}, \odot \in \{<, \leq, \geq, >, =\}, n \in \mathbb{N}$, and $CA(\mathcal{C})$ (or CA) denotes the set of all clock assignments and $CC(\mathcal{C})$ (or CC) the set of all clock constraints.

Example 2. Now let us consider expiring FIFO-1 channels in Figure 4, which extend from FIFO-1 with a max time constraint for the data residing in the buffer. After that time, the data is forced to be lost. A clock x is declared in TCA. A time constraint $\leq t$ equipped under the buffer in Reo circuit is used to specify the expiring time aspect, which is denoted by a clock constraint $CC(x)$ as an invariance condition for location $l'(d)$ in TCA. The two edges from $l'(d)$ to l represent the event where a data item is discarded upon timer expiration and the event where B reads the data out of the buffer respectively. △

III-B Probabilistic Constraint Automata

In [5], Baier introduced *probabilistic constraint automata* (PCA) for describing probabilistic connectors in Reo circuits built out of unreliable channels with known failure probabilities, so as to support the modelling of probabilistic lossy synchronous channels or randomized synchronous channels. Baier also defines a probabilistic model, called *simple probabilistic constraint automata* (SPCA), that appear natural to model various kinds of unreliable FIFO channels. SPCA only treat probabilistic choices over configurations but fail to model two important aspects: (i) Channels where synchronous I/O-operations might fail with some probability, such as in probabilistic lossy synchronous channels; (ii) Coin tossing actions where different data items are produced through sink nodes. PCA allow to model these common aspects of IoT systems. In this paper, we consider probabilistic aspects as those representable in PCA.

Merger structures are (again) used for joining sink or mixed nodes in a preprocessing step. The `product` operation on PCA constitutes the semantics for the `join` operation (i.e., joining one source node with another one) in Reo circuits. This enables to construct complex connectors out of simpler ones. According to Baier, generating the product of two PCA is much more difficult compared to

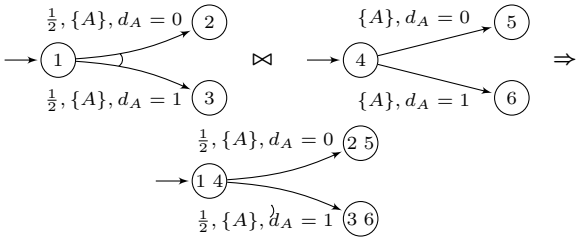


Figure 5: A case for product for PCA $\mathcal{A}_1$ and $\mathcal{A}_2$.

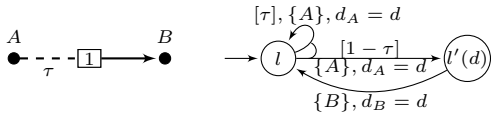


Figure 6: Probabilistic Reo circuit and PCA for a faulty FIFO-1 channel.

SPCA that we might need to match (i) one transition with one transition in the other PCA deterministically, or (ii) one transition where none of the sink or mixed nodes of the other PCA is involved with several transitions in the other PCA. This is rooted in the fact that PCA allow distinct I/O-operations on probabilistic branches. Let us consider the case in Figure 5: $\mathcal{A}_1$ offers I/O-operations with a probabilistic effect for a sink node A that in turn is a source node in $\mathcal{A}_2$. Then $\mathcal{A}_2$ can react on the outcome of $\mathcal{A}_1$'s probabilistic choice (namely $d_A = 0$ or $d_A = 1$). In this case, the conventional product operation is not handleable, and $\mathcal{A}_1$ could be a perfect example for the inability of SPCA.

Notation 3 (Probability distribution)**.** A (discrete) *probability distribution* over a countable set S is a function $\pi : S \to [0, 1]$ satisfying $\sum_{s \in S} \pi(s) = 1$. We call the elements of S *probabilistic events*. We use $Distr(S)$ to denote the set of distributions over S.

Example 3. We consider another variant of FIFO-1, called faulty FIFO-1 in Figure 6, where the data fails to be written into the buffer with some probability τ. In PCA, the 'arc' notation is used to connect probabilistic branches of one transition. Since I/O-operations for the writing failure and success are different (modelled by $\langle \{A\}, \emptyset \rangle$ and $\langle \{A\}, d_A = d \rangle$ on each pair of probabilistic branches out of l), SPCA are not expressive enough for such situations. △

IV Priced Probabilistic Timed Constraint Automata

Real time aspects (e.g., a pen being able to indicate to the outside ink tank to stop refilling no more than 0.01s after reaching its maximum capacity bound) and probabilistic aspects (e.g., the chance of the ink tank failing to release ink being at most 5% after a request from the pen) are two vital aspects to be considered in models of IoT. In these contexts, distributed components (i.e., physical *things*) coordinate with each other using different kinds of communication channels. We intend to employ Reo and CA as the scripting language

to specify components as nodes in Reo and connections as channels or connectors. To this end, they are to be enhanced to support those two features properly. For instance, let us consider a faulty expiring FIFO-1 channel where the data fails to be written into the buffer with some probability and the data is going to be lost after certain time units once it has entered the buffer. On the technical level, we can model the time expiration using TCA, and the possibility to lose data by PCA respectively, but there is still no variant of CA that combines these two aspects effectively. In practice, another vital characteristic of IoT systems is that they need to meet quantitative requirements, because *things* are operating with limited power, memory, computation budgets and other resources. One might be interested in computing some expected values(e.g., the average ink usage costs per refill). A way to handle such quantitative requirements is to use reward/cost notation to specify interesting pricing informations.

This motivates us to propose a model, called *Priced Probabilistic Timed Constraint Automata* (pPTCA), which combines PCA and TCA and is equipped with prices to model nondeterministic, probabilistic, real-time and reward/cost aspects of a system composed of channel-based component (distributed) connectors. Inspired from weighted [10] and priced [11] timed automata, in pPTCA, price rates are attached to locations, indicating the cost or reward to reside in a location per time unit. Furthermore, prices are attached to transitions, indicating the cost to take the transition or the reward produced, when changing from one location to another. In the following definition, we explicitly use data assignments instead of the symbolic representation data constraints on the transitions in our model to be consistent with the definition of PCA in [5]. We use $\Sigma \subseteq (2^{\mathcal{N}} \times DA \times 2^{\mathcal{C}} \times L)$ to denote the set of probabilistic events, and Ct for the set of costs (or rewards).

Definition 2 (pPTCA)**.** *A* priced probabilistic timed constraint automaton (pPTCA) *is a tuple* $\mathcal{A} = (L, \mathcal{C}, \mathcal{N}, \longrightarrow, L_0, ic, \rho)$ *where* L *is a countable set of locations,* $L_0 \subseteq L$ *the set of initial locations,* $\mathcal{C}$ *a finite set of clocks, and* $\mathcal{N}$ *is a finite set of nodes disjointly partitioned into* $\mathcal{N}^{src}, \mathcal{N}^{snk}$ *and* $\mathcal{N}^{mix}$*.* $ic : L \to CC$ *is a function that assigns to any location* l *an invariance condition, and* $\longrightarrow \subseteq L \times CC \times Distr(2^{\mathcal{N}} \times DA \times 2^{\mathcal{C}} \times L)$*.* $\rho : Ct \to (L \cup \Sigma \to \mathbb{R})$ *is a price function, for each* cost $ct \in Ct$*, assigns* price rate *to the locations and* price *to its probabilistic branches. We require that for any transition* $l \xrightarrow{cc} \Pi$*, we have: If* $\Pi(N_1, \delta_1, C_1, l_1) > 0$ *and* $\Pi(N_2, \delta_2, C_2, l_2) > 0$ *then* $N_1 \cap \mathcal{N}^{src} = N_2 \cap \mathcal{N}^{src}$ *and* $\delta_1.A = \delta_2.A$ *for all source nodes* $A \in N_1 \cap \mathcal{N}^{src}$*.*

We claim that the probability distribution does not effect the data items source nodes take in, which is formalized in the last statement.

A transition fires if data items are observed in the respective nodes of the component and the clock constraint is satisfied, and the data assignment is performed (except for the empty node-set). We write $l \xrightarrow{cc} \Pi(N, \delta, C, l')$ instead of $(l, cc, \Pi(N, \delta, C, l')) \in \longrightarrow$, where N denotes the nodes, $\delta \in DA(N)$ the data assignment on N, cc the clock constraint, C the set of clocks to be reset, l and l' represent the source and target locations respectively. Transitions with Dirac distribution (i.e., $\mathcal{D}(N, \delta, C, l') = 1$) are called *Dirac transitions*, and we simply write $l \xrightarrow{cc} \mathcal{D}$.

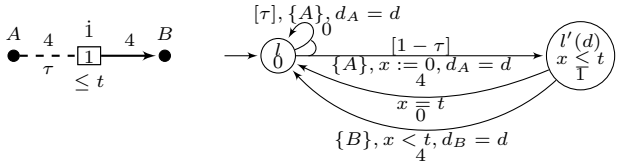


Figure 7: Reo circuit and pPTCA for a faulty expiring FIFO-1 channel with energy consumption.

Intuitively, a pPTCA behaves as follows: It starts in an initial location $l_0 \in L_0$. Then, whenever a location l is occupied with a valid invariance condition ic and each cost ct incurred until now, it is chosen nondeterministically whether to delay or to take a transition which satisfies the above observation constraint and data/clock requirements. Delaying will increase each clock by the delay units and each accumulated cost by the product of delay units and corresponding price rate $\rho(ct)(l)$. A transition instance $l \xrightarrow{cc} \Pi(e)$ (where $e = \langle N, \delta, C, l' \rangle \in \Sigma$) is taken, resetting each clock $x \in C$ to 0, and increasing the cost by $\rho(ct)(e)$. The configuration moves to l' with probability $\Pi(N, \delta, C, l')$. Note that it is reasonable practice (rooted in the work of Segala [12]) that each transition has a single clock constraint, while the sub-probabilistic branches can be equipped with different data assignments and prices. In the sequel, clock assignment $cc = true$, data constraint $\delta = \emptyset$ and price or price rate 0 are often left out for simplicity.

Example 4. Figure 7 presents a faulty expiring FIFO-1 channel equipped with prices to reflect the energy consumption on channels and within the buffer, representing that the system needs to consume energy for transmitting data as well for keeping data in the buffer. The semantic pPTCA for this Reo circuit is depicted on the right. This simplified parametric model combines the features from TCA and PCA. Beyond these, in this Reo circuit, the positive constant 4 above the channels represents the instantaneous energy cost for data transmission from node A to the buffer and from the buffer to node B respectively. We use the form $\dot{\$}$ (so as to distinguish it from price) in pictorial Reo circuits to denote the price rate $\$ \in \mathbb{R}$. Thus, $\dot{1}$ above the buffer specifies the energy cost increases by 1 each time unit. Accordingly in the pPTCA, the price rate 0 on location l results from no energy being consumed when the buffer is empty, and 1 on location $l'(d)$ to represent 1 energy unit cost per time unit. The energy consumption for data transmission

is encoded into the price 4 on the probabilistic branch from l to $l'(d)$ and on the lowest transition. $\triangle$

Target run: Given a pPTCA $\mathcal{A} = (L, \mathcal{C}, \mathcal{N}, \longrightarrow, L_0, ic, \rho)$, a *target run* r for $\mathcal{A}$ represents a finite sequence of consecutive transition instances that ends up at a target location without delaying of the following form,

$$\mathsf{r} = l_0 \xrightarrow{cc_0, t_0, \Pi_0} l_1 \xrightarrow{cc_1, t_1, \Pi_1} \dots \xrightarrow{cc_{n-1}, t_{n-1}, \Pi_{n-1}} l_n$$

where $l_i \in L, cc_i \in CC, (l_i, cc_i, \Pi_i) \in \longrightarrow$, and there exist $N_i \subseteq \mathcal{N}$, $\delta_i \in DA(N_i)$, $C_i \in \mathcal{C}$, $e_i = \langle N_i, \delta_i, C_i, l_{i+1} \rangle$, such that $\Pi_i(e_i) > 0$, and $t_i > 0$ satisfying:

$$\begin{aligned} &\text{(i)} && \nu_i + t'_i \models ic(l_i) \text{ for all } 0 < t'_i \leq t_i \\ &\text{(ii)} && (\nu_i + t_i)[C_i := 0] \models ic(l_{i+1}) \text{ and} \\ &\text{(iii)} && \nu_i + t_i \models cc_i \end{aligned}$$

where $\nu_i \models ic(l_i), 0 \leq i \leq n$. We denote the target location as $\mathsf{Last}(\mathsf{r}) = l_n$. The total cost of r corresponding to $ct \in Ct$ is $\mathsf{TC}_{ct}(\mathsf{r}) = \sum_{i=0}^{n-1} (\rho(ct)(l_i) \cdot t_i + \rho(ct)(e_i))$, i.e., the sum of accumulated prices on locations and instantaneous prices on transitions. The probability for reaching the target location along r is defined as: $\mathsf{P}(\mathsf{r}) = \prod_i \Pi_i(e_i)$.

IV-A Priced probabilistic timed Reo circuits

Electric energy is the key resource needed for *things* to sense, to calculate, to store and to interact under IoT, and thus each step of those *things* consumes electric power. We detail the pricing model from this very natural IoT perspective now. Nevertheless we mention that we take energy resources as an example, users are nevertheless free to build and work with any pricing structure they wish.

We usually want *things* to accomplish a task or sustain a series of interactions with a limited budget of energy. We furthermore might want to test or verify whether the energy consumption is acceptable (possibly under some fairness conditions) for each *thing* or for the whole IoT system. For this purpose, we extend several useful primitive channels in the Reo framework by adding prices and price rates together with timing constraints and probabilistic choices for enhancement of their I/O-operations. We let $b, b_1, b_2 \in \mathbb{R}$ be instantaneous prices on channels (or on transitions in pPTCA), and $\dot{\$}$ (represented by $\$ \in \mathbb{R}$ on the locations in pPTCA) be price rate on buffers or time counting nodes.

Basic channels: In Figure 8 we present conservative extensions of part of interesting channels from the original Reo publication [1]. We also present a new communication channel named Zigbee, defined for the purpose of modelling realistic connection means, and a new timer, named $t_{\geq}$-*timer*, introduced to support lower-bound timers (i.e., the time for alerting time-out, represented by "TO", is greater than or equal to t). Naturally, a Zigbee channel is equipped with data transmission time t through the channel, failure probability τ and energy consumption b. These channels will reappear in the case study section that follows.

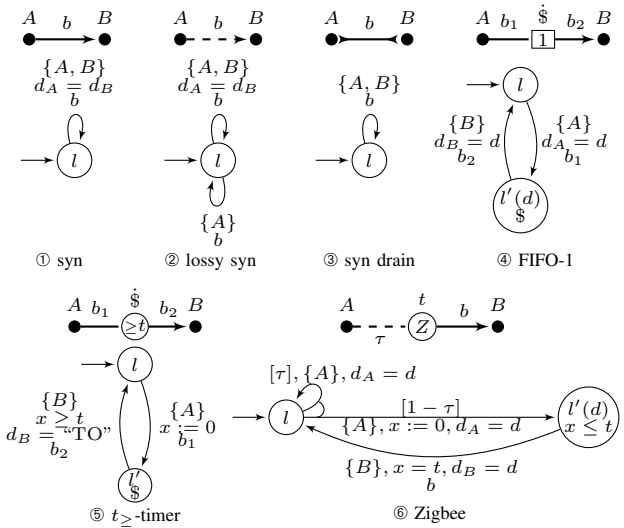


Figure 8: Six basic Reo channels and their pPTCA.

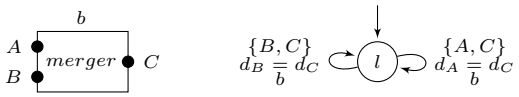


Figure 9: *Merger*.

Merger: For the *merger* structure to be equipped with prices, we decide to equip it with a price rate of 0 on the location and some energy cost $b \in \mathbb{R}$ on the transitions, as depicted in Figure 9. This is necessary because the merger has a genuine logic functionality, trying to control the nondeterministic choices between two sink or mixed nodes, as well as implementing the join of two synchronous channels (with ends A, C and B, C respectively).

IV-B Product and hiding of pPTCA

After using common mergers to reprocess the automata taking care of the join of non-source nodes, we build the product of two pPTCA to semantically specify the join of one source node with another node in Reo circuits. As mentioned above, one transition is either matched by one or several transitions in another pPTCA. In the latter case, we require all matched transitions have the same clock guard for technical consideration. Assuming independence of energy consumption, for each considered cost, the synchronisation of two pPTCA induces the sum of the price rates on each sub-location to be used as the price rate for the combined location. Once two matching transitions (or one transition matching several transitions) fire at the same time, the price on every probabilistic branch of the combined transition is set to the sum of that on each original branch.

Definition 3 (Product)**.** *Given two* pPTCA $\mathcal{A}_i = (L_i, \mathcal{C}_i, \mathcal{N}_i, \longrightarrow_i, L_{0,i}, ic_i, \rho_i)$, $i = 1, 2$, *with* $\mathcal{C}_1 \cap \mathcal{C}_2 = \emptyset$, *the* product of $\mathcal{A}_1$ and $\mathcal{A}_2$ *is* $\mathcal{A}_1 \bowtie \mathcal{A}_2 = (L_1 \times L_2, \mathcal{C}_1 \cup \mathcal{C}_2, \mathcal{N}_1 \cup \mathcal{N}_2, \longrightarrow, L_{0,1} \times L_{0,2}, ic, \rho)$ *where* $ic(\langle l_1, l_2 \rangle) = ic_1(l_1) \wedge ic_2(l_2)$, $\rho(ct)(\langle l_1, l_2 \rangle) = \rho_1(ct)(l_1) + \rho_2(ct)(l_2)$

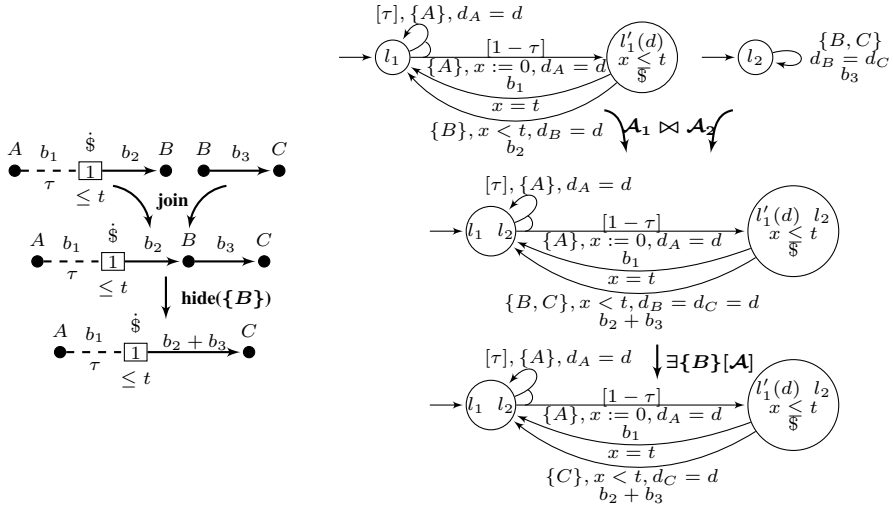


Figure 10: Example on join and hide, and their semantics.

for each $ct \in Ct$, $\longrightarrow$ is defined by the following rules:

$$\frac{l_1 \xrightarrow{cc_1}_1 \Pi_1, \bigwedge_{j=1}^{j=k}(l_2 \xrightarrow{cc_2}_2 \Pi_{2,j})}{\langle l_1, l_2\rangle \xrightarrow{cc_1 \wedge cc_2} \Pi} \quad \text{(initialized by } l_1\text{)}$$

where $k \in \mathbb{N}^+$, and $\bigwedge_{j=1}^{j=k}(l_2 \xrightarrow{cc_2}_2 \Pi_{2,j})$ stands for all the transitions in $\mathcal{A}_2$ with same current location l_2, if *for each pair of $(M, \sigma) = (N_1 \cap \mathcal{N}_2, \delta_1)$ where $\Pi_1(N_1, \delta_1, C_1, l_1') > 0$, either*

(i) *$M \neq \emptyset$, there exists a probability distribution $\Pi_{2,j}$ such that $\Pi_{2,j}(N_{2,j}, \delta_{2,j}, C_{2,j}, l_{2,j}') > 0$ implies $N_2 \cap \mathcal{N}_1 = M$ and $\delta_{2,j}.A = \sigma.A$ for all $A \in M$, then*

$$\Pi(\overbrace{N_1 \cup N_{2,j}, \delta_1 \uplus \delta_{2,j}, C_1 \cup C_{2,j}, \langle l_1', l_{2,j}'\rangle}^{e}) = \Pi_1(\underbrace{N_1, \delta_1, C_1, l_1'}_{e_1}) \cdot \Pi_{2,j}(\underbrace{N_{2,j}, \delta_{2,j}, C_{2,j}, l_{2,j}'}_{e_2})$$

where $\rho(ct)(e) = \rho_1(ct)(e_1) + \rho_2(ct)(e_2)$,

(ii) *$M = \emptyset$, then*

$$\Pi(\underbrace{N_1, \delta_1, C_1, \langle l_1', l_2\rangle}_{e}) = \Pi_1(\underbrace{N_1, \delta_1, C_1, l_1'}_{e_1})$$

where $\rho(ct)(e) = \rho_1(ct)(e_1)$.

Transitions out of $\langle l_1, l_2\rangle$ initialized by l_2 are defined in the symmetric way.

The above definition is a conservative extension of the one for PCA [5] to timed and priced features, with the handling of prices being motivated by [13]. It reformulates the original PCA definition and thereby enables to drop a restriction that originally excludes composition of transitions that are neither input-independent nor I/O-deterministic. Support for this case arises naturally from the reformulation.

The hiding structure for pPTCA is extended from that of the PCA by augmenting it with a timed and a price structure. Given one pPTCA $\mathcal{A} = (L, \mathcal{C}, \mathcal{N}, \longrightarrow, L_0, ic, \rho)$, a new clock $y \notin \mathcal{C}$ and a non-empty node-set $M \subseteq \mathcal{N}^{mix}$. Then, the `hide` operation $\mathrm{hide}(\mathcal{A}, M)$ on $\mathcal{A}$ results a pPTCA $\exists M[\mathcal{A}] = (L, \mathcal{C} \cup \{y\}, \mathcal{N} \setminus M, \longrightarrow_M, L_0, ic, \rho_M)$ where $\longrightarrow_M$ is given by the rules:

$$\frac{l \xrightarrow{cc} \Pi, (N = \emptyset \vee N \setminus M \neq \emptyset)}{l \xrightarrow{cc}_M \Pi_M} \quad \text{or} \quad \frac{l \xrightarrow{cc} \Pi, \emptyset \neq N \subseteq M}{l \xrightarrow{cc \wedge (y>0)}_M \Pi_M}$$

where $\Pi_M(\overbrace{N \setminus M, \delta|_{N\setminus M}, C \cup \{y\}, l'}^{e_M}) = \Pi(\overbrace{N, \delta, C, l'}^{e})$, and for each $ct \in Ct$, $\rho_M(ct)(e_M) = \rho(ct)(e), \rho_M(ct)(l) = \rho(ct)(l)$ for all e_M and $l \in L$. The second rule is used to specify the case that when all the nodes and corresponding data constraints are hidden (as the result of a hiding operation), we need to ensure that this transition can fire only after a positive delay. This is achieved by introducing an additional clock.

Example 5. Figure 10 illustrates the process of building a connector out of a priced faulty expiring FIFO-1 channel and a priced synchronous channel. On the level of the Reo circuit, these two channels join over B, then hiding it leads to the absence of B and the sum of b_2 and b_3 as the new price associated to the buffer to C. The semantic operations for those behaviours are denoted by the product of the two pPTCA and hiding B and all data constraints involved with d_B on transitions. △

V Case study

In this section, we consider an Internet of Things scenario where a remote and isolated factory unit answers to customers' requests. We assume in this IoT system that one *sink device* **SD** is in charge of receiving orders and distributing the tasks to *robots*. The robots are deployed in hostile environments and their main duty is to sense, process and transmit data. They need to be able to operate as long as possible. Due to cost economics, they are small in size and provide limited power (supported by batteries). At the

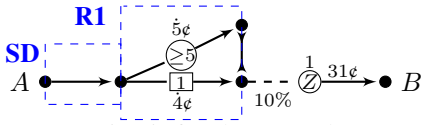


Figure 11: Reo circuit for the exemplary IoT system.

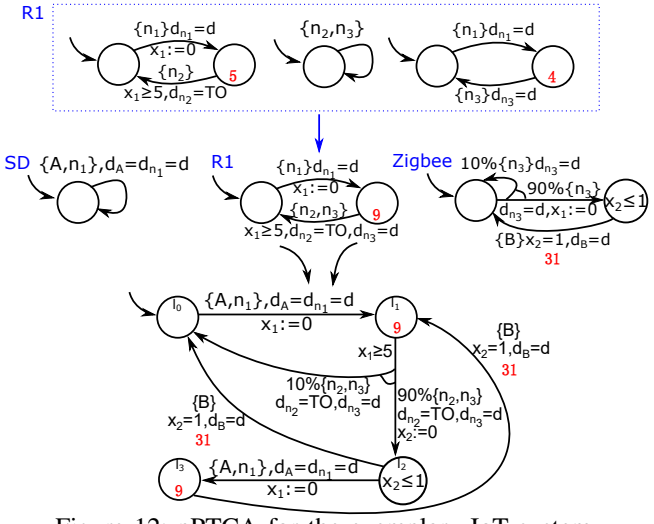


Figure 12: pPTCA for the exemplary IoT system.

beginning, only one robot named **R1** is deployed, and can handle one task at a time. After completing the task, **R1** sends data to one nearby *workstation* **W** through Zigbee channels for further processing. In this case, we consider one energy cost $ct \in Ct = \{ct\}$. For simplicity, $\rho(ct)$ is written directly as ρ.

V-A *Reo circuit for the system*

We can easily construct the model of connecting every *thing* in this IoT system as layed out in Figure 11. We use one connector composed of one FIFO-1, one $t_{\geq}$-timer and one synchronous drain channel to model task processing in **R1**, where the data from the buffer can only be fetched after the execution time (i.e. ≥ 5). The drain channel restricts time-out and buffer-clear to coincide. Price rates equipping FIFO-1 and $t_{\geq}$-timer represent the energy consumption rate for data storing and processing respectively. The energy consumption for data transmission is encoded in Zigbee channels. Due to the unreliable wireless network, these communication channels have a loss probabilities associated. **SD** writes data through A and **W** reads data from B. We use numbers with ¢(i.e. micro Watt) to denote energy consumption, and numbers with no units for time.

Based on the pPTCA corresponding to this example (provided in Figure 12), one can identify the target run that transmits data to **W** with lowest energy cost as follows:

$$\mathsf{r}_1 = l_0 \xrightarrow{true,t_1,\Pi_0} l_1 \xrightarrow{x_1=5,5,\Pi_1} l_2 \xrightarrow{x_2=1,1,\Pi_2} l_0$$

where $e_0 = \langle \{A,\mathrm{n}_1\}, d_A = d_{\mathrm{n}_1} = d, x_1 := 0, l_1\rangle$, $e_1 = \langle \{\mathrm{n}_2,\mathrm{n}_3\}, d_{\mathrm{n}_2} = \mathrm{TO} \wedge d_{\mathrm{n}_3}{=}d, x_1{:=}0, l_2\rangle$, $e_2{=}\langle \{B\}, d_B{=}d, \emptyset, l_0\rangle$, $\Pi_0(e_0){=}1, \Pi_1(e_1){=} 0.9, \Pi_2(e_2){=}1$. The energy cost of r_1 is $ct_\rho(\mathsf{r}_1) = (\rho(l_0)\cdot t_1 + \rho(e_0)) + (\rho(l_1)\cdot 5 + \rho(e_1)) + (\rho(l_2)\cdot 1 + \rho(e_2)) = 76¢$. The reach probability is $\mathsf{P} = 1 \cdot 0.9 \cdot 1 = 0.9$.

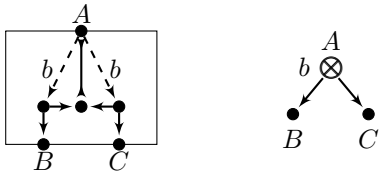


Figure 13: Router connector and its instance.

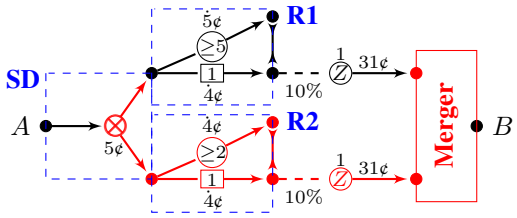


Figure 14: Reo circuit for the updated exemplary system.

V-B *Updating the system*

Now let us consider an update to this system: Due to the increased financial budget and technical development, a new advanced robot **R2** is deployed in this unit. **R2** is faster and more effective (modelled with lower execution time and energy consumption rate for processing).

We first construct a compositional connector called *Router* in Figure 13 to model the task distribution where incoming tasks are routed to one of the robots. A Router is built out of four synchronous channels, two lossy synchronous channels, and one synchronous drain channel. A inputs some data item d, and either B or C outputs d simultaneously. The nondeterminism for the case that both B and C are ready is determined by the middle mixed node taking the data from only one of the two synchronous channels coinciding on it. On the right a simple mark is used to refer to the instance for Router. b represents the energy price for effectuating the distribution.

The Reo circuit for this updated system is shown in Figure 14. Notably, realizing such a flexible and adaptive communication and interaction pattern is readily possible by means of priced probabilistic timed Reo. The updated connectors are depicted in red. The semantic pPTCA (provided in Figure 15) can be mechanically inspected to compute the lowest energy consumption and the corresponding probability for transmitting data to **W** either by **R1** or **R2**. We can find the following target runs:

$$\mathsf{r}_2 = l_0 \xrightarrow{true,t_2,\Pi_0} l_1 \xrightarrow{x_1=5,5,\Pi_1} l_2 \xrightarrow{x_3=1,1,\Pi_2} l_0$$

where $e_0 = \langle \{A,\mathrm{n}_1,\mathrm{n}_2\}, d_A = d_{\mathrm{n}_1} = d_{\mathrm{n}_2} = d, x_1 := 0, l_1\rangle, e_1 = \langle \{\mathrm{n}_3,\mathrm{n}_4\}, d_{\mathrm{n}_3}{=}\mathrm{TO}{\wedge}d_{\mathrm{n}_4}{=}d, x_3{:=}0, l_2\rangle, e_2{=}\langle \{B,\mathrm{n}_8\}, d_B{=}d_{\mathrm{n}_8}{=}d, \emptyset, l_0\rangle$, $\Pi_0(e_0){=}1, \Pi_1(e_1){=}0.9, \Pi_2(e_2){=}1$, and

$$\mathsf{r}_3 = l_0 \xrightarrow{true,t_3,\Pi_0} l_4 \xrightarrow{x_2=2,2,\Pi_1} l_8 \xrightarrow{x_4=1,1,\Pi_2} l_0$$

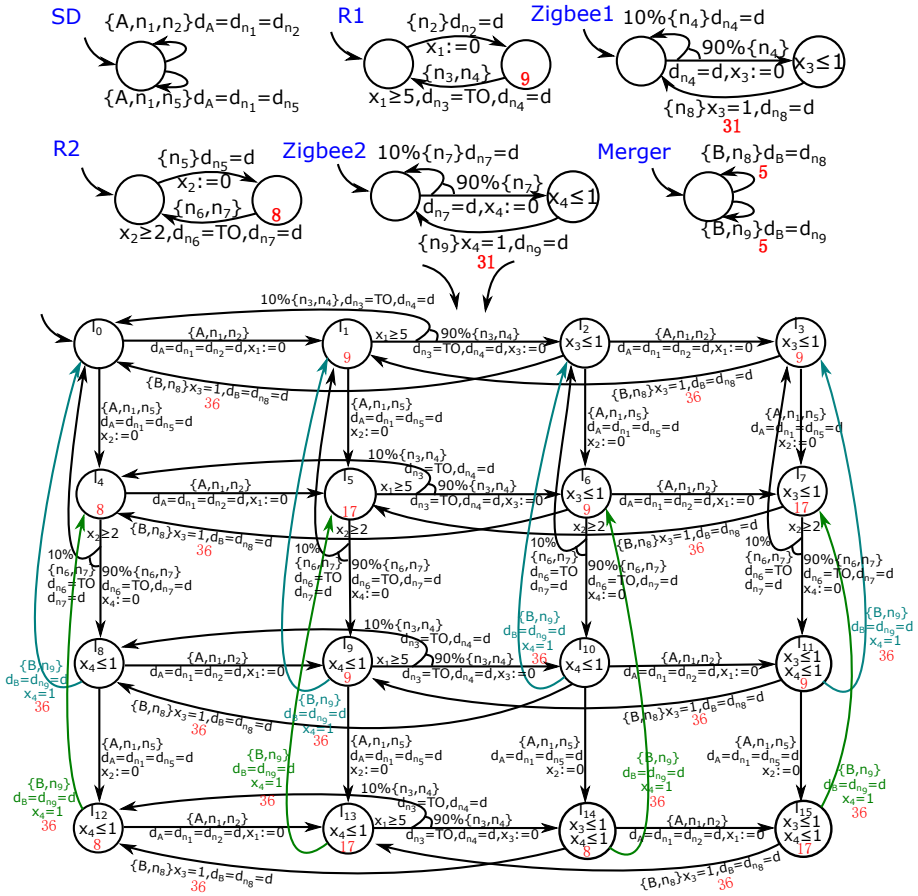


Figure 15: pPTCA for the exemplary IoT system.

where $e_0 = \langle\{A,n_1,n_5\}, d_A = d_{n_1} = d_{n_5} = d, x_2 := 0, l_4\rangle$, $e_1 = \langle\{n_6,n_7\}, d_{n_6}{=}\mathrm{TO} \wedge d_{n_7}{=}d, x_4{:=}0, l_8\rangle$, $e_2{=}\langle\{B,n_9\}, d_B{=}d_{n_9}{=}d, \emptyset, l_0\rangle$, $\Pi_0(e_0){=}1, \Pi_1(e_1){=}0.9, \Pi_2(e_2){=}1$. Due to the task distribution, the average energy cost is $\frac{ct_\rho(r_2)+ct_\rho(r_3)}{2} = 66.5¢$. And probability for each run is $P = 0.9$. Obviously, in this case the updated system consumes lower energy while maintaining the same probability of success.

Based on the pPTCA, one can explore more evaluations and also apply exhaustive verification techniques. For instance, we might be interested in the situation where **R1** and **R2** both have just completed a task and the resulting data is in transmission while no new tasks come in (modelled by location l_{10} in Figure 15). It is not difficult to calculate the minimal expected time for reaching that situation. Moreover the numbers of complete task execution (i.e., reaching l_0 from l_2 or l_8) under some bounded time could be computed, just as many other quantities of interest.

VI Related Work

In recent years, varied efforts have been devoted to develop the formal modelling involving pricing information in the IoT domain. Li and Jin et al. developed an approach to model the reliability and cost of service composition in the IoT on the basis of Markov Decision Processes with cost structure [14]. Then in [15], Li and Wei et al. extended this work to model real-time constraints, where the IoT services and their corresponding environment can be described in Probabilistic Timed Automata. However, they focus on a level of services in SOC-based IoT. Martinez et al. [16] proposed a methodology for the power consumption of wireless network devices at the system level. Through this approach, application engineers can foretell how parameters impact power consumption and make estimates without a complete implementation of the application. This pricing model exclusively aims at analysing the energy life-cycles in applications, making its limitations on IoT systems with probabilistic and timed aspects. Specifically, Lee has claimed that Reo plays a significant role on an emerging means to model Cyber Physical Systems (CPS) at the component interaction level [17]. Palomar et al. [18] developed a case study on the scalable smart city systems in CPS using Reo as the modelling language. A closely related model considering non-function requirements is the so-called resource-sensitive TCA (RSTCA) [19] where execution times for interactions are made dependent on resource availability and timeout behaviour. Relative to our approach, RSTCA seems more restrictive, using implicit clocks on each transition. Another model, called Quantitative Constraint Automata [20], con-

siders quantitative aspects, which can be specified by so-called Q-algebra as constraints on transition. The motivation of this model is close in spirit to the one considered here, but it avoids to consider global time advance explicitly. In [21], Baier and Wolf developed Continuous-time CA (CCA) as extensions of CA with soft (memoryless) time, instead of hard time bounds (i.e., exact upper and lower bounds of time). Stochastic Reo [22] is a variant of Reo appended with data arrival rates and processing delay rates. The approach targets soft real-time behaviour, too, and is thus similar in capabilities to the CCA approach. Unfortunately CCA does not support representing nondeterminism. Motivated by the IoT context, our model represents hard real-time behaviour mixed with probabilistic effects and nondeterministic behaviour.

VII Conclusion

In this paper, we develop a priced, probabilistic and timed extension of Reo and Constraint Automata (which forms pPTCA) for the purpose of enabling a faithful modelling of IoT systems. Therefore, from the level of Reo, the modeller can easily construct (possible) large scalable IoT systems. Within the framework of pPTCA, where cost, time and probabilities are taken into consideration, the modeller can describe, on a single model, different aspects of an IoT system, and analyse real-time properties, performance, QoS and reliability properties. A small example has demonstrated the principal expressiveness and modelling conveniences. We are currently exploring semantics preserving translations to Modest [23], [24], a well-designed and very rich framework for the analysis of real-time, distributed and stochastic systems, to enable model checking as well as discrete event simulation of the model families we developed for IoT.

Acknowledgment

This work is supported by the National Natural Science Foundation of China (Grant No. 61370100 and Grant No. 61321064), and Shanghai Knowledge Service Platform for Trustworthy Internet of Things (Grant No. ZF1213), by the ERC Advanced Grant 695614 (POWVER), and by the Sino-German Center for Research Project CAP (GZ 1023). The authors would like to thank the anonymous referees for their valuable comments and suggestions.

References

[1] F. Arbab, "Reo: a channel-based coordination model for component composition," *Mathematical Structures in Computer Science*, vol. 14, no. 3, pp. 329–366, 2004.

[2] C. Baier, M. Sirjani, F. Arbab, and J. J. M. M. Rutten, "Modeling component connectors in reo by constraint automata," *Sci. Comput. Program.*, vol. 61, no. 2, pp. 75–113, 2006.

[3] S. T. Q. Jongmans and F. Arbab, "Overview of thirty semantic formalisms for reo," *Sci. Ann. Comp. Sci.*, vol. 22, no. 1, pp. 201–251, 2012.

[4] F. Arbab, C. Baier, F. S. de Boer, and J. J. M. M. Rutten, "Models and temporal logics for timed component connectors," in *2nd International Conference on Software Engineering and Formal Methods (SEFM)*, 2004, pp. 198–207.

[5] C. Baier, "Probabilistic models for reo connector circuits," *J. UCS*, vol. 11, no. 10, pp. 1718–1748, 2005.

[6] "Extensible coordination tools," http://reo.project.cwi.nl/reo/wiki/Tools.

[7] F. Arbab and J. J. M. M. Rutten, "A coinductive calculus of component connectors," in *16th International Workshop on Recent Trends in Algebraic Development Techniques (WADT)*, 2002, pp. 34–55.

[8] R. Alur and D. L. Dill, "A theory of timed automata," *Theor. Comput. Sci.*, vol. 126, no. 2, pp. 183–235, 1994.

[9] T. A. Henzinger, X. Nicollin, J. Sifakis, and S. Yovine, "Symbolic model checking for real-time systems," *Inf. Comput.*, vol. 111, no. 2, pp. 193–244, 1994.

[10] R. Alur, S. La Torre, and G. J. Pappas, "Optimal paths in weighted timed automata," *Theor. Comput. Sci.*, vol. 318, no. 3, pp. 297–322, 2004.

[11] G. Behrmann, A. Fehnker, T. Hune, K. G. Larsen, P. Pettersson, J. Romijn, and F. W. Vaandrager, "Minimum-cost reachability for priced timed automata," in *4th International Workshop on Hybrid Systems: Computation and Control (HSCC)*, 2001, pp. 147–161.

[12] R. Segala and N. A. Lynch, "Probabilistic simulations for probabilistic processes," *Nord. J. Comput.*, vol. 2, no. 2, pp. 250–273, 1995.

[13] A. Turrini and H. Hermanns, "Cost preserving bisimulations for probabilistic automata," *Logical Methods in Computer Science*, vol. 10, no. 4, 2014.

[14] L. Li, Z. Jin, G. Li, L. Zheng, and Q. Wei, "Modeling and analyzing the reliability and cost of service composition in the iot: A probabilistic approach," in *19th International Conference on Web Services (ICWS)*, 2012, pp. 584–591.

[15] G. Li, Q. Wei, L. Li, Z. Jin, Y. Xu, and L. Zheng, "Environment based modeling approach for services in the internet of things," *Scientia Sinica*, vol. 43, no. 10, p. 1198, 2013.

[16] B. Martinez, M. Montón, I. Vilajosana, and J. D. Prades, "The power of models: Modeling power consumption for IoT devices," *IEEE Sensors Journal*, vol. 15, no. 10, pp. 5777–5789, 2015.

[17] E. A. Lee, "Cyber physical systems: Design challenges," in *11th IEEE International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC)*, 2008, pp. 363–369.

[18] E. Palomar, X. Chen, Z. Liu, S. Maharjan, and J. P. Bowen, "Component-based modelling for scalable smart city systems interoperability: A case study on integrating energy demand response systems," *Sensors*, vol. 16, no. 11, p. 1810, 2016.

[19] S. Meng and F. Arbab, "On resource-sensitive timed component connectors," in *9th IFIP WG 6.1 International Conference on Formal Methods for Open Object-Based Distributed Systems (FMOODS)*, 2007, pp. 301–316.

[20] F. Arbab, T. Chothia, S. Meng, and Y. Moon, "Component connectors with qos guarantees," in *9th International Conference on Coordination Models and Languages (COORDINATION)*, 2007, pp. 286–304.

[21] C. Baier and V. Wolf, "Stochastic reasoning about channel-based component connectors," in *8th International Conference on Coordination Models and Languages (COORDINATION)*, 2006, pp. 1–15.

[22] Y. Moon, A. Silva, C. Krause, and F. Arbab, "A compositional semantics for stochastic reo connectors," in *9th International Workshop on the Foundations of Coordination Languages and Software Architectures (FOCLASA)*, 2010, pp. 93–107.

[23] H. C. Bohnenkamp, P. R. D'Argenio, H. Hermanns, and J. Katoen, "MODEST: A compositional modeling formalism for hard and softly timed systems," *IEEE Trans. Software Eng.*, vol. 32, no. 10, pp. 812–830, 2006.

[24] E. M. Hahn, A. Hartmanns, H. Hermanns, and J. Katoen, "A compositional modelling and analysis framework for stochastic hybrid systems," *Formal Methods in System Design*, vol. 43, no. 2, pp. 191–232, 2013.