IEICE TRANS. INF. & SYST., VOL.E100–D, NO.9 SEPTEMBER 2017

PAPER

A Formal Model to Enforce Trustworthiness Requirements in Service Composition

Ning FU[†a)], ***Member***, **Yingfeng ZHANG**[†], **Lijun SHAN**[†], **Zhiqiang LIU**[†], ***and*** **Han PENG**[†], ***Nonmembers***

SUMMARY With the in-depth development of service computing, it has become clear that when constructing service applications in an open dynamic network environment, greater attention must be paid to trustworthiness under the premise of functions' realization. Trustworthy computing requires theories for business process modeling in terms of both behavior and trustworthiness. In this paper, a calculus for ensuring the satisfaction of trustworthiness requirements in service-oriented systems is proposed. We investigate a calculus called QPi, for representing both the behavior and the trustworthiness property of concurrent systems. QPi is the combination of pi-calculus and a constraint semiring, which has a feature when problems with multi-dimensional properties must be tackled. The concept of the quantified bisimulation of processes provides us a measure of the degree of equivalence of processes based on the bisimulation distance. The QPi related properties of bisimulation and bisimilarity are also discussed. A specific modeling example is given to illustrate the effectiveness of the algebraic method.

key words: *trustworthy software, process algebra, pi-calculus, Q-algebra, semi-ring*

1. Introduction

Service-oriented Computing (SOC), which is currently mainly led by Web Services, is the contemporary mainstream model and an important development direction of network computing. Service Computing refers to a development paradigm and relevant theories, methods, technologies, and supporting environment, for creating applications based on services and their compositions. In recent years, with the in-depth development of related research and applications, it has become clear that when creating service applications in an open dynamic network environment, greater attention must be paid to their trustworthiness under the premise of system functions' realization. The Trusted Computing Group (TCG) defined trusted computing as a computer system which will consistently behave in expected ways, and whose behavior can be enforced by computer hardware and software [1]. From the corresponding definitions [1], [2] and related research [3], it is evident that trusted computing is a measure of the non-functional properties of a system, such as real-time performance, reliability, availability, security, and so on. We believe that the concept of trustworthiness of a service computing system has two aspects: one is that the functions of the system are fully realized, and the other is that the non-functional properties of the system meet users' needs.

Computer science theory and formal methods are considered to constitute an important approach to ensure software trustworthiness [4]. Pi-calculus [5], which focuses on inter-process mobile communications, is a theory for modeling concurrent systems and can be considered as a continuation of Milner's work on the process calculus CCS. It allows channel names to be communicated along the channels themselves, and in this way it is able to describe concurrent computations whose network configuration may change during the computation. It also provides a strong formal theory for modeling service computing systems and analyzing their dynamic characteristics.

Based on a development of Milner's pi-calculus that explicitly investigates how the compositional features of process algebra may translate to the trustworthiness domain, we present a calculus for modeling processes' behavior and trustworthiness properties. We extend the pi-calculus by introducing an algebra called Q-algebra [6], which adds a new multiplicative operator to the constraint semiring. Q-algebra allows the description of multi-dimensional trustworthiness properties in a uniform way. We combine Q-algebra and pi-calculus and associate each pi-calculus action with a trustworthiness tuple. Using this method, we model the behavior and trustworthiness properties of service computing systems in a unified manner. The operational semantics of QPi are also combined with trustworthy properties, and the behavior and trustworthiness of the modeled system can be deduced from the structure of processes and the inter-operations between these processes. We introduce the concept of quantified bi-simulation of processes, which provides us with a measure of the degree of equivalence of processes' behavior and trustworthiness properties. The theory of quantified bisimulation can not only be used to analyze the equivalence of process behaviors qualitatively, but also to analyze the degree of similarity between process behavior and trustworthiness quantitatively. The feasibility and effectiveness of the proposed calculus is demonstrated through a modeling and analysis example.

This paper is organized as follows: in the next section we begin with a brief introduction on Q-algebra. Then, we present the QPi language in Sect. 3. The operational semantics of QPi are discussed in Sect. 4, while in Sect. 5 we define a quantified version of bisimulation in QPi and demonstrate some of its properties. Section 6 introduces a temporal logic that allows one to associate trustworthiness constraints with

Manuscript received September 17, 2016.
Manuscript revised March 28, 2017.
Manuscript publicized June 20, 2017.

[†]The authors are with Northwestern Polytechnical University, Xi'an, China.
a) E-mail: funingg@163.com

DOI: 10.1587/transinf.2016EDP7400

fragments of behaviors. In Sect. 7, we present the QPi model of the ABP (Alternating Bit Protocol) communication protocol, where the effectiveness of the calculus is illustrated through the analysis of the QELTS of the system. In Sect. 8 we discuss related work. Finally, we conclude with Sect. 9.

2. Q-Algebra

In order to construct a calculus for modeling service computing system behaviors and trustworthiness properties, we need a method to describe the trustworthiness features of the services and their activities. The trustworthiness of service entities is a measure of its non-functional properties along multiple dimensions, such as reliability, security, scalability, performance (in terms of, for example, response time), and so on. Different aspects of trustworthiness properties may belong to different domain types; for instance, the real-time performance of a service can be described as a real number belonging to R+. The security property may be a set of permissions that describe whether or not it is permitted to perform an action. The Q-Algebra proposed in [6] allows the description of multi-dimensional trustworthiness properties in a uniform manner. Q-algebra is an extension of the constraint semiring, which adds a new multiplicative operator to the constraint semiring. First, we introduce the basic concepts of the constraint semiring [7].

Definition 2.1 (*c-semiring*). A *c-semiring* is a tuple $S = <A, +, \times, \mathbf{0}, \mathbf{1}>$ such that

- A is a set and $\mathbf{0}$, $\mathbf{1} \in A$;
- $+$, called the additive operation, is a commutative, associative and idempotent operation such that $\mathbf{0}$ is its unit element and $\mathbf{1}$ is its absorbing element;
- $\times$, called the multiplicative operation, is an associative, commutative operation such that $\mathbf{1}$ is its unit element and $\mathbf{0}$ is its absorbing element;
- $\times$ distributes over $+$.

We call $+$ the additive operator and $\times$ the multiplicative operator, while A is called the domain set of the c-semiring S. The additive operation is used to compare the elements in the set A, whereas the multiplicative operation is used to describe the composition of two elements. 0 is the minimum or 'worst' element of the set A, while 1 is the maximum or 'best' element of the set A. The additive operation can be considered as a "selection" operation. It suggests that $a + b$ means to select the better element between a and b. A partial order $\leq$ can be defined based on the $+$ operation on set A, such that $a_1 \leq a_2$ holds if and only if $a_1 + a_2 = a_2$ holds. That is to say, two trustworthiness tuples can be compared based on the relation $\leq$, if and only if one of them can be obtained by applying the $+$ operation on them. The partial order $\leq$ is used to classify the elements in set A according to their "qualities". Actually, the additive operation, when applied on two elements, always yields the lower up boundary of these elements as a result. A detailed discussion of c-semiring properties can be found in [7].

When modeling a service computing system using process algebra, there will be two patterns of composition between processes, the sequential and the concurrent. Q-algebra adds a new multiplicative operator to the constraint semiring and defines the different composition operations of trustworthiness values with respect to sequential and concurrent execution. In this section, we present some of the basic concepts Q-algebra.

Definition 2.2 A *Q-algebra* is an algebraic structure $R = <C, +, \times, \odot, 0, 1>$ such that $R_\times = <C, +, \times, 0, 1>$ and $R_\odot = <C, +, \odot, 0, 1>$ are both *c-semirings*. C is a set of trustworthiness values which is called the domain of R.

The $+$ operator bears the same meaning as in the definition of c-semiring, and is used to compare or select the trustworthiness values. The $\times$ operator is used to compute the trustworthiness of a service which is composed of two services executed in sequence, while the $\odot$ operator is used to compute the trustworthiness of a service which consists of two concurrently executed services. Different Q-algebras are introduced for the description of different trustworthiness dimensions. For example, the Q-algebra $C_{rt} = <R_+ \cup \{+\infty\}, min, +, max, +\infty, 0>$ can be used to describe the real-time property of service. Specifically, the real-time property of a service performing an activity can be defined as a non-negative real number in R_+, where $+\infty$ is the worst possible value and 0 is the best value for the domain. The *min* operator indicates that when two execution times are compared, the shorter one is better. The $+$ operator means that when two activities are executed sequentially, the execution time of the composed service is the sum of the duration of each activity. The *max* operation means that when two activities are executed concurrently, the execution time of the composed service is equal to the duration of the longer activity. The Q-algebra $<2^N, \cup, \cap, \cap, \phi, N>$ can be used to describe the security property of services, such as whether a service has a set of permissions to perform an operation.

Several Q-algebras can be composed through the product operation:

Definition 2.3 The product of n given Q-algebras $R_i = <C_i, +_i, \times_i, \odot_i, 0_i, 1_i>$, $(i = 1, \ldots, n)$, is an algebraic structure $Comp(R_1, \ldots, R_n) = <C, +, \times, \odot, 0, 1>$ defined by:

- $C = C_1 \times C_2 \times \ldots C_n$
- $(a_1, a_2, \ldots, a_n) + (b_1, b_2, \ldots, b_n) = (a_1 +_1 b_1, a_2 +_2 b_2, \ldots, a_n +_n b_n)$
- $(a_1, a_2, \ldots, a_n) \times (b_1, b_2, \ldots, b_n) = (a_1 \times_1 b_1, a_2 \times_2 b_2, \ldots, a_n \times_n b_n)$
- $(a_1, a_2, \ldots, a_n) \odot (b_1, b_2, \ldots, b_n) = (a_1 \odot_1 b_1, a_2 \odot_2 b_2, \ldots, a_n \odot_n b_n)$
- $0 = (0_1, 0_2, \ldots, 0_n)$
- $1 = (1_1, 1_2, \ldots, 1_n)$

Theorem 2.1 Given n Q-algebras $R_i = <C_i, +_i, \times_i, \odot_i, 0_i, 1_i>$, $(i = 1, \ldots, n)$, the structure $Comp(R_1, \ldots, R_n)$ is a Q-algebra.

The proof of Theorem 2.1 is omitted and can be found in [8]

In light of the above mentioned Q-algebra features, we can describe the trustworthiness of services and their activi-

ties. We introduce different Q-algebras for the descriptions of different aspects of service trustworthiness. The trustworthiness property of a specific dimension is described as an element in the domain set. The $\times$ operation describes the trustworthiness of sequential compositions of services for a specific dimension, whereas the $\odot$ operation does so for concurrent compositions. Several single dimensional Q-algebras can be combined into a multi-dimensional Q-algebra to obtain a unified description of the trustworthiness of a service.

In order to describe the meaning of an element in a specific position of a trustworthiness tuple, to each element of the tuple we attach a label. This definition allows us to write values such as (*responsetime*:0.5, *availability*:0.99) for the description of the real-time and availability properties rather than (0.5, 0.99). The Q-algebras with this form are called labeled Q-algebras.

Definition 2.4 Suppose for all $1 \leq i \leq n$ and with $R_i = <C_i, +_i, \times_i, \odot_i, 0_i, 1_i>$ being a Q-algebra, a distinct label l_i is associated with each R_i, where $i \neq j$ if $l_i \neq l_j$. Then, $R = <C, +, \times, \odot, 0, 1>$ is a labeled Q-algebra if

- $C = (\{l_1\} \times C_1) \times \ldots \times (\{l_n\} \times C_n)$
- $0 = (l_1{:}0_1, \ldots, l_n{:}0_n)$
- $1 = (l_1{:}1_1, \ldots, l_n{:}1_n)$
- $(l_1{:}c_1, \ldots, l_n{:}c_n) + (l_1{:}c_1', \ldots, l_n{:}c_n') = (l_1{:}c_1 +_1 c_1', \ldots, l_n{:}c_n +_n c_n')$
- $(l_1{:}c_1, \ldots, l_n{:}c_n) \times (l_1{:}c_1', \ldots, l_n{:}c_n') = (l_1{:}c_1 \times_1 c_1', \ldots, l_n{:}c_n \times_n c_n')$
- $(l_1{:}c_1, \ldots, l_n{:}c_n) \odot (l_1{:}c_1', \ldots, l_n{:}c_n') = (l_1{:}c_1 \odot_1 c_1', \ldots, l_n{:}c_n \odot_n c_n')$

3. Quantified Pi Calculus

When defining the quantified pi-calculus, we endeavor to combine the features of Q-algebra and those of pi-calculus. The aim is to extend pi-calculus using Q-algebra to obtain a calculus that allow us to model both the (functional) behavior and (non-functional) trustworthiness properties of a composed system. Each action in pi-calculus is associated with a trustworthiness tuple, which describes the trustworthiness property of the action, such like real-time performance, cost, probability, etc. In QPi, we use the term activity instead of the usual process algebra concept of an action (like in CCS). The syntax of QPi is defined as follows.

In this paper, we adopt the following conventions:

N is an infinite set of names, ranged over by u, v, w, x, y, z, etc. $\mathcal{K}$ is a set of process identifiers, ranged over by A, B, C.... P, Q, R..., representing a process or an expression of a process. $\mathcal{A} = \{\bar{x}y, x(y), \tau \mid x, y \in N\}$ is the set of all actions. $\bar{x}y$ and $x(y)$ denote input and output actions respectively, while τ is a special name which denotes internal actions. $\mathbb{Q} = <\mathbb{C}, +, \times, \odot, 0, 1>$ is the Q-algebra for describing the trustworthiness domain. $\mathbb{C}$ is a set of trustworthiness values which is called the domain of $\mathbb{Q}$. An activity set in *QPi* calculus is defined as $Act := \mathcal{A} \times \mathbb{C}$, ranged over by α, β, γ. Thus, each activity α is defined as a pair $<a, c>$ where $a := \bar{x}<y> \mid x(y) \mid \tau$ is the action and $c \in \mathbb{C}$ is the activity trustworthiness value that belongs to the trustworthiness domain $\mathbb{C}$.

The set of processes is defined as follows:

$$P ::= \mathbf{0} \mid \alpha.P \mid P_1 + P_2 \mid P_1 \mid P_2 \mid (x)P \mid [x = y]P \mid A(y_1, \ldots, y_n)$$

We consider the following language constructs and their intended interpretations in some detail:

- The empty agent 0, which cannot perform any actions;
- Prefix $\alpha.P$. Prefixes are the basic mechanism through which the behaviors of services are constructed. $\alpha.P$ refers to a communication (or silent) action with a corresponding trustworthiness value. The service subsequently behaves as service P. Each $\alpha \in Act$ is an activity which has the form as (a, c).
  Here $a ::= \bar{x}y \mid x(y) \mid \tau$ is an action, while c is a value that describes the trustworthiness of the action.
  $(\bar{x}y, c)$ is called a negative prefix. A name x may be thought of as an output port of an process which contains it; $(\bar{x}y, c).P$ outputs the name y at port x and then behaves like P.
  $(x(y), c)$ is called a positive prefix. $(x(y), c).P$ inputs an arbitrary name z at port y and then behaves like $P\{z/y\}$. The trustworthiness of action $x(y)$ is c. The name y is bound by the positive prefix $(x(y), c)$.
  (τ, c) is called a silent prefix. $(\tau, c).P$ performs the silent action τ and then behaves like P. τ is an internal action which is unobservable outside the process.
  $P\{z/y\}$ denotes the substitution of z for all free occurrences of y in P;
- A summation $P_1 + P_2$. This represents a process which may behave either as P_1 or as P_2. It allows all the current activities of P_1 and all the current activities of P_2;
- $P_1 \mid P_2$ denotes the concurrent behavior of P_1 and P_2, where P_1 and P_2 synchronize on complementary activities. For each new activity obtained as a combination of its component process' activities, the corresponding trustworthiness value is computed from the values of its constituents using the $\odot$ operator of the corresponding Q-algebra;
- A restriction $(x)P$. This agent behaves as P but the name x is local, meaning it cannot be immediately used as a communication port between P and its environment. However, it can be used for communication between components within P;
- A match $[x = y]P$. This agent behaves like P if the names x and y are identical; otherwise it does nothing;
- A defined agent $A(y_1, \ldots, y_n)$. For any agent identifier A (with arity n) used thus, there must be a unique defining equation $A(x_1, x_2, \ldots, x_n) ::= P$, where the names $x_1, x_2, \ldots, x_n$ are pairwise distinct and are the only names which may occur free in P. Then $A(y_1, y_2, \ldots, y_n)$ behaves like $P\{y_1/x_1, \ldots, y_n/x_n\}$. Defining equations in this manner allows us to perform recursion, since P may contain

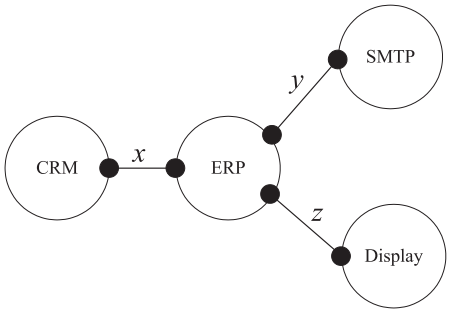


Fig. 1 QPi-calculus flow chart of a business process

any agent identifier, even A itself.

For each agent in one of the forms $(x(y), c).P$ and $(y)P$, the occurrence of y with in parentheses is a binding occurrence, and in each case the scope of the occurrence is P. An occurrence of y in an agent is said to be free if it does not lie within the scope of a binding occurrence of y. The set of free names occurring in P is denoted as $fn(P)$. The set of bound names occurring in P is denoted by $bn(P)$. The set $fn(P) \cup bn(P)$ of all names occurring in P is denoted as $n(P)$. We then have the following properties:

$fn(0) = \emptyset$; $fn(P \mid Q) = fn(P) \cup fn(Q)$; $fn(P + Q) = fn(P) \cup fn(Q)$; $fn((\tau, c).P) = fn(P)$;

$fn((\bar{x}y, c).P) = \{x, y\} \cup fn(P)$; For output, where x, y are free names

$fn((x(y), c).P) = \{x\} \cup (fn(P) - \{y\})$; For input, where x is free name while y is bound name.

$fn((x).P) = fn(P) - \{x\}$; For restriction, where x is not a free name.

This intuitive explanation can be elaborated in terms of the structural operational semantics discussed in Sect. 4, which define a labelled transition system for QPi processes. Before that, we present a simple example, illustrating how the language may be used to describe a system in terms of both behavior and trustworthiness.

As shown in Fig. 1, the *CRM* service fetches purchase orders from a *CRM* application system and sends them to an *ERP* service. When it receives a purchase order, the *ERP* service makes a judgment according the content of the order. If the purchase order is approved it will be forwarded to the *SMTP* service, which will send a confirmation e-mail to the customer. Otherwise, the rejected order will be displayed on an employee's terminal via a display component. Historical statistical information available to the system indicates that there is a 90% probability that orders sent to the *ERP* service will be approved. The services and system are thus defined as:

CRM: let $N_c = \{x, PO\}$,
$CRM(N_c) = (\bar{x}\langle PO\rangle, (prob{:}1, time{:}1)).CRM(N_c)$
ERP: let $N_e = \{x, y, z, PO, Acc, Rej\}$,
$ERP(N_e) = (x(Msg), (prob{:}1, time{:}1)).[Msg = PO]((\bar{y}\langle Acc\rangle, (prob{:}0.9, time{:}1)).ERP(N_e) + (\bar{z}\langle Rej\rangle, (prob{:}0.1, time{:}1)).ERP(N_e))$
SMTP: let $N_s = \{y, Acc\}$, $SMTP(N_s) = (y(Msg), (prob{:}1, time{:}1)).[Msg = Acc](\tau, (prob{:}1, time{:}1)).SMTP(N_s)$
Display: let $N_d = \{z, Rej\}$,
$Display(N_d) = (z(Msg), (prob{:}1, time{:}1)).[Msg = Rej](\tau, (prob{:}1, time{:}1)).Display(N_d)$

Let $N_P = \{x, y, z, PO, Acc, Rej\}$, in which case the whole business process is defined as: $Process(N_P) = CRM(N_c) \mid ERP(N_e) \mid SMTP(N_s) \mid Display(N_d)$

Each action in *QPi* is associated with a value which describes its trustworthiness property. The given example describes the trustworthiness property of a business process from the aspects of activity occurrence probability and execution time. The *CRM* service outputs message *PO* via channel x with a probability of 100% and requires one time unit for this. The *ERP* service approves orders and sends confirmation e-mails with a probability 90%, while the probability of an order refusal occurrence is 10%.

4. Operational Semantics

The semantics of QPi are defined through a quality extended labelled transition system (QELTS). A QELTS is like a general labelled transition system, but the transitions are labelled using a tuple containing the action a and its corresponding trustworthiness value c, which describes the behavior of the services and takes trustworthiness into account. The transition form in *QPi* is $P \xrightarrow{(a,c)} Q$. Here P and Q are process expressions and $(a, c) \in Act$ is an element in an activity set. $P \xrightarrow{(a,c)} Q$ means that P does activity (a, c) and then evolves to Q. $a := \bar{x}y \mid x(y) \mid \tau$ describes the action of the activity and c describes the trustworthiness of the action.

Definition 4.1 A Quality Extended Labelled Transition System is defined as a tuple $QLTS = \langle S, s_0, \mathcal{L}, \rightarrow\rangle$, where

- S is a set of states,
- $s_0 \in S$ is the initial state,
- $\mathcal{L} \subseteq Act$ is a set of labels denoting activities,
- $\rightarrow \subseteq (S \times Act \times S)$ is a set of labelled transitions. Each transition is a quadruple consisting of a source state s, an action a, a trustworthiness value c, and a target state s' and it is denoted as $s \xrightarrow{(a,c)} s'$. Intuitively, a transition means that the transitioning system, when in a current state s, can change its state to s' by performing action a with trustworthiness value c.

Definition 4.2 The semantics for a QPi service P are given by a $QELTS(E^{QPi}, P, Act, \rightarrow)$ where E^{QPi} is a set of *QPi* expressions and the transition relation $\rightarrow \subseteq (E^{QPi} \times Act \times E^{QPi})$ is the least relation satisfying the rules in Table 1.

The operational semantics listed in Table 1 give the evolution rules of the QPi calculus process. Each activity completion brings about a transition in the system, and each activity has its own trustworthiness property and this property is represented explicitly in the transition.

The evolution of a system modeled in QPi is represented by a path through the corresponding QELTS$(S, s_0, \mathcal{L}, \rightarrow)$. A path ω is a non-empty sequence of the form $s_1 \xrightarrow{(a_1,c_1)} s_2 \xrightarrow{(a_2,c_2)} s_3 \ldots$ where $s_i \in S$,

Table 1 Operational semantics of *QPi* calculus

TAU-ACT: $\frac{}{(\tau,c).P \xrightarrow{(\tau,c)} P}$	OUTPUT-ACT: $\frac{}{(\bar{x}y,c).P \xrightarrow{(\bar{x}y,c)} P}$
INPUT-ACT: $\frac{}{(x(z),c)P \xrightarrow{(x(w),c)} P\{w/z\}}\ w \notin \mathrm{fn}((z)P)$	
SUM: $\frac{P \xrightarrow{(\alpha,c)} P'}{P+Q \xrightarrow{(\alpha,c)} P'}$	MATCH: $\frac{P \xrightarrow{(\alpha,c)} P'}{[x=x]P \xrightarrow{(\alpha,c)} P'}$
IDE: $\frac{P\{\tilde{y}/\tilde{x}\} \xrightarrow{(\alpha,c)} P'}{A(\tilde{y}) \xrightarrow{(\alpha,c)} P'}\quad A(\tilde{x}) \overset{def}{=} P$	
PAR: $\frac{P \xrightarrow{(\alpha,c)} P'}{P \mid Q \xrightarrow{(\alpha,c)} P' \mid Q}\ \mathrm{bn}(\alpha)\cap\mathrm{fn}(Q)=\emptyset$	
COM: $\frac{P \xrightarrow{(\bar{x}y,c1)} P' \quad Q \xrightarrow{(x(z),c2)} Q'}{P \mid Q \xrightarrow{(\tau,c1\odot c2)} P' \mid Q'\{y/z\}}$	CLOSE: $\frac{P \xrightarrow{(\bar{x}(w),c1)} P' \quad Q \xrightarrow{(x(w),c2)} Q'}{P \mid Q \xrightarrow{(\tau,c1\otimes c2)} (w)(P' \mid Q')}$
RES: $\frac{P \xrightarrow{(\alpha,c)} P'}{(y)P \xrightarrow{(\alpha,c)} (y)P'}\ y \notin \mathrm{n}(\alpha)$	OPEN: $\frac{P \xrightarrow{(\bar{x}y,c)} P'}{(y)P \xrightarrow{(\bar{x}w,c)} P'\{w/y\}}\ y\neq x,\ w \notin \mathrm{fn}((y)P')$

$(a_i, c_i) \in \mathcal{L}$ for all $i \geq 1$. $\omega(i)$ denotes the i-th state of a path ω. $Path(s)$ denotes the set of all paths beginning at state s. The length $l(\omega)$ of a finite path $\omega = s_1 \xrightarrow{(a_1,c_1)} s_2 \xrightarrow{(a_2,c_2)} s_3 \ldots s_n \xrightarrow{(a_n,c_n)} s_{n+1}$ is defined as the number of transitions n, while its trustworthiness is defined as $q(\omega) = c_1 \odot c_2 \odot \cdots \odot c_n$. Correspondingly, the length of an infinite path $\omega = s_1 \xrightarrow{(a_1,c_1)} s_2 \xrightarrow{(a_2,c_2)} s_3 \ldots$ is $l(\omega) = \infty$. Furthermore, for an arbitrary path $\omega = s_1 \xrightarrow{(a_1,c_1)} s_2 \xrightarrow{(a_2,c_2)} s_3 \ldots$, if $l(\omega) > n$, then we define $head_n(\omega)$ to be the finite path $\omega = s_1 \xrightarrow{(a_1,c_1)} s_2 \xrightarrow{(a_2,c_2)} s_3 \ldots s_n \xrightarrow{(a_n,c_n)} s_{n+1}$ which consists of the first n transitions of ω, and $tail_n(\omega)$ as the remainder of ω, i.e., the sub-path of ω starting at s_{n+1}.

5. Quantified Bisimulation

Bisimulation theory is used to study if two concurrent processes have the same behavior and forms an important part of pi-calculus theory. When introducing the trustworthiness property into pi-calculus, we are not only interested in assessing whether or not two concurrent systems have the same behavior, but we are concerned with the degree to which the two systems can simulate each other, and to which degree the two systems differ in their trustworthiness property. For these reasons, we introduce the concept of quantified bisimulation, which allows us to assess the degree of equivalence between services, and which is very suitable for determining whether a service can replace another without any reduction in trustworthiness. We first consider strong bisimulation, which is often defined as follows:

Definition 5.1 A binary relation $\mathfrak{R} \subseteq S \times S$ is a bisimulation for $QELTS(S, s_0, \mathcal{L}, \rightarrow)$ if $P\mathfrak{R}Q$ implies:

1. if $P \xrightarrow{(a,c)} P'$, and a is a free action, then for some Q', $Q \xrightarrow{(a,c)} Q'$ and $P'\mathfrak{R}Q'$
2. if $P \xrightarrow{(x(y),c)} P'$, and $y \notin \mathrm{n}(P,Q)$, then for some Q', $Q \xrightarrow{(x(y),c)} Q'$ and for all w, $P'\{w/y\}\mathfrak{R}Q'\{w/y\}$
3. if $P \xrightarrow{(\bar{x}(y),c)} P'$ and $y \notin \mathrm{n}(P,Q)$, then for some Q', $Q \xrightarrow{(\bar{x}(y),c)} Q'$ and $P'\mathfrak{R}Q'$

Definition 5.1 is an extension of the pi-calculus strong bisimulation definition found in [5]. We see that this definition matches two transitions only when the actions are same and the corresponding trustworthiness values are identical. In fact, this is not a robust relation for considering trustworthiness aspects. For example, services that differ by a very small amount in terms of execution time would be considered just as different as services that exhibit completely different behavior. To find a more suitable approach to differentiate services according to both behavior and trustworthiness aspects, we borrow from mathematics the concept of metrics and define a quality metric space of activities as follows:

Definition 5.2 A quality metric space on activities is a pair $<Act, \rho>$, in which $Act \subseteq \mathcal{A} \times \mathbb{C}$ is a set of activities and ρ is a mapping from $Act \times Act$ to $[0, \infty]$ such that

- $\rho((a,c),(b,c')) = \infty$ *if* $a \neq b$,
- $\rho((a,c),(a,c')) = 0$ *iff* $c = c'$,
- $\rho((a,c),(a,c')) = \rho((a,c'),(a,c))$
- For any a, b, $d \in \mathcal{A}$, and c, c', $c'' \in \mathbb{C}$, $\rho((a,c),(b,c')) \leq \rho((a,c),(d,c'')) + \rho((d,c''),(b,c'))$

Quality metric provides a quantitative measure of the

difference between two services, which not only allows the assessment of behavioral equivalence, but also the similarity of trustworthiness. If the quality metric between two activities is positive infinity, this means that the two activities have different actions and are not similar. If ρ is a positive integer, then the two activities have the same action, but their trustworthiness differs. The smaller ρ is, the closer the trustworthiness features of the two services are. A value of ρ equal to 0 means that two activities are fully consistent in both behavior and trustworthiness features. The following definition introduces the concept of distance between services.

Definition 5.3 For a given $QELTS(S, s_0, \mathcal{L}, \rightarrow)$, $R \subseteq S \times S$ and a quality metric ρ on $\mathcal{L}$, we have $d_R(t, s, t', \alpha) = glb\{\rho(\alpha,\beta) \mid \beta \in \mathcal{L}$, where α, β satisfy one of the following three conditions$\}$

1. α is a free action, $t \xrightarrow{\alpha} t'$, $\exists s' \in S$ *s.t.* $s \xrightarrow{\beta} s'$ and $t'Rs'$
2. α in a $x(y)$ form, $t \xrightarrow{(x(y),c)} t'$ and $y \notin \mathrm{n}(t, s)$, $\exists s' \in S$ *s.t.* $s \xrightarrow{(x(y),c')} s'$, β is $(x(y), c')$ and for all w, $t'\{w/y\}Rs'\{w/y\}$
3. α in a $\bar{x}(y)$ form, $t \xrightarrow{(\bar{x}(y),c)} t'$ and $y \notin \mathrm{n}(t, s)$, $\exists s' \in S$ *s.t.* $s \xrightarrow{(\bar{x}(y),c')} s'$, β is $(\bar{x}(y), c')$ and $t'Rs'$

Further, for any t, s, $t' \in S$ and $\alpha \in \mathcal{L}$, $d_R(t, s) = lub\{d_R(t, s, t', \alpha) \mid t, s, t' \in S, \alpha \in \mathcal{L}$, and $t \xrightarrow{\alpha} t'\}$

Finally, $d_R = lub\{max\{d_R(s_1, s_2), {d_R}^{-1}(s_1, s_2)\} \mid s_1 R s_2\}$ is called the bisimulation distance of R.

In Definition5.3, *glb* refers to greatest lower bound, *lub* refers to smallest upper bound. Intuitively, for given $t, s, t' \in S$, tRs, $t \xrightarrow{\alpha} t'$, $d_R(t, s, t', \alpha)$ is the greatest lower bound for the distances between α and β, where $t \xrightarrow{\alpha} t'$, $s \xrightarrow{\beta} s'$ and $t'Rs'$. $d_R(t, s)$ describes the degree of equivalence between process s and t. It is determined through the successive investigation of each $t \xrightarrow{\alpha} t'$ transition of process t, and by selecting the maximum of $d_R(t, s, t', \alpha)$ as $d_R(t, s)$.

Furthermore, for a given $<s_1, s_2> \in R$, $d_R(s_1, s_2)$ and ${d_R}^{-1}(s_1, s_2)$ describe the degree of equivalence between s_1 and s_2. We choose $max\{d_R(s_1, s_2), {d_R}^{-1}(s_1, s_2)\}$ as the value of the degree to which s_1 and s_2 simulate each other. For all $s_1 R s_2$, $lub\{max\{d_R(s_1, s_2), {d_R}^{-1}(s_1, s_2)\} \mid s_1 R s_2\}$ is the maximum equivalence distance for tuples in R. Therefore, d_R describes the degree to which R is a bisimulation.

$d_R = \infty$ means that at least one pair of processes in R cannot simulate each other's behavior. If d_R is a constant, then all pairs of processes are able to simulate each other's behavior, but the trustworthiness values of the behaviors are different. The smaller d_R is, the more similar the trustworthiness of the processes are. d_R =0 means all pairs of processes in R can fully simulate each other's behavior, and the corresponding trustworthiness values of the behaviors are exactly the same.

Proposition 5.1 The bisimulation distance satisfies the following properties:

1. *If R is a bisimulation then we have $d_R = 0$.*
2. $d_R = {d_R}^{-1}$
3. $d_{R_1 \circ R_2} \leq d_{R_1} + d_{R_2}$
4. $d_{\cup_i R_i} \leq lub_i d_{R_i}$

Proof: Properties 1 and 2 are immediately evident from the definition of bisimulation distance. For property 3, if $d_{R1} = \infty$ or $d_{R2} = \infty$, the conclusion is clearly satisfied. We assume that $d_{R1} < \infty$ and $d_{R2} < \infty$. For any s_1, $s_3 \in S$, if $s_1 R_1 \circ R_2 s_3$, then there exists $s_2 \in S$ with $s_1 R_1 s_2$ and $s_2 R_2 s_3$. For any $s_1' \in S$ and $\alpha \in Act$, if $s_1 \xrightarrow{\alpha} s_1'$ then from $d_{R1} < \infty$ we have $s_2' \in S$ and $\gamma \in Act$ such that $s_2 \xrightarrow{\beta} s_2'$, $s_1' R_1 s_2'$, and $\rho(\alpha,\beta) \leq d_{R1}$. Similarly, from $d_{R2} < \infty$, we have $s_3' \in S$ and $\gamma \in Act$ such that $s_3 \xrightarrow{\gamma} s_3'$, $s_2' R_2 s_3'$, and $\rho(\beta,\gamma) \leq d_{R2}$. Therefore, $s_1' R_1 \circ R_2 s_3'$, $\rho(\alpha,\gamma) \leq \rho(\alpha,\beta) + \rho(\beta,\gamma) \leq d_{R1} + d_{R2}$, $d_{R1 \circ R2} \leq d_{R1} + d_{R2}$. Property 4 can be proved in a similar manner.

The first property indicates that the bisimulation distance of a bisimulation is 0. Property 2 says that the bisimulation distance of a relation and that of its inverse are the same, while property 3 means that the bisimulation distance of the composition of two relations is not greater than the sum of the corresponding distances of the two relations. Property 4 means that if the degree to which R_i is a bisimulation is equal or greater than some value for all i, then the degree to which $\cup_i R_i$ is a bisimulation is equal or greater than this value.

Since each relation between states is assigned a bisimulation distance, all relations can be classified according to their distances.

Definition 5.4 For a given $QELTS(S, s_0, \mathcal{L}, \rightarrow)$, $R \subseteq S \times S$, and $\lambda \in [0, \infty]$, if $d_R \leq \lambda$, then R is called a λ-bisimulation.

λ is a real number in $[0, \infty]$. The relation R is a λ-bisimulation means $d_R \leq \lambda$. This denotes that the trustworthiness distances of the tuples of relation R are not more than λ.

From a quantitative perspective, λ describes the similarity of the state tuples in relation R. Obviously, every relation R is a ∞-bisimulation. If R is a λ-bisimulation and $\lambda \leq \mu$, then it is also a μ-bisimulation. If R is a λ_i-bisimulation for $i \in I$, then R is a $glb_{i \in I}\lambda_i$-bisimulation.

The following properties can be easily derived from Proposition 5.1 [8].

Proposition 5.2 For λ-bisimulation, we have

1. If R is a bisimulation, then R is a 0-bisimulation.
2. R is a λ-bisimulation *iff* R^{-1} is a λ-bisimulation.
3. If R_i is a λ_i-bisimulation for $i = 1, 2$, then $R_1 \circ R_2$ is a $(\lambda_1 + \lambda_2)$-bisimulation.
4. If R_i is a λ_i-bisimulation, then $\cup_i R_i$ is a $max_i\{\lambda_i\}$-bisimulation.

Proof: If R is a bisimulation then we have $d_R = 0$, so R is a 0-bisimulation. If R is a λ-bisimulation then we have $d_R \leq \lambda$. Since $d_R = {d_R}^{-1}$, we have ${d_R}^{-1} \leq \lambda$, so R^{-1} is a λ-bisimulation. Also, $d_{R_1 \circ R_2} \leq d_{R_1} + d_{R_2}$ and $d_{R_1} + d_{R_2} = (\lambda_1 + \lambda_2)$, so we have $d_{R_1 \circ R_2} \leq \lambda_1 + \lambda_2$, and $R_1 \circ R_2$ is a $(\lambda_1 + \lambda_2)$-bisimulation. From Proposition 5.1 we have $d_{\cup_i R_i} \leq lub_i d_{R_i}$, which gives us $\cup_i R_i$ is a $max_i\{\lambda_i\}-$ bisimulation.

Based on the concept of the λ-bisimulation, we are able to define the concept of λ-bisimilarity, which is the largest λ-bisimulation.

Definition 5.5 For any $\lambda \in [0, \infty]$, λ-bisimilarity is defined as $\sim_\lambda = \bigcup_{R \text{ is } \lambda\text{-bisimulation}} R$

In other words, two states s_1, $s_2 \in S$ are said to be λ-bisimilar whenever there exists a λ-bisimulation containing the pair (s_1, s_2).

Proposition 5.3 λ-bisimilarity satisfies:

1. $\backsim \subseteq \backsim_{\mathbf{0}}$
2. If $\lambda_1 \leq \lambda_2$, then $\backsim_{\lambda 1} \subseteq \backsim_{\lambda 2}$
3. For any $\lambda \in [0, \infty]$, $\backsim_\lambda$ is a λ-bisimulation, and it is reflexive and symmetric
4. $\backsim_{\lambda 1} \circ \backsim_{\lambda 2} \subseteq \backsim_{\lambda 1 + \lambda 2}$

Proof: Proposition 5.3 can be derived from proposition 5.2 and definitions 5.4 and 5.5 [8].

$\backsim$ refers to strong bisimulation, so $d_{\backsim} = 0$, which means that $\backsim$ is a 0-bisimulation, and consequently $\backsim \subseteq \backsim_{\mathbf{0}}$. The meaning of property 2 is that if $\lambda_1 \leq \lambda_2$ then the largest λ_1-bisimulation relation set will be included in the largest λ_2-bisimulation set. Property 3's meaning is that even for the largest λ-bisimulation relation, distances between processes of its tuples are less than or equal to λ, and the relation set is reflexive and symmetric. Property 4 means that for the relation $\backsim_{\lambda 1} \circ \backsim_{\lambda 2}$ composed by the largest λ_1-bisimulation and λ_2-bisimulation, its simulation distance will always less than or equal to $\lambda_1 + \lambda_2$.

Proposition 5.4 Let $P_1 \sim_\lambda P_2$. Then

1. $\alpha_1.P_1 \sim_{\boldsymbol{max}\{\lambda, \rho(\alpha 1, \alpha 2)\}} \alpha_2.P_2$
2. $(x)P_1 \sim_\lambda (x)P_2$, $x \notin fn(P_1, P_2) \,|\, i \in I$
3. *Let* $P_{1i} \sim_{\lambda i} P_{2i}$ for $i \in I$,

$$\text{then } \left(\sum_{i \in I} P_{1i}\right) \sim max\{\lambda_i \,|\, i \in I\} \left(\sum_{i \in I} P_{2i}\right)$$

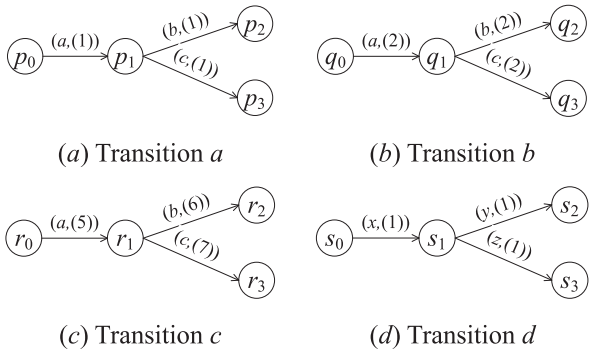
The proof is omitted due to its simplicity.

The meaning of property 1 is that if the bisimulation distance between P_1 and P_2 is not greater than λ, then the bisimulation distance between processes $\alpha_1.P_1$ and $\alpha_2.P_2$ is not greater than either λ or $\rho(\alpha_1, \alpha_2)$. Property 2 means name restrictions applied on two process will not affect their bisimilarity. Property 3 means that if the bisimulation distance between P_{1i} is P_{2i} not greater than λ_i then the bisimulation distance between $(\sum_{i \in I} P_{1i})$ and $(\sum_{i \in I} P_{2i})$ is not greater than $max\{\lambda_i \,|\, i \in I\}$.

For a detailed discussion of the above theorems and properties, the reader is referred to [8].

The concept of bisimulation distance is suitable for comparing services in terms of both behavior and trustworthiness. In practical applications, for a certain aspect of trustworthiness, a specific bisimulation distance function can be introduced to scale similar levels of trustworthiness properties. For example, for the Q-algebra for execution time $<R_+ \cup \{\infty\}, min, +, max, \infty, 0>$, the quality metric can be defined as:

- $\rho((a, t), (b, t')) = \infty$ if $a \neq b$
- $\rho((a, t), (b, t')) = |t - t'|$



(a) Transition a (b) Transition b

(c) Transition c (d) Transition d

Fig. 2 Example of bisimulation

Note that the distance defined by such a metric is ∞ for two activities with different actions. For activities with same action, the distance is exactly the difference between their execution times. In general, we can have a simple characterization of λ–bisimulation in QPi, which means that in a λ–bisimulation, an activity must be simulated by another activity with identical action, but its trustworthiness value may be matched by another approximate value. Two processes of QPi satisfying λ–bisimulation have the same actions and their trustworthiness values are approximate to the extent of a distance smaller than or equal to λ.

The LTS shown in Fig. 2 describes a system's behavior, and the corresponding trustworthiness values are labeled with values for each action. Considering (p_1, s_1) in relation $R_1 = \{(p_0, s_0), (p_1, s_1), (p_2, s_2), (p_3, s_3)\}$, because p_1, s_1 have different actions and cannot simulate each other's behavior, we have $d_{R1} = \infty$. For relation $R_2 = \{(p_0, q_0), (p_1, q_1), (p_2, q_2), (p_3, q_3)\}$, let the bisimulation distance function be $\rho = |x_1 - x_2|$. In this case, the compared states of tuples in R_2 can simulate each other's actions, but their trustworthiness values are different. Using the bisimulation distance function, we obtain $d_{R2} = 1$. For relation $R_3 = \{(p_0, r_0), (p_1, r_1), (p_2, r_2), (p_3, r_3)\}$, $d_{R3} = 6$. $d_{R2} < d_{R3}$ means that the activities of the states in R_2 are more similar than the activities of the states in R_3. In Fig. 2, from the perspective of behavior, (a), (b) and (c) are same, while from the perspective of trustworthiness, the distance between (b) and (a) is smaller than that between (c) and (a). Therefore (b) is a more similar LTS to (a) than (c). This observation is certainly consistent with our intuition.

6. QCTL: A Logic for Specifying Trustworthiness

There is a great advantage in being able to verify the correctness of a service composition application. For this purpose, we introduce a specification language for describing the properties to be verified. The presented formalism is called the Quantified Computation Tree Logic (QCTL). The syntax of QCTL is defined as follows:

Definition 6.1 As for QPi, assume a set of activities Act ranged over by α. Let c be the trustworthiness parameter, range over the domain C. Let $\bowtie \in \{\leq, <, \geq, >, =\}$. The syntax of QCTL formulas is defined inductively as follows:

$\varphi ::= true \mid \varphi \wedge \varphi \mid \neg\varphi \mid [E\boldsymbol{\psi}]_{c\bowtie n} \mid [A\boldsymbol{\psi}]_{c\bowtie n}$

$\boldsymbol{\psi} ::= <\pi> \mid X\varphi \mid \varphi \cup^{c\bowtie n} \varphi \mid \varphi \cup \varphi$

Where $n \in C$ is some concrete cost value, π is an expression built by the grammar:

$\pi ::= a \mid \pi;\pi \mid \pi^* \mid \pi + \pi$

QCTL formulae are interpreted over a specific QELTS. Each atomic proposition σ must be obtained from the set used to label the states of this QELTS. QCTL formulae are classified as state formulae φ and path formulae $\boldsymbol{\psi}$, which are evaluated over states and paths respectively. A property of a QPi model to be verified will always be represented by a state formula. Path formulae can only appear as parameters of the formulae $[E\boldsymbol{\psi}]_{c\bowtie n}$ and $[A\boldsymbol{\psi}]_{c\bowtie n}$. Intuitively, a state s satisfies $[E\boldsymbol{\psi}]_{c\bowtie n}$ if there exists a path from s satisfying $\boldsymbol{\psi}$ and the trustworthiness of the path is in the domain specified by $\bowtie n$. Similarly, a state s satisfies $[A\boldsymbol{\psi}]_{c\bowtie n}$ if all the paths from s satisfy $\boldsymbol{\psi}$ and the trustworthiness of each such path is in the domain specified by $\bowtie n$.

The path formulae are standard in temporal logic except $<\pi>$. Intuitively, $X\varphi$ holds if φ is satisfied in every next state. The formula $\varphi_1 U \varphi_2$ holds over a path if φ_1 holds continuously until φ_2 holds. $\varphi_1 \cup^{c\bowtie n} \varphi_2$ is true if φ_2 is satisfied within a finite path whose trustworthiness is in the domain specified by $\bowtie n$ and φ_1 is true until that point. The expression $<\pi>$ represents a sequence of actions. The meanings of $\pi;\pi$ (i.e. sequential composition), π^* (finitely many repetitions), and $\pi + \pi$ (choice) are similar as in the case of pi-calculus. a represents a single action in Act. In the following, we define the semantics of QCTL over the QELTS.

For a given $QELTS = <S, s_0, \mathcal{L}, \rightarrow>$, state $s \in S$ and QCTL formula φ, we use the satisfaction relation $s \vDash \varphi$ to indicate that a formula φ is holds in state s. Similarly, for a path ω satisfying a path formula ψ, we write $\omega \vDash \psi$. The satisfaction relation is defined as the least relation such that for a path $\omega = s_1 \xrightarrow{(a_1,c_1)} s_2 \xrightarrow{(a_2,c_2)} s_3 \ldots$, the following hold:

$\omega \vDash <\pi> \Leftrightarrow <\pi> = a_1; a_2; \cdots$

$\omega \vDash X\varphi \Leftrightarrow s_2 \vDash \varphi$

$\omega \vDash \varphi_1 \cup^{c\bowtie n} \varphi_2 \Leftrightarrow \exists s_k.((s_k \vDash \varphi_2) \wedge (\forall j < k.s_j \vDash \varphi_1) \wedge (c_1 \times c_2 \times \cdots \times c_{k-1} \bowtie n))$

$\omega \vDash \varphi_1 \cup \varphi_2 \Leftrightarrow \exists s_k.((s_k \vDash \varphi_2) \wedge (\forall j < k.s_j \vDash \varphi_1))$

and for state $s \in S$:

$s \vDash true$ for all $s \in S$

$s \vDash \varphi_1 \wedge \varphi_2 \Leftrightarrow s \vDash \varphi_1 \wedge s \vDash \varphi_2$

$s \vDash \neg\varphi \Leftrightarrow s \nvDash \varphi$

$s \vDash [E\psi]_{c\bowtie n} \Leftrightarrow \exists \omega \in Path(s),\ k \in \mathbb{N}.tail_k(\omega) \vDash \psi \wedge cost(head_k(\omega)) \bowtie n$

$s \vDash [A\psi]_{c\bowtie n} \Leftrightarrow \forall \omega \in Path(s).(\exists k \in \mathbb{N}.tail_k(\omega) \vDash \psi \wedge cost(head_k(\omega)) \bowtie n)$

A number of additional useful operators can be derived from the basic syntax of QCTL. For example, we have the following obvious logic equivalences: $false \equiv \neg true$, $\varphi_1 \bigvee \varphi_2 \equiv \neg(\neg\varphi_1 \wedge \neg\varphi_2)$, and $\varphi_1 \Rightarrow \varphi_2 \equiv \neg(\neg\varphi_1 \bigvee \varphi_2)$. In a similar manner to temporal logic, we can also define the modalities *eventually* $\Diamond$ and *always* $\Box$. In addition, we also allow the bounded variant of the eventually operator $\Diamond c \bowtie n$. Intuitively, $\Diamond c \bowtie n\varphi$ means that φ is satisfied with a cost in a domain specified by $\bowtie n$. These modal operators can be expressed as follows:

$\Diamond \equiv true \cup \varphi$

$\Diamond_{c\bowtie n}\varphi \equiv true \cup^{c\bowtie n} \varphi$

$\Box\varphi \equiv \neg\Diamond\neg\varphi$

7. An Example Model in QPi

Taking the Alternating Bit Protocol (ABP) as example, we use our model to specify the ABP and demonstrate the descriptive ability of QPi. ABP is a simple communication protocol that provides error-free communications over a medium that may cause message loss. The description consists of four services: a sender and a receiver which are connected via two media. The structure of the communication system is shown in Fig. 3.

In this protocol, a retransmission mechanism is used to overcome the unreliability of the medium. For the sender, when the delay between issuing a message to the medium via the *in* channel and receiving an acknowledgement via the *acks* channel is too long, a retransmission is initiated. In our model, we assume that 10% of the messages are lost and time delays occur in the media.

Modeling this system requires taking the reliability and time properties into account. We introduce a labeled Q-algebra $Q_t = <R_+ \cup \{+\infty\}, min, +, max, +\infty, 0>$ to describe the trustworthiness of the time dimension and $Q_r = <[0,1], max, \cdot, \cdot, 0, 1>$ for the reliability dimension. The composition of Q_t and Q_r, denoted as $Comp(Q_t, Q_r)$, is used to describe the trustworthiness domain. The real-time and reliability dimensions are labeled by *prob* and *time*, respectively.

We now specify the system as follows:

Sender:

$S = (send(Msg), (t{:}1, p{:}1)).S_1$

$S_1 = (\overline{in}(Msg_0), (t{:}0.5, p{:}1)).((acks(Ack_0), (t{:}0.5, p{:}1)).S_2 + (\tau, (t{:}10, p{:}1)).S_1)$

$S_2 = (send(Msg), (t{:}1, p{:}1)).S_3$

$S_3 = (\overline{in}(Msg_1), (t{:}0.5, p{:}1)).((acks(Ack_1), (t{:}0.5, p{:}1)).S + (\tau, (t{:}10, p{:}1)).S_3)$

Media:

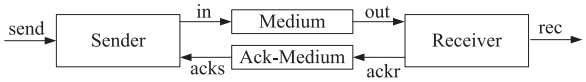


Fig. 3 Structure of ABP

$M = (in(Msg), (t{:}0.5, p{:}1)).M_1$

$M_1 = (\tau, (t{:}0.5, p{:}0.1)).M + (\tau, (t{:}0.5, p{:}0.9)).M_3$

$M_3 = (\overline{out}(Msg), (t{:}1, p{:}1)).M$

$AM = (ackr(AckMsg), (t{:}0.5, p{:}1))).AM_1$

$AM_1 = (\tau, (t{:}0.5, p{:}0.1)).AM + (\tau, (t{:}0.5, p{:}0.9)).AM_3$

$AM_3 = (\overline{acks}(AckMsg), (t{:}1, p{:}1)).AM$

Receiver:

$R = (out(Msg), (t{:}1, p{:}1)).([Msg = Msg_0](\overline{rec}(Msg),$
$(t{:}1, p{:}1)).(\overline{ackr}(Ack_0), (t{:}1, p{:}1)).R_1 +$
$[Msg = Msg_1](\overline{ackr}(Ack_1), (t{:}1, p{:}1)).R)$

$R_1 = (out(Msg), (t{:}1, p{:}1)).([Msg = Msg_1](\overline{rec}(Msg),$
$(t{:}1, p{:}1)).(\overline{ackr}(Ack_1), (t{:}1, p{:}1)).R +$
$[Msg = Msg_0](\overline{ackr}(Ack_0), (t{:}1, p{:}1)).R_1)$

The composition of these services is as follows: $ABP(send, rec) = (in, out, ackr, acks)(S \mid M \mid AM \mid R)$

From the above example, we see that by attaching to each action a trustworthiness tuple, QPi describes the behavior and trustworthiness of a system in a unified manner. Moreover, we can deduce the evolution of the system based on the optional semantics of QPi to verify whether the system satisfies some expected properties. For the sender service, the QELTS in Fig. 4 describes its behavior and states' transitions. Each transition describes an action of the service and the corresponding trustworthiness of the action. In this manner, we can analyze the behavior, states' transitions and trustworthiness of the whole communication system. Figure 5 illustrates a part of the ABP system's state transitions.

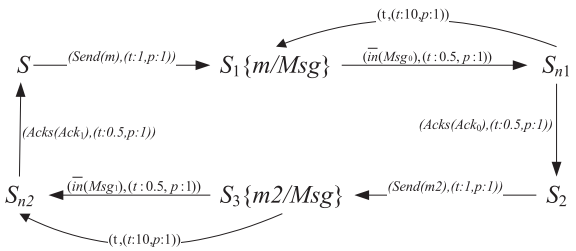


Fig. 4 State transition of sender

Using QELTS, we can conduct an analysis of the behavior and trustworthiness of the system. The properties to be verified are described in QCTL.

Propterty 1. $(<send> \Rightarrow (true \cup^{t\leq 5, p\geq 0.9} <\overline{rec}>)) \cup^{t\leq\infty, p\geq 1}$ *false*

The property requests that during the system's evolution, if action <*send*> occurs then <$\overline{rec}$> will occur within 5 time units with probability not less than 0.9. This corresponds to establishing "whether at least 90% of the packages can be sent from the sender to the receiver in 5 time units." This property can be verified by analyzing the QELTS established by QPi. In Fig. 5, there exists a path $\omega_1 = ABP(send, rec) \xrightarrow{Send(Msg),(t:1,p:1)} S_1|M|AM|R \xrightarrow{(\tau,(t:0.5,p:1))} S_4 \mid M_1 \mid AM \mid R \xrightarrow{(\tau,(t:0.5,p:0.9))} S_4 \mid M_3 \mid AM \mid R \xrightarrow{(\tau,(t:1,p:1))} S_4 \mid M \mid AM \mid R_3 \xrightarrow{(\overline{rec}(Msg0),(t:1,p:1))} S_4 \mid M \mid AM \mid R_4$. The composition of trustworthiness values on the path is $(t{:}1, p{:}1)\times(t{:}0.5, p{:}1)\times(t{:}0.5, p{:}0.9)\times(t{:}1, p{:}1)\times(t{:}1, p{:}1) = (t{:}4, p{:}0.9)$. This shows that there exists an evolution path that begins with activity $(Send(Msg), (t{:}1, p{:}1))$ and eventually causes activity $(\overline{rec}(Msg_0), (t{:}1, p{:}1))$ to occur, while the trustworthiness of this path is $(t{:}4, p{:}0.9)$. This value satisfied the constraint of $t \leq 5$, $p \geq 0.9$, so property 1 is satisfied. This states that the possibility that a message is received within 4 time units after it has been sent is not less than 90 percent. This parameter can be regarded as one of the performance indicators of the system.

Propterty 2. $(<send> \Rightarrow X(true \cup^{t\leq 10, p\geq 0.8} <send>)) \cup^{t\leq\infty, p\geq 1}$ *false*

Property 2 states that at the sending end, the probability of sending two consecutive packages in 10 time units should be greater than or equal to 0.8. In Fig. 5, there exists a path $\omega_2 = ABP(send, rec) \xrightarrow{Send(Msg),(t:1,p:1)} S_1 \mid M \mid AM \mid R \xrightarrow{(\tau,(t:0.5,p:1))} S_4 \mid M_1 \mid AM \mid R \xrightarrow{(\tau,(t:0.5,p:0.9))} S_4|M_3|AM|R \xrightarrow{(\tau,(t:1,p:1))} S_4|M|AM|R_3 \xrightarrow{(\overline{rec}(Msg0),(t:1,p:1))} S_4 \mid M \mid AM \mid R_4 \xrightarrow{\tau,(t:1,p:1)} S_4 \mid M \mid AM_1 \mid R_1 \xrightarrow{(\tau,(t:0.5,p:0.9))} S_4|M|AM_3|R_1 \xrightarrow{\tau,(t:1,p:1)} S_2|M|AM|R_1 \xrightarrow{Send(Msg),(t:1,p:1)}$

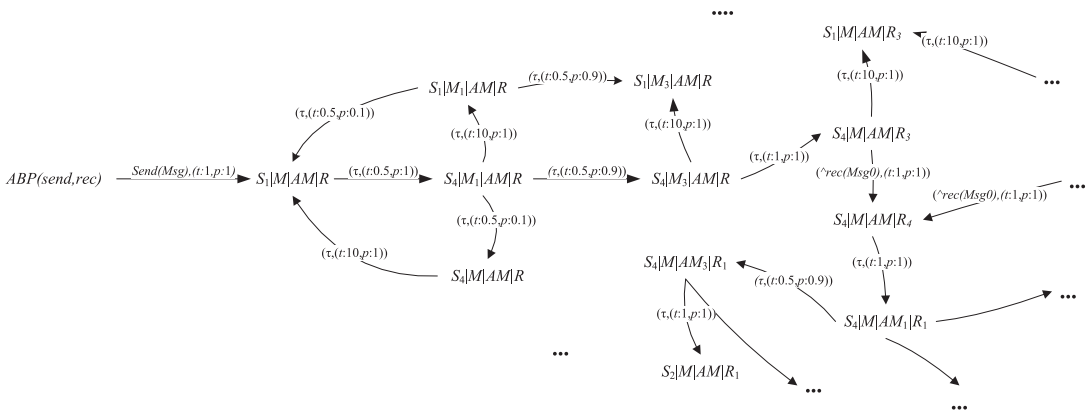


Fig. 5 State transitions of ABP

$S_3 \mid M \mid AM \mid R_1$. The composition of trustworthiness values on the path is (t:7.5, p:0.81), which satisfies the constraint of (t: ≤ 10, p: ≥ 0.8), so property 2 is satisfied.

Property 3: $(<send> \Rightarrow (true \cup^{t\leq\infty, p\geq 1} <\overline{rec}>)) \cup^{t\leq\infty, p\geq 1} false$

Property 3 means that during the system's evolution, an input action <*send*> will eventually cause the output action $\overline{rec}$ to occur. This implies that an input package can always be sent to the receiver. By the analysis of property 1, we have that there exists a path that makes the possibility of a message received within 4 time units equal to 0.9. Besides, in the QELTS there exists a path $\omega = ABP(send, rec) \xrightarrow{Send(Msg),(t:1,p:1)} S_1|M|AM|R \xrightarrow{(\tau,(t:0.5,p:1))} S_4|M_1|AM|R \xrightarrow{(\tau,(t:0.5,p:0.1))} S_4|M|AM|R \xrightarrow{(\tau,(t:10,p:1))} S_1|M|AM|R \xrightarrow{(\tau,(t:0.5,p:0.9))} S_4|M_3|AM|R \xrightarrow{(\tau,(t:1,p:1))} S_4|M|AM|R_3 \xrightarrow{(\overline{rec}(Msg0),(t:1,p:1))} S_4 \mid M \mid AM \mid R_4$, the trustworthiness of which is (t:14.5, p:0.09). Further investigation shows there exist evolution paths with trustworthiness (t:25, p:0.009), (t:35.5, p:0.0009), (t:46, p:0.00009)... in the QELTS, which begin with activity ($Send(Msg)$, (t:1, p:1)) and eventually cause activity ($\overline{rec}(\mathrm{Msg}_0)$, (t:1, p:1)) to occur. Without considering the time dimension constraints, the probability that $Send(Msg)$ occurs and eventually makes $\overline{rec}(Msg)$ occur is the sum of the probabilities of all these paths. Therefore, we have $P_n = \frac{a_1(1-q^n)}{1-q} = \frac{0.9(1-0.1^n)}{1-0.1}$ and $\lim_{n\to\infty} P_n = 1$. This suggests that an input action always causes an output action, and that data can always be sent from the transmitting end to the receiving end. The above analysis indicates that the timeout wait/resend mechanism ensures that data will not be lost during transmission.

Let us now consider which one of services *Sender*2 and *Sender*3 given below is more suitable to replace service *Sender*1?

Sender2:

$$P = (send(Msg), (t{:}1.1, p{:}1)).P_1$$
$$P_1 = (\overline{in}(Msg_0), (t{:}0.7, p{:}1)).((acks(Ack_0), (t{:}0.7, p{:}1)).P_2 + (\tau, (t{:}10, p{:}1)).P_1)$$
$$P_2 = (send(Msg), (t{:}1.1, p{:}1)).P_3$$
$$P_3 = (\overline{in}(Msg_1), (t{:}0.7, p{:}1)).((acks(Ack_1), (t{:}0.7, p{:}1)).P + (\tau, (t{:}10, p{:}1)).P_3)$$

Sender3:

$$Q = (send(Msg), (t{:}1.5, p{:}1)).Q_1$$
$$Q_1 = (\overline{in}(Msg_0), (t{:}0.8, p{:}1)).((acks(Ack_0), (t{:}0.8, p{:}1)).Q_2 + (\tau, (t{:}10, p{:}1)).Q_1)$$
$$Q_2 = (send(Msg), (t{:}1.5, p{:}1)).Q_3$$
$$Q_3 = (\overline{in}(Msg_1), (t{:}0.8, p{:}1)).((acks(Ack_1), (t{:}0.8, p{:}1)).Q + (\tau, (t{:}10, p{:}1)).Q_3)$$

This can be analyzed by calculating and comparing the bisimulation distances of service sets. We investigate the similarity distance of $R_1 = \{(S, P), (S_1, P_1), (S_2, P_2), (S_3, P_3)\}$ and $R_2 = \{(S, Q), (S_1, Q_1), (S_2, Q_2), (S_3, Q_3)\}$ respectively. We define the quality metric function as:

- $\rho((a, t, p), (b, t', p')) = \infty$, *if* $a \neq b$
- $\rho((a, t, p), (b, t', p')) = |t - t'| + |p - p'|$

According to the definition of bisimulation distance, we have $d_{R1} = max\{d_{R1}(S, P) = 0.1, d_{R1}(S_1, P_1) = 0.2, d_{R1}(S_2, P_2) = 0.1, d_{R1}(S_3, P_3) = 0.2\} = 0.2$. and $d_{R2} = max\{d_{R2}(S, Q) = 0.5, d_{R2}(S_1, Q_1) = 0.3, d_{R2}(S_2, Q_2) = 0.5, d_{R2}(S_3, Q_3) = 0.3\} = 0.5$.

It is evident that *Sender*2 more similar to *Sender*1 than *Sender*3 is. Therefore, if there are only the two services to be considered for the replacement of *Sender*1, *Sender*2 should be chosen. Through observation, we can found that the executable actions for these three services are the same, but there are differences in their trustworthiness properties. In the real-time dimension, the real-time property of *Sender*2 is better than *Sender*3, thus making it more suitable as a replacement for *Sender*1. The calculated result is as same as our intuition.

From the perspective of λ-bisimulation, R_1 is a 0.2 bisimulation, and R_2 is a 0.5 bisimulation. The upper bound of the bisimulation distance of the process tuples in R_1 is less than the process tuples in R_2, so the service tuples in R_1 are more similar than R_2.

8. Related Work

Formal method is considered to be effective method for complex system modeling, analysis and verification. Related research in this field has decades of history and has achieved rich theoretical results. Among them, the process algebra represented by CCS and CSP is the most important theoretical method for modeling and analyzing concurrent systems. Researchers proposed various extensions of process algebra to enhance its descriptive abilities, such as timed process algebras [9], probabilistic process algebra [10], and non-deterministic process algebra [11], [12]. Similar works also include [13], who proposed a CCS-based process algebra which can be used for concurrent resource usage modeling and analysis.

By introducing the constraint semi-ring [14] described the multi-dimensional QoS property in a uniform manner, and proposed a process algebra for describing distributed application. By combining Synchronised Hyper edge Replacement (SHR) with constraint-semirings [15] presented a formal framework for specifying service systems that handle abstract high-level QoS aspects. SHR is a (hyper) graph rewriting mechanism for modelling the mobility and reconfiguration of systems. [6] introduced a new operator into the constraint semi-ring to describe QoS composition in different manner, called Q-algebra. Then, Q-algebra was combined with automata to propose an automata-based formal method for modeling systems with properties in multiple non-functional dimensions. [16] extended CCS by introducing Q-algebra and proposed a process algebra to enforce QoS requirements in service computing.

[17] proposed the concept of the bisimulation index for common labelled transition systems by using metrics on

actions in process algebra and applying the concept to timed CCS and real time ACP. The definition we present in this paper is similar to the concept of the bisimulation index presented in [17], but the metric in this paper is more general and can be applied to both actions and different trustworthiness aspects. [18] proposed action-labelled quantitative transition systems as a general framework for combining qualitative and quantitative analysis, which further extends bisimulation to quantitative systems. The deeper connection between [17], [18] and our work needs to be investigated in further works.

In this paper, we introduce the concept of trustworthiness. We believe that the concept of trustworthiness of a service computing system has two aspects: one is the functions of the system are fully realized, and the other is that the non-functional properties of the system meet users' needs. The notion of trustworthiness is a measure of the non-functional properties of a system. Some dimensions of trustworthiness may be probabilistic properties, such like reliability, availability, and so on, but some are not, such as the real-time property, cost, security, and so on. The introduction of Q-algebra allows the description of multi-dimensional trustworthiness properties in a uniform way.

In comparison to the existing works, the difference of our approach is that we introduce Q-algebra to describe multi-dimensional trustworthiness properties in a uniform manner. Based on this feature, we extend the pi-calculus for modeling and reasoning the structure, behavior, and trustworthiness of a concurrent system in a unified abstract level. In a QPi modelled service computing system, actions correspond to the system functions' implementation and the trustworthiness values added with correspond to the non-functional properties of the system.

Secondly, QPi is a combination of pi calculus and Q algebra, and allows the creation and removal of communication links between processes. By allowing links to be created and deleted, QPi is more suitable for modeling service computing systems with a dynamic structure.

9. Conclusion

Trustworthy computing requires modeling service systems in terms of both behavioral aspects and trustworthiness. We extend the pi-calculus by combining it with Q algebra for modeling the structure, behavior, and trustworthiness of a concurrent system in a unified abstract level. QPi attaches a trustworthiness tuple to process actions, and the former that allows the modeling of the system's behavior while also describing its trustworthiness. We also propose the operational semantics of QPi. The concept of bisimulation distance provides a proper candidate for comparing systems quantitatively from both the behavioural and the trustworthiness perspective. Additionally, QCTL is also introduced as a logic for reasoning about both behaviour and trustworthiness of service oriented systems specified in QPi. A specific example is described and analyzed to illustrate the effectiveness of the algebraic method.

As to future work, we are particularly interested in investigating the expressivity and complexity of QCTL. It is also worth studying the related properties of quantified bisimulation and an efficient algorithm for determining whether a service and its traces satisfy a trustworthiness constraint. We would also like to exploit existing model checkers to conduct a formal verification of the trustworthiness properties of services presented in our model.

Acknowledgements

This research was supported by the National Natural Science Foundation of China (No. 61472327), and the Aviation Science Foundation of China (No. 2015ZD53050). It also was supported by the Fundamental Research Funds for the Central Universities (No. 3102017jc04001), and the 111 Project Grant of NPU (No. B13044).

References

[1] Trusted Computing Group, "TCG Specification Architecture Overview," Trusted Computing Group, http://www.trustedcomputinggroup.org/tcg-architecture-overvie-version-1-4/, accessed Sept. 10. 2016.
[2] S. Pearson, Trusted Computing Platforms, the Next Security Solution, HP Laboratories, Bristol, 2002.
[3] M.T. Zhou and L. Tan, "Progress in Trusted Computing," Journal of University of Electronic Science and Technology of China, vol.35, no.4, pp.686–697, 2006.
[4] H.W. Chen, J. Wang, and W. Dong, "High Confidence Software Engineering Technologies," Acta Electronica Sinica, vol.31, no.12, pp.1933–1938, 2003.
[5] R. Milner, "A Calculus of Mobile Processes, Parts I and II," Information and Computation, vol.100, no.1, pp.1–77, 1992.
[6] T. Chothia and J. Kleijn, "Q-Automata: Modelling the Resource Usage of Concurrent Components," Electronic Notes in Theoretical Computer Science, vol.175, no.2, pp.153–167, 2007.
[7] S. Bistarelli, U. Montanari, and F. Rossi, "Semiring-Based Constraint Satisfaction and Optimization," Journal of the ACM, vol.44, no.2, pp.201–236, 1997.
[8] N. Fu, "The Research on Key Technologies of Trustworthy Service Cooperation Environment," Ph.D. Thesis, School of Computer, Northwestern Polytechnical University, Xi'an, 2010.
[9] X. Nicollin and J. Sifakis, "An Overview and Synthesis on Timed Process Algebras," Computer Aided Verification, ed. K.G. Larsen and A. Skou, pp.376–398, Berlin Springer, 1992.
[10] G. Lowe, "Probabilistic and Prioritized Models of Timed CSP," Theoretical Computer Science, vol.138, no.2, pp.315–352, 1995.
[11] J. Hillston, A Compositional Approach to Performance Modelling, Cambridge University Press, 1996.
[12] M. Bernardo and R. Gorrieri, "A Tutorial on EMPA: A Theory of Concurrent Processes with Nondeterminism, Priorities, Probabilities and Time," Theoretical Computer Science, vol.202, no.1-2, pp.1–54, 1998.
[13] D. Pym and C. Tofts, "A Calculus and Logic of Resources and Processes," Formal Aspects of Computing, vol.18, no.4, pp.495–517, 2006.
[14] R.D. Nicola, G. Ferrari, U. Montanari, R. Pugliese, and E. Tuosto, "A Formal Basis for Reasoning on Programmable QoS," in Verification: Theory and Practice, ed. N. Dershowitz, pp.185–187, Springer, Berlin, 2004.
[15] D. Hirsch and E. Tuosto, "SHReQ: Coordinating Application Level QoS," Proc. of 3rd IEEE International Conference on Software Engineering and Formal Methods, Koblenz, Germany, pp.425–434,

2005.
[16] M. Sun, "QCCS: A Formal Model to Enforce QoS Requirements in Service Composition," Proc. of the 1st Joint IEEE/IFIP Symposium on Theoretical Aspects of Software Engineering, Washington, USA. pp.389–400, 2007.
[17] M. Ying, "Bisimulation Indexes and Their Applications," Theoretical Computer Science, vol.275, no.1-2, pp.1–68, 2002.
[18] Y. Deng, T. Chothia, C. Palamidessi, and J. Pang, "Metrics for Action-labelled Quantitative Transition Systems," Electronic Notes in Theoretical Computer Science, vol.153, no.2, pp.79–96, 2006.

Ning Fu is an assistant research fellow at the School of Computer, Northwestern Polytechnical University, Xi'an. He received his Ph.D. (2010) degree in computer science and technology from Northwestern Polytechnical University. His main research interests focus on modeling and verification embedded system by formal methods.

Yingfeng Zhang is a professor, Ph.D. supervisor at the School of Mechanical Engineering, Northwestern Polytechnical University. His main research interests include intelligent manufacturing and Cyber-Physical system modeling & analysis.

Lijun Shan is a lecturer at the School of Computer, Northwestern Polytechnical University, Xi'an. She received her Ph.D. degree in computer science and technology from National University of Defense Technology. Her main research interests include Model-Driven Development Method and Formal Method.

Zhiqiang Liu is a associate professor at the School of Software and Microelectronics, Northwestern Polytechnical University, Xi'an. His current research interests focus on modeling and verification embedded system by formal methods.

Han Peng is a PhD candidate at the School of Computer, Northwestern Polytechnical University, Xi'an. His current research interests focus on modeling and verification Cyber-Physical system by formal methods.