Algebraic Laws for True Concurrency

Yong Wang

College of Computer Science and Technology,
Faculty of Information Technology,
Beijing University of Technology, Beijing, China

Abstract. We find the algebraic laws for true concurrency. Eventually, we establish a whole axiomatization for true concurrency called $APTC$ (Algebra for Parallelism in True Concurrency). The theory $APTC$ has four modules: $BATC$ (Basic Algebra for True Concurrency), $APTC$ (Algebra for Parallelism in True Concurrency), recursion and abstraction. And also, we show the applications and extensions of $APTC$.

Keywords: True Concurrency; Behaviorial Equivalence; Prime Event Structure; Axiomatization

1. Introduction

Parallelism and concurrency [7] are the core concepts within computer science. There are mainly two camps in capturing concurrency: the interleaving concurrency and the true concurrency.

The representative of interleaving concurrency is bisimulation/rooted branching bisimulation equivalences. CCS (Calculus of Communicating Systems) [3] is a calculus based on bisimulation semantics model. Hennessy and Milner (HM) logic for bisimulation equivalence is also designed. Later, algebraic laws to capture computational properties modulo bisimulation equivalence was introduced in [1], this work eventually founded the comprehensive axiomatization modulo bisimulation equivalence – ACP (Algebra of Communicating Algebra) [4].

The other camp of concurrency is true concurrency. The researches on true concurrency are still active. Firstly, there are several truly concurrent bisimulation equivalence, the representatives are: pomset bisimulation equivalence, step bisimulation equivalence, history-preserving (hp-) bisimulation equivalence, and especially hereditary history-preserving (hhp-) bisimulation equivalence [8] [9], the well-known finest truly concurrent bisimulation equivalence. These truly concurrent bisimulations are studied in different structures [5] [6] [7]: Petri nets, event structures, domains, and also a uniform form called TSI (Transition System with Independence) [13]. There are also several logics based on different truly concurrent bisimulation equivalences, for example, SFL (Separation Fixpoint Logic) and TFL (Trace Fixpoint Logic) [13] are extensions on true concurrency of mu-calculi [10] on bisimulation equivalence, and also a logic with reverse modalities [11] [12] based on the so-called reverse bisimulations with a reverse flavor. It must be pointed out that, a uniform

Correspondence and offprint requests to: Yong Wang, Pingleyuan 100, Chaoyang District, Beijing, China. e-mail: wangy@bjut.edu.cn

logic for true concurrency [14] [15] was represented several years ago, which used a logical framework to unify several truly concurrent bisimulation equivalences, including pomset bisimulation, step bisimulation, hp-bisimulation and hhp-bisimulation.

There are simple comparisons between HM logic and bisimulation equivalence as the uniform logic [14] [15] and truly concurrent bisimulation equivalences, the algebraic laws [1], ACP [4] and bisimulation equivalence, as truly concurrent bisimulation equivalences and *what*, which is still missing.

Yes, we try to find the algebraic laws for true concurrency following the way paved by ACP for bisimulation equivalence. And finally, we establish a whole axiomatization for true concurrency called $APTC$. The theory $APTC$ has four modules: $BATC$ (Basic Algebra for True Concurrency), $APTC$ (Algebra for Parallelism in True Concurrency), recursion and abstraction.

This paper is organized as follows. In section 2, we introduce some preliminaries, including a brief introduction to ACP, and also preliminaries on true concurrency. We introduce $BATC$ in section 3, $APTC$ in section 4, recursion in section 5, abstraction in section 6. In section 7, we show the applications of $APTC$ by an example called alternating bit protocol. We show the modularity and extension mechanism of $APTC$ in section 8. Finally, in section 9, we conclude this paper.

2. Backgrounds

2.1. Process Algebra

In this subsection, we introduce the preliminaries on process algebra ACP [4], which is based on the interleaving bisimulation semantics. ACP has an almost perfect axiomatization to capture laws on bisimulation equivalence, including equational logic and bisimulation semantics, and also the soundness and completeness bridged between them.

2.1.1. ACP

ACP captures several computational properties in the form of algebraic laws, and proves the soundness and completeness modulo bisimulation/rooted branching bisimulation equivalence. These computational properties are organized in a modular way by use of the concept of conservational extension, which include the following modules, note that, every algebra are composed of constants and operators, the constants are the computational objects, while operators capture the computational properties.

1. BPA **(Basic Process Algebras)**. BPA has sequential composition $\cdot$ and alternative composition $+$ to capture sequential computation and nondeterminacy. The constants are ranged over A, the set of atomic actions. The algebraic laws on $\cdot$ and $+$ are sound and complete modulo bisimulation equivalence.
2. ACP **(Algebra of Communicating Processes)**. ACP uses the parallel operator $\parallel$, the auxiliary binary left merge $\between$ to model parallelism, and the communication merge $|$ to model communications among different parallel branches. Since a communication may be blocked, a new constant called deadlock δ is extended to A, and also a new unary encapsulation operator ∂_H is introduced to eliminate δ, which may exist in the processes. The algebraic laws on these operators are also sound and complete modulo bisimulation equivalence. Note that, these operators in a process can be eliminated by deductions on the process using axioms of ACP, and eventually be steadied by $\cdot$ and $+$, this is also why bisimulation is called an *interleaving* semantics.
3. **Recursion**. To model infinite computation, recursion is introduced into ACP. In order to obtain a sound and complete theory, guarded recursion and linear recursion are needed. The corresponding axioms are RSP (Recursive Specification Principle) and RDP (Recursive Definition Principle), RDP says the solutions of a recursive specification can represent the behaviors of the specification, while RSP says that a guarded recursive specification has only one solution, they are sound with respect to ACP with guarded recursion modulo bisimulation equivalence, and they are complete with respect to ACP with linear recursion modulo bisimulation equivalence.
4. **Abstraction**. To abstract away internal implementations from the external behaviors, a new constant τ called silent step is added to A, and also a new unary abstraction operator τ_I is used to rename actions in I into τ (the resulted ACP with silent step and abstraction operator is called ACP_τ). The recursive specification is adapted to guarded linear recursion to prevent infinite τ-loops specifically. The axioms

for τ and τ_I are sound modulo rooted branching bisimulation equivalence (a kind of weak bisimulation equivalence). To eliminate infinite τ-loops caused by τ_I and obtain the completeness, $CFAR$ (Cluster Fair Abstraction Rule) is used to prevent infinite τ-loops in a constructible way.

ACP can be used to verify the correctness of system behaviors, by deduction on the description of the system using the axioms of ACP. Base on the modularity of ACP, it can be extended easily and elegantly. For more details, please refer to the book of ACP [4].

2.1.2. Operational Semantics

The semantics of ACP is based on bisimulation/rooted branching bisimulation equivalences, and the modularity of ACP relies on the concept of conservative extension, for the conveniences, we introduce some concepts and conclusions on them.

Definition 2.1 (Bisimulation). A bisimulation relation R is a binary relation on processes such that: (1) if pRq and $p \xrightarrow{a} p'$ then $q \xrightarrow{a} q'$ with $p'Rq'$; (2) if pRq and $q \xrightarrow{a} q'$ then $p \xrightarrow{a} p'$ with $p'Rq'$; (3) if pRq and pP, then qP; (4) if pRq and qP, then pP. Two processes p and q are bisimilar, denoted by $p \sim_{HM} q$, if there is a bisimulation relation R such that pRq.

Definition 2.2 (Congruence). Let Σ be a signature. An equivalence relation R on $\mathcal{T}(\Sigma)$ is a congruence if for each $f \in \Sigma$, if $s_i R t_i$ for $i \in \{1, \cdots, ar(f)\}$, then $f(s_1, \cdots, s_{ar(f)}) R f(t_1, \cdots, t_{ar(f)})$.

Definition 2.3 (Branching bisimulation). A branching bisimulation relation R is a binary relation on the collection of processes such that: (1) if pRq and $p \xrightarrow{a} p'$ then either $a \equiv \tau$ and $p'Rq$ or there is a sequence of (zero or more) τ-transitions $q \xrightarrow{\tau} \cdots \xrightarrow{\tau} q_0$ such that pRq_0 and $q_0 \xrightarrow{a} q'$ with $p'Rq'$; (2) if pRq and $q \xrightarrow{a} q'$ then either $a \equiv \tau$ and pRq' or there is a sequence of (zero or more) τ-transitions $p \xrightarrow{\tau} \cdots \xrightarrow{\tau} p_0$ such that p_0Rq and $p_0 \xrightarrow{a} p'$ with $p'Rq'$; (3) if pRq and pP, then there is a sequence of (zero or more) τ-transitions $q \xrightarrow{\tau} \cdots \xrightarrow{\tau} q_0$ such that pRq_0 and q_0P; (4) if pRq and qP, then there is a sequence of (zero or more) τ-transitions $p \xrightarrow{\tau} \cdots \xrightarrow{\tau} p_0$ such that p_0Rq and p_0P. Two processes p and q are branching bisimilar, denoted by $p \approx_{bHM} q$, if there is a branching bisimulation relation R such that pRq.

Definition 2.4 (Rooted branching bisimulation). A rooted branching bisimulation relation R is a binary relation on processes such that: (1) if pRq and $p \xrightarrow{a} p'$ then $q \xrightarrow{a} q'$ with $p' \approx_{bHM} q'$; (2) if pRq and $q \xrightarrow{a} q'$ then $p \xrightarrow{a} p'$ with $p' \approx_{bHM} q'$; (3) if pRq and pP, then qP; (4) if pRq and qP, then pP. Two processes p and q are rooted branching bisimilar, denoted by $p \approx_{rbHM} q$, if there is a rooted branching bisimulation relation R such that pRq.

Definition 2.5 (Conservative extension). Let T_0 and T_1 be TSSs (transition system specifications) over signatures Σ_0 and Σ_1, respectively. The TSS $T_0 \oplus T_1$ is a conservative extension of T_0 if the LTSs (labeled transition systems) generated by T_0 and $T_0 \oplus T_1$ contain exactly the same transitions $t \xrightarrow{a} t'$ and tP with $t \in \mathcal{T}(\Sigma_0)$.

Definition 2.6 (Source-dependency). The source-dependent variables in a transition rule of ρ are defined inductively as follows: (1) all variables in the source of ρ are source-dependent; (2) if $t \xrightarrow{a} t'$ is a premise of ρ and all variables in t are source-dependent, then all variables in t' are source-dependent. A transition rule is source-dependent if all its variables are. A TSS is source-dependent if all its rules are.

Definition 2.7 (Freshness). Let T_0 and T_1 be TSSs over signatures Σ_0 and Σ_1, respectively. A term in $\mathbb{T}(T_0 \oplus T_1)$ is said to be fresh if it contains a function symbol from $\Sigma_1 \setminus \Sigma_0$. Similarly, a transition label or predicate symbol in T_1 is fresh if it does not occur in T_0.

Theorem 2.8 (Conservative extension). Let T_0 and T_1 be TSSs over signatures Σ_0 and Σ_1, respectively, where T_0 and $T_0 \oplus T_1$ are positive after reduction. Under the following conditions, $T_0 \oplus T_1$ is a conservative extension of T_0. (1) T_0 is source-dependent. (2) For each $\rho \in T_1$, either the source of ρ is fresh, or ρ has a premise of the form $t \xrightarrow{a} t'$ or tP, where $t \in \mathbb{T}(\Sigma_0)$, all variables in t occur in the source of ρ and t', a or P is fresh.

2.1.3. Proof Techniques

In this subsection, we introduce the concepts and conclusions about elimination, which is very important in the proof of completeness theorem.

Definition 2.9 (Elimination property). Let a process algebra with a defined set of basic terms as a subset of the set of closed terms over the process algebra. Then the process algebra has the elimination to basic terms property if for every closed term s of the algebra, there exists a basic term t of the algebra such that the algebra$\vdash s = t$.

Definition 2.10 (Strongly normalizing). A term s_0 is called strongly normalizing if does not an infinite series of reductions beginning in s_0.

Definition 2.11. We write $s >_{lpo} t$ if $s \to^+ t$ where $\to^+$ is the transitive closure of the reduction relation defined by the transition rules of a algebra.

Theorem 2.12 (Strong normalization). Let a term rewriting (TRS) system with finitely many rewriting rules and let $>$ be a well-founded ordering on the signature of the corresponding algebra. If $s >_{lpo} t$ for each rewriting rule $s \to t$ in the TRS, then the term rewriting system is strongly normalizing.

2.2. True Concurrency

In this subsection, we introduce the concepts of prime event structure, and also concurrent behavior equivalence [5] [6] [7], and also we extend prime event structure with silent event τ, and explain the concept of weakly true concurrency, i.e., concurrent behaviorial equivalence with considering silent event τ [16].

2.2.1. Event Structure

We give the definition of prime event structure (PES) [5] [6] [7] extended with the silent event τ as follows.

Definition 2.13 (Prime event structure with silent event). Let Λ be a fixed set of labels, ranged over $a, b, c, \cdots$ and τ. A (Λ-labelled) prime event structure with silent event τ is a tuple $\mathcal{E} = \langle \mathbb{E}, \leq, \sharp, \lambda \rangle$, where $\mathbb{E}$ is a denumerable set of events, including the silent event τ. Let $\hat{\mathbb{E}} = \mathbb{E} \backslash \{\tau\}$, exactly excluding τ, it is obvious that $\hat{\tau^*} = \epsilon$, where ϵ is the empty event. Let $\lambda : \mathbb{E} \to \Lambda$ be a labelling function and let $\lambda(\tau) = \tau$. And $\leq$, $\sharp$ are binary relations on $\mathbb{E}$, called causality and conflict respectively, such that:

1. $\leq$ is a partial order and $\lceil e \rceil = \{e' \in \mathbb{E} | e' \leq e\}$ is finite for all $e \in \mathbb{E}$. It is easy to see that $e \leq \tau^* \leq e' = e \leq \tau \leq \cdots \leq \tau \leq e'$, then $e \leq e'$.
2. $\sharp$ is irreflexive, symmetric and hereditary with respect to $\leq$, that is, for all $e, e', e'' \in \mathbb{E}$, if $e \sharp e' \leq e''$, then $e \sharp e''$.

Then, the concepts of consistency and concurrency can be drawn from the above definition:

1. $e, e' \in \mathbb{E}$ are consistent, denoted as $e \frown e'$, if $\neg(e \sharp e')$. A subset $X \subseteq \mathbb{E}$ is called consistent, if $e \frown e'$ for all $e, e' \in X$.
2. $e, e' \in \mathbb{E}$ are concurrent, denoted as $e \parallel e'$, if $\neg(e \leq e')$, $\neg(e' \leq e)$, and $\neg(e \sharp e')$.

The prime event structure without considering silent event τ is the original one in [5] [6] [7].

Definition 2.14 (Configuration). Let $\mathcal{E}$ be a PES. A (finite) configuration in $\mathcal{E}$ is a (finite) consistent subset of events $C \subseteq \mathcal{E}$, closed with respect to causality (i.e. $\lceil C \rceil = C$). The set of finite configurations of $\mathcal{E}$ is denoted by $\mathcal{C}(\mathcal{E})$. We let $\hat{C} = C \backslash \{\tau\}$.

A consistent subset of $X \subseteq \mathbb{E}$ of events can be seen as a pomset. Given $X, Y \subseteq \mathbb{E}$, $\hat{X} \sim \hat{Y}$ if $\hat{X}$ and $\hat{Y}$ are isomorphic as pomsets. In the following of the paper, we say $C_1 \sim C_2$, we mean $\hat{C_1} \sim \hat{C_2}$.

Definition 2.15 (Pomset transitions and step). Let $\mathcal{E}$ be a PES and let $C \in \mathcal{C}(\mathcal{E})$, and $\varnothing \neq X \subseteq \mathbb{E}$, if $C \cap X = \varnothing$ and $C' = C \cup X \in \mathcal{C}(\mathcal{E})$, then $C \xrightarrow{X} C'$ is called a pomset transition from C to C'. When the events in X are pairwise concurrent, we say that $C \xrightarrow{X} C'$ is a step.

Definition 2.16 (Weak pomset transitions and weak step). Let $\mathcal{E}$ be a PES and let $C \in \mathcal{C}(\mathcal{E})$, and $\varnothing \neq X \subseteq \hat{\mathbb{E}}$, if $C \cap X = \varnothing$ and $\hat{C'} = \hat{C} \cup X \in \mathcal{C}(\mathcal{E})$, then $C \stackrel{X}{\Longrightarrow} C'$ is called a weak pomset transition from C to C', where we define $\stackrel{e}{\Longrightarrow} \triangleq \stackrel{\tau^*}{\longrightarrow}\stackrel{e}{\rightarrow}\stackrel{\tau^*}{\longrightarrow}$. And $\stackrel{X}{\Longrightarrow} \triangleq \stackrel{\tau^*}{\longrightarrow}\stackrel{e}{\rightarrow}\stackrel{\tau^*}{\longrightarrow}$, for every $e \in X$. When the events in X are pairwise concurrent, we say that $C \stackrel{X}{\Longrightarrow} C'$ is a weak step.

We will also suppose that all the PESs in this paper are image finite, that is, for any PES $\mathcal{E}$ and $C \in \mathcal{C}(\mathcal{E})$ and $a \in \Lambda$, $\{e \in \mathbb{E} | C \stackrel{e}{\rightarrow} C' \wedge \lambda(e) = a\}$ and $\{e \in \hat{\mathbb{E}} | C \stackrel{e}{\Longrightarrow} C' \wedge \lambda(e) = a\}$ is finite.

2.2.2. Concurrent Behavioral Equivalence

Definition 2.17 (Pomset, step bisimulation). Let $\mathcal{E}_1$, $\mathcal{E}_2$ be PESs. A pomset bisimulation is a relation $R \subseteq \mathcal{C}(\mathcal{E}_1) \times \mathcal{C}(\mathcal{E}_2)$, such that if $(C_1, C_2) \in R$, and $C_1 \stackrel{X_1}{\longrightarrow} C_1'$ then $C_2 \stackrel{X_2}{\longrightarrow} C_2'$, with $X_1 \subseteq \mathbb{E}_1$, $X_2 \subseteq \mathbb{E}_2$, $X_1 \sim X_2$ and $(C_1', C_2') \in R$, and vice-versa. We say that $\mathcal{E}_1$, $\mathcal{E}_2$ are pomset bisimilar, written $\mathcal{E}_1 \sim_p \mathcal{E}_2$, if there exists a pomset bisimulation R, such that $(\varnothing, \varnothing) \in R$. By replacing pomset transitions with steps, we can get the definition of step bisimulation. When PESs $\mathcal{E}_1$ and $\mathcal{E}_2$ are step bisimilar, we write $\mathcal{E}_1 \sim_s \mathcal{E}_2$.

Definition 2.18 (Weak pomset, step bisimulation). Let $\mathcal{E}_1$, $\mathcal{E}_2$ be PESs. A weak pomset bisimulation is a relation $R \subseteq \mathcal{C}(\mathcal{E}_1) \times \mathcal{C}(\mathcal{E}_2)$, such that if $(C_1, C_2) \in R$, and $C_1 \stackrel{X_1}{\Longrightarrow} C_1'$ then $C_2 \stackrel{X_2}{\Longrightarrow} C_2'$, with $X_1 \subseteq \hat{\mathbb{E}_1}$, $X_2 \subseteq \hat{\mathbb{E}_2}$, $X_1 \sim X_2$ and $(C_1', C_2') \in R$, and vice-versa. We say that $\mathcal{E}_1$, $\mathcal{E}_2$ are weak pomset bisimilar, written $\mathcal{E}_1 \approx_p \mathcal{E}_2$, if there exists a weak pomset bisimulation R, such that $(\varnothing, \varnothing) \in R$. By replacing weak pomset transitions with weak steps, we can get the definition of weak step bisimulation. When PESs $\mathcal{E}_1$ and $\mathcal{E}_2$ are weak step bisimilar, we write $\mathcal{E}_1 \approx_s \mathcal{E}_2$.

Definition 2.19 (Posetal product). Given two PESs $\mathcal{E}_1$, $\mathcal{E}_2$, the posetal product of their configurations, denoted $\mathcal{C}(\mathcal{E}_1) \overline{\times} \mathcal{C}(\mathcal{E}_2)$, is defined as

$$\{(C_1, f, C_2) | C_1 \in \mathcal{C}(\mathcal{E}_1), C_2 \in \mathcal{C}(\mathcal{E}_2), f : C_1 \to C_2 \text{ isomorphism}\}.$$

A subset $R \subseteq \mathcal{C}(\mathcal{E}_1) \overline{\times} \mathcal{C}(\mathcal{E}_2)$ is called a posetal relation. We say that R is downward closed when for any $(C_1, f, C_2), (C_1', f', C_2') \in \mathcal{C}(\mathcal{E}_1) \overline{\times} \mathcal{C}(\mathcal{E}_2)$, if $(C_1, f, C_2) \subseteq (C_1', f', C_2')$ pointwise and $(C_1', f', C_2') \in R$, then $(C_1, f, C_2) \in R$.

For $f : X_1 \to X_2$, we define $f[x_1 \mapsto x_2] : X_1 \cup \{x_1\} \to X_2 \cup \{x_2\}$, $z \in X_1 \cup \{x_1\}$,(1)$f[x_1 \mapsto x_2](z) = x_2$,if $z = x_1$;(2)$f[x_1 \mapsto x_2](z) = f(z)$, otherwise. Where $X_1 \subseteq \mathbb{E}_1$, $X_2 \subseteq \mathbb{E}_2$, $x_1 \in \mathbb{E}_1$, $x_2 \in \mathbb{E}_2$.

Definition 2.20 (Weakly posetal product). Given two PESs $\mathcal{E}_1$, $\mathcal{E}_2$, the weakly posetal product of their configurations, denoted $\mathcal{C}(\mathcal{E}_1) \overline{\times} \mathcal{C}(\mathcal{E}_2)$, is defined as

$$\{(C_1, f, C_2) | C_1 \in \mathcal{C}(\mathcal{E}_1), C_2 \in \mathcal{C}(\mathcal{E}_2), f : \hat{C_1} \to \hat{C_2} \text{ isomorphism}\}.$$

A subset $R \subseteq \mathcal{C}(\mathcal{E}_1) \overline{\times} \mathcal{C}(\mathcal{E}_2)$ is called a weakly posetal relation. We say that R is downward closed when for any $(C_1, f, C_2), (C_1', f, C_2') \in \mathcal{C}(\mathcal{E}_1) \overline{\times} \mathcal{C}(\mathcal{E}_2)$, if $(C_1, f, C_2) \subseteq (C_1', f', C_2')$ pointwise and $(C_1', f', C_2') \in R$, then $(C_1, f, C_2) \in R$.

For $f : X_1 \to X_2$, we define $f[x_1 \mapsto x_2] : X_1 \cup \{x_1\} \to X_2 \cup \{x_2\}$, $z \in X_1 \cup \{x_1\}$,(1)$f[x_1 \mapsto x_2](z) = x_2$,if $z = x_1$;(2)$f[x_1 \mapsto x_2](z) = f(z)$, otherwise. Where $X_1 \subseteq \hat{\mathbb{E}_1}$, $X_2 \subseteq \hat{\mathbb{E}_2}$, $x_1 \in \hat{\mathbb{E}_1}$, $x_2 \in \hat{\mathbb{E}_2}$. Also, we define $f(\tau^*) = f(\tau^*)$.

Definition 2.21 ((Hereditary) history-preserving bisimulation). A history-preserving (hp-) bisimulation is a posetal relation $R \subseteq \mathcal{C}(\mathcal{E}_1) \overline{\times} \mathcal{C}(\mathcal{E}_2)$ such that if $(C_1, f, C_2) \in R$, and $C_1 \stackrel{e_1}{\longrightarrow} C_1'$, then $C_2 \stackrel{e_2}{\longrightarrow} C_2'$, with $(C_1', f[e_1 \mapsto e_2], C_2') \in R$, and vice-versa. $\mathcal{E}_1, \mathcal{E}_2$ are history-preserving (hp-)bisimilar and are written $\mathcal{E}_1 \sim_{hp} \mathcal{E}_2$ if there exists a hp-bisimulation R such that $(\varnothing, \varnothing, \varnothing) \in R$.

A hereditary history-preserving (hhp-)bisimulation is a downward closed hp-bisimulation. $\mathcal{E}_1, \mathcal{E}_2$ are hereditary history-preserving (hhp-)bisimilar and are written $\mathcal{E}_1 \sim_{hhp} \mathcal{E}_2$.

Definition 2.22 (Weak (hereditary) history-preserving bisimulation). A weak history-preserving (hp-) bisimulation is a weakly posetal relation $R \subseteq \mathcal{C}(\mathcal{E}_1) \overline{\times} \mathcal{C}(\mathcal{E}_2)$ such that if $(C_1, f, C_2) \in R$, and $C_1 \stackrel{e_1}{\Longrightarrow} C_1'$, then

No.	Axiom
$A1$	$x + y = y + x$
$A2$	$(x + y) + z = x + (y + z)$
$A3$	$x + x = x$
$A4$	$(x + y) \cdot z = x \cdot z + y \cdot z$
$A5$	$(x \cdot y) \cdot z = x \cdot (y \cdot z)$

Table 1. Axioms of $BATC$

$C_2 \stackrel{e_2}{\Longrightarrow} C_2'$, with $(C_1', f[e_1 \mapsto e_2], C_2') \in R$, and vice-versa. $\mathcal{E}_1, \mathcal{E}_2$ are weak history-preserving (hp-)bisimilar and are written $\mathcal{E}_1 \approx_{hp} \mathcal{E}_2$ if there exists a hp-bisimulation R such that $(\varnothing, \varnothing, \varnothing) \in R$.

A weakly hereditary history-preserving (hhp-)bisimulation is a downward closed weak hp-bisimulation. $\mathcal{E}_1, \mathcal{E}_2$ are weakly hereditary history-preserving (hhp-)bisimilar and are written $\mathcal{E}_1 \approx_{hhp} \mathcal{E}_2$.

Proposition 2.23 (Weakly concurrent behavioral equivalence). (Strongly) concurrent behavioral equivalences imply weakly concurrent behavioral equivalences. That is, $\sim_p$ implies $\approx_p$, $\sim_s$ implies $\approx_s$, $\sim_{hp}$ implies $\approx_{hp}$, $\sim_{hhp}$ implies $\approx_{hhp}$.

Proof. From the definition of weak pomset transition, weak step transition, weakly posetal product and weakly concurrent behavioral equivalence, it is easy to see that $\stackrel{e}{\rightarrow} = \stackrel{\epsilon}{\rightarrow} \stackrel{e}{\rightarrow} \stackrel{\epsilon}{\rightarrow}$ for $e \in \mathbb{E}$, where ϵ is the empty event. □

Note that in the above definitions, truly concurrent behavioral equivalences are defined by events $e \in \mathcal{E}$ and prime event structure $\mathcal{E}$, in contrast to interleaving behavioral equivalences by actions $a, b \in \mathcal{P}$ and process (graph) $\mathcal{P}$. Indeed, they have correspondences, in [13], models of concurrency, including Petri nets, transition systems and event structures, are unified in a uniform representation – TSI (Transition System with Independence). If x is a process, let $C(x)$ denote the corresponding configuration (the already executed part of the process x, of course, it is free of conflicts), when $x \stackrel{e}{\rightarrow} x'$, the corresponding configuration $C(x) \stackrel{e}{\rightarrow} C(x')$ with $C(x') = C(x) \cup \{e\}$, where e may be caused by some events in $C(x)$ and concurrent with the other events in $C(x)$, or entirely concurrent with all events in $C(x)$, or entirely caused by all events in $C(x)$. Though the concurrent behavioral equivalences (Definition 2.17, 2.18, 2.21 and 2.22) are defined based on configurations (pasts of processes), they can also be defined based on processes (futures of configurations), we omit the concrete definitions. One key difference between definitions based on configurations and processes is that, the definitions based on configurations are stressing the structures of two equivalent configurations and the concrete atomic events may be different, but the definitions based on processes require not only the equivalent structures, but also the same atomic events by their labels, since we try to establish the algebraic equations modulo the corresponding concurrent behavior equivalences.

With a little abuse of concepts, in the following of the paper, we will not distinguish actions and events, prime event structures and processes, also concurrent behavior equivalences based on configurations and processes, and use them freely, unless they have specific meanings. Usually, in congruence theorem and soundness, we show them in a structure only flavor (equivalences based on configuration); but in proof of the completeness theorem, we must require not only the equivalent structure, but also the same set of atomic events.

3. Basic Algebra for True Concurrency

In this section, we will discuss the algebraic laws for prime event structure $\mathcal{E}$, exactly for causality $\leq$ and conflict $\sharp$. We will follow the conventions of process algebra, using $\cdot$ instead of $\leq$ and $+$ instead of $\sharp$. The resulted algebra is called Basic Algebra for True Concurrency, abbreviated $BATC$.

3.1. Axiom System of $BATC$

In the following, let $e_1, e_2, e_1', e_2' \in \mathbb{E}$, and let variables x, y, z range over the set of terms for true concurrency, p, q, s range over the set of closed terms. The set of axioms of $BATC$ consists of the laws given in Table 1.

No.	Rewriting Rule
$RA3$	$x + x \to x$
$RA4$	$(x + y) \cdot z \to x \cdot z + y \cdot z$
$RA5$	$(x \cdot y) \cdot z \to x \cdot (y \cdot z)$

Table 2. Term rewrite system of $BATC$

Intuitively, the axiom $A1$ says that the binary operator + satisfies commutative law. The axiom $A2$ says that + satisfies associativity. $A3$ says that + satisfies idempotency. The axiom $A4$ is the right distributivity of the binary operator $\cdot$ to +. And $A5$ is the associativity of $\cdot$.

3.2. Properties of $BATC$

Definition 3.1 (Basic terms of $BATC$). The set of basic terms of $BATC$, $\mathcal{B}(BATC)$, is inductively defined as follows:

1. $\mathbb{E} \subset \mathcal{B}(BATC)$;
2. if $e \in \mathbb{E}, t \in \mathcal{B}(BATC)$ then $e \cdot t \in \mathcal{B}(BATC)$;
3. if $t, s \in \mathcal{B}(BATC)$ then $t + s \in \mathcal{B}(BATC)$.

Theorem 3.2 (Elimination theorem of $BATC$). Let p be a closed $BATC$ term. Then there is a basic $BATC$ term q such that $BATC \vdash p = q$.

Proof. (1) Firstly, suppose that the following ordering on the signature of $BATC$ is defined: $\cdot > +$ and the symbol $\cdot$ is given the lexicographical status for the first argument, then for each rewrite rule $p \to q$ in Table 2 relation $p >_{lpo} q$ can easily be proved. We obtain that the term rewrite system shown in Table 2 is strongly normalizing, for it has finitely many rewriting rules, and $>$ is a well-founded ordering on the signature of $BATC$, and if $s >_{lpo} t$, for each rewriting rule $s \to t$ is in Table 2 (see Theorem 2.12).

(2) Then we prove that the normal forms of closed $BATC$ terms are basic $BATC$ terms.

Suppose that p is a normal form of some closed $BATC$ term and suppose that p is not a basic term. Let p' denote the smallest sub-term of p which is not a basic term. It implies that each sub-term of p' is a basic term. Then we prove that p is not a term in normal form. It is sufficient to induct on the structure of p':

- Case $p' \equiv e, e \in \mathbb{E}$. p' is a basic term, which contradicts the assumption that p' is not a basic term, so this case should not occur.
- Case $p' \equiv p_1 \cdot p_2$. By induction on the structure of the basic term p_1:
  - Subcase $p_1 \in \mathbb{E}$. p' would be a basic term, which contradicts the assumption that p' is not a basic term;
  - Subcase $p_1 \equiv e \cdot p_1'$. $RA5$ rewriting rule can be applied. So p is not a normal form;
  - Subcase $p_1 \equiv p_1' + p_1''$. $RA4$ rewriting rule can be applied. So p is not a normal form.
- Case $p' \equiv p_1 + p_2$. By induction on the structure of the basic terms both p_1 and p_2, all subcases will lead to that p' would be a basic term, which contradicts the assumption that p' is not a basic term.

□

3.3. Structured Operational Semantics of $BATC$

In this subsection, we will define a term-deduction system which gives the operational semantics of $BATC$. We give the operational transition rules for operators $\cdot$ and + as Table 3 shows. And the predicate $\xrightarrow{e} \surd$ represents successful termination after execution of the event e.

$$\frac{}{e \xrightarrow{e} \surd}$$

$$\frac{x \xrightarrow{e} \surd}{x + y \xrightarrow{e} \surd} \quad \frac{x \xrightarrow{e} x'}{x + y \xrightarrow{e} x'} \quad \frac{y \xrightarrow{e} \surd}{x + y \xrightarrow{e} \surd} \quad \frac{y \xrightarrow{e} y'}{x + y \xrightarrow{e} y'}$$

$$\frac{x \xrightarrow{e} \surd}{x \cdot y \xrightarrow{e} y} \quad \frac{x \xrightarrow{e} x'}{x \cdot y \xrightarrow{e} x' \cdot y}$$

Table 3. Single event transition rules of $BATC$

$$\frac{}{X \xrightarrow{X} \surd}$$

$$\frac{x \xrightarrow{X} \surd}{x + y \xrightarrow{X} \surd}(X \subseteq x) \quad \frac{x \xrightarrow{X} x'}{x + y \xrightarrow{X} x'}(X \subseteq x) \quad \frac{y \xrightarrow{Y} \surd}{x + y \xrightarrow{Y} \surd}(Y \subseteq y) \quad \frac{y \xrightarrow{Y} y'}{x + y \xrightarrow{Y} y'}(Y \subseteq y)$$

$$\frac{x \xrightarrow{X} \surd}{x \cdot y \xrightarrow{X} y}(X \subseteq x) \quad \frac{x \xrightarrow{X} x'}{x \cdot y \xrightarrow{X} x' \cdot y}(X \subseteq x)$$

Table 4. Pomset transition rules of $BATC$

Theorem 3.3 (Congruence of $BATC$ with respect to bisimulation equivalence). Bisimulation equivalence $\sim_{HM}$ is a congruence with respect to $BATC$.

Proof. The axioms in Table 1 of $BATC$ are the same as the axioms of BPA (Basic Process Algebra) [1] [2] [4], so, bisimulation equivalence $\sim_{HM}$ is a congruence with respect to $BATC$. □

Theorem 3.4 (Soundness of $BATC$ modulo bisimulation equivalence). Let x and y be $BATC$ terms. If $BATC \vdash x = y$, then $x \sim_{HM} y$.

Proof. The axioms in Table 1 of $BATC$ are the same as the axioms of BPA (Basic Process Algebra) [1] [2] [4], so, $BATC$ is sound modulo bisimulation equivalence. □

Theorem 3.5 (Completeness of $BATC$ modulo bisimulation equivalence). Let p and q be closed $BATC$ terms, if $p \sim_{HM} q$ then $p = q$.

Proof. The axioms in Table 1 of $BATC$ are the same as the axioms of BPA (Basic Process Algebra) [1] [2] [4], so, $BATC$ is complete modulo bisimulation equivalence. □

The pomset transition rules are shown in Table 4, different to single event transition rules in Table 3, the pomset transition rules are labeled by pomsets, which are defined by causality $\cdot$ and conflict $+$.

Theorem 3.6 (Congruence of $BATC$ with respect to pomset bisimulation equivalence). Pomset bisimulation equivalence $\sim_p$ is a congruence with respect to $BATC$.

Proof. It is easy to see that pomset bisimulation is an equivalent relation on $BATC$ terms, we only need to prove that $\sim_p$ is preserved by the operators $\cdot$ and $+$.

- Causality operator $\cdot$. Let x_1, x_2 and y_1, y_2 be $BATC$ processes, and $x_1 \sim_p y_1$, $x_2 \sim_p y_2$, it is sufficient to prove that $x_1 \cdot x_2 \sim_p y_1 \cdot y_2$.
  By the definition of pomset bisimulation $\sim_p$ (Definition 2.17), $x_1 \sim_p y_1$ means that

$$x_1 \xrightarrow{X_1} x_1' \quad y_1 \xrightarrow{Y_1} y_1'$$

  with $X_1 \subseteq x_1$, $Y_1 \subseteq y_1$, $X_1 \sim Y_1$ and $x_1' \sim_p y_1'$. The meaning of $x_2 \sim_p y_2$ is similar.
  By the pomset transition rules for causality operator $\cdot$ in Table 4, we can get

$$x_1 \cdot x_2 \xrightarrow{X_1} x_2 \quad y_1 \cdot y_2 \xrightarrow{Y_1} y_2$$

with $X_1 \subseteq x_1$, $Y_1 \subseteq y_1$, $X_1 \sim Y_1$ and $x_2 \sim_p y_2$, so, we get $x_1 \cdot x_2 \sim_p y_1 \cdot y_2$, as desired.
Or, we can get

$$x_1 \cdot x_2 \xrightarrow{X_1} x_1' \cdot x_2 \quad y_1 \cdot y_2 \xrightarrow{Y_1} y_1' \cdot y_2$$

with $X_1 \subseteq x_1$, $Y_1 \subseteq y_1$, $X_1 \sim Y_1$ and $x_1' \sim_p y_1'$, $x_2 \sim_p y_2$, so, we get $x_1 \cdot x_2 \sim_p y_1 \cdot y_2$, as desired.

- Conflict operator +. Let x_1, x_2 and y_1, y_2 be $BATC$ processes, and $x_1 \sim_p y_1$, $x_2 \sim_p y_2$, it is sufficient to prove that $x_1 + x_2 \sim_p y_1 + y_2$. The meanings of $x_1 \sim_p y_1$ and $x_2 \sim_p y_2$ are the same as the above case, according to the definition of pomset bisimulation $\sim_p$ in Definition 2.17.
  By the pomset transition rules for conflict operator + in Table 4, we can get four cases:

$$x_1 + x_2 \xrightarrow{X_1} \surd \quad y_1 + y_2 \xrightarrow{Y_1} \surd$$

with $X_1 \subseteq x_1$, $Y_1 \subseteq y_1$, $X_1 \sim Y_1$, so, we get $x_1 + x_2 \sim_p y_1 + y_2$, as desired.
Or, we can get

$$x_1 + x_2 \xrightarrow{X_1} x_1' \quad y_1 + y_2 \xrightarrow{Y_1} y_1'$$

with $X_1 \subseteq x_1$, $Y_1 \subseteq y_1$, $X_1 \sim Y_1$, and $x_1' \sim_p y_1'$, so, we get $x_1 + x_2 \sim_p y_1 + y_2$, as desired.
Or, we can get

$$x_1 + x_2 \xrightarrow{X_2} \surd \quad y_1 + y_2 \xrightarrow{Y_2} \surd$$

with $X_2 \subseteq x_2$, $Y_2 \subseteq y_2$, $X_2 \sim Y_2$, so, we get $x_1 + x_2 \sim_p y_1 + y_2$, as desired.
Or, we can get

$$x_1 + x_2 \xrightarrow{X_2} x_2' \quad y_1 + y_2 \xrightarrow{Y_2} y_2'$$

with $X_2 \subseteq x_2$, $Y_2 \subseteq y_2$, $X_2 \sim Y_2$, and $x_2' \sim_p y_2'$, so, we get $x_1 + x_2 \sim_p y_1 + y_2$, as desired.

□

Theorem 3.7 (Soundness of $BATC$ modulo pomset bisimulation equivalence). Let x and y be $BATC$ terms. If $BATC \vdash x = y$, then $x \sim_p y$.

Proof. Since pomset bisimulation $\sim_p$ is both an equivalent and a congruent relation, we only need to check if each axiom in Table 1 is sound modulo pomset bisimulation equivalence.

- **Axiom** $A1$. Let p, q be $BATC$ processes, and $p + q = q + p$, it is sufficient to prove that $p + q \sim_p q + p$. By the pomset transition rules for operator + in Table 4, we get

$$\frac{p \xrightarrow{P} \surd}{p + q \xrightarrow{P} \surd}(P \subseteq p) \quad \frac{p \xrightarrow{P} \surd}{q + p \xrightarrow{P} \surd}(P \subseteq p)$$

$$\frac{p \xrightarrow{P} p'}{p + q \xrightarrow{P} p'}(P \subseteq p) \quad \frac{p \xrightarrow{P} p'}{q + p \xrightarrow{P} p'}(P \subseteq p)$$

$$\frac{q \xrightarrow{Q} \surd}{p + q \xrightarrow{Q} \surd}(Q \subseteq q) \quad \frac{q \xrightarrow{Q} \surd}{q + p \xrightarrow{Q} \surd}(Q \subseteq q)$$

$$\frac{q \xrightarrow{Q} q'}{p + q \xrightarrow{Q} q'}(Q \subseteq q) \quad \frac{q \xrightarrow{Q} q'}{q + p \xrightarrow{Q} q'}(Q \subseteq q)$$

  So, $p + q \sim_p q + p$, as desired.
- **Axiom** $A2$. Let p, q, s be $BATC$ processes, and $(p + q) + s = p + (q + s)$, it is sufficient to prove that $(p + q) + s \sim_p p + (q + s)$. By the pomset transition rules for operator + in Table 4, we get

$$\frac{p \xrightarrow{P} \surd}{(p+q)+s \xrightarrow{P} \surd}(P \subseteq p) \quad \frac{p \xrightarrow{P} \surd}{p+(q+s) \xrightarrow{P} \surd}(P \subseteq p)$$

$$\frac{p \xrightarrow{P} p'}{(p+q)+s \xrightarrow{P} p'}(P \subseteq p) \quad \frac{p \xrightarrow{P} p'}{p+(q+s) \xrightarrow{P} p'}(P \subseteq p)$$

$$\frac{q \xrightarrow{Q} \surd}{(p+q)+s \xrightarrow{Q} \surd}(Q \subseteq q) \quad \frac{q \xrightarrow{Q} \surd}{p+(q+s) \xrightarrow{Q} \surd}(Q \subseteq q)$$

$$\frac{q \xrightarrow{Q} q'}{(p+q)+s \xrightarrow{Q} q'}(Q \subseteq q) \quad \frac{q \xrightarrow{Q} q'}{p+(q+s) \xrightarrow{Q} q'}(Q \subseteq q)$$

$$\frac{s \xrightarrow{S} \surd}{(p+q)+s \xrightarrow{S} \surd}(S \subseteq s) \quad \frac{s \xrightarrow{S} \surd}{p+(q+s) \xrightarrow{S} \surd}(S \subseteq s)$$

$$\frac{s \xrightarrow{S} s'}{(p+q)+s \xrightarrow{S} s'}(S \subseteq s) \quad \frac{s \xrightarrow{S} s'}{p+(q+s) \xrightarrow{S} s'}(S \subseteq s)$$

So, $(p+q)+s \sim_p p+(q+s)$, as desired.

- **Axiom** $A3$. Let p be a $BATC$ process, and $p+p=p$, it is sufficient to prove that $p+p \sim_p p$. By the pomset transition rules for operator $+$ in Table 4, we get

$$\frac{p \xrightarrow{P} \surd}{p+p \xrightarrow{P} \surd}(P \subseteq p) \quad \frac{p \xrightarrow{P} \surd}{p \xrightarrow{P} \surd}(P \subseteq p)$$

$$\frac{p \xrightarrow{P} p'}{p+p \xrightarrow{P} p'}(P \subseteq p) \quad \frac{p \xrightarrow{P} p'}{p \xrightarrow{P} p'}(P \subseteq p)$$

So, $p+p \sim_p p$, as desired.

- **Axiom** $A4$. Let p, q, s be $BATC$ processes, and $(p+q) \cdot s = p \cdot s + q \cdot s$, it is sufficient to prove that $(p+q) \cdot s \sim_p p \cdot s + q \cdot s$. By the pomset transition rules for operators $+$ and $\cdot$ in Table 4, we get

$$\frac{p \xrightarrow{P} \surd}{(p+q) \cdot s \xrightarrow{P} s}(P \subseteq p) \quad \frac{p \xrightarrow{P} \surd}{p \cdot s + q \cdot s \xrightarrow{P} s}(P \subseteq p)$$

$$\frac{p \xrightarrow{P} p'}{(p+q) \cdot s \xrightarrow{P} p' \cdot s}(P \subseteq p) \quad \frac{p \xrightarrow{P} p'}{p \cdot s + q \cdot s \xrightarrow{P} p' \cdot s}(P \subseteq p)$$

$$\frac{q \xrightarrow{Q} \surd}{(p+q) \cdot s \xrightarrow{Q} s}(Q \subseteq q) \quad \frac{q \xrightarrow{Q} \surd}{p \cdot s + q \cdot s \xrightarrow{Q} s}(Q \subseteq q)$$

$$\frac{q \xrightarrow{Q} q'}{(p+q) \cdot s \xrightarrow{Q} q' \cdot s}(Q \subseteq q) \quad \frac{q \xrightarrow{Q} q'}{p \cdot s + q \cdot s \xrightarrow{Q} q' \cdot s}(Q \subseteq q)$$

So, $(p+q) \cdot s \sim_p p \cdot s + q \cdot s$, as desired.

- **Axiom** $A5$. Let p,q,s be $BATC$ processes, and $(p\cdot q)\cdot s = p\cdot(q\cdot s)$, it is sufficient to prove that $(p\cdot q)\cdot s \sim_p p\cdot(q\cdot s)$. By the pomset transition rules for operator $\cdot$ in Table 4, we get

$$\frac{p\xrightarrow{P}\surd}{(p\cdot q)\cdot s\xrightarrow{P}q\cdot s}(P\subseteq p)\quad \frac{p\xrightarrow{P}\surd}{p\cdot(q\cdot s)\xrightarrow{P}q\cdot s}(P\subseteq p)$$

$$\frac{p\xrightarrow{P}p'}{(p\cdot q)\cdot s\xrightarrow{P}(p'\cdot q)\cdot s}(P\subseteq p)\quad \frac{p\xrightarrow{P}p'}{p\cdot(q\cdot s)\xrightarrow{P}p'\cdot(q\cdot s)}(P\subseteq p)$$

  With an assumption $(p'\cdot q)\cdot s = p'\cdot(q\cdot s)$, so, $(p\cdot q)\cdot s\sim_p p\cdot(q\cdot s)$, as desired.

□

Theorem 3.8 (Completeness of $BATC$ modulo pomset bisimulation equivalence). Let p and q be closed $BATC$ terms, if $p\sim_p q$ then $p = q$.

Proof. Firstly, by the elimination theorem of $BATC$, we know that for each closed $BATC$ term p, there exists a closed basic $BATC$ term p', such that $BATC\vdash p = p'$, so, we only need to consider closed basic $BATC$ terms.

The basic terms (see Definition 3.1) modulo associativity and commutativity (AC) of conflict + (defined by axioms $A1$ and $A2$ in Table 1), and this equivalence is denoted by $=_{AC}$. Then, each equivalence class s modulo AC of + has the following normal form

$$s_1+\cdots+s_k$$

with each s_i either an atomic event or of the form $t_1\cdot t_2$, and each s_i is called the summand of s.

Now, we prove that for normal forms n and n', if $n\sim_p n'$ then $n =_{AC} n'$. It is sufficient to induct on the sizes of n and n'.

- Consider a summand e of n. Then $n\xrightarrow{e}\surd$, so $n\sim_p n'$ implies $n'\xrightarrow{e}\surd$, meaning that n' also contains the summand e.
- Consider a summand $t_1\cdot t_2$ of n. Then $n\xrightarrow{t_1}t_2$, so $n\sim_p n'$ implies $n'\xrightarrow{t_1}t_2'$ with $t_2\sim_p t_2'$, meaning that n' contains a summand $t_1\cdot t_2'$. Since t_2 and t_2' are normal forms and have sizes smaller than n and n', by the induction hypotheses $t_2\sim_p t_2'$ implies $t_2 =_{AC} t_2'$.

So, we get $n =_{AC} n'$.

Finally, let s and t be basic terms, and $s\sim_p t$, there are normal forms n and n', such that $s = n$ and $t = n'$. The soundness theorem of $BATC$ modulo pomset bisimulation equivalence (see Theorem 3.7) yields $s\sim_p n$ and $t\sim_p n'$, so $n\sim_p s\sim_p t\sim_p n'$. Since if $n\sim_p n'$ then $n =_{AC} n'$, $s = n =_{AC} n' = t$, as desired. □

The step transition rules are defined in Table 5, different to pomset transition rules, the step transition rules are labeled by steps, in which every event is pairwise concurrent.

Theorem 3.9 (Congruence of $BATC$ with respect to step bisimulation equivalence). Step bisimulation equivalence $\sim_s$ is a congruence with respect to $BATC$.

Proof. It is easy to see that step bisimulation is an equivalent relation on $BATC$ terms, we only need to prove that $\sim_s$ is preserved by the operators $\cdot$ and +.

- Causality operator $\cdot$. Let x_1,x_2 and y_1,y_2 be $BATC$ processes, and $x_1\sim_s y_1$, $x_2\sim_s y_2$, it is sufficient to prove that $x_1\cdot x_2\sim_s y_1\cdot y_2$.
  By the definition of step bisimulation $\sim_s$ (Definition 2.17), $x_1\sim_s y_1$ means that

$$x_1\xrightarrow{X_1}x_1'\quad y_1\xrightarrow{Y_1}y_1'$$

  with $X_1\subseteq x_1$, $\forall e_1,e_2\in X_1$ are pairwise concurrent, $Y_1\subseteq y_1$, $\forall e_1,e_2\in Y_1$ are pairwise concurrent, $X_1\sim Y_1$ and $x_1'\sim_s y_1'$. The meaning of $x_2\sim_s y_2$ is similar.
  By the step transition rules for causality operator $\cdot$ in Table 5, we can get

$$\frac{}{X \xrightarrow{X} \surd}(\forall e_1, e_2 \in X \text{ are pairwise concurrent.})$$

$$\frac{x \xrightarrow{X} \surd}{x + y \xrightarrow{X} \surd}(X \subseteq x, \forall e_1, e_2 \in X \text{ are pairwise concurrent.})$$

$$\frac{x \xrightarrow{X} x'}{x + y \xrightarrow{X} x'}(X \subseteq x, \forall e_1, e_2 \in X \text{ are pairwise concurrent.})$$

$$\frac{y \xrightarrow{Y} \surd}{x + y \xrightarrow{Y} \surd}(Y \subseteq y, \forall e_1, e_2 \in Y \text{ are pairwise concurrent.})$$

$$\frac{y \xrightarrow{Y} y'}{x + y \xrightarrow{Y} y'}(Y \subseteq y, \forall e_1, e_2 \in Y \text{ are pairwise concurrent.})$$

$$\frac{x \xrightarrow{X} \surd}{x \cdot y \xrightarrow{X} y}(X \subseteq x, \forall e_1, e_2 \in X \text{ are pairwise concurrent.})$$

$$\frac{x \xrightarrow{X} x'}{x \cdot y \xrightarrow{X} x' \cdot y}(X \subseteq x, \forall e_1, e_2 \in X \text{ are pairwise concurrent.})$$

Table 5. Step transition rules of $BATC$

$$x_1 \cdot x_2 \xrightarrow{X_1} x_2 \quad y_1 \cdot y_2 \xrightarrow{Y_1} y_2$$

with $X_1 \subseteq x_1$, $\forall e_1, e_2 \in X_1$ are pairwise concurrent, $Y_1 \subseteq y_1$, $\forall e_1, e_2 \in Y_1$ are pairwise concurrent, $X_1 \sim Y_1$ and $x_2 \sim_s y_2$, so, we get $x_1 \cdot x_2 \sim_s y_1 \cdot y_2$, as desired.
Or, we can get

$$x_1 \cdot x_2 \xrightarrow{X_1} x_1' \cdot x_2 \quad y_1 \cdot y_2 \xrightarrow{Y_1} y_1' \cdot y_2$$

with $X_1 \subseteq x_1$, $\forall e_1, e_2 \in X_1$ are pairwise concurrent., $Y_1 \subseteq y_1$, $\forall e_1, e_2 \in Y_1$ are pairwise concurrent., $X_1 \sim Y_1$ and $x_1' \sim_s y_1'$, $x_2 \sim_s y_2$, so, we get $x_1 \cdot x_2 \sim_s y_1 \cdot y_2$, as desired.

- Conflict operator +. Let x_1, x_2 and y_1, y_2 be $BATC$ processes, and $x_1 \sim_s y_1$, $x_2 \sim_s y_2$, it is sufficient to prove that $x_1 + x_2 \sim_s y_1 + y_2$. The meanings of $x_1 \sim_s y_1$ and $x_2 \sim_s y_2$ are the same as the above case, according to the definition of step bisimulation $\sim_s$ in Definition 2.17.
  By the step transition rules for conflict operator + in Table 5, we can get four cases:

$$x_1 + x_2 \xrightarrow{X_1} \surd \quad y_1 + y_2 \xrightarrow{Y_1} \surd$$

with $X_1 \subseteq x_1$, $\forall e_1, e_2 \in X_1$ are pairwise concurrent, $Y_1 \subseteq y_1$, $\forall e_1, e_2 \in Y_1$ are pairwise concurrent, $X_1 \sim Y_1$, so, we get $x_1 + x_2 \sim_s y_1 + y_2$, as desired.
Or, we can get

$$x_1 + x_2 \xrightarrow{X_1} x_1' \quad y_1 + y_2 \xrightarrow{Y_1} y_1'$$

with $X_1 \subseteq x_1$, $\forall e_1, e_2 \in X_1$ are pairwise concurrent, $Y_1 \subseteq y_1$, $\forall e_1, e_2 \in Y_1$ are pairwise concurrent, $X_1 \sim Y_1$, and $x_1' \sim_s y_1'$, so, we get $x_1 + x_2 \sim_s y_1 + y_2$, as desired.
Or, we can get

$$x_1 + x_2 \xrightarrow{X_2} \surd \quad y_1 + y_2 \xrightarrow{Y_2} \surd$$

with $X_2 \subseteq x_2$, $\forall e_1, e_2 \in X_2$ are pairwise concurrent, $Y_2 \subseteq y_2$, $\forall e_1, e_2 \in Y_2$ are pairwise concurrent, $X_2 \sim Y_2$, so, we get $x_1 + x_2 \sim_s y_1 + y_2$, as desired.
Or, we can get

$$x_1 + x_2 \xrightarrow{X_2} x_2' \quad y_1 + y_2 \xrightarrow{Y_2} y_2'$$

with $X_2 \subseteq x_2$, $\forall e_1, e_2 \in X_2$ are pairwise concurrent, $Y_2 \subseteq y_2$, $\forall e_1, e_2 \in Y_2$ are pairwise concurrent, $X_2 \sim Y_2$, and $x_2' \sim_s y_2'$, so, we get $x_1 + x_2 \sim_s y_1 + y_2$, as desired.

□

Theorem 3.10 (Soundness of $BATC$ modulo step bisimulation equivalence). Let x and y be $BATC$ terms. If $BATC \vdash x = y$, then $x \sim_s y$.

Proof. Since step bisimulation $\sim_s$ is both an equivalent and a congruent relation, we only need to check if each axiom in Table 1 is sound modulo step bisimulation equivalence.

- **Axiom** $A1$. Let p, q be $BATC$ processes, and $p + q = q + p$, it is sufficient to prove that $p + q \sim_s q + p$. By the step transition rules for operator + in Table 5, we get

$$\frac{p \xrightarrow{P} \surd}{p + q \xrightarrow{P} \surd}(P \subseteq p, \forall e_1, e_2 \in P \text{ are pairwise concurrent.})$$

$$\frac{p \xrightarrow{P} \surd}{q + p \xrightarrow{P} \surd}(P \subseteq p, \forall e_1, e_2 \in P \text{ are pairwise concurrent.})$$

$$\frac{p \xrightarrow{P} p'}{p + q \xrightarrow{P} p'}(P \subseteq p, \forall e_1, e_2 \in P \text{ are pairwise concurrent.})$$

$$\frac{p \xrightarrow{P} p'}{q + p \xrightarrow{P} p'}(P \subseteq p, \forall e_1, e_2 \in P \text{ are pairwise concurrent.})$$

$$\frac{q \xrightarrow{Q} \surd}{p + q \xrightarrow{Q} \surd}(Q \subseteq q, \forall e_1, e_2 \in Q \text{ are pairwise concurrent.})$$

$$\frac{q \xrightarrow{Q} \surd}{q + p \xrightarrow{Q} \surd}(Q \subseteq q, \forall e_1, e_2 \in Q \text{ are pairwise concurrent.})$$

$$\frac{q \xrightarrow{Q} q'}{p + q \xrightarrow{Q} q'}(Q \subseteq q, \forall e_1, e_2 \in Q \text{ are pairwise concurrent.})$$

$$\frac{q \xrightarrow{Q} q'}{q + p \xrightarrow{Q} q'}(Q \subseteq q, \forall e_1, e_2 \in Q \text{ are pairwise concurrent.})$$

  So, $p + q \sim_s q + p$, as desired.
- **Axiom** $A2$. Let p, q, s be $BATC$ processes, and $(p + q) + s = p + (q + s)$, it is sufficient to prove that $(p + q) + s \sim_s p + (q + s)$. By the step transition rules for operator + in Table 5, we get

$$\frac{p \xrightarrow{P} \surd}{(p + q) + s \xrightarrow{P} \surd}(P \subseteq p, \forall e_1, e_2 \in P \text{ are pairwise concurrent.})$$

$$\frac{p \xrightarrow{P} \surd}{p + (q + s) \xrightarrow{P} \surd}(P \subseteq p, \forall e_1, e_2 \in P \text{ are pairwise concurrent.})$$

$$\frac{p \xrightarrow{P} p'}{(p+q)+s \xrightarrow{P} p'}(P \subseteq p, \forall e_1, e_2 \in P \text{ are pairwise concurrent.})$$

$$\frac{p \xrightarrow{P} p'}{p+(q+s) \xrightarrow{P} p'}(P \subseteq p, \forall e_1, e_2 \in P \text{ are pairwise concurrent.})$$

$$\frac{q \xrightarrow{Q} \surd}{(p+q)+s \xrightarrow{Q} \surd}(Q \subseteq q, \forall e_1, e_2 \in Q \text{ are pairwise concurrent.})$$

$$\frac{q \xrightarrow{Q} \surd}{p+(q+s) \xrightarrow{Q} \surd}(Q \subseteq q, \forall e_1, e_2 \in Q \text{ are pairwise concurrent.})$$

$$\frac{q \xrightarrow{Q} q'}{(p+q)+s \xrightarrow{Q} q'}(Q \subseteq q, \forall e_1, e_2 \in Q \text{ are pairwise concurrent.})$$

$$\frac{q \xrightarrow{Q} q'}{p+(q+s) \xrightarrow{Q} q'}(Q \subseteq q, \forall e_1, e_2 \in Q \text{ are pairwise concurrent.})$$

$$\frac{s \xrightarrow{S} \surd}{(p+q)+s \xrightarrow{S} \surd}(S \subseteq s, \forall e_1, e_2 \in S \text{ are pairwise concurrent.})$$

$$\frac{s \xrightarrow{S} \surd}{p+(q+s) \xrightarrow{S} \surd}(S \subseteq s, \forall e_1, e_2 \in S \text{ are pairwise concurrent.})$$

$$\frac{s \xrightarrow{S} s'}{(p+q)+s \xrightarrow{S} s'}(S \subseteq s, \forall e_1, e_2 \in S \text{ are pairwise concurrent.})$$

$$\frac{s \xrightarrow{S} s'}{p+(q+s) \xrightarrow{S} s'}(S \subseteq s, \forall e_1, e_2 \in S \text{ are pairwise concurrent.})$$

So, $(p+q)+s \sim_s p+(q+s)$, as desired.

- **Axiom** $A3$. Let p be a $BATC$ process, and $p+p=p$, it is sufficient to prove that $p+p \sim_s p$. By the step transition rules for operator + in Table 5, we get

$$\frac{p \xrightarrow{P} \surd}{p+p \xrightarrow{P} \surd}(P \subseteq p, \forall e_1, e_2 \in P \text{ are pairwise concurrent.})$$

$$\frac{p \xrightarrow{P} \surd}{p \xrightarrow{P} \surd}(P \subseteq p, \forall e_1, e_2 \in P \text{ are pairwise concurrent.})$$

$$\frac{p \xrightarrow{P} p'}{p+p \xrightarrow{P} p'}(P \subseteq p, \forall e_1, e_2 \in P \text{ are pairwise concurrent.})$$

$$\frac{p \xrightarrow{P} p'}{p \xrightarrow{P} p'}(P \subseteq p, \forall e_1, e_2 \in P \text{ are pairwise concurrent.})$$

So, $p+p \sim_s p$, as desired.

- **Axiom** $A4$. Let p,q,s be $BATC$ processes, and $(p+q)\cdot s = p\cdot s + q\cdot s$, it is sufficient to prove that $(p+q)\cdot s \sim_s p\cdot s + q\cdot s$. By the step transition rules for operators + and $\cdot$ in Table 5, we get

$$\frac{p \xrightarrow{P} \surd}{(p+q)\cdot s \xrightarrow{P} s}(P \subseteq p, \forall e_1,e_2 \in P \text{ are pairwise concurrent.})$$

$$\frac{p \xrightarrow{P} \surd}{p\cdot s + q\cdot s \xrightarrow{P} s}(P \subseteq p, \forall e_1,e_2 \in P \text{ are pairwise concurrent.})$$

$$\frac{p \xrightarrow{P} p'}{(p+q)\cdot s \xrightarrow{P} p'\cdot s}(P \subseteq p, \forall e_1,e_2 \in P \text{ are pairwise concurrent.})$$

$$\frac{p \xrightarrow{P} p'}{p\cdot s + q\cdot s \xrightarrow{P} p'\cdot s}(P \subseteq p, \forall e_1,e_2 \in P \text{ are pairwise concurrent.})$$

$$\frac{q \xrightarrow{Q} \surd}{(p+q)\cdot s \xrightarrow{Q} s}(Q \subseteq q, \forall e_1,e_2 \in Q \text{ are pairwise concurrent.})$$

$$\frac{q \xrightarrow{Q} \surd}{p\cdot s + q\cdot s \xrightarrow{Q} s}(Q \subseteq q, \forall e_1,e_2 \in Q \text{ are pairwise concurrent.})$$

$$\frac{q \xrightarrow{Q} q'}{(p+q)\cdot s \xrightarrow{Q} q'\cdot s}(Q \subseteq q, \forall e_1,e_2 \in Q \text{ are pairwise concurrent.})$$

$$\frac{q \xrightarrow{Q} q'}{p\cdot s + q\cdot s \xrightarrow{Q} q'\cdot s}(Q \subseteq q, \forall e_1,e_2 \in Q \text{ are pairwise concurrent.})$$

So, $(p+q)\cdot s \sim_s p\cdot s + q\cdot s$, as desired.

- **Axiom** $A5$. Let p,q,s be $BATC$ processes, and $(p\cdot q)\cdot s = p\cdot(q\cdot s)$, it is sufficient to prove that $(p\cdot q)\cdot s \sim_s p\cdot(q\cdot s)$. By the step transition rules for operator $\cdot$ in Table 5, we get

$$\frac{p \xrightarrow{P} \surd}{(p\cdot q)\cdot s \xrightarrow{P} q\cdot s}(P \subseteq p, \forall e_1,e_2 \in P \text{ are pairwise concurrent.})$$

$$\frac{p \xrightarrow{P} \surd}{p\cdot(q\cdot s) \xrightarrow{P} q\cdot s}(P \subseteq p, \forall e_1,e_2 \in P \text{ are pairwise concurrent.})$$

$$\frac{p \xrightarrow{P} p'}{(p\cdot q)\cdot s \xrightarrow{P} (p'\cdot q)\cdot s}(P \subseteq p, \forall e_1,e_2 \in P \text{ are pairwise concurrent.})$$

$$\frac{p \xrightarrow{P} p'}{p\cdot(q\cdot s) \xrightarrow{P} p'\cdot(q\cdot s)}(P \subseteq p, \forall e_1,e_2 \in P \text{ are pairwise concurrent.})$$

With an assumption $(p'\cdot q)\cdot s = p'\cdot(q\cdot s)$, so, $(p\cdot q)\cdot s \sim_s p\cdot(q\cdot s)$, as desired.

□

$$\frac{}{e \xrightarrow{e} \surd}$$

$$\frac{x \xrightarrow{e} \surd}{x+y \xrightarrow{e} \surd} \quad \frac{x \xrightarrow{e} x'}{x+y \xrightarrow{e} x'} \quad \frac{y \xrightarrow{e} \surd}{x+y \xrightarrow{e} \surd} \quad \frac{y \xrightarrow{e} y'}{x+y \xrightarrow{e} y'}$$

$$\frac{x \xrightarrow{e} \surd}{x \cdot y \xrightarrow{e} y} \quad \frac{x \xrightarrow{e} x'}{x \cdot y \xrightarrow{e} x' \cdot y}$$

Table 6. (Hereditary) hp-transition rules of $BATC$

Theorem 3.11 (Completeness of $BATC$ modulo step bisimulation equivalence). Let p and q be closed $BATC$ terms, if $p \sim_s q$ then $p = q$.

Proof. Firstly, by the elimination theorem of $BATC$, we know that for each closed $BATC$ term p, there exists a closed basic $BATC$ term p', such that $BATC \vdash p = p'$, so, we only need to consider closed basic $BATC$ terms.

The basic terms (see Definition 3.1) modulo associativity and commutativity (AC) of conflict $+$ (defined by axioms $A1$ and $A2$ in Table 1), and this equivalence is denoted by $=_{AC}$. Then, each equivalence class s modulo AC of $+$ has the following normal form

$$s_1 + \cdots + s_k$$

with each s_i either an atomic event or of the form $t_1 \cdot t_2$, and each s_i is called the summand of s.

Now, we prove that for normal forms n and n', if $n \sim_s n'$ then $n =_{AC} n'$. It is sufficient to induct on the sizes of n and n'.

- Consider a summand e of n. Then $n \xrightarrow{e} \surd$, so $n \sim_s n'$ implies $n' \xrightarrow{e} \surd$, meaning that n' also contains the summand e.
- Consider a summand $t_1 \cdot t_2$ of n. Then $n \xrightarrow{t_1} t_2 (\forall e_1, e_2 \in t_1$ are pairwise concurrent), so $n \sim_s n'$ implies $n' \xrightarrow{t_1} t_2' (\forall e_1, e_2 \in t_1$ are pairwise concurrent) with $t_2 \sim_s t_2'$, meaning that n' contains a summand $t_1 \cdot t_2'$. Since t_2 and t_2' are normal forms and have sizes smaller than n and n', by the induction hypotheses if $t_2 \sim_s t_2'$ then $t_2 =_{AC} t_2'$.

So, we get $n =_{AC} n'$.

Finally, let s and t be basic terms, and $s \sim_s t$, there are normal forms n and n', such that $s = n$ and $t = n'$. The soundness theorem of $BATC$ modulo step bisimulation equivalence (see Theorem 3.10) yields $s \sim_s n$ and $t \sim_s n'$, so $n \sim_s s \sim_s t \sim_s n'$. Since if $n \sim_s n'$ then $n =_{AC} n'$, $s = n =_{AC} n' = t$, as desired. □

The transition rules for (hereditary) hp-bisimulation of $BATC$ are defined in Table 6, they are the same as single event transition rules in Table 3.

Theorem 3.12 (Congruence of $BATC$ with respect to hp-bisimulation equivalence). Hp-bisimulation equivalence $\sim_{hp}$ is a congruence with respect to $BATC$.

Proof. It is easy to see that history-preserving bisimulation is an equivalent relation on $BATC$ terms, we only need to prove that $\sim_{hp}$ is preserved by the operators $\cdot$ and $+$.

- Causality operator $\cdot$. Let x_1, x_2 and y_1, y_2 be $BATC$ processes, and $x_1 \sim_{hp} y_1$, $x_2 \sim_{hp} y_2$, it is sufficient to prove that $x_1 \cdot x_2 \sim_{hp} y_1 \cdot y_2$.
  By the definition of hp-bisimulation $\sim_{hp}$ (Definition 2.21), $x_1 \sim_{hp} y_1$ means that there is a posetal relation $(C(x_1), f, C(y_1)) \in \sim_{hp}$, and

$$x_1 \xrightarrow{e_1} x_1' \quad y_1 \xrightarrow{e_2} y_1'$$

  with $(C(x_1'), f[e_1 \mapsto e_2], C(y_1')) \in \sim_{hp}$. The meaning of $x_2 \sim_{hp} y_2$ is similar.
  By the hp-transition rules for causality operator $\cdot$ in Table 6, we can get

$$x_1 \cdot x_2 \xrightarrow{e_1} x_2 \quad y_1 \cdot y_2 \xrightarrow{e_2} y_2$$

with $x_2 \leftrightarrows_{hp} y_2$, so, we get $x_1 \cdot x_2 \leftrightarrows_{hp} y_1 \cdot y_2$, as desired.
Or, we can get

$$x_1 \cdot x_2 \xrightarrow{e_1} x_1' \cdot x_2 \quad y_1 \cdot y_2 \xrightarrow{e_2} y_1' \cdot y_2$$

with $x_1' \leftrightarrows_{hp} y_1'$, $x_2 \leftrightarrows_{hp} y_2$, so, we get $x_1 \cdot x_2 \leftrightarrows_{hp} y_1 \cdot y_2$, as desired.

- Conflict operator +. Let x_1, x_2 and y_1, y_2 be $BATC$ processes, and $x_1 \leftrightarrows_{hp} y_1$, $x_2 \leftrightarrows_{hp} y_2$, it is sufficient to prove that $x_1 + x_2 \leftrightarrows_{hp} y_1 + y_2$. The meanings of $x_1 \leftrightarrows_{hp} y_1$ and $x_2 \leftrightarrows_{hp} y_2$ are the same as the above case, according to the definition of hp-bisimulation $\leftrightarrows_{hp}$ in Definition 2.21.
  By the hp-transition rules for conflict operator + in Table 6, we can get four cases:

$$x_1 + x_2 \xrightarrow{e_1} \surd \quad y_1 + y_2 \xrightarrow{e_2} \surd$$

so, we get $x_1 + x_2 \leftrightarrows_{hp} y_1 + y_2$, as desired.
Or, we can get

$$x_1 + x_2 \xrightarrow{e_1} x_1' \quad y_1 + y_2 \xrightarrow{e_2} y_1'$$

with $x_1' \leftrightarrows_{hp} y_1'$, so, we get $x_1 + x_2 \leftrightarrows_{hp} y_1 + y_2$, as desired.
Or, we can get

$$x_1 + x_2 \xrightarrow{e_1'} \surd \quad y_1 + y_2 \xrightarrow{e_2'} \surd$$

so, we get $x_1 + x_2 \leftrightarrows_{hp} y_1 + y_2$, as desired.
Or, we can get

$$x_1 + x_2 \xrightarrow{e_1'} x_2' \quad y_1 + y_2 \xrightarrow{e_2'} y_2'$$

with $x_2' \leftrightarrows_{hp} y_2'$, so, we get $x_1 + x_2 \leftrightarrows_{hp} y_1 + y_2$, as desired.

□

Theorem 3.13 (Soundness of $BATC$ modulo hp-bisimulation equivalence). Let x and y be $BATC$ terms. If $BATC \vdash x = y$, then $x \leftrightarrows_{hp} y$.

Proof. Since hp-bisimulation $\leftrightarrows_{hp}$ is both an equivalent and a congruent relation, we only need to check if each axiom in Table 1 is sound modulo hp-bisimulation equivalence.

- **Axiom** $A1$. Let p, q be $BATC$ processes, and $p + q = q + p$, it is sufficient to prove that $p + q \leftrightarrows_{hp} q + p$. By the hp-transition rules for operator + in Table 6, we get

$$\frac{p \xrightarrow{e_1} \surd}{p + q \xrightarrow{P} \surd} \quad \frac{p \xrightarrow{e_1} \surd}{q + p \xrightarrow{e_1} \surd}$$

$$\frac{p \xrightarrow{e_1} p'}{p + q \xrightarrow{e_1} p'} \quad \frac{p \xrightarrow{e_1} p'}{q + p \xrightarrow{P} p'}$$

$$\frac{q \xrightarrow{e_2} \surd}{p + q \xrightarrow{e_2} \surd} \quad \frac{q \xrightarrow{e_2} \surd}{q + p \xrightarrow{e_2} \surd}$$

$$\frac{q \xrightarrow{e_2} q'}{p + q \xrightarrow{e_2} q'} \quad \frac{q \xrightarrow{e_2} q'}{q + p \xrightarrow{e_2} q'}$$

  So, for $(C(p + q), f, C(q + p)) \in \leftrightarrows_{hp}$, $(C((p + q)'), f[e_1 \mapsto e_1], C((q + p)')) \in \leftrightarrows_{hp}$ and $(C((p + q)'), f[e_2 \mapsto e_2], C((q + p)')) \in \leftrightarrows_{hp}$, that is, $p + q \leftrightarrows_{hp} q + p$, as desired.

- **Axiom** $A2$. Let p, q, s be $BATC$ processes, and $(p + q) + s = p + (q + s)$, it is sufficient to prove that $(p + q) + s \sim_{hp} p + (q + s)$. By the hp-transition rules for operator + in Table 6, we get

$$\frac{p \xrightarrow{e_1} \surd}{(p + q) + s \xrightarrow{e_1} \surd} \quad \frac{p \xrightarrow{e_1} \surd}{p + (q + s) \xrightarrow{e_1} \surd}$$

$$\frac{p \xrightarrow{e_1} p'}{(p + q) + s \xrightarrow{e_1} p'} \quad \frac{p \xrightarrow{e_1} p'}{p + (q + s) \xrightarrow{e_1} p'}$$

$$\frac{q \xrightarrow{e_2} \surd}{(p + q) + s \xrightarrow{e_2} \surd} \quad \frac{q \xrightarrow{e_2} \surd}{p + (q + s) \xrightarrow{e_2} \surd}$$

$$\frac{q \xrightarrow{e_2} q'}{(p + q) + s \xrightarrow{e_2} q'} \quad \frac{q \xrightarrow{e_2} q'}{p + (q + s) \xrightarrow{e_2} q'}$$

$$\frac{s \xrightarrow{e_3} \surd}{(p + q) + s \xrightarrow{e_3} \surd} \quad \frac{s \xrightarrow{e_3} \surd}{p + (q + s) \xrightarrow{e_3} \surd}$$

$$\frac{s \xrightarrow{e_3} s'}{(p + q) + s \xrightarrow{e_3} s'} \quad \frac{s \xrightarrow{e_3} s'}{p + (q + s) \xrightarrow{e_3} s'}$$

  So, for $(C((p + q) + s), f, C(p + (q + s))) \in \sim_{hp}$, $(C(((p + q) + s)'), f[e_1 \mapsto e_1], C((p + (q + s))')) \in \sim_{hp}$ and $(C(((p + q) + s)'), f[e_2 \mapsto e_2], C((p + (q + s))')) \in \sim_{hp}$ and $(C(((p + q) + s)'), f[e_3 \mapsto e_3], C((p + (q + s))')) \in \sim_{hp}$, that is, $(p + q) + s \sim_{hp} p + (q + s)$, as desired.
- **Axiom** $A3$. Let p be a $BATC$ process, and $p + p = p$, it is sufficient to prove that $p + p \sim_{hp} p$. By the hp-transition rules for operator + in Table 6, we get

$$\frac{p \xrightarrow{e_1} \surd}{p + p \xrightarrow{e_1} \surd} \quad \frac{p \xrightarrow{e_1} \surd}{p \xrightarrow{e_1} \surd}$$

$$\frac{p \xrightarrow{e_1} p'}{p + p \xrightarrow{e_1} p'} \quad \frac{p \xrightarrow{e_1} p'}{p \xrightarrow{e_1} p'}$$

  So, for $(C(p + p), f, C(p)) \in \sim_{hp}$, $(C((p + p)'), f[e_1 \mapsto e_1], C((p)')) \in \sim_{hp}$, that is, $p + p \sim_{hp} p$, as desired.
- **Axiom** $A4$. Let p, q, s be $BATC$ processes, and $(p + q) \cdot s = p \cdot s + q \cdot s$, it is sufficient to prove that $(p + q) \cdot s \sim_{hp} p \cdot s + q \cdot s$. By the hp-transition rules for operators + and $\cdot$ in Table 6, we get

$$\frac{p \xrightarrow{e_1} \surd}{(p + q) \cdot s \xrightarrow{e_1} s} \quad \frac{p \xrightarrow{e_1} \surd}{p \cdot s + q \cdot s \xrightarrow{P} s}$$

$$\frac{p \xrightarrow{e_1} p'}{(p + q) \cdot s \xrightarrow{e_1} p' \cdot s} \quad \frac{p \xrightarrow{e_1} p'}{p \cdot s + q \cdot s \xrightarrow{e_1} p' \cdot s}$$

$$\frac{q \xrightarrow{e_2} \surd}{(p + q) \cdot s \xrightarrow{e_2} s} \quad \frac{q \xrightarrow{e_2} \surd}{p \cdot s + q \cdot s \xrightarrow{e_2} s}$$

$$\frac{q \xrightarrow{e_2} q'}{(p + q) \cdot s \xrightarrow{e_2} q' \cdot s} \quad \frac{q \xrightarrow{e_2} q'}{p \cdot s + q \cdot s \xrightarrow{Q} q' \cdot s}$$

  So, for $(C((p + q) \cdot s), f, C(p \cdot s + q \cdot s)) \in \sim_{hp}$, $(C(((p + q) \cdot s)'), f[e_1 \mapsto e_1], C((p \cdot s + q \cdot s)')) \in \sim_{hp}$ and $(C(((p + q) \cdot s)'), f[e_2 \mapsto e_2], C((p \cdot s + q \cdot s)')) \in \sim_{hp}$, that is, $(p + q) \cdot s \sim_{hp} p \cdot s + q \cdot s$, as desired.

- **Axiom** $A5$. Let p,q,s be $BATC$ processes, and $(p\cdot q)\cdot s = p\cdot(q\cdot s)$, it is sufficient to prove that $(p\cdot q)\cdot s \sim_{hp} p\cdot(q\cdot s)$. By the hp-transition rules for operator $\cdot$ in Table 6, we get

$$\frac{p\xrightarrow{e_1}\surd}{(p\cdot q)\cdot s\xrightarrow{e_1}q\cdot s}\quad\frac{p\xrightarrow{e_1}\surd}{p\cdot(q\cdot s)\xrightarrow{e_1}q\cdot s}$$

$$\frac{p\xrightarrow{e_1}p'}{(p\cdot q)\cdot s\xrightarrow{e_1}(p'\cdot q)\cdot s}\quad\frac{p\xrightarrow{e_1}p'}{p\cdot(q\cdot s)\xrightarrow{e_1}p'\cdot(q\cdot s)}$$

  With an assumption $(p'\cdot q)\cdot s = p'\cdot(q\cdot s)$, for $(C((p\cdot q)\cdot s),f,C(p\cdot(q\cdot s)))\in\sim_{hp}$, $(C(((p\cdot q)\cdot s)'),f[e_1\mapsto e_1],C((p\cdot(q\cdot s))'))\in\sim_{hp}$, that is, so, $(p\cdot q)\cdot s\sim_{hp}p\cdot(q\cdot s)$, as desired.

□

Theorem 3.14 (Completeness of $BATC$ modulo hp-bisimulation equivalence). Let p and q be closed $BATC$ terms, if $p\sim_{hp}q$ then $p=q$.

Proof. Firstly, by the elimination theorem of $BATC$, we know that for each closed $BATC$ term p, there exists a closed basic $BATC$ term p', such that $BATC\vdash p=p'$, so, we only need to consider closed basic $BATC$ terms.

The basic terms (see Definition 3.1) modulo associativity and commutativity (AC) of conflict + (defined by axioms $A1$ and $A2$ in Table 1), and this equivalence is denoted by $=_{AC}$. Then, each equivalence class s modulo AC of + has the following normal form

$$s_1+\cdots+s_k$$

with each s_i either an atomic event or of the form $t_1\cdot t_2$, and each s_i is called the summand of s.

Now, we prove that for normal forms n and n', if $n\sim_{hp}n'$ then $n=_{AC}n'$. It is sufficient to induct on the sizes of n and n'.

- Consider a summand e of n. Then $n\xrightarrow{e}\surd$, so $n\sim_{hp}n'$ implies $n'\xrightarrow{e}\surd$, meaning that n' also contains the summand e.
- Consider a summand $e\cdot s$ of n. Then $n\xrightarrow{e}s$, so $n\sim_{hp}n'$ implies $n'\xrightarrow{e}t$ with $s\sim_{hp}t$, meaning that n' contains a summand $e\cdot t$. Since s and t are normal forms and have sizes smaller than n and n', by the induction hypotheses $s\sim_{hp}t$ implies $s=_{AC}t$.

So, we get $n=_{AC}n'$.

Finally, let s and t be basic terms, and $s\sim_{hp}t$, there are normal forms n and n', such that $s=n$ and $t=n'$. The soundness theorem of $BATC$ modulo hp-bisimulation equivalence (see Theorem 3.13) yields $s\sim_{hp}n$ and $t\sim_{hp}n'$, so $n\sim_{hp}s\sim_{hp}t\sim_{hp}n'$. Since if $n\sim_{hp}n'$ then $n=_{AC}n'$, $s=n=_{AC}n'=t$, as desired. □

Theorem 3.15 (Congruence of $BATC$ with respect to hhp-bisimulation equivalence). Hhp-bisimulation equivalence $\sim_{hhp}$ is a congruence with respect to $BATC$.

Proof. It is easy to see that hhp-bisimulation is an equivalent relation on $BATC$ terms, we only need to prove that $\sim_{hhp}$ is preserved by the operators $\cdot$ and +.

- Causality operator $\cdot$. Let x_1,x_2 and y_1,y_2 be $BATC$ processes, and $x_1\sim_{hhp}y_1$, $x_2\sim_{hhp}y_2$, it is sufficient to prove that $x_1\cdot x_2\sim_{hhp}y_1\cdot y_2$.
  By the definition of hhp-bisimulation $\sim_{hhp}$ (Definition 2.21), $x_1\sim_{hhp}y_1$ means that there is a posetal relation $(C(x_1),f,C(y_1))\in\sim_{hhp}$, and

$$x_1\xrightarrow{e_1}x_1'\quad y_1\xrightarrow{e_2}y_1'$$

  with $(C(x_1'),f[e_1\mapsto e_2],C(y_1'))\in\sim_{hhp}$. The meaning of $x_2\sim_{hhp}y_2$ is similar.
  By the hhp-transition rules for causality operator $\cdot$ in Table 6, we can get

$$x_1\cdot x_2\xrightarrow{e_1}x_2\quad y_1\cdot y_2\xrightarrow{e_2}y_2$$

  with $x_2\sim_{hhp}y_2$, so, we get $x_1\cdot x_2\sim_{hhp}y_1\cdot y_2$, as desired.

Or, we can get

$$x_1 \cdot x_2 \xrightarrow{e_1} x_1' \cdot x_2 \quad y_1 \cdot y_2 \xrightarrow{e_2} y_1' \cdot y_2$$

with $x_1' \sim_{hhp} y_1'$, $x_2 \sim_{hhp} y_2$, so, we get $x_1 \cdot x_2 \sim_{hhp} y_1 \cdot y_2$, as desired.

- Conflict operator +. Let x_1, x_2 and y_1, y_2 be $BATC$ processes, and $x_1 \sim_{hhp} y_1$, $x_2 \sim_{hhp} y_2$, it is sufficient to prove that $x_1 + x_2 \sim_{hhp} y_1 + y_2$. The meanings of $x_1 \sim_{hhp} y_1$ and $x_2 \sim_{hhp} y_2$ are the same as the above case, according to the definition of hhp-bisimulation $\sim_{hhp}$ in Definition 2.21.
  By the hhp-transition rules for conflict operator + in Table 6, we can get four cases:

$$x_1 + x_2 \xrightarrow{e_1} \surd \quad y_1 + y_2 \xrightarrow{e_2} \surd$$

so, we get $x_1 + x_2 \sim_{hhp} y_1 + y_2$, as desired.
Or, we can get

$$x_1 + x_2 \xrightarrow{e_1} x_1' \quad y_1 + y_2 \xrightarrow{e_2} y_1'$$

with $x_1' \sim_{hhp} y_1'$, so, we get $x_1 + x_2 \sim_{hhp} y_1 + y_2$, as desired.
Or, we can get

$$x_1 + x_2 \xrightarrow{e_1'} \surd \quad y_1 + y_2 \xrightarrow{e_2'} \surd$$

so, we get $x_1 + x_2 \sim_{hhp} y_1 + y_2$, as desired.
Or, we can get

$$x_1 + x_2 \xrightarrow{e_1'} x_2' \quad y_1 + y_2 \xrightarrow{e_2'} y_2'$$

with $x_2' \sim_{hhp} y_2'$, so, we get $x_1 + x_2 \sim_{hhp} y_1 + y_2$, as desired.

□

Theorem 3.16 (Soundness of $BATC$ modulo hhp-bisimulation equivalence). Let x and y be $BATC$ terms. If $BATC \vdash x = y$, then $x \sim_{hhp} y$.

Proof. Since hhp-bisimulation $\sim_{hhp}$ is both an equivalent and a congruent relation, we only need to check if each axiom in Table 1 is sound modulo hhp-bisimulation equivalence.

- **Axiom** $A1$. Let p, q be $BATC$ processes, and $p + q = q + p$, it is sufficient to prove that $p + q \sim_{hhp} q + p$. By the hhp-transition rules for operator + in Table 6, we get

$$\frac{p \xrightarrow{e_1} \surd}{p + q \xrightarrow{P} \surd} \quad \frac{p \xrightarrow{e_1} \surd}{q + p \xrightarrow{e_1} \surd}$$

$$\frac{p \xrightarrow{e_1} p'}{p + q \xrightarrow{e_1} p'} \quad \frac{p \xrightarrow{e_1} p'}{q + p \xrightarrow{P} p'}$$

$$\frac{q \xrightarrow{e_2} \surd}{p + q \xrightarrow{e_2} \surd} \quad \frac{q \xrightarrow{e_2} \surd}{q + p \xrightarrow{e_2} \surd}$$

$$\frac{q \xrightarrow{e_2} q'}{p + q \xrightarrow{e_2} q'} \quad \frac{q \xrightarrow{e_2} q'}{q + p \xrightarrow{e_2} q'}$$

  So, for $(C(p+q), f, C(q+p)) \in \sim_{hhp}$, $(C((p+q)'), f[e_1 \mapsto e_1], C((q+p)')) \in \sim_{hhp}$ and $(C((p+q)'), f[e_2 \mapsto e_2], C((q+p)')) \in \sim_{hhp}$, that is, $p + q \sim_{hhp} q + p$, as desired.
- **Axiom** $A2$. Let p, q, s be $BATC$ processes, and $(p + q) + s = p + (q + s)$, it is sufficient to prove that $(p + q) + s \sim_{hhp} p + (q + s)$. By the hhp-transition rules for operator + in Table 6, we get

$$\frac{p \xrightarrow{e_1} \surd}{(p+q)+s \xrightarrow{e_1} \surd} \quad \frac{p \xrightarrow{e_1} \surd}{p+(q+s) \xrightarrow{e_1} \surd}$$

$$\frac{p \xrightarrow{e_1} p'}{(p+q)+s \xrightarrow{e_1} p'} \quad \frac{p \xrightarrow{e_1} p'}{p+(q+s) \xrightarrow{e_1} p'}$$

$$\frac{q \xrightarrow{e_2} \surd}{(p+q)+s \xrightarrow{e_2} \surd} \quad \frac{q \xrightarrow{e_2} \surd}{p+(q+s) \xrightarrow{e_2} \surd}$$

$$\frac{q \xrightarrow{e_2} q'}{(p+q)+s \xrightarrow{e_2} q'} \quad \frac{q \xrightarrow{e_2} q'}{p+(q+s) \xrightarrow{e_2} q'}$$

$$\frac{s \xrightarrow{e_3} \surd}{(p+q)+s \xrightarrow{e_3} \surd} \quad \frac{s \xrightarrow{e_3} \surd}{p+(q+s) \xrightarrow{e_3} \surd}$$

$$\frac{s \xrightarrow{e_3} s'}{(p+q)+s \xrightarrow{e_3} s'} \quad \frac{s \xrightarrow{e_3} s'}{p+(q+s) \xrightarrow{e_3} s'}$$

So, for $(C((p+q)+s), f, C(p+(q+s))) \in\sim_{hhp}$, $(C(((p+q)+s)'), f[e_1 \mapsto e_1], C((p+(q+s))')) \in\sim_{hhp}$ and $(C(((p+q)+s)'), f[e_2 \mapsto e_2], C((p+(q+s))')) \in\sim_{hhp}$ and $(C(((p+q)+s)'), f[e_3 \mapsto e_3], C((p+(q+s))')) \in\sim_{hhp}$, that is, $(p+q)+s \sim_{hhp} p+(q+s)$, as desired.

- **Axiom** $A3$. Let p be a $BATC$ process, and $p+p=p$, it is sufficient to prove that $p+p \sim_{hhp} p$. By the hhp-transition rules for operator $+$ in Table 6, we get

$$\frac{p \xrightarrow{e_1} \surd}{p+p \xrightarrow{e_1} \surd} \quad \frac{p \xrightarrow{e_1} \surd}{p \xrightarrow{e_1} \surd}$$

$$\frac{p \xrightarrow{e_1} p'}{p+p \xrightarrow{e_1} p'} \quad \frac{p \xrightarrow{e_1} p'}{p \xrightarrow{e_1} p'}$$

So, for $(C(p+p), f, C(p)) \in\sim_{hhp}$, $(C((p+p)'), f[e_1 \mapsto e_1], C((p)')) \in\sim_{hhp}$, that is, $p+p \sim_{hhp} p$, as desired.

- **Axiom** $A4$. Let p, q, s be $BATC$ processes, and $(p+q) \cdot s = p \cdot s + q \cdot s$, it is sufficient to prove that $(p+q) \cdot s \sim_{hhp} p \cdot s + q \cdot s$. By the hhp-transition rules for operators $+$ and $\cdot$ in Table 6, we get

$$\frac{p \xrightarrow{e_1} \surd}{(p+q) \cdot s \xrightarrow{e_1} s} \quad \frac{p \xrightarrow{e_1} \surd}{p \cdot s + q \cdot s \xrightarrow{P} s}$$

$$\frac{p \xrightarrow{e_1} p'}{(p+q) \cdot s \xrightarrow{e_1} p' \cdot s} \quad \frac{p \xrightarrow{e_1} p'}{p \cdot s + q \cdot s \xrightarrow{e_1} p' \cdot s}$$

$$\frac{q \xrightarrow{e_2} \surd}{(p+q) \cdot s \xrightarrow{e_2} s} \quad \frac{q \xrightarrow{e_2} \surd}{p \cdot s + q \cdot s \xrightarrow{e_2} s}$$

$$\frac{q \xrightarrow{e_2} q'}{(p+q) \cdot s \xrightarrow{e_2} q' \cdot s} \quad \frac{q \xrightarrow{e_2} q'}{p \cdot s + q \cdot s \xrightarrow{e_2} q' \cdot s}$$

So, for $(C((p+q) \cdot s), f, C(p \cdot s + q \cdot s)) \in\sim_{hhp}$, $(C(((p+q) \cdot s)'), f[e_1 \mapsto e_1], C((p \cdot s + q \cdot s)')) \in\sim_{hhp}$ and $(C(((p+q) \cdot s)'), f[e_2 \mapsto e_2], C((p \cdot s + q \cdot s)')) \in\sim_{hhp}$, that is, $(p+q) \cdot s \sim_{hhp} p \cdot s + q \cdot s$, as desired.

- **Axiom** $A5$. Let p, q, s be $BATC$ processes, and $(p \cdot q) \cdot s = p \cdot (q \cdot s)$, it is sufficient to prove that $(p \cdot q) \cdot s \sim_{hhp} p \cdot (q \cdot s)$. By the hhp-transition rules for operator $\cdot$ in Table 6, we get

$$\frac{p \xrightarrow{e_1} \surd}{(p \cdot q) \cdot s \xrightarrow{e_1} q \cdot s} \quad \frac{p \xrightarrow{e_1} \surd}{p \cdot (q \cdot s) \xrightarrow{e_1} q \cdot s}$$

$$\frac{p \xrightarrow{e_1} p'}{(p \cdot q) \cdot s \xrightarrow{e_1} (p' \cdot q) \cdot s} \quad \frac{p \xrightarrow{e_1} p'}{p \cdot (q \cdot s) \xrightarrow{e_1} p' \cdot (q \cdot s)}$$

With an assumption $(p' \cdot q) \cdot s = p' \cdot (q \cdot s)$, for $(C((p \cdot q) \cdot s), f, C(p \cdot (q \cdot s))) \in \sim_{hhp}$, $(C(((p \cdot q) \cdot s)'), f[e_1 \mapsto e_1], C((p \cdot (q \cdot s))')) \in \sim_{hhp}$, that is, so, $(p \cdot q) \cdot s \sim_{hhp} p \cdot (q \cdot s)$, as desired.

□

Theorem 3.17 (Completeness of $BATC$ modulo hhp-bisimulation equivalence). Let p and q be closed $BATC$ terms, if $p \sim_{hhp} q$ then $p = q$.

Proof. Firstly, by the elimination theorem of $BATC$, we know that for each closed $BATC$ term p, there exists a closed basic $BATC$ term p', such that $BATC \vdash p = p'$, so, we only need to consider closed basic $BATC$ terms.

The basic terms (see Definition 3.1) modulo associativity and commutativity (AC) of conflict + (defined by axioms $A1$ and $A2$ in Table 1), and this equivalence is denoted by $=_{AC}$. Then, each equivalence class s modulo AC of + has the following normal form

$$s_1 + \cdots + s_k$$

with each s_i either an atomic event or of the form $t_1 \cdot t_2$, and each s_i is called the summand of s.

Now, we prove that for normal forms n and n', if $n \sim_{hhp} n'$ then $n =_{AC} n'$. It is sufficient to induct on the sizes of n and n'.

- Consider a summand e of n. Then $n \xrightarrow{e} \surd$, so $n \sim_{hhp} n'$ implies $n' \xrightarrow{e} \surd$, meaning that n' also contains the summand e.
- Consider a summand $e \cdot s$ of n. Then $n \xrightarrow{e} s$, so $n \sim_{hhp} n'$ implies $n' \xrightarrow{e} t$ with $s \sim_{hhp} t$, meaning that n' contains a summand $e \cdot t$. Since s and t are normal forms and have sizes smaller than n and n', by the induction hypotheses $s \sim_{hhp} t$ implies $s =_{AC} t$.

So, we get $n =_{AC} n'$.

Finally, let s and t be basic terms, and $s \sim_{hhp} t$, there are normal forms n and n', such that $s = n$ and $t = n'$. The soundness theorem of $BATC$ modulo history-preserving bisimulation equivalence (see Theorem 3.16) yields $s \sim_{hhp} n$ and $t \sim_{hhp} n'$, so $n \sim_{hhp} s \sim_{hhp} t \sim_{hhp} n'$. Since if $n \sim_{hhp} n'$ then $n =_{AC} n'$, $s = n =_{AC} n' = t$, as desired. □

4. Algebra for Parallelism in True Concurrency

In this section, we will discuss parallelism in true concurrency. We know that parallelism can be modeled by left merge and communication merge in ACP (Algebra of Communicating Process) [1] [4] with an interleaving bisimulation semantics. Parallelism in true concurrency is quite different to that in interleaving bisimulation: it is a fundamental computational pattern (modeled by parallel operator $\parallel$) and cannot be merged (replaced by other operators). The resulted algebra is called Algebra for Parallelism in True Concurrency, abbreviated $APTC$.

4.1. Parallelism as a Fundamental Computational Pattern

Through several propositions, we show that parallelism is a fundamental computational pattern. Firstly, we give the transition rules for parallel operator $\parallel$ as follows, it is suitable for all truly concurrent behavioral equivalence, including pomset bisimulation, step bisimulation, hp-bisimulation and hhp-bisimulation.

$$\frac{x \xrightarrow{e_1} \surd \quad y \xrightarrow{e_2} \surd}{x \parallel y \xrightarrow{\{e_1,e_2\}} \surd} \qquad \frac{x \xrightarrow{e_1} x' \quad y \xrightarrow{e_2} \surd}{x \parallel y \xrightarrow{\{e_1,e_2\}} x'}$$

$$\frac{x \xrightarrow{e_1} \surd \quad y \xrightarrow{e_2} y'}{x \parallel y \xrightarrow{\{e_1,e_2\}} y'} \qquad \frac{x \xrightarrow{e_1} x' \quad y \xrightarrow{e_2} y'}{x \parallel y \xrightarrow{\{e_1,e_2\}} x' \parallel y'}$$

Table 7. Transition rules of parallel operator $\parallel$

We will show that Milner's expansion law [1] does not hold modulo any truly concurrent behavioral equivalence, as the following proposition shows.

Proposition 4.1 (Milner's expansion law modulo truly concurrent behavioral equivalence). Milner's expansion law does not hold modulo any truly concurrent behavioral equivalence, that is:

1. For atomic event e_1 and e_2,
   (a) $e_1 \parallel e_2 \nsim_p e_1 \cdot e_2 + e_2 \cdot e_1$;
   (b) $e_1 \parallel e_2 \nsim_s e_1 \cdot e_2 + e_2 \cdot e_1$;
   (c) $e_1 \parallel e_2 \nsim_{hp} e_1 \cdot e_2 + e_2 \cdot e_1$;
   (d) $e_1 \parallel e_2 \nsim_{hhp} e_1 \cdot e_2 + e_2 \cdot e_1$;
2. Specially, for auto-concurrency, let e be an atomic event,
   (a) $e \parallel e \nsim_p e \cdot e$;
   (b) $e \parallel e \nsim_s e \cdot e$;
   (c) $e \parallel e \nsim_{hp} e \cdot e$;
   (d) $e \parallel e \nsim_{hhp} e \cdot e$.

Proof. In nature, it is caused by $e_1 \parallel e_2$ and $e_1 \cdot e_2 + e_2 \cdot e_1$ (specially $e \parallel e$ and $e \cdot e$) having different causality structure. They are based on the following obvious facts according to transition rules for parallel operator in Table 7:

1. $e_1 \parallel e_2 \xrightarrow{\{e_1,e_2\}} \surd$, while $e_1 \cdot e_2 + e_2 \cdot e_1 \not\rightarrow^{\{e_1,e_2\}}$;
2. specially, $e \parallel e \xrightarrow{\{e,e\}} \surd$, while $e \cdot e \not\rightarrow^{\{e,e\}}$.

□

In the following, we show that the elimination theorem does not hold for truly concurrent processes combined the operators $\cdot$, $+$ and $\parallel$. Firstly, we define the basic terms for $APTC$.

Definition 4.2 (Basic terms of $APTC$). The set of basic terms of $APTC$, $\mathcal{B}(APTC)$, is inductively defined as follows:

1. $\mathbb{E} \subset \mathcal{B}(APTC)$;
2. if $e \in \mathbb{E}, t \in \mathcal{B}(APTC)$ then $e \cdot t \in \mathcal{B}(APTC)$;
3. if $t, s \in \mathcal{B}(APTC)$ then $t + s \in \mathcal{B}(APTC)$;
4. if $t, s \in \mathcal{B}(APTC)$ then $t \parallel s \in \mathcal{B}(APTC)$.

Proposition 4.3 (About elimination theorem of $APTC$). 1. Let p be a closed $APTC$ term. Then there may not be a closed $BATC$ term q such that $APTC \vdash p = q$;
2. Let p be a closed $APTC$ term. Then there may not be a closed basic $APTC$ term q such that $APTC \vdash p = q$.

Proof. 1. By Proposition 4.1;
2. We show this property through two aspects:
   (a) The left and right distributivity of $\cdot$ to $\parallel$, and $\parallel$ to $\cdot$, do not hold modulo any truly concurrent bisimulation equivalence.

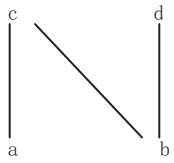

Fig. 1.

Left distributivity of $\cdot$ to $\parallel$: $(e_1 \cdot e_2) \parallel (e_1 \cdot e_3) \xrightarrow{\{e_1,e_1\}} e_2 \parallel e_3$, while $e_1 \cdot (e_2 \parallel e_3) \not\rightarrow^{\{e_1,e_1\}}$.
Right distributivity of $\cdot$ to $\parallel$: $(e_1 \cdot e_3) \parallel (e_2 \cdot e_3) \xrightarrow{\{e_1,e_2\}} e_3 \parallel e_3 \xrightarrow{\{e_3,e_3\}} \surd$, while $(e_1 \parallel e_2) \cdot e_3 \xrightarrow{\{e_1,e_2\}} e_3 \not\rightarrow^{\{e_3,e_3\}}$.
Left distributivity of $\parallel$ to $\cdot$: $(e_1 \parallel e_2) \cdot (e_1 \parallel e_3) \xrightarrow{\{e_1,e_2\}} e_1 \parallel e_3 \xrightarrow{\{e_1,e_3\}} \surd$, while $e_1 \parallel (e_2 \cdot e_3) \xrightarrow{\{e_1,e_2\}} e_3 \not\rightarrow^{\{e_1,e_3\}}$.
Right distributivity of $\parallel$ to $\cdot$: $(e_1 \parallel e_3) \cdot (e_2 \parallel e_3) \xrightarrow{\{e_1,e_3\}} e_2 \parallel e_3 \xrightarrow{\{e_2,e_3\}} \surd$, while $(e_1 \cdot e_2) \parallel e_3 \xrightarrow{\{e_1,e_3\}} e_2 \not\rightarrow^{\{e_2,e_3\}}$.
This means that there are not normal forms for the closed basic $APTC$ terms.

(b) There are causality relations among different parallel branches can not be expressed by closed basic $APTC$ terms.
We consider the graph as Fig. 1 illustrates. There are four events labeled a, b, c, d, and there are three causality relations: c after a, d after b, and c after b. This graph can not be expressed by basic $APTC$ terms. a and b are in parallel, c after a, so c and a are in the same parallel branch; d after b, so d and b are in the same parallel branch; so c and d are in different parallel branches. But, c after b means that c and d are in the same parallel branch. This causes contradictions, it means that the graph in Fig. 1 can not be expressed by closed basic $APTC$ terms.

□

Until now, we see that parallelism acts as a fundamental computational pattern, and any elimination theorem does not hold any more. In nature, an event structure $\mathcal{E}$ (see Definition 2.13) is a graph defined by causality and conflict relations among events, while concurrency and consistency are implicitly defined by causality and conflict. The above conclusions say that an event structure $\mathcal{E}$ cannot be fully structured, the explicit parallel operator $\parallel$ in a fully structured event structure combined by $\cdot$, $+$ and $\parallel$ can not be replaced by $\cdot$ and $+$, and a fully structured event structure combined by $\cdot$, $+$ and $\parallel$ has no a normal form.

The above propositions mean that a perfectly sound and complete axiomatization of parallelism for truly concurrent bisimulation equivalence (like ACP [4] for bisimulation equivalence) *cannot* be established. Then, what can we do for $APTC$?

4.2. Axiom System of Parallelism

Though a fully sound and complete axiomatization for $APTC$ seems impossible, we must and can do something, we believe. We also believe that the future is fully implied by the history, let us reconsider parallelism in interleaving bisimulation. In ACP [4], the full parallelism is captured by an auxiliary left merge and communication merge, left merge captures the interleaving bisimulation semantics, while communication merge expresses the communications among parallel branches. In true concurrency, if we try to define parallelism explicitly like $APTC$, the left merge captured Milner's expansion law does not hold any more, while communications among different parallel branches captured by communication merge still stand there. So, it is reasonable to assume that causality relations among different parallel branches are all communications among them. The communication between two parallel branches is defined as a communicating function between two communicating events $e_1, e_2 \in \mathbb{E}$, $\gamma(e_1, e_2) : \mathbb{E} \times \mathbb{E} \to \mathbb{E}$.

$$\frac{x \xrightarrow{e_1} \surd \quad y \xrightarrow{e_2} \surd}{x \mid y \xrightarrow{\gamma(e_1,e_2)} \surd} \qquad \frac{x \xrightarrow{e_1} x' \quad y \xrightarrow{e_2} \surd}{x \mid y \xrightarrow{\gamma(e_1,e_2)} x'}$$

$$\frac{x \xrightarrow{e_1} \surd \quad y \xrightarrow{e_2} y'}{x \mid y \xrightarrow{\gamma(e_1,e_2)} y'} \qquad \frac{x \xrightarrow{e_1} x' \quad y \xrightarrow{e_2} y'}{x \mid y \xrightarrow{\gamma(e_1,e_2)} x' \between y'}$$

Table 8. Transition rules of communication operator $\mid$

$$\frac{x \xrightarrow{e_1} \surd \quad (\natural(e_1,e_2))}{\Theta(x) \xrightarrow{e_1} \surd} \qquad \frac{x \xrightarrow{e_2} \surd \quad (\natural(e_1,e_2))}{\Theta(x) \xrightarrow{e_2} \surd}$$

$$\frac{x \xrightarrow{e_1} x' \quad (\natural(e_1,e_2))}{\Theta(x) \xrightarrow{e_1} \Theta(x')} \qquad \frac{x \xrightarrow{e_2} x' \quad (\natural(e_1,e_2))}{\Theta(x) \xrightarrow{e_2} \Theta(x')}$$

$$\frac{x \xrightarrow{e_1} \surd \quad y \not\rightarrow^{e_2} \quad (\natural(e_1,e_2))}{x \lhd y \xrightarrow{\tau} \surd} \qquad \frac{x \xrightarrow{e_1} x' \quad y \not\rightarrow^{e_2} \quad (\natural(e_1,e_2))}{x \lhd y \xrightarrow{\tau} x'}$$

$$\frac{x \xrightarrow{e_1} \surd \quad y \not\rightarrow^{e_3} \quad (\natural(e_1,e_2), e_2 \leq e_3)}{x \lhd y \xrightarrow{e_1} \surd} \qquad \frac{x \xrightarrow{e_1} x' \quad y \not\rightarrow^{e_3} \quad (\natural(e_1,e_2), e_2 \leq e_3)}{x \lhd y \xrightarrow{e_1} x'}$$

$$\frac{x \xrightarrow{e_3} \surd \quad y \not\rightarrow^{e_2} \quad (\natural(e_1,e_2), e_1 \leq e_3)}{x \lhd y \xrightarrow{\tau} \surd} \qquad \frac{x \xrightarrow{e_3} x' \quad y \not\rightarrow^{e_2} \quad (\natural(e_1,e_2), e_1 \leq e_3)}{x \lhd y \xrightarrow{\tau} x'}$$

Table 9. Transition rules of conflict elimination

The communications among parallel branches are still defined by the communication operator $\mid$, which is expressed by four transition rules in Table 8. The whole parallelism semantics is modeled by the parallel operator $\parallel$ and communication operator $\mid$, we denote the whole parallel operator as $\between$ (for the transition rules of $\between$, we omit them).

Note that the last transition rule for the parallel operator $\parallel$ in Table 7 should be modified to the following one.

$$\frac{x \xrightarrow{e_1} x' \quad y \xrightarrow{e_2} y'}{x \parallel y \xrightarrow{\{e_1,e_2\}} x' \between y'}$$

By communication operator $\mid$, the causality relation among different parallel branches are structured (we will show the algebra laws on communication operator in the following). Now, let us consider conflicts in parallelism. The conflicts exist within the same parallel branches can be captured by $+$ by a structured way, but, how to express conflicts among events in different parallel branches? The conflict relation is also a binary relation between two events $e_1, e_2 \in \mathbb{E}$, $\natural(e_1,e_2) : \mathbb{E} \times \mathbb{E} \to \mathbb{E}$, and we know that $\natural$ is irreflexive, symmetric and hereditary with respect to $\cdot$, that is, for all $e, e', e'' \in \mathbb{E}$, if $e \natural e' \cdot e''$, then $e \natural e''$ (see Definition 2.13).

These conflicts among different parallel branches must be eliminated to make the concurrent process structured. We are inspired by modeling of priority in ACP [4], the conflict elimination is also captured by two auxiliary operators, the unary conflict elimination operator Θ and the binary unless operator $\lhd$. The transition rules for Θ and $\lhd$ are expressed by ten transition rules in Table 9.

In four transition rules in Table 9, there is a new constant τ called silent step (see section 6), this makes the semantics of conflict elimination is really based on weakly true concurrency (see Definition 2.18 and Definition 2.22), and we should move it to section 6. But the movement would make $APTC$ incomplete (conflicts among different parallel branches cannot be expressed), let us forget this regret and just remember that τ can be eliminated, without anything on weakly true concurrency.

Ok, causality relations and conflict relations among events in different parallel branches are structured. In the following, we prove the congruence theorem.

Theorem 4.4 (Congruence theorem of $APTC$). Truly concurrent bisimulation equivalences $\sim_p$, $\sim_s$, $\sim_{hp}$ and $\sim_{hhp}$ are all congruences with respect to $APTC$.

Proof. (1) Case pomset bisimulation equivalence $\sim_p$.

- Case parallel operator $\parallel$. Let x_1, x_2 and y_1, y_2 be $APTC$ processes, and $x_1 \sim_p y_1$, $x_2 \sim_p y_2$, it is sufficient to prove that $x_1 \parallel x_2 \sim_p y_1 \parallel y_2$.
  By the definition of pomset bisimulation $\sim_p$ (Definition 2.17), $x_1 \sim_p y_1$ means that

$$x_1 \xrightarrow{X_1} x_1' \quad y_1 \xrightarrow{Y_1} y_1'$$

  with $X_1 \subseteq x_1$, $Y_1 \subseteq y_1$, $X_1 \sim Y_1$ and $x_1' \sim_p y_1'$. The meaning of $x_2 \sim_p y_2$ is similar.
  By the pomset transition rules for parallel operator $\parallel$ in Table 7, we can get

$$x_1 \parallel x_2 \xrightarrow{\{X_1,X_2\}} \surd \quad y_1 \parallel y_2 \xrightarrow{\{Y_1,Y_2\}} \surd$$

  with $X_1 \subseteq x_1$, $Y_1 \subseteq y_1$, $X_2 \subseteq x_2$, $Y_2 \subseteq y_2$, $X_1 \sim Y_1$ and $X_2 \sim Y_2$, so, we get $x_1 \parallel x_2 \sim_p y_1 \parallel y_2$, as desired.
  Or, we can get

$$x_1 \parallel x_2 \xrightarrow{\{X_1,X_2\}} x_1' \quad y_1 \parallel y_2 \xrightarrow{\{Y_1,Y_2\}} y_1'$$

  with $X_1 \subseteq x_1$, $Y_1 \subseteq y_1$, $X_2 \subseteq x_2$, $Y_2 \subseteq y_2$, $X_1 \sim Y_1$, $X_2 \sim Y_2$, and $x_1' \sim_p y_1'$, so, we get $x_1 \parallel x_2 \sim_p y_1 \parallel y_2$, as desired.
  Or, we can get

$$x_1 \parallel x_2 \xrightarrow{\{X_1,X_2\}} x_2' \quad y_1 \parallel y_2 \xrightarrow{\{Y_1,Y_2\}} y_2'$$

  with $X_1 \subseteq x_1$, $Y_1 \subseteq y_1$, $X_2 \subseteq x_2$, $Y_2 \subseteq y_2$, $X_1 \sim Y_1$, $X_2 \sim Y_2$, and $x_2' \sim_p y_2'$, so, we get $x_1 \parallel x_2 \sim_p y_1 \parallel y_2$, as desired.
  Or, we can get

$$x_1 \parallel x_2 \xrightarrow{\{X_1,X_2\}} x_1' \between x_2' \quad y_1 \parallel y_2 \xrightarrow{\{Y_1,Y_2\}} y_1' \between y_2'$$

  with $X_1 \subseteq x_1$, $Y_1 \subseteq y_1$, $X_2 \subseteq x_2$, $Y_2 \subseteq y_2$, $X_1 \sim Y_1$, $X_2 \sim Y_2$, $x_1' \sim_p y_1'$ and $x_2' \sim_p y_2'$, and also the assumption $x_1' \between x_2' \sim_p y_1' \between y_2'$, so, we get $x_1 \parallel x_2 \sim_p y_1 \parallel y_2$, as desired.
- Case communication operator $|$. It can be proved similarly to the case of parallel operator $\parallel$, we omit it. Note that, a communication is defined between two single communicating events.
- Case conflict elimination operator Θ. It can be proved similarly to the above cases, we omit it. Note that the conflict elimination operator Θ is a unary operator.
- Case unless operator $\triangleleft$. It can be proved similarly to the case of parallel operator $\parallel$, we omit it. Note that, a conflict relation is defined between two single events.

(2) The cases of step bisimulation $\sim_s$, hp-bisimulation $\sim_{hp}$ and hhp-bisimulation $\sim_{hhp}$ can be proven similarly, we omit them. □

So, we design the axioms of parallelism in Table 10, including algebraic laws for parallel operator $\parallel$, communication operator $|$, conflict elimination operator Θ and unless operator $\triangleleft$, and also the whole parallel operator $\between$. Since the communication between two communicating events in different parallel branches may cause deadlock (a state of inactivity), which is caused by mismatch of two communicating events or the imperfectness of the communication channel. We introduce a new constant δ to denote the deadlock, and let the atomic event $e \in \mathbb{E} \cup \{\delta\}$.

We explain the intuitions of the axioms of parallelism in Table 10 in the following. The axiom $A6$ says

No.	Axiom
$A6$	$x + \delta = x$
$A7$	$\delta \cdot x = \delta$
$P1$	$x \between y = x \parallel y + x \mid y$
$P2$	$x \parallel y = y \parallel x$
$P3$	$(x \parallel y) \parallel z = x \parallel (y \parallel z)$
$P4$	$e_1 \parallel (e_2 \cdot y) = (e_1 \parallel e_2) \cdot y$
$P5$	$(e_1 \cdot x) \parallel e_2 = (e_1 \parallel e_2) \cdot x$
$P6$	$(e_1 \cdot x) \parallel (e_2 \cdot y) = (e_1 \parallel e_2) \cdot (x \between y)$
$P7$	$(x + y) \parallel z = (x \parallel z) + (y \parallel z)$
$P8$	$x \parallel (y + z) = (x \parallel y) + (x \parallel z)$
$P9$	$\delta \parallel x = \delta$
$P10$	$x \parallel \delta = \delta$
$C11$	$e_1 \mid e_2 = \gamma(e_1, e_2)$
$C12$	$e_1 \mid (e_2 \cdot y) = \gamma(e_1, e_2) \cdot y$
$C13$	$(e_1 \cdot x) \mid e_2 = \gamma(e_1, e_2) \cdot x$
$C14$	$(e_1 \cdot x) \mid (e_2 \cdot y) = \gamma(e_1, e_2) \cdot (x \between y)$
$C15$	$(x + y) \mid z = (x \mid z) + (y \mid z)$
$C16$	$x \mid (y + z) = (x \mid y) + (x \mid z)$
$C17$	$\delta \mid x = \delta$
$C18$	$x \mid \delta = \delta$
$CE19$	$\Theta(e) = e$
$CE20$	$\Theta(\delta) = \delta$
$CE21$	$\Theta(x + y) = \Theta(x) \lhd y + \Theta(y) \lhd x$
$CE22$	$\Theta(x \cdot y) = \Theta(x) \cdot \Theta(y)$
$CE23$	$\Theta(x \parallel y) = ((\Theta(x) \lhd y) \parallel y) + ((\Theta(y) \lhd x) \parallel x)$
$CE24$	$\Theta(x \mid y) = ((\Theta(x) \lhd y) \mid y) + ((\Theta(y) \lhd x) \mid x)$
$U25$	$(\sharp(e_1, e_2)) \quad e_1 \lhd e_2 = \tau$
$U26$	$(\sharp(e_1, e_2), e_2 \leq e_3) \quad e_1 \lhd e_3 = e_1$
$U27$	$(\sharp(e_1, e_2), e_2 \leq e_3) \quad e3 \lhd e_1 = \tau$
$U28$	$e \lhd \delta = e$
$U29$	$\delta \lhd e = \delta$
$U30$	$(x + y) \lhd z = (x \lhd z) + (y \lhd z)$
$U31$	$(x \cdot y) \lhd z = (x \lhd z) \cdot (y \lhd z)$
$U32$	$(x \parallel y) \lhd z = (x \lhd z) \parallel (y \lhd z)$
$U33$	$(x \mid y) \lhd z = (x \lhd z) \mid (y \lhd z)$
$U34$	$x \lhd (y + z) = (x \lhd y) \lhd z$
$U35$	$x \lhd (y \cdot z) = (x \lhd y) \lhd z$
$U36$	$x \lhd (y \parallel z) = (x \lhd y) \lhd z$
$U37$	$x \lhd (y \mid z) = (x \lhd y) \lhd z$

Table 10. Axioms of parallelism

that the deadlock δ is redundant in the process term $t + \delta$. $A7$ says that the deadlock blocks all behaviors of the process term $\delta \cdot t$.

The axiom $P1$ is the definition of the whole parallelism $\between$, which says that $s \between t$ either is the form of $s \parallel t$ or $s \mid t$. $P2$ says that $\parallel$ satisfies commutative law, while $P3$ says that $\parallel$ satisfies associativity. $P4$, $P5$ and $P6$ are the defining axioms of $\parallel$, say the $s \parallel t$ executes s and t concurrently. $P7$ and $P8$ are the right and left distributivity of $\parallel$ to $+$. $P9$ and $P10$ say that both $\delta \parallel t$ and $t \parallel \delta$ all block any event.

$C11$, $C12$, $C13$ and $C14$ are the defining axioms of the communication operator $\mid$ which say that $s \mid t$ makes a communication between s and t. $C15$ and $C16$ are the right and left distributivity of $\mid$ to $+$. $C17$ and $C18$ say that both $\delta \mid t$ and $t \mid \delta$ all block any event.

$CE19$ and $CE20$ say that the conflict elimination operator Θ leaves atomic events and the deadlock unchanged. $CE21 - CE24$ are the functions of Θ acting on the operators $+$, $\cdot$, $\parallel$ and $\mid$. $U25$, $U26$ and $U27$ are the defining laws of the unless operator $\lhd$, in $U25$ and $U27$, there is a new constant τ, the silent step, we will discuss τ in details in section 6, in these two axioms, we just need to remember that τ really keeps silent. $U28$ says that the deadlock δ cannot block any event in the process term $e \lhd \delta$, while $U29$ says that $\delta \lhd e$ does not exhibit any behavior. $U30 - U37$ are the disguised right and left distributivity of $\lhd$ to the operators $+$, $\cdot$, $\parallel$ and $\mid$.

No.	Rewriting Rule
$RA6$	$x + \delta \to x$
$RA7$	$\delta \cdot x \to \delta$
$RP1$	$x \between y \to x \parallel y + x \mid y$
$RP2$	$x \parallel y \to y \parallel x$
$RP3$	$(x \parallel y) \parallel z \to x \parallel (y \parallel z)$
$RP4$	$e_1 \parallel (e_2 \cdot y) \to (e_1 \parallel e_2) \cdot y$
$RP5$	$(e_1 \cdot x) \parallel e_2 \to (e_1 \parallel e_2) \cdot x$
$RP6$	$(e_1 \cdot x) \parallel (e_2 \cdot y) \to (e_1 \parallel e_2) \cdot (x \between y)$
$RP7$	$(x + y) \parallel z \to (x \parallel z) + (y \parallel z)$
$RP8$	$x \parallel (y + z) \to (x \parallel y) + (x \parallel z)$
$RP9$	$\delta \parallel x \to \delta$
$RP10$	$x \parallel \delta \to \delta$
$RC11$	$e_1 \mid e_2 \to \gamma(e_1, e_2)$
$RC12$	$e_1 \mid (e_2 \cdot y) \to \gamma(e_1, e_2) \cdot y$
$RC13$	$(e_1 \cdot x) \mid e_2 \to \gamma(e_1, e_2) \cdot x$
$RC14$	$(e_1 \cdot x) \mid (e_2 \cdot y) \to \gamma(e_1, e_2) \cdot (x \between y)$
$RC15$	$(x + y) \mid z \to (x \mid z) + (y \mid z)$
$RC16$	$x \mid (y + z) \to (x \mid y) + (x \mid z)$
$RC17$	$\delta \mid x \to \delta$
$RC18$	$x \mid \delta \to \delta$
$RCE19$	$\Theta(e) \to e$
$RCE20$	$\Theta(\delta) \to \delta$
$RCE21$	$\Theta(x + y) \to \Theta(x) \lhd y + \Theta(y) \lhd x$
$RCE22$	$\Theta(x \cdot y) \to \Theta(x) \cdot \Theta(y)$
$RCE23$	$\Theta(x \parallel y) \to ((\Theta(x) \lhd y) \parallel y) + ((\Theta(y) \lhd x) \parallel x)$
$RCE24$	$\Theta(x \mid y) \to ((\Theta(x) \lhd y) \mid y) + ((\Theta(y) \lhd x) \mid x)$
$RU25$	$(\sharp(e_1, e_2)) \quad e_1 \lhd e_2 \to \tau$
$RU26$	$(\sharp(e_1, e_2), e_2 \cdot e_3) \quad e_1 \lhd e_3 \to e_1$
$RU27$	$(\sharp(e_1, e_2), e_2 \cdot e_3) \quad e3 \lhd e_1 \to \tau$
$RU28$	$e \lhd \delta \to e$
$RU29$	$\delta \lhd e \to \delta$
$RU30$	$(x + y) \lhd z \to (x \lhd z) + (y \lhd z)$
$RU31$	$(x \cdot y) \lhd z \to (x \lhd z) \cdot (y \lhd z)$
$RU32$	$(x \parallel y) \lhd z \to (x \lhd z) \parallel (y \lhd z)$
$RU33$	$(x \mid y) \lhd z \to (x \lhd z) \mid (y \lhd z)$
$RU34$	$x \lhd (y + z) \to (x \lhd y) \lhd z$
$RU35$	$x \lhd (y \cdot z) \to (x \lhd y) \lhd z$
$RU36$	$x \lhd (y \parallel z) \to (x \lhd y) \lhd z$
$RU37$	$x \lhd (y \mid z) \to (x \lhd y) \lhd z$

Table 11. Term rewrite system of $APTC$

4.3. Properties of Parallelism

Based on the definition of basic terms for $APTC$ (see Definition 4.2) and axioms of parallelism (see Table 10), we can prove the elimination theorem of parallelism.

Theorem 4.5 (Elimination theorem of parallelism). Let p be a closed $APTC$ term. Then there is a basic $APTC$ term q such that $APTC \vdash p = q$.

Proof. (1) Firstly, suppose that the following ordering on the signature of $APTC$ is defined: $\parallel > \cdot > +$ and the symbol $\parallel$ is given the lexicographical status for the first argument, then for each rewrite rule $p \to q$ in Table 11 relation $p >_{lpo} q$ can easily be proved. We obtain that the term rewrite system shown in Table 11 is strongly normalizing, for it has finitely many rewriting rules, and $>$ is a well-founded ordering on the signature of $APTC$, and if $s >_{lpo} t$, for each rewriting rule $s \to t$ is in Table 11 (see Theorem 2.12).

(2) Then we prove that the normal forms of closed $APTC$ terms are basic $APTC$ terms.

Suppose that p is a normal form of some closed $APTC$ term and suppose that p is not a basic $APTC$ term. Let p' denote the smallest sub-term of p which is not a basic $APTC$ term. It implies that each sub-term of p' is a basic $APTC$ term. Then we prove that p is not a term in normal form. It is sufficient to induct on the structure of p':

- Case $p' \equiv e, e \in \mathbb{E}$. p' is a basic $APTC$ term, which contradicts the assumption that p' is not a basic $APTC$ term, so this case should not occur.
- Case $p' \equiv p_1 \cdot p_2$. By induction on the structure of the basic $APTC$ term p_1:
  - Subcase $p_1 \in \mathbb{E}$. p' would be a basic $APTC$ term, which contradicts the assumption that p' is not a basic $APTC$ term;
  - Subcase $p_1 \equiv e \cdot p_1'$. $RA5$ rewriting rule in Table 2 can be applied. So p is not a normal form;
  - Subcase $p_1 \equiv p_1' + p_1''$. $RA4$ rewriting rule in Table 2 can be applied. So p is not a normal form;
  - Subcase $p_1 \equiv p_1' \between p_1''$. p' would be a basic $APTC$ term, which contradicts the assumption that p' is not a basic $APTC$ term;
  - Subcase $p_1 \equiv p_1' \mid p_1''$. $RC11$ rewrite rule in Table 11 can be applied. So p is not a normal form;
  - Subcase $p_1 \equiv \Theta(p_1')$. $RCE19$ and $RCE20$ rewrite rules in Table 11 can be applied. So p is not a normal form.
- Case $p' \equiv p_1 + p_2$. By induction on the structure of the basic $APTC$ terms both p_1 and p_2, all subcases will lead to that p' would be a basic $APTC$ term, which contradicts the assumption that p' is not a basic $APTC$ term.
- Case $p' \equiv p_1 \between p_2$. By induction on the structure of the basic $APTC$ terms both p_1 and p_2, all subcases will lead to that p' would be a basic $APTC$ term, which contradicts the assumption that p' is not a basic $APTC$ term.
- Case $p' \equiv p_1 \mid p_2$. By induction on the structure of the basic $APTC$ terms both p_1 and p_2, all subcases will lead to that p' would be a basic $APTC$ term, which contradicts the assumption that p' is not a basic $APTC$ term.
- Case $p' \equiv \Theta(p_1)$. By induction on the structure of the basic $APTC$ term p_1, $RCE19 - RCE24$ rewrite rules in Table 11 can be applied. So p is not a normal form.
- Case $p' \equiv p_1 \lhd p_2$. By induction on the structure of the basic $APTC$ terms both p_1 and p_2, all subcases will lead to that p' would be a basic $APTC$ term, which contradicts the assumption that p' is not a basic $APTC$ term.

□

4.4. Structured Operational Semantics of Parallelism

It is quite a challenge to prove the algebraic laws in Table 10 is sound/complete or unsound/incomplete modulo truly concurrent behavioral equivalence (pomset bisimulation equivalence, step bisimulation equivalence, hp-bisimulation equivalence and hhp-bisimulation equivalence), in this subsection, we try to do these.

Theorem 4.6 (Conservativity of the algebra for parallelism with respect to $BATC$). The algebra for parallelism is a conservative extension of $BATC$.

Proof. It follows from the following two facts (see Theorem 2.8).

1. The transition rules of $BATC$ in section 3 are all source-dependent;
2. The sources of the transition rules for the algebra for parallelism contain an occurrence of $\between$, or $\parallel$, or $\mid$, or Θ, or $\lhd$.

So, the algebra for parallelism is a conservative extension of $BATC$, as desired. □

Theorem 4.7 (Soundness of parallelism modulo step bisimulation equivalence). Let x and y be $APTC$ terms. If $APTC \vdash x = y$, then $x \sim_s y$.

Proof. Since step bisimulation $\sim_s$ is both an equivalent and a congruent relation with respect to the operators $\between$, $\parallel$, $\mid$, Θ and $\lhd$, we only need to check if each axiom in Table 10 is sound modulo step bisimulation equivalence.

Though transition rules in Table 7, 8, and 9 are defined in the flavor of single event, they can be modified into a step (a set of events within which each event is pairwise concurrent), we omit them. If we treat a single

event as a step containing just one event, the proof of this soundness theorem does not exist any problem, so we use this way and still use the transition rules in Table 7, 8, and 9.

We omit the defining axioms, including axioms $P1$, $C11$, $CE19$, $CE20$, $U25-U27$ (the soundness of $U25$ and $U27$ is remained to section 6); we also omit the trivial axioms related to δ, including axioms $A6$, $A7$, $P9$, $P10$, $C17$, $C18$, $U28$ and $U29$; in the following, we only prove the soundness of the non-trivial axioms, including axioms $P2-P8$, $C12-C16$, $CE21-CE24$ and $U30-U37$.

- **Axiom** $P2$. Let p,q be $APTC$ processes, and $p \parallel q = q \parallel p$, it is sufficient to prove that $p \parallel q \sim_s q \parallel p$. By the transition rules for operator $\parallel$ in Table 7, we get

$$\frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} \surd}{p \parallel q \xrightarrow{\{e_1,e_2\}} \surd} \quad \frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} \surd}{q \parallel p \xrightarrow{\{e_1,e_2\}} \surd}$$

$$\frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} \surd}{p \parallel q \xrightarrow{\{e_1,e_2\}} p'} \quad \frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} \surd}{q \parallel p \xrightarrow{\{e_1,e_2\}} p'}$$

$$\frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} q'}{p \parallel q \xrightarrow{\{e_1,e_2\}} q'} \quad \frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} q'}{q \parallel p \xrightarrow{\{e_1,e_2\}} q'}$$

$$\frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} q'}{p \parallel q \xrightarrow{\{e_1,e_2\}} p' \between q'} \quad \frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} q'}{q \parallel p \xrightarrow{\{e_1,e_2\}} q' \between p'}$$

  So, with the assumption $p' \between q' = q' \between p'$, $p \parallel q \sim_s q \parallel p$, as desired.

- **Axiom** $P3$. Let p,q,r be $APTC$ processes, and $(p \parallel q) \parallel r = p \parallel (q \parallel r)$, it is sufficient to prove that $(p \parallel q) \parallel r \sim_s p \parallel (q \parallel r)$. By the transition rules for operator $\parallel$ in Table 7, we get

$$\frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} \surd \quad r \xrightarrow{e_3} \surd}{(p \parallel q) \parallel r \xrightarrow{\{e_1,e_2,e_3\}} \surd} \quad \frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} \surd \quad r \xrightarrow{e_3} \surd}{p \parallel (q \parallel r) \xrightarrow{\{e_1,e_2,e_3\}} \surd}$$

$$\frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} \surd \quad r \xrightarrow{e_3} \surd}{(p \parallel q) \parallel r \xrightarrow{\{e_1,e_2,e_3\}} p'} \quad \frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} \surd \quad r \xrightarrow{e_3} \surd}{p \parallel (q \parallel r) \xrightarrow{\{e_1,e_2,e_3\}} p'}$$

  There are also two cases that two process terms successfully terminate, we omit them.

$$\frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} q' \quad r \xrightarrow{e_3} \surd}{(p \parallel q) \parallel r \xrightarrow{\{e_1,e_2,e_3\}} p' \between q'} \quad \frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} q' \quad r \xrightarrow{e_3} \surd}{p \parallel (q \parallel r) \xrightarrow{\{e_1,e_2,e_3\}} p' \between q'}$$

  There are also other cases that just one process term successfully terminate, we also omit them.

$$\frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} q' \quad r \xrightarrow{e_3} r'}{(p \parallel q) \parallel r' \xrightarrow{\{e_1,e_2,e_3\}} (p' \between q') \between r'} \quad \frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} q' \quad r \xrightarrow{e_3} r'}{p \parallel (q \parallel r) \xrightarrow{\{e_1,e_2,e_3\}} p' \between (q' \between r')}$$

  So, with the assumption $(p' \between q') \between r' = p' \between (q' \between r')$, $(p \parallel q) \parallel r \sim_s p \parallel (q \parallel r)$, as desired.

- **Axiom** $P4$. Let q be an $APTC$ process, and $e_1 \parallel (e_2 \cdot q) = (e_1 \parallel e_2) \cdot q$, it is sufficient to prove that $e_1 \parallel (e_2 \cdot q) \sim_s (e_1 \parallel e_2) \cdot q$. By the transition rules for operator $\parallel$ in Table 7, we get

$$\frac{e_1 \xrightarrow{e_1} \surd \quad e_2 \cdot q \xrightarrow{e_2} q}{e_1 \parallel (e_2 \cdot q) \xrightarrow{\{e_1,e_2\}} q}$$

$$\frac{e_1 \xrightarrow{e_1} \surd \quad e_2 \xrightarrow{e_2} \surd}{(e_1 \parallel e_2) \cdot q \xrightarrow{\{e_1,e_2\}} q}$$

So, $e_1 \parallel (e_2 \cdot q) \sim_s (e_1 \parallel e_2) \cdot q$, as desired.

- **Axiom** $P5$. Let p be an $APTC$ process, and $(e_1 \cdot p) \parallel e_2 = (e_1 \parallel e_2) \cdot p$, it is sufficient to prove that $(e_1 \cdot p) \parallel e_2 \sim_s (e_1 \parallel e_2) \cdot p$. By the transition rules for operator $\parallel$ in Table 7, we get

$$\frac{e_1 \cdot p \xrightarrow{e_1} p \quad e_2 \xrightarrow{e_2} \surd}{(e_1 \cdot p) \parallel e_2 \xrightarrow{\{e_1,e_2\}} p}$$

$$\frac{e_1 \xrightarrow{e_1} \surd \quad e_2 \xrightarrow{e_2} \surd}{(e_1 \parallel e_2) \cdot p \xrightarrow{\{e_1,e_2\}} p}$$

So, $(e_1 \cdot p) \parallel e_2 \sim_s (e_1 \parallel e_2) \cdot p$, as desired.

- **Axiom** $P6$. Let p, q be $APTC$ processes, and $(e_1 \cdot p) \parallel (e_2 \cdot q) = (e_1 \parallel e_2) \cdot (p \between q)$, it is sufficient to prove that $(e_1 \cdot p) \parallel (e_2 \cdot q) \sim_s (e_1 \parallel e_2) \cdot (p \between q)$. By the transition rules for operator $\parallel$ in Table 7, we get

$$\frac{e_1 \cdot p \xrightarrow{e_1} p \quad e_2 \cdot q \xrightarrow{e_2} q}{(e_1 \cdot p) \parallel (e_2 \cdot q) \xrightarrow{\{e_1,e_2\}} p \between q}$$

$$\frac{e_1 \xrightarrow{e_1} \surd \quad e_2 \xrightarrow{e_2} \surd}{(e_1 \parallel e_2) \cdot (p \between q) \xrightarrow{\{e_1,e_2\}} p \between q}$$

So, $(e_1 \cdot p) \parallel (e_2 \cdot q) \sim_s (e_1 \parallel e_2) \cdot (p \between q)$, as desired.

- **Axiom** $P7$. Let p, q, r be $APTC$ processes, and $(p + q) \parallel r = (p \parallel r) + (q \parallel r)$, it is sufficient to prove that $(p + q) \parallel r \sim_s (p \parallel r) + (q \parallel r)$. By the transition rules for operators $+$ and $\parallel$ in Table 5 and 7, we get

$$\frac{p \xrightarrow{e_1} \surd \quad r \xrightarrow{e_2} \surd}{(p + q) \parallel r \xrightarrow{\{e_1,e_2\}} \surd} \qquad \frac{p \xrightarrow{e_1} \surd \quad r \xrightarrow{e_2} \surd}{(p \parallel r) + (q \parallel r) \xrightarrow{\{e_1,e_2\}} \surd}$$

$$\frac{q \xrightarrow{e_1} \surd \quad r \xrightarrow{e_2} \surd}{(p + q) \parallel r \xrightarrow{\{e_1,e_2\}} \surd} \qquad \frac{q \xrightarrow{e_1} \surd \quad r \xrightarrow{e_2} \surd}{(p \parallel r) + (q \parallel r) \xrightarrow{\{e_1,e_2\}} \surd}$$

$$\frac{p \xrightarrow{e_1} p' \quad r \xrightarrow{e_2} \surd}{(p + q) \parallel r \xrightarrow{\{e_1,e_2\}} p'} \qquad \frac{p \xrightarrow{e_1} p' \quad r \xrightarrow{e_2} \surd}{(p \parallel r) + (q \parallel r) \xrightarrow{\{e_1,e_2\}} p'}$$

$$\frac{q \xrightarrow{e_1} q' \quad r \xrightarrow{e_2} \surd}{(p + q) \parallel r \xrightarrow{\{e_1,e_2\}} q'} \qquad \frac{q \xrightarrow{e_1} q' \quad r \xrightarrow{e_2} \surd}{(p \parallel r) + (q \parallel r) \xrightarrow{\{e_1,e_2\}} q'}$$

$$\frac{p \xrightarrow{e_1} \surd \quad r \xrightarrow{e_2} r'}{(p + q) \parallel r \xrightarrow{\{e_1,e_2\}} r'} \qquad \frac{p \xrightarrow{e_1} \surd \quad r \xrightarrow{e_2} r'}{(p \parallel r) + (q \parallel r) \xrightarrow{\{e_1,e_2\}} r'}$$

$$\frac{q \xrightarrow{e_1} \surd \quad r \xrightarrow{e_2} r'}{(p + q) \parallel r \xrightarrow{\{e_1,e_2\}} r'} \qquad \frac{q \xrightarrow{e_1} \surd \quad r \xrightarrow{e_2} r'}{(p \parallel r) + (q \parallel r) \xrightarrow{\{e_1,e_2\}} r'}$$

$$\frac{p \xrightarrow{e_1} p' \quad r \xrightarrow{e_2} r'}{(p + q) \parallel r \xrightarrow{\{e_1,e_2\}} p' \between r'} \qquad \frac{p \xrightarrow{e_1} p' \quad r \xrightarrow{e_2} r'}{(p \parallel r) + (q \parallel r) \xrightarrow{\{e_1,e_2\}} p' \between r'}$$

$$\frac{q \xrightarrow{e_1} q' \quad r \xrightarrow{e_2} r'}{(p+q) \parallel r \xrightarrow{\{e_1,e_2\}} q' \between r'} \qquad \frac{q \xrightarrow{e_1} q' \quad r \xrightarrow{e_2} r'}{(p \parallel r) + (q \parallel r) \xrightarrow{\{e_1,e_2\}} q' \between r'}$$

So, $(p+q) \parallel r \sim_s (p \parallel r) + (q \parallel r)$, as desired.

- **Axiom** $P8$. Let p, q, r be $APTC$ processes, and $p \parallel (q+r) = (p \parallel q) + (p \parallel r)$, it is sufficient to prove that $p \parallel (q+r) \sim_s (p \parallel q) + (p \parallel r)$. By the transition rules for operators + and $\parallel$ in Table 5 and 7, we get

$$\frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} \surd}{p \parallel (q+r) \xrightarrow{\{e_1,e_2\}} \surd} \qquad \frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} \surd}{(p \parallel q) + (p \parallel r) \xrightarrow{\{e_1,e_2\}} \surd}$$

$$\frac{p \xrightarrow{e_1} \surd \quad r \xrightarrow{e_2} \surd}{p \parallel (q+r) \xrightarrow{\{e_1,e_2\}} \surd} \qquad \frac{p \xrightarrow{e_1} \surd \quad r \xrightarrow{e_2} \surd}{(p \parallel q) + (p \parallel r) \xrightarrow{\{e_1,e_2\}} \surd}$$

$$\frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} \surd}{p \parallel (q+r) \xrightarrow{\{e_1,e_2\}} p'} \qquad \frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} \surd}{(p \parallel q) + (p \parallel r) \xrightarrow{\{e_1,e_2\}} p'}$$

$$\frac{p \xrightarrow{e_1} p' \quad r \xrightarrow{e_2} \surd}{p \parallel (q+r) \xrightarrow{\{e_1,e_2\}} p'} \qquad \frac{p \xrightarrow{e_1} p' \quad r \xrightarrow{e_2} \surd}{(p \parallel q) + (p \parallel r) \xrightarrow{\{e_1,e_2\}} p'}$$

$$\frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} q'}{p \parallel (q+r) \xrightarrow{\{e_1,e_2\}} q'} \qquad \frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} q'}{(p \parallel q) + (p \parallel r) \xrightarrow{\{e_1,e_2\}} q'}$$

$$\frac{p \xrightarrow{e_1} \surd \quad r \xrightarrow{e_2} r'}{p \parallel (q+r) \xrightarrow{\{e_1,e_2\}} r'} \qquad \frac{p \xrightarrow{e_1} \surd \quad r \xrightarrow{e_2} r'}{(p \parallel q) + (p \parallel r) \xrightarrow{\{e_1,e_2\}} r'}$$

$$\frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} q'}{p \parallel (q+r) \xrightarrow{\{e_1,e_2\}} p' \between q'} \qquad \frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} q'}{(p \parallel q) + (p \parallel r) \xrightarrow{\{e_1,e_2\}} p' \between q'}$$

$$\frac{p \xrightarrow{e_1} p' \quad r \xrightarrow{e_2} r'}{p \parallel (q+r) \xrightarrow{\{e_1,e_2\}} p' \between r'} \qquad \frac{p \xrightarrow{e_1} p' \quad r \xrightarrow{e_2} r'}{(p \parallel q) + (p \parallel r) \xrightarrow{\{e_1,e_2\}} p' \between r'}$$

So, $p \parallel (q+r) \sim_s (p \parallel q) + (p \parallel r)$, as desired.

- **Axiom** $C12$. Let q be an $APTC$ process, and $e_1 \mid (e_2 \cdot q) = \gamma(e_1, e_2) \cdot q$, it is sufficient to prove that $e_1 \mid (e_2 \cdot q) \sim_s \gamma(e_1, e_2) \cdot q$. By the transition rules for operator $\mid$ in Table 8, we get

$$\frac{e_1 \xrightarrow{e_1} \surd \quad e_2 \cdot q \xrightarrow{e_2} q}{e_1 \mid (e_2 \cdot q) \xrightarrow{\gamma(e_1,e_2)} q}$$

$$\frac{e_1 \xrightarrow{e_1} \surd \quad e_2 \xrightarrow{e_2} \surd}{\gamma(e_1, e_2) \cdot q \xrightarrow{\gamma(e_1,e_2)} q}$$

So, $e_1 \mid (e_2 \cdot q) \sim_s \gamma(e_1, e_2) \cdot q$, as desired.

- **Axiom** $C13$. Let p be an $APTC$ process, and $(e_1 \cdot p) \mid e_2 = \gamma(e_1, e_2) \cdot p$, it is sufficient to prove that $(e_1 \cdot p) \mid e_2 \sim_s \gamma(e_1, e_2) \cdot p$. By the transition rules for operator $\mid$ in Table 8, we get

$$\frac{e_1 \cdot p \xrightarrow{e_1} p \quad e_2 \xrightarrow{e_2} \surd}{(e_1 \cdot p) \mid e_2 \xrightarrow{\gamma(e_1,e_2)} p}$$

$$\frac{e_1 \xrightarrow{e_1} \surd \quad e_2 \xrightarrow{e_2} \surd}{\gamma(e_1,e_2)\cdot p \xrightarrow{\gamma(e_1,e_2)} p}$$

So, $(e_1 \cdot p) \mid e_2 \sim_s \gamma(e_1,e_2)\cdot p$, as desired.

- **Axiom** $C14$. Let p,q be $APTC$ processes, and $(e_1\cdot p) \mid (e_2\cdot q) = \gamma(e_1,e_2)\cdot(p \between q)$, it is sufficient to prove that $(e_1 \cdot p) \mid (e_2 \cdot q) \sim_s \gamma(e_1,e_2)\cdot(p \between q)$. By the transition rules for operator $\mid$ in Table 8, we get

$$\frac{e_1\cdot p \xrightarrow{e_1} p \quad e_2\cdot q \xrightarrow{e_2} q}{(e_1\cdot p)\mid(e_2\cdot q) \xrightarrow{\gamma(e_1,e_2)} p \between q}$$

$$\frac{e_1 \xrightarrow{e_1} \surd \quad e_2 \xrightarrow{e_2} \surd}{\gamma(e_1,e_2)\cdot(p \between q) \xrightarrow{\gamma(e_1,e_2)} p \between q}$$

So, $(e_1 \cdot p) \mid (e_2 \cdot q) \sim_s \gamma(e_1,e_2)\cdot(p \between q)$, as desired.

- **Axiom** $C15$. Let p,q,r be $APTC$ processes, and $(p+q)\mid r = (p\mid r)+(q\mid r)$, it is sufficient to prove that $(p+q)\mid r \sim_s (p\mid r)+(q\mid r)$. By the transition rules for operators $+$ and $\mid$ in Table 5 and 8, we get

$$\frac{p \xrightarrow{e_1} \surd \quad r \xrightarrow{e_2} \surd}{(p+q)\mid r \xrightarrow{\gamma(e_1,e_2)} \surd} \qquad \frac{p \xrightarrow{e_1} \surd \quad r \xrightarrow{e_2} \surd}{(p\mid r)+(q\mid r) \xrightarrow{\gamma(e_1,e_2)} \surd}$$

$$\frac{q \xrightarrow{e_1} \surd \quad r \xrightarrow{e_2} \surd}{(p+q)\mid r \xrightarrow{\gamma(e_1,e_2)} \surd} \qquad \frac{q \xrightarrow{e_1} \surd \quad r \xrightarrow{e_2} \surd}{(p\mid r)+(q\mid r) \xrightarrow{\gamma(e_1,e_2)} \surd}$$

$$\frac{p \xrightarrow{e_1} p' \quad r \xrightarrow{e_2} \surd}{(p+q)\mid r \xrightarrow{\gamma(e_1,e_2)} p'} \qquad \frac{p \xrightarrow{e_1} p' \quad r \xrightarrow{e_2} \surd}{(p\mid r)+(q\mid r) \xrightarrow{\gamma(e_1,e_2)} p'}$$

$$\frac{q \xrightarrow{e_1} q' \quad r \xrightarrow{e_2} \surd}{(p+q)\mid r \xrightarrow{\gamma(e_1,e_2)} q'} \qquad \frac{q \xrightarrow{e_1} q' \quad r \xrightarrow{e_2} \surd}{(p\mid r)+(q\mid r) \xrightarrow{\gamma(e_1,e_2)} q'}$$

$$\frac{p \xrightarrow{e_1} \surd \quad r \xrightarrow{e_2} r'}{(p+q)\mid r \xrightarrow{\gamma(e_1,e_2)} r'} \qquad \frac{p \xrightarrow{e_1} \surd \quad r \xrightarrow{e_2} r'}{(p\mid r)+(q\mid r) \xrightarrow{\gamma(e_1,e_2)} r'}$$

$$\frac{q \xrightarrow{e_1} \surd \quad r \xrightarrow{e_2} r'}{(p+q)\mid r \xrightarrow{\gamma(e_1,e_2)} r'} \qquad \frac{q \xrightarrow{e_1} \surd \quad r \xrightarrow{e_2} r'}{(p\mid r)+(q\mid r) \xrightarrow{\gamma(e_1,e_2)} r'}$$

$$\frac{p \xrightarrow{e_1} p' \quad r \xrightarrow{e_2} r'}{(p+q)\mid r \xrightarrow{\gamma(e_1,e_2)} p' \between r'} \qquad \frac{p \xrightarrow{e_1} p' \quad r \xrightarrow{e_2} r'}{(p\mid r)+(q\mid r) \xrightarrow{\gamma(e_1,e_2)} p' \between r'}$$

$$\frac{q \xrightarrow{e_1} q' \quad r \xrightarrow{e_2} r'}{(p+q)\mid r \xrightarrow{\gamma(e_1,e_2)} q' \between r'} \qquad \frac{q \xrightarrow{e_1} q' \quad r \xrightarrow{e_2} r'}{(p\mid r)+(q\mid r) \xrightarrow{\gamma(e_1,e_2)} q' \between r'}$$

So, $(p+q)\mid r \sim_s (p\mid r)+(q\mid r)$, as desired.

- **Axiom** $C16$. Let p,q,r be $APTC$ processes, and $p\mid(q+r) = (p\mid q)+(p\mid r)$, it is sufficient to prove that $p\mid(q+r) \sim_s (p\mid q)+(p\mid r)$. By the transition rules for operators $+$ and $\mid$ in Table 5 and 8, we get

$$\frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} \surd}{p\mid(q+r) \xrightarrow{\gamma(e_1,e_2)} \surd} \qquad \frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} \surd}{(p\mid q)+(p\mid r) \xrightarrow{\gamma(e_1,e_2)} \surd}$$

$$\frac{p \xrightarrow{e_1} \surd \quad r \xrightarrow{e_2} \surd}{p \mid (q+r) \xrightarrow{\gamma(e_1,e_2)} \surd} \qquad \frac{p \xrightarrow{e_1} \surd \quad r \xrightarrow{e_2} \surd}{(p \mid q) + (p \mid r) \xrightarrow{\gamma(e_1,e_2)} \surd}$$

$$\frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} \surd}{p \mid (q+r) \xrightarrow{\gamma(e_1,e_2)} p'} \qquad \frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} \surd}{(p \mid q) + (p \mid r) \xrightarrow{\gamma(e_1,e_2)} p'}$$

$$\frac{p \xrightarrow{e_1} p' \quad r \xrightarrow{e_2} \surd}{p \mid (q+r) \xrightarrow{\gamma(e_1,e_2)} p'} \qquad \frac{p \xrightarrow{e_1} p' \quad r \xrightarrow{e_2} \surd}{(p \mid q) + (p \mid r) \xrightarrow{\gamma(e_1,e_2)} p'}$$

$$\frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} q'}{p \mid (q+r) \xrightarrow{\gamma(e_1,e_2)} q'} \qquad \frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} q'}{(p \mid q) + (p \mid r) \xrightarrow{\gamma(e_1,e_2)} q'}$$

$$\frac{p \xrightarrow{e_1} \surd \quad r \xrightarrow{e_2} r'}{p \mid (q+r) \xrightarrow{\gamma(e_1,e_2)} r'} \qquad \frac{p \xrightarrow{e_1} \surd \quad r \xrightarrow{e_2} r'}{(p \mid q) + (p \mid r) \xrightarrow{\gamma(e_1,e_2)} r'}$$

$$\frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} q'}{p \mid (q+r) \xrightarrow{\gamma(e_1,e_2)} p' \between q'} \qquad \frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} q'}{(p \mid q) + (p \mid r) \xrightarrow{\gamma(e_1,e_2)} p' \between q'}$$

$$\frac{p \xrightarrow{e_1} p' \quad r \xrightarrow{e_2} r'}{p \mid (q+r) \xrightarrow{\gamma(e_1,e_2)} p' \between r'} \qquad \frac{p \xrightarrow{e_1} p' \quad r \xrightarrow{e_2} r'}{(p \mid q) + (p \mid r) \xrightarrow{\gamma(e_1,e_2)} p' \between r'}$$

So, $p \mid (q+r) \sim_s (p \mid q) + (p \mid r)$, as desired.

- **Axiom** $CE21$. Let p, q be $APTC$ processes, and $\Theta(p+q) = \Theta(p) \lhd q + \Theta(q) \lhd p$, it is sufficient to prove that $\Theta(p+q) \sim_s \Theta(p) \lhd q + \Theta(q) \lhd p$. By the transition rules for operators $+$ in Table 5, and Θ and $\lhd$ in Table 9, we get

$$\frac{p \xrightarrow{e_1} \surd(\sharp(e_1,e_2))}{\Theta(p+q) \xrightarrow{e_1} \surd} \qquad \frac{p \xrightarrow{e_1} \surd(\sharp(e_1,e_2))}{\Theta(p) \lhd q + \Theta(q) \lhd p \xrightarrow{e_1} \surd}$$

$$\frac{q \xrightarrow{e_2} \surd(\sharp(e_1,e_2))}{\Theta(p+q) \xrightarrow{e_2} \surd} \qquad \frac{q \xrightarrow{e_2} \surd(\sharp(e_1,e_2))}{\Theta(p) \lhd q + \Theta(q) \lhd p \xrightarrow{e_2} \surd}$$

$$\frac{p \xrightarrow{e_1} p'(\sharp(e_1,e_2))}{\Theta(p+q) \xrightarrow{e_1} \Theta(p')} \qquad \frac{p \xrightarrow{e_1} p'(\sharp(e_1,e_2))}{\Theta(p) \lhd q + \Theta(q) \lhd p \xrightarrow{e_1} \Theta(p')}$$

$$\frac{q \xrightarrow{e_2} q'(\sharp(e_1,e_2))}{\Theta(p+q) \xrightarrow{e_2} \Theta(q')} \qquad \frac{q \xrightarrow{e_2} q'(\sharp(e_1,e_2))}{\Theta(p) \lhd q + \Theta(q) \lhd p \xrightarrow{e_2} \Theta(q')}$$

So, $\Theta(p+q) \sim_s \Theta(p) \lhd q + \Theta(q) \lhd p$, as desired.

- **Axiom** $CE22$. Let p, q be $APTC$ processes, and $\Theta(p \cdot q) = \Theta(p) \cdot \Theta(q)$, it is sufficient to prove that $\Theta(p \cdot q) \sim_s \Theta(p) \cdot \Theta(q)$. By the transition rules for operators $\cdot$ in Table 5, and Θ in Table 9, we get

$$\frac{p \xrightarrow{e_1} \surd}{\Theta(p \cdot q) \xrightarrow{e_1} \Theta(q)} \qquad \frac{p \xrightarrow{e_1} \surd}{\Theta(p) \cdot \Theta(q) \xrightarrow{e_1} \Theta(q)}$$

$$\frac{p \xrightarrow{e_1} p'}{\Theta(p \cdot q) \xrightarrow{e_1} \Theta(p' \cdot q)} \qquad \frac{p \xrightarrow{e_1} p'}{\Theta(p) \cdot \Theta(q) \xrightarrow{e_1} \Theta(p') \cdot \Theta(q)}$$

So, with the assumption $\Theta(p' \cdot q) = \Theta(p') \cdot \Theta(q)$, $\Theta(p \cdot q) \sim_s \Theta(p) \cdot \Theta(q)$, as desired.

- **Axiom** $CE23$. Let p, q be $APTC$ processes, and $\Theta(p \parallel q) = ((\Theta(p) \lhd q) \parallel q) + ((\Theta(q) \lhd p) \parallel p)$, it is sufficient to prove that $\Theta(p \parallel q) \sim_s ((\Theta(p) \lhd q) \parallel q) + ((\Theta(q) \lhd p) \parallel p)$. By the transition rules for operators $+$ in Table 5, and Θ and $\lhd$ in Table 9, and $\parallel$ in Table 7 we get

$$\frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} \surd}{\Theta(p \parallel q) \xrightarrow{\{e_1,e_2\}} \surd}$$

$$\frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} \surd}{((\Theta(p) \lhd q) \parallel q) + ((\Theta(q) \lhd p) \parallel p) \xrightarrow{\{e_1,e_2\}} \surd}$$

$$\frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} \surd}{\Theta(p \parallel q) \xrightarrow{\{e_1,e_2\}} \Theta(p')}$$

$$\frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} \surd}{((\Theta(p) \lhd q) \parallel q) + ((\Theta(q) \lhd p) \parallel p) \xrightarrow{\{e_1,e_2\}} \Theta(p')}$$

$$\frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} q'}{\Theta(p \parallel q) \xrightarrow{\{e_1,e_2\}} \Theta(q')}$$

$$\frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} q'}{((\Theta(p) \lhd q) \parallel q) + ((\Theta(q) \lhd p) \parallel p) \xrightarrow{\{e_1,e_2\}} \Theta(q')}$$

$$\frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} q'}{\Theta(p \parallel q) \xrightarrow{\{e_1,e_2\}} \Theta(p' \between q')}$$

$$\frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} q'}{((\Theta(p) \lhd q) \parallel q) + ((\Theta(q) \lhd p) \parallel p) \xrightarrow{\{e_1,e_2\}} ((\Theta(p') \lhd q') \between q') + ((\Theta(q') \lhd p') \between p')}$$

So, with the assumption $\Theta(p' \between q') = ((\Theta(p') \lhd q') \between q') + ((\Theta(q') \lhd p') \between p')$, $\Theta(p \parallel q) \sim_s ((\Theta(p) \lhd q) \parallel q) + ((\Theta(q) \lhd p) \parallel p)$, as desired.

- **Axiom** $CE24$. Let p, q be $APTC$ processes, and $\Theta(p \mid q) = ((\Theta(p) \lhd q) \mid q) + ((\Theta(q) \lhd p) \mid p)$, it is sufficient to prove that $\Theta(p \mid q) \sim_s ((\Theta(p) \lhd q) \mid q) + ((\Theta(q) \lhd p) \mid p)$. By the transition rules for operators $+$ in Table 5, and Θ and $\lhd$ in Table 9, and $\mid$ in Table 8 we get

$$\frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} \surd}{\Theta(p \mid q) \xrightarrow{\gamma(e_1,e_2)} \surd}$$

$$\frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} \surd}{((\Theta(p) \lhd q) \mid q) + ((\Theta(q) \lhd p) \mid p) \xrightarrow{\gamma(e_1,e_2)} \surd}$$

$$\frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} \surd}{\Theta(p \mid q) \xrightarrow{\gamma(e_1,e_2)} \Theta(p')}$$

$$\frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} \surd}{((\Theta(p) \lhd q) \mid q) + ((\Theta(q) \lhd p) \mid p) \xrightarrow{\gamma(e_1,e_2)} \Theta(p')}$$

$$\frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} q'}{\Theta(p \mid q) \xrightarrow{\gamma(e_1,e_2)} \Theta(q')}$$

$$\frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} q'}{((\Theta(p) \lhd q) \mid q) + ((\Theta(q) \lhd p) \mid p) \xrightarrow{\gamma(e_1,e_2)} \Theta(q')}$$

$$\frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} q'}{\Theta(p \mid q) \xrightarrow{\gamma(e_1,e_2)} \Theta(p' \between q')}$$

$$\frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} q'}{((\Theta(p) \lhd q) \mid q) + ((\Theta(q) \lhd p) \mid p) \xrightarrow{\gamma(e_1,e_2)} ((\Theta(p') \lhd q') \between q') + ((\Theta(q') \lhd p') \between p')}$$

So, with the assumption $\Theta(p' \between q') = ((\Theta(p') \lhd q') \between q') + ((\Theta(q') \lhd p') \between p')$, $\Theta(p \mid q) \sim_s ((\Theta(p) \lhd q) \mid q) + ((\Theta(q) \lhd p) \mid p)$, as desired.

- **Axiom** $U30$. Let p, q, r be $APTC$ processes, and $(p + q) \lhd r = (p \lhd r) + (q \lhd r)$, it is sufficient to prove that $(p + q) \lhd r \sim_s (p \lhd r) + (q \lhd r)$. By the transition rules for operators $+$ and $\lhd$ in Table 5 and 9, we get

$$\frac{p \xrightarrow{e_1} \surd}{(p + q) \lhd r \xrightarrow{e_1} \surd} \quad \frac{p \xrightarrow{e_1} \surd}{(p \lhd r) + (q \lhd r) \xrightarrow{e_1} \surd}$$

$$\frac{q \xrightarrow{e_2} \surd}{(p + q) \lhd r \xrightarrow{e_2} \surd} \quad \frac{q \xrightarrow{e_2} \surd}{(p \lhd r) + (q \lhd r) \xrightarrow{e_2} \surd}$$

$$\frac{p \xrightarrow{e_1} p'}{(p + q) \lhd r \xrightarrow{e_1} p' \lhd r} \quad \frac{p \xrightarrow{e_1} p'}{(p \lhd r) + (q \lhd r) \xrightarrow{e_1} p' \lhd r}$$

$$\frac{q \xrightarrow{e_2} q'}{(p + q) \lhd r \xrightarrow{e_2} q' \lhd r} \quad \frac{q \xrightarrow{e_2} q'}{(p \lhd r) + (q \lhd r) \xrightarrow{e_2} q' \lhd r}$$

Let us forget anything about τ. So, $(p + q) \lhd r \sim_s (p \lhd r) + (q \lhd r)$, as desired.

- **Axiom** $U31$. Let p, q, r be $APTC$ processes, and $(p \cdot q) \lhd r = (p \lhd r) \cdot (q \lhd r)$, it is sufficient to prove that $(p \cdot q) \lhd r \sim_s (p \lhd r) \cdot (q \lhd r)$. By the transition rules for operators $\cdot$ and $\lhd$ in Table 5 and 9, we get

$$\frac{p \xrightarrow{e_1} \surd}{(p \cdot q) \lhd r \xrightarrow{e_1} q \lhd r} \quad \frac{p \xrightarrow{e_1} \surd}{(p \lhd r) \cdot (q \lhd r) \xrightarrow{e_1} q \lhd r}$$

$$\frac{p \xrightarrow{e_1} p'}{(p \cdot q) \lhd r \xrightarrow{e_1} (p' \cdot q) \lhd r} \quad \frac{p \xrightarrow{e_1} p'}{(p \lhd r) \cdot (q \lhd r) \xrightarrow{e_1} (p' \lhd r) \cdot (q \lhd r)}$$

Let us forget anything about τ. With the assumption $(p' \cdot q) \lhd r = (p' \lhd r) \cdot (q \lhd r)$, so, $(p \cdot q) \lhd r \sim_s (p \lhd r) \cdot q$, as desired.

- **Axiom** $U32$. Let p, q, r be $APTC$ processes, and $(p \parallel q) \lhd r = (p \lhd r) \parallel (q \lhd r)$, it is sufficient to prove that $(p \parallel q) \lhd r \sim_s (p \lhd r) \parallel (q \lhd r)$. By the transition rules for operators $\parallel$ and $\lhd$ in Table 7 and 9, we get

$$\frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} \surd}{(p \parallel q) \lhd r \xrightarrow{\{e_1,e_2\}} \surd} \quad \frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} \surd}{(p \lhd r) \parallel (q \lhd r) \xrightarrow{\{e_1,e_2\}} \surd}$$

$$\frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} \surd}{(p \parallel q) \lhd r \xrightarrow{\{e_1,e_2\}} p' \lhd r} \qquad \frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} \surd}{(p \lhd r) \parallel (q \lhd r) \xrightarrow{\{e_1,e_2\}} p' \lhd r}$$

$$\frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} q'}{(p \parallel q) \lhd r \xrightarrow{\{e_1,e_2\}} q' \lhd r} \qquad \frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} q'}{(p \lhd r) \parallel (q \lhd r) \xrightarrow{\{e_1,e_2\}} q' \lhd r}$$

$$\frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} q'}{(p \parallel q) \lhd r \xrightarrow{\{e_1,e_2\}} (p' \between q') \lhd r} \qquad \frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} q'}{(p \lhd r) \parallel (q \lhd r) \xrightarrow{\{e_1,e_2\}} (p' \lhd r) \between (q' \lhd r)}$$

Let us forget anything about τ. With the assumption $(p' \between q') \lhd r = (p' \lhd r) \between (q' \lhd r)$, so, $(p \parallel q) \lhd r \sim_s (p \lhd r) \parallel (q \lhd r)$, as desired.

- **Axiom** $U33$. Let p, q, r be $APTC$ processes, and $(p \mid q) \lhd r = (p \lhd r) \mid (q \lhd r)$, it is sufficient to prove that $(p \mid q) \lhd r \sim_s (p \lhd r) \mid (q \lhd r)$. By the transition rules for operators $\mid$ and $\lhd$ in Table 8 and 9, we get

$$\frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} \surd}{(p \mid q) \lhd r \xrightarrow{\gamma(e_1,e_2)} \surd} \qquad \frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} \surd}{(p \lhd r) \mid (q \lhd r) \xrightarrow{\gamma(e_1,e_2)} \surd}$$

$$\frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} \surd}{(p \mid q) \lhd r \xrightarrow{\gamma(e_1,e_2)} p' \lhd r} \qquad \frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} \surd}{(p \lhd r) \mid (q \lhd r) \xrightarrow{\gamma(e_1,e_2)} p' \lhd r}$$

$$\frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} q'}{(p \mid q) \lhd r \xrightarrow{\gamma(e_1,e_2)} q' \lhd r} \qquad \frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} q'}{(p \lhd r) \mid (q \lhd r) \xrightarrow{\gamma(e_1,e_2)} q' \lhd r}$$

$$\frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} q'}{(p \mid q) \lhd r \xrightarrow{\gamma(e_1,e_2)} (p' \between q') \lhd r} \qquad \frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} q'}{(p \lhd r) \mid (q \lhd r) \xrightarrow{\gamma(e_1,e_2)} (p' \lhd r) \between (q' \lhd r)}$$

Let us forget anything about τ. With the assumption $(p' \between q') \lhd r = (p' \lhd r) \between (q' \lhd r)$, so, $(p \mid q) \lhd r \sim_s (p \lhd r) \mid (q \lhd r)$, as desired.

- **Axiom** $U34$. Let p, q, r be $APTC$ processes, and $p \lhd (q + r) = (p \lhd q) \lhd r$, it is sufficient to prove that $p \lhd (q + r) \sim_s (p \lhd q) \lhd r$. By the transition rules for operators $+$ and $\lhd$ in Table 5 and 9, we get

$$\frac{p \xrightarrow{e_1} \surd}{p \lhd (q + r) \xrightarrow{e_1} \surd} \qquad \frac{p \xrightarrow{e_1} \surd}{(p \lhd q) \lhd r \xrightarrow{e_1} \surd}$$

$$\frac{p \xrightarrow{e_1} p'}{p \lhd (q + r) \xrightarrow{e_1} p' \lhd (q + r)} \qquad \frac{p \xrightarrow{e_1} p'}{(p \lhd q) \lhd r \xrightarrow{e_1} (p' \lhd q) \lhd r}$$

Let us forget anything about τ. With the assumption $p' \lhd (q+r) = (p' \lhd q) \lhd r$, so, $p \lhd (q+r) \sim_s (p \lhd q) \lhd r$, as desired.

- **Axiom** $U35$. Let p, q, r be $APTC$ processes, and $p \lhd (q \cdot r) = (p \lhd q) \lhd r$, it is sufficient to prove that $p \lhd (q \cdot r) \sim_s (p \lhd q) \lhd r$. By the transition rules for operators $\cdot$ and $\lhd$ in Table 5 and 9, we get

$$\frac{p \xrightarrow{e_1} \surd}{p \lhd (q \cdot r) \xrightarrow{e_1} \surd} \qquad \frac{p \xrightarrow{e_1} \surd}{(p \lhd q) \lhd r \xrightarrow{e_1} \surd}$$

$$\frac{p \xrightarrow{e_1} p'}{p \lhd (q \cdot r) \xrightarrow{e_1} p' \lhd (q \cdot r)} \qquad \frac{p \xrightarrow{e_1} p'}{(p \lhd q) \lhd r \xrightarrow{e_1} (p' \lhd q) \lhd r}$$

Let us forget anything about τ. With the assumption $p' \lhd (q \cdot r) = (p' \lhd q) \lhd r$, so, $p \lhd (q \cdot r) \sim_s p \lhd q$, as desired.

- **Axiom** $U36$. Let p, q, r be $APTC$ processes, and $p \lhd (q \parallel r) = (p \lhd q) \lhd r$, it is sufficient to prove that $p \lhd (q \parallel r) \sim_s (p \lhd q) \lhd r$. By the transition rules for operators $\parallel$ and $\lhd$ in Table 7 and 9, we get

$$\frac{p \xrightarrow{e_1} \surd}{p \lhd (q \parallel r) \xrightarrow{e_1} \surd} \quad \frac{p \xrightarrow{e_1} \surd}{(p \lhd q) \lhd r \xrightarrow{e_1} \surd}$$

$$\frac{p \xrightarrow{e_1} p'}{p \lhd (q \parallel r) \xrightarrow{e_1} p' \lhd (q \parallel r)} \quad \frac{p \xrightarrow{e_1} p'}{(p \lhd q) \lhd r \xrightarrow{e_1} (p' \lhd q) \lhd r}$$

Let us forget anything about τ. With the assumption $p' \lhd (q \parallel r) = (p' \lhd q) \lhd r$, so, $p \lhd (q \parallel r) \sim_s (p \lhd q) \lhd r$, as desired.

- **Axiom** $U37$. Let p, q, r be $APTC$ processes, and $p \lhd (q \mid r) = (p \lhd q) \lhd r$, it is sufficient to prove that $p \lhd (q \mid r) \sim_s (p \lhd q) \lhd r$. By the transition rules for operators $\mid$ and $\lhd$ in Table 8 and 9, we get

$$\frac{p \xrightarrow{e_1} \surd}{p \lhd (q \mid r) \xrightarrow{e_1} \surd} \quad \frac{p \xrightarrow{e_1} \surd}{(p \lhd q) \lhd r \xrightarrow{e_1} \surd}$$

$$\frac{p \xrightarrow{e_1} p'}{p \lhd (q \mid r) \xrightarrow{e_1} p' \lhd (q \mid r)} \quad \frac{p \xrightarrow{e_1} p'}{(p \lhd q) \lhd r \xrightarrow{e_1} (p' \lhd q) \lhd r}$$

Let us forget anything about τ. With the assumption $p' \lhd (q \mid r) = (p' \lhd q) \lhd r$, so, $p \lhd (q \mid r) \sim_s (p \lhd q) \lhd r$, as desired.

□

Theorem 4.8 (Completeness of parallelism modulo step bisimulation equivalence). Let p and q be closed $APTC$ terms, if $p \sim_s q$ then $p = q$.

Proof. Firstly, by the elimination theorem of $APTC$ (see Theorem 4.5), we know that for each closed $APTC$ term p, there exists a closed basic $APTC$ term p', such that $APTC \vdash p = p'$, so, we only need to consider closed basic $APTC$ terms.

The basic terms (see Definition 4.2) modulo associativity and commutativity (AC) of conflict + (defined by axioms $A1$ and $A2$ in Table 1) and associativity and commutativity (AC) of parallel $\parallel$ (defined by axioms $P2$ and $P3$ in Table 10), and these equivalences is denoted by $=_{AC}$. Then, each equivalence class s modulo AC of + and $\parallel$ has the following normal form

$$s_1 + \cdots + s_k$$

with each s_i either an atomic event or of the form

$$t_1 \cdot \cdots \cdot t_m$$

with each t_j either an atomic event or of the form

$$u_1 \parallel \cdots \parallel u_n$$

with each u_l an atomic event, and each s_i is called the summand of s.

Now, we prove that for normal forms n and n', if $n \sim_s n'$ then $n =_{AC} n'$. It is sufficient to induct on the sizes of n and n'.

- Consider a summand e of n. Then $n \xrightarrow{e} \surd$, so $n \sim_s n'$ implies $n' \xrightarrow{e} \surd$, meaning that n' also contains the summand e.
- Consider a summand $t_1 \cdot t_2$ of n,
  - if $t_1 \equiv e'$, then $n \xrightarrow{e'} t_2$, so $n \sim_s n'$ implies $n' \xrightarrow{e'} t_2'$ with $t_2 \sim_s t_2'$, meaning that n' contains a summand

$e' \cdot t_2'$. Since t_2 and t_2' are normal forms and have sizes smaller than n and n', by the induction hypotheses if $t_2 \backsim_s t_2'$ then $t_2 =_{AC} t_2'$;

– if $t_1 \equiv e_1 \parallel \cdots \parallel e_n$, then $n \xrightarrow{\{e_1,\cdots,e_n\}} t_2$, so $n \backsim_s n'$ implies $n' \xrightarrow{\{e_1,\cdots,e_n\}} t_2'$ with $t_2 \backsim_s t_2'$, meaning that n' contains a summand $(e_1 \parallel \cdots \parallel e_n) \cdot t_2'$. Since t_2 and t_2' are normal forms and have sizes smaller than n and n', by the induction hypotheses if $t_2 \backsim_s t_2'$ then $t_2 =_{AC} t_2'$.

So, we get $n =_{AC} n'$.

Finally, let s and t be basic $APTC$ terms, and $s \backsim_s t$, there are normal forms n and n', such that $s = n$ and $t = n'$. The soundness theorem of parallelism modulo step bisimulation equivalence (see Theorem 4.7) yields $s \backsim_s n$ and $t \backsim_s n'$, so $n \backsim_s s \backsim_s t \backsim_s n'$. Since if $n \backsim_s n'$ then $n =_{AC} n'$, $s = n =_{AC} n' = t$, as desired. □

Theorem 4.9 (Soundness of parallelism modulo pomset bisimulation equivalence). Let x and y be $APTC$ terms. If $APTC \vdash x = y$, then $x \backsim_p y$.

Proof. Since pomset bisimulation $\backsim_p$ is both an equivalent and a congruent relation with respect to the operators $\between$, $\parallel$, $|$, Θ and $\lhd$, we only need to check if each axiom in Table 10 is sound modulo pomset bisimulation equivalence.

From the definition of pomset bisimulation (see Definition 2.17), we know that pomset bisimulation is defined by pomset transitions, which are labeled by pomsets. In a pomset transition, the events in the pomset are either within causality relations (defined by $\cdot$) or in concurrency (implicitly defined by $\cdot$ and $+$, and explicitly defined by $\between$), of course, they are pairwise consistent (without conflicts). In Theorem 4.7, we have already proven the case that all events are pairwise concurrent, so, we only need to prove the case of events in causality. Without loss of generality, we take a pomset of $P = \{e_1, e_2 : e_1 \cdot e_2\}$. Then the pomset transition labeled by the above P is just composed of one single event transition labeled by e_1 succeeded by another single event transition labeled by e_2, that is, $\xrightarrow{P} = \xrightarrow{e_1} \xrightarrow{e_2}$.

Similarly to the proof of soundness of parallelism modulo step bisimulation equivalence (see Theorem 4.7), we can prove that each axiom in Table 10 is sound modulo pomset bisimulation equivalence, we omit them. □

Theorem 4.10 (Completeness of parallelism modulo pomset bisimulation equivalence). Let p and q be closed $APTC$ terms, if $p \backsim_p q$ then $p = q$.

Proof. Firstly, by the elimination theorem of $APTC$ (see Theorem 4.5), we know that for each closed $APTC$ term p, there exists a closed basic $APTC$ term p', such that $APTC \vdash p = p'$, so, we only need to consider closed basic $APTC$ terms.

The basic terms (see Definition 4.2) modulo associativity and commutativity (AC) of conflict $+$ (defined by axioms $A1$ and $A2$ in Table 1) and associativity and commutativity (AC) of parallel $\parallel$ (defined by axioms $P2$ and $P3$ in Table 10), and these equivalences is denoted by $=_{AC}$. Then, each equivalence class s modulo AC of $+$ and $\parallel$ has the following normal form

$$s_1 + \cdots + s_k$$

with each s_i either an atomic event or of the form

$$t_1 \cdot \cdots \cdot t_m$$

with each t_j either an atomic event or of the form

$$u_1 \parallel \cdots \parallel u_n$$

with each u_l an atomic event, and each s_i is called the summand of s.

Now, we prove that for normal forms n and n', if $n \backsim_p n'$ then $n =_{AC} n'$. It is sufficient to induct on the sizes of n and n'.

- Consider a summand e of n. Then $n \xrightarrow{e} \surd$, so $n \backsim_p n'$ implies $n' \xrightarrow{e} \surd$, meaning that n' also contains the summand e.
- Consider a summand $t_1 \cdot t_2$ of n,

  – if $t_1 \equiv e'$, then $n \xrightarrow{e'} t_2$, so $n \backsim_p n'$ implies $n' \xrightarrow{e'} t_2'$ with $t_2 \backsim_p t_2'$, meaning that n' contains a summand

$e' \cdot t'_2$. Since t_2 and t'_2 are normal forms and have sizes smaller than n and n', by the induction hypotheses if $t_2 \leftrightarrows_p t'_2$ then $t_2 =_{AC} t'_2$;

– if $t_1 \equiv e_1 \parallel \cdots \parallel e_n$, then $n \xrightarrow{\{e_1,\cdots,e_n\}} t_2$, so $n \leftrightarrows_p n'$ implies $n' \xrightarrow{\{e_1,\cdots,e_n\}} t'_2$ with $t_2 \leftrightarrows_p t'_2$, meaning that n' contains a summand $(e_1 \parallel \cdots \parallel e_n) \cdot t'_2$. Since t_2 and t'_2 are normal forms and have sizes smaller than n and n', by the induction hypotheses if $t_2 \leftrightarrows_p t'_2$ then $t_2 =_{AC} t'_2$.

So, we get $n =_{AC} n'$.

Finally, let s and t be basic $APTC$ terms, and $s \leftrightarrows_p t$, there are normal forms n and n', such that $s = n$ and $t = n'$. The soundness theorem of parallelism modulo pomset bisimulation equivalence (see Theorem 4.9) yields $s \leftrightarrows_p n$ and $t \leftrightarrows_p n'$, so $n \leftrightarrows_p s \leftrightarrows_p t \leftrightarrows_p n'$. Since if $n \leftrightarrows_p n'$ then $n =_{AC} n'$, $s = n =_{AC} n' = t$, as desired. □

Theorem 4.11 (Soundness of parallelism modulo hp-bisimulation equivalence). Let x and y be $APTC$ terms. If $APTC \vdash x = y$, then $x \leftrightarrows_{hp} y$.

Proof. Since hp-bisimulation $\leftrightarrows_{hp}$ is both an equivalent and a congruent relation with respect to the operators $\between$, $\parallel$, $\mid$, Θ and $\lhd$, we only need to check if each axiom in Table 10 is sound modulo hp-bisimulation equivalence.

From the definition of hp-bisimulation (see Definition 2.21), we know that hp-bisimulation is defined on the posetal product $(C_1, f, C_2), f : C_1 \to C_2$ isomorphism. Two process terms s related to C_1 and t related to C_2, and $f : C_1 \to C_2$ isomorphism. Initially, $(C_1, f, C_2) = (\varnothing, \varnothing, \varnothing)$, and $(\varnothing, \varnothing, \varnothing) \in \leftrightarrows_{hp}$. When $s \xrightarrow{e} s'$ ($C_1 \xrightarrow{e} C'_1$), there will be $t \xrightarrow{e} t'$ ($C_2 \xrightarrow{e} C'_2$), and we define $f' = f[e \mapsto e]$. Then, if $(C_1, f, C_2) \in \leftrightarrows_{hp}$, then $(C'_1, f', C'_2) \in \leftrightarrows_{hp}$.

Similarly to the proof of soundness of parallelism modulo pomset bisimulation equivalence (see Theorem 4.9), we can prove that each axiom in Table 10 is sound modulo hp-bisimulation equivalence, we just need additionally to check the above conditions on hp-bisimulation, we omit them. □

Theorem 4.12 (Completeness of parallelism modulo hp-bisimulation equivalence). Let p and q be closed $APTC$ terms, if $p \leftrightarrows_{hp} q$ then $p = q$.

Proof. Firstly, by the elimination theorem of $APTC$ (see Theorem 4.5), we know that for each closed $APTC$ term p, there exists a closed basic $APTC$ term p', such that $APTC \vdash p = p'$, so, we only need to consider closed basic $APTC$ terms.

The basic terms (see Definition 4.2) modulo associativity and commutativity (AC) of conflict + (defined by axioms $A1$ and $A2$ in Table 1) and associativity and commutativity (AC) of parallel $\parallel$ (defined by axioms $P2$ and $P3$ in Table 10), and these equivalences is denoted by $=_{AC}$. Then, each equivalence class s modulo AC of + and $\parallel$ has the following normal form

$$s_1 + \cdots + s_k$$

with each s_i either an atomic event or of the form

$$t_1 \cdot \cdots \cdot t_m$$

with each t_j either an atomic event or of the form

$$u_1 \parallel \cdots \parallel u_n$$

with each u_l an atomic event, and each s_i is called the summand of s.

Now, we prove that for normal forms n and n', if $n \leftrightarrows_{hp} n'$ then $n =_{AC} n'$. It is sufficient to induct on the sizes of n and n'.

- Consider a summand e of n. Then $n \xrightarrow{e} \surd$, so $n \leftrightarrows_{hp} n'$ implies $n' \xrightarrow{e} \surd$, meaning that n' also contains the summand e.
- Consider a summand $t_1 \cdot t_2$ of n,

  – if $t_1 \equiv e'$, then $n \xrightarrow{e'} t_2$, so $n \leftrightarrows_{hp} n'$ implies $n' \xrightarrow{e'} t'_2$ with $t_2 \leftrightarrows_{hp} t'_2$, meaning that n' contains a summand $e' \cdot t'_2$. Since t_2 and t'_2 are normal forms and have sizes smaller than n and n', by the induction hypotheses if $t_2 \leftrightarrows_{hp} t'_2$ then $t_2 =_{AC} t'_2$;

– if $t_1 \equiv e_1 \parallel \cdots \parallel e_n$, then $n \xrightarrow{\{e_1,\cdots,e_n\}} t_2$, so $n \sim_{hp} n'$ implies $n' \xrightarrow{\{e_1,\cdots,e_n\}} t_2'$ with $t_2 \sim_{hp} t_2'$, meaning that n' contains a summand $(e_1 \parallel \cdots \parallel e_n) \cdot t_2'$. Since t_2 and t_2' are normal forms and have sizes smaller than n and n', by the induction hypotheses if $t_2 \sim_{hp} t_2'$ then $t_2 =_{AC} t_2'$.

So, we get $n =_{AC} n'$.

Finally, let s and t be basic $APTC$ terms, and $s \sim_{hp} t$, there are normal forms n and n', such that $s = n$ and $t = n'$. The soundness theorem of parallelism modulo hp-bisimulation equivalence (see Theorem 4.9) yields $s \sim_{hp} n$ and $t \sim_{hp} n'$, so $n \sim_{hp} s \sim_{hp} t \sim_{hp} n'$. Since if $n \sim_{hp} n'$ then $n =_{AC} n'$, $s = n =_{AC} n' = t$, as desired. □

Proposition 4.13 (About Soundness and Completeness of parallelism modulo hhp-bisimulation equivalence).
1. Let x and y be $APTC$ terms. If $APTC \vdash x = y \Rightarrow x \sim_{hhp} y$;

2. If p and q are closed $APTC$ terms, then $p \sim_{hhp} q \Rightarrow p = q$.

Proof. Imperfectly, the algebraic laws in Table 10 are not sound and complete modulo hhp-bisimulation equivalence, we just need enumerate several key axioms in Table 10 are not sound modulo hhp-bisimulation equivalence.

From the definition of hhp-bisimulation (see Definition 2.21), we know that an hhp-bisimulation is a downward closed hp-bisimulation. That is, for any posetal products (C_1, f, C_2) and (C_1', f, C_2'), if $(C_1, f, C_2) \subseteq (C_1', f', C_2')$ pointwise and $(C_1', f', C_2') \in \sim_{hhp}$, then $(C_1, f, C_2) \in \sim_{hhp}$.

Now, let us consider the axioms $P7$ and $P8$ (the right and left distributivity of $\parallel$ to $+$). Let $s_1 = (a+b) \parallel c$, $t_1 = (a \parallel c) + (b \parallel c)$, and $s_2 = a \parallel (b + c)$, $t_2 = (a \parallel b) + (a \parallel c)$. We know that $s_1 \sim_{hp} t_1$ and $s_2 \sim_{hp} t_2$ (by Theorem 4.11), we prove that $s_1 \not\sim_{hhp} t_1$ and $s_2 \not\sim_{hhp} t_2$. Let $(C(s_1), f_1, C(t_1))$ and $(C(s_2), f_2, C(t_2))$ are the corresponding posetal products.

- **Axiom** $P7$. $s_1 \xrightarrow{\{a,c\}} \surd(s_1')$ $(C(s_1) \xrightarrow{\{a,c\}} C(s_1'))$, then $t_1 \xrightarrow{\{a,c\}} \surd(t_1')$ $(C(t_1) \xrightarrow{\{a,c\}} C(t_1'))$, we define $f_1' = f_1[a \mapsto a, c \mapsto c]$, obviously, $(C(s_1), f_1, C(t_1)) \in \sim_{hp}$ and $(C(s_1'), f_1', C(t_1')) \in \sim_{hp}$. But, $(C(s_1), f_1, C(t_1)) \in \sim_{hhp}$ and $(C(s_1'), f_1', C(t_1')) \notin \sim_{hhp}$, just because they are not downward closed. Let $(C(s_1''), f_1'', C(t_1''))$, and $f_1'' = f_1[c \mapsto c]$, $s_1 \xrightarrow{c} s_1''$ $(C(s_1) \xrightarrow{c} C(s_1''))$, $t_1 \xrightarrow{c} t_1''$ $(C(t_1) \xrightarrow{c} C(t_1''))$, it is easy to see that $(C(s_1''), f_1'', C(t_1'')) \subseteq (C(s_1'), f_1', C(t_1'))$ pointwise, while $(C(s_1''), f_1'', C(t_1'')) \notin \sim_{hp}$, because s_1'' and $C(s_1'')$ exist, but t_1'' and $C(t_1'')$ do not exist.
- **Axiom** $P8$. $s_2 \xrightarrow{\{a,c\}} \surd(s_2')$ $(C(s_2) \xrightarrow{\{a,c\}} C(s_2'))$, then $t_2 \xrightarrow{\{a,c\}} \surd(t_2')$ $(C(t_2) \xrightarrow{\{a,c\}} C(t_2'))$, we define $f_2' = f_2[a \mapsto a, c \mapsto c]$, obviously, $(C(s_2), f_2, C(t_2)) \in \sim_{hp}$ and $(C(s_2'), f_2', C(t_2')) \in \sim_{hp}$. But, $(C(s_2), f_2, C(t_2)) \in \sim_{hhp}$ and $(C(s_2'), f_2', C(t_2')) \notin \sim_{hhp}$, just because they are not downward closed. Let $(C(s_2''), f_2'', C(t_2''))$, and $f_2'' = f_2[a \mapsto a]$, $s_2 \xrightarrow{a} s_2''$ $(C(s_2) \xrightarrow{a} C(s_2''))$, $t_2 \xrightarrow{a} t_2''$ $(C(t_2) \xrightarrow{a} C(t_2''))$, it is easy to see that $(C(s_2''), f_2'', C(t_2'')) \subseteq (C(s_2'), f_2', C(t_2'))$ pointwise, while $(C(s_2''), f_2'', C(t_2'')) \notin \sim_{hp}$, because s_2'' and $C(s_2'')$ exist, but t_2'' and $C(t_2'')$ do not exist.

The unsoundness of parallelism modulo hhp-bisimulation equivalence makes the completeness of parallelism modulo hhp-bisimulation equivalence meaningless. Further more, unsoundness of $P7$ and $P8$ lead to the elimination theorem of $APTC$ (see Theorem 4.5) failing, so, the non-existence of normal form also makes the completeness impossible. □

A soundness and completeness axiomatization modulo hhp-bisimulation equivalence may exist or may not exist, we do not know, let it be an open problem. In following sections, we will discuss nothing about hhp-bisimulation, because the following encapsulation, recursion and abstraction are based on the algebraic laws in this section.

Finally, let us explain the so-called absorption law [18] in a straightforward way. Process term $P = a \parallel (b + c) + a \parallel b + b \parallel (a + c)$, and process term $Q = a \parallel (b + c) + b \parallel (a + c)$, equated by the absorption law.

Modulo $\sim_s$, $\sim_p$, and $\sim_{hp}$, by use of the axioms of $BATC$ and $APTC$, we have the following deductions:

$$\begin{aligned}
P &= a \parallel (b + c) + a \parallel b + b \parallel (a + c)\\
&\overset{P8}{=} a \parallel b + a \parallel c + a \parallel b + b \parallel a + b \parallel c\\
&\overset{P2}{=} a \parallel b + a \parallel c + a \parallel b + a \parallel b + b \parallel c\\
&\overset{A3}{=} a \parallel b + a \parallel c + b \parallel c
\end{aligned}$$

$$\frac{x \xrightarrow{e} \surd}{\partial_H(x) \xrightarrow{e} \surd} \quad (e \notin H) \qquad \frac{x \xrightarrow{e} x'}{\partial_H(x) \xrightarrow{e} \partial_H(x')} \quad (e \notin H)$$

Table 12. Transition rules of encapsulation operator ∂_H

No.	Axiom
$D1$	$e \notin H \quad \partial_H(e) = e$
$D2$	$e \in H \quad \partial_H(e) = \delta$
$D3$	$\partial_H(\delta) = \delta$
$D4$	$\partial_H(x + y) = \partial_H(x) + \partial_H(y)$
$D5$	$\partial_H(x \cdot y) = \partial_H(x) \cdot \partial_H(y)$
$D6$	$\partial_H(x \parallel y) = \partial_H(x) \parallel \partial_H(y)$

Table 13. Axioms of encapsulation operator

$$\begin{aligned} Q &= a \parallel (b + c) + b \parallel (a + c) \\ &\overset{P8}{=} a \parallel b + a \parallel c + b \parallel a + b \parallel c \\ &\overset{P2}{=} a \parallel b + a \parallel c + a \parallel b + b \parallel c \\ &\overset{A3}{=} a \parallel b + a \parallel c + b \parallel c \end{aligned}$$

It means that $P = Q$ modulo $\sim_s$, $\sim_p$, and $\sim_{hp}$, that is, $P \sim_s Q$, $P \sim_p Q$ and $P \sim_{hp} Q$. But, $P \neq Q$ modulo $\sim_{hhp}$, which means that $P \not\sim_{hhp} Q$.

4.5. Encapsulation

The mismatch of two communicating events in different parallel branches can cause deadlock, so the deadlocks in the concurrent processes should be eliminated. Like ACP [4], we also introduce the unary encapsulation operator ∂_H for set H of atomic events, which renames all atomic events in H into δ. The whole algebra including parallelism for true concurrency in the above subsections, deadlock δ and encapsulation operator ∂_H, is called Algebra for Parallelism in True Concurrency, abbreviated $APTC$.

The transition rules of encapsulation operator ∂_H are shown in Table 12.

Based on the transition rules for encapsulation operator ∂_H in Table 12, we design the axioms as Table 13 shows.

The axioms $D1 - D3$ are the defining laws for the encapsulation operator ∂_H, $D1$ leaves atomic events outside H unchanged, $D2$ renames atomic events in H into δ, and $D3$ says that it leaves δ unchanged. $D4 - D6$ say that in term $\partial_H(t)$, all transitions of t labeled with atomic events in H are blocked.

Theorem 4.14 (Conservativity of $APTC$ with respect to the algebra for parallelism). $APTC$ is a conservative extension of the algebra for parallelism.

Proof. It follows from the following two facts (see Theorem 2.8).

1. The transition rules of the algebra for parallelism in the above subsections are all source-dependent;
2. The sources of the transition rules for the encapsulation operator contain an occurrence of ∂_H.

So, $APTC$ is a conservative extension of the algebra for parallelism, as desired. □

Theorem 4.15 (Congruence theorem of encapsulation operator ∂_H). Truly concurrent bisimulation equivalences $\sim_p$, $\sim_s$, $\sim_{hp}$ and $\sim_{hhp}$ are all congruences with respect to encapsulation operator ∂_H.

No.	Rewriting Rule
$RD1$	$e \notin H \quad \partial_H(e) \to e$
$RD2$	$e \in H \quad \partial_H(e) \to \delta$
$RD3$	$\partial_H(\delta) \to \delta$
$RD4$	$\partial_H(x + y) \to \partial_H(x) + \partial_H(y)$
$RD5$	$\partial_H(x \cdot y) \to \partial_H(x) \cdot \partial_H(y)$
$RD6$	$\partial_H(x \parallel y) \to \partial_H(x) \parallel \partial_H(y)$

Table 14. Term rewrite system of encapsulation operator ∂_H

Proof. (1) Case pomset bisimulation equivalence $\sim_p$.

Let x and y be $APTC$ processes, and $x \sim_p y$, it is sufficient to prove that $\partial_H(x) \sim_p \partial_H(y)$.

By the definition of pomset bisimulation $\sim_p$ (Definition 2.17), $x \sim_p y$ means that

$$x \xrightarrow{X} x' \quad y \xrightarrow{Y} y'$$

with $X \subseteq x$, $Y \subseteq y$, $X \sim Y$ and $x' \sim_p y'$.

By the pomset transition rules for encapsulation operator ∂_H in Table 12, we can get

$$\partial_H(x) \xrightarrow{X} \surd(X \nsubseteq H) \quad \partial_H(y) \xrightarrow{Y} \surd(Y \nsubseteq H)$$

with $X \subseteq x$, $Y \subseteq y$, and $X \sim Y$, so, we get $\partial_H(x) \sim_p \partial_H(y)$, as desired.

Or, we can get

$$\partial_H(x) \xrightarrow{X} \partial_H(x')(X \nsubseteq H) \quad \partial_H(y) \xrightarrow{Y} \partial_H(y')(Y \nsubseteq H)$$

with $X \subseteq x$, $Y \subseteq y$, $X \sim Y$, $x' \sim_p y'$ and the assumption $\partial_H(x') \sim_p \partial_H(y')$, so, we get $\partial_H(x) \sim_p \partial_H(y)$, as desired.

(2) The cases of step bisimulation $\sim_s$, hp-bisimulation $\sim_{hp}$ and hhp-bisimulation $\sim_{hhp}$ can be proven similarly, we omit them. □

Theorem 4.16 (Elimination theorem of $APTC$). Let p be a closed $APTC$ term including the encapsulation operator ∂_H. Then there is a basic $APTC$ term q such that $APTC \vdash p = q$.

Proof. (1) Firstly, suppose that the following ordering on the signature of $APTC$ is defined: $\parallel > \cdot > +$ and the symbol $\parallel$ is given the lexicographical status for the first argument, then for each rewrite rule $p \to q$ in Table 14 relation $p >_{lpo} q$ can easily be proved. We obtain that the term rewrite system shown in Table 14 is strongly normalizing, for it has finitely many rewriting rules, and $>$ is a well-founded ordering on the signature of $APTC$, and if $s >_{lpo} t$, for each rewriting rule $s \to t$ is in Table 14 (see Theorem 2.12).

(2) Then we prove that the normal forms of closed $APTC$ terms including encapsulation operator ∂_H are basic $APTC$ terms.

Suppose that p is a normal form of some closed $APTC$ term and suppose that p is not a basic $APTC$ term. Let p' denote the smallest sub-term of p which is not a basic $APTC$ term. It implies that each sub-term of p' is a basic $APTC$ term. Then we prove that p is not a term in normal form. It is sufficient to induct on the structure of p', following from Theorem 4.3, we only prove the new case $p' \equiv \partial_H(p_1)$:

- Case $p_1 \equiv e$. The transition rules $RD1$ or $RD2$ can be applied, so p is not a normal form;
- Case $p_1 \equiv \delta$. The transition rules $RD3$ can be applied, so p is not a normal form;
- Case $p_1 \equiv p_1' + p_1''$. The transition rules $RD4$ can be applied, so p is not a normal form;
- Case $p_1 \equiv p_1' \cdot p_1''$. The transition rules $RD5$ can be applied, so p is not a normal form;
- Case $p_1 \equiv p_1' \parallel p_1''$. The transition rules $RD6$ can be applied, so p is not a normal form.

□

Theorem 4.17 (Soundness of $APTC$ modulo step bisimulation equivalence). Let x and y be $APTC$ terms including encapsulation operator ∂_H. If $APTC \vdash x = y$, then $x \sim_s y$.

Proof. Since step bisimulation $\sim_s$ is both an equivalent and a congruent relation with respect to the operator ∂_H, we only need to check if each axiom in Table 13 is sound modulo step bisimulation equivalence.

Though transition rules in Table 12 are defined in the flavor of single event, they can be modified into a step (a set of events within which each event is pairwise concurrent), we omit them. If we treat a single event as a step containing just one event, the proof of this soundness theorem does not exist any problem, so we use this way and still use the transition rules in Table 12.

We omit the defining axioms, including axioms $D1-D3$, and we only prove the soundness of the non-trivial axioms, including axioms $D4-D6$.

- **Axiom** $D4$. Let p,q be $APTC$ processes, and $\partial_H(p+q)=\partial_H(p)+\partial_H(q)$, it is sufficient to prove that $\partial_H(p+q)\sim_s\partial_H(p)+\partial_H(q)$. By the transition rules for operator $+$ in Table 5 and ∂_H in Table 12, we get

$$\frac{p\xrightarrow{e_1}\surd\quad(e_1\notin H)}{\partial_H(p+q)\xrightarrow{e_1}\surd}\qquad\frac{p\xrightarrow{e_1}\surd\quad(e_1\notin H)}{\partial_H(p)+\partial_H(q)\xrightarrow{e_1}\surd}$$

$$\frac{q\xrightarrow{e_2}\surd\quad(e_2\notin H)}{\partial_H(p+q)\xrightarrow{e_2}\surd}\qquad\frac{q\xrightarrow{e_2}\surd\quad(e_2\notin H)}{\partial_H(p)+\partial_H(q)\xrightarrow{e_2}\surd}$$

$$\frac{p\xrightarrow{e_1}p'\quad(e_1\notin H)}{\partial_H(p+q)\xrightarrow{e_1}\partial_H(p')}\qquad\frac{p\xrightarrow{e_1}p'\quad(e_1\notin H)}{\partial_H(p)+\partial_H(q)\xrightarrow{e_1}\partial_H(p')}$$

$$\frac{q\xrightarrow{e_2}q'\quad(e_2\notin H)}{\partial_H(p+q)\xrightarrow{e_2}\partial_H(q')}\qquad\frac{q\xrightarrow{e_2}q'\quad(e_2\notin H)}{\partial_H(p)+\partial_H(q)\xrightarrow{e_2}\partial_H(q')}$$

  So, $\partial_H(p+q)\sim_s\partial_H(p)+\partial_H(q)$, as desired.
- **Axiom** $D5$. Let p,q be $APTC$ processes, and $\partial_H(p\cdot q)=\partial_H(p)\cdot\partial_H(q)$, it is sufficient to prove that $\partial_H(p\cdot q)\sim_s\partial_H(p)\cdot\partial_H(q)$. By the transition rules for operator $\cdot$ in Table 5 and ∂_H in Table 12, we get

$$\frac{p\xrightarrow{e_1}\surd\quad(e_1\notin H)}{\partial_H(p\cdot q)\xrightarrow{e_1}\partial_H(q)}\qquad\frac{p\xrightarrow{e_1}\surd\quad(e_1\notin H)}{\partial_H(p)\cdot\partial_H(q)\xrightarrow{e_1}\partial_H(q)}$$

$$\frac{p\xrightarrow{e_1}p'\quad(e_1\notin H)}{\partial_H(p\cdot q)\xrightarrow{e_1}\partial_H(p'\cdot q)}\qquad\frac{p\xrightarrow{e_1}p'\quad(e_1\notin H)}{\partial_H(p)\cdot\partial_H(q)\xrightarrow{e_1}\partial_H(p')\cdot\partial_H(q)}$$

  So, with the assumption $\partial_H(p'\cdot q)=\partial_H(p')\cdot\partial_H(q)$, $\partial_H(p\cdot q)\sim_s\partial_H(p)\cdot\partial_H(q)$, as desired.
- **Axiom** $D6$. Let p,q be $APTC$ processes, and $\partial_H(p\parallel q)=\partial_H(p)\parallel\partial_H(q)$, it is sufficient to prove that $\partial_H(p\parallel q)\sim_s\partial_H(p)\parallel\partial_H(q)$. By the transition rules for operator $\parallel$ in Table 7 and ∂_H in Table 12, we get

$$\frac{p\xrightarrow{e_1}\surd\quad q\xrightarrow{e_2}\surd\quad(e_1,e_2\notin H)}{\partial_H(p\parallel q)\xrightarrow{\{e_1,e_2\}}\surd}\qquad\frac{p\xrightarrow{e_1}\surd\quad q\xrightarrow{e_2}\surd\quad(e_1,e_2\notin H)}{\partial_H(p)\parallel\partial_H(q)\xrightarrow{\{e_1,e_2\}}\surd}$$

$$\frac{p\xrightarrow{e_1}p'\quad q\xrightarrow{e_2}\surd\quad(e_1,e_2\notin H)}{\partial_H(p\parallel q)\xrightarrow{\{e_1,e_2\}}\partial_H(p')}\qquad\frac{p\xrightarrow{e_1}p'\quad q\xrightarrow{e_2}\surd\quad(e_1,e_2\notin H)}{\partial_H(p)\parallel\partial_H(q)\xrightarrow{\{e_1,e_2\}}\partial_H(p')}$$

$$\frac{p\xrightarrow{e_1}\surd\quad q\xrightarrow{e_2}q'\quad(e_1,e_2\notin H)}{\partial_H(p\parallel q)\xrightarrow{\{e_1,e_2\}}\partial_H(q')}\qquad\frac{p\xrightarrow{e_1}\surd\quad q\xrightarrow{e_2}q'\quad(e_1,e_2\notin H)}{\partial_H(p)\parallel\partial_H(q)\xrightarrow{\{e_1,e_2\}}\partial_H(q')}$$

$$\frac{p\xrightarrow{e_1}p'\quad q\xrightarrow{e_2}q'\quad(e_1,e_2\notin H)}{\partial_H(p\parallel q)\xrightarrow{\{e_1,e_2\}}\partial_H(p'\between q')}\qquad\frac{p\xrightarrow{e_1}p'\quad q\xrightarrow{e_2}q'\quad(e_1,e_2\notin H)}{\partial_H(p)\parallel\partial_H(q)\xrightarrow{\{e_1,e_2\}}\partial_H(p')\between\partial_H(q')}$$

  So, with the assumption $\partial_H(p'\between q')=\partial_H(p')\between\partial_H(q')$, $\partial_H(p\parallel q)\sim_s\partial_H(p)\parallel\partial_H(q)$, as desired.

□

Theorem 4.18 (Completeness of $APTC$ modulo step bisimulation equivalence). Let p and q be closed $APTC$ terms including encapsulation operator ∂_H, if $p \sim_s q$ then $p = q$.

Proof. Firstly, by the elimination theorem of $APTC$ (see Theorem 4.16), we know that the normal form of $APTC$ does not contain ∂_H, and for each closed $APTC$ term p, there exists a closed basic $APTC$ term p', such that $APTC \vdash p = p'$, so, we only need to consider closed basic $APTC$ terms.

Similarly to Theorem 4.8, we can prove that for normal forms n and n', if $n \sim_s n'$ then $n =_{AC} n'$.

Finally, let s and t be basic $APTC$ terms, and $s \sim_s t$, there are normal forms n and n', such that $s = n$ and $t = n'$. The soundness theorem of $APTC$ modulo step bisimulation equivalence (see Theorem 4.17) yields $s \sim_s n$ and $t \sim_s n'$, so $n \sim_s s \sim_s t \sim_s n'$. Since if $n \sim_s n'$ then $n =_{AC} n'$, $s = n =_{AC} n' = t$, as desired. □

Theorem 4.19 (Soundness of $APTC$ modulo pomset bisimulation equivalence). Let x and y be $APTC$ terms including encapsulation operator ∂_H. If $APTC \vdash x = y$, then $x \sim_p y$.

Proof. Since pomset bisimulation $\sim_p$ is both an equivalent and a congruent relation with respect to the operator ∂_H, we only need to check if each axiom in Table 13 is sound modulo pomset bisimulation equivalence.

From the definition of pomset bisimulation (see Definition 2.17), we know that pomset bisimulation is defined by pomset transitions, which are labeled by pomsets. In a pomset transition, the events in the pomset are either within causality relations (defined by $\cdot$) or in concurrency (implicitly defined by $\cdot$ and $+$, and explicitly defined by $\between$), of course, they are pairwise consistent (without conflicts). In Theorem 4.17, we have already proven the case that all events are pairwise concurrent, so, we only need to prove the case of events in causality. Without loss of generality, we take a pomset of $P = \{e_1, e_2 : e_1 \cdot e_2\}$. Then the pomset transition labeled by the above P is just composed of one single event transition labeled by e_1 succeeded by another single event transition labeled by e_2, that is, $\xrightarrow{P} = \xrightarrow{e_1} \xrightarrow{e_2}$.

Similarly to the proof of soundness of $APTC$ modulo step bisimulation equivalence (see Theorem 4.17), we can prove that each axiom in Table 13 is sound modulo pomset bisimulation equivalence, we omit them. □

Theorem 4.20 (Completeness of $APTC$ modulo pomset bisimulation equivalence). Let p and q be closed $APTC$ terms including encapsulation operator ∂_H, if $p \sim_p q$ then $p = q$.

Proof. Firstly, by the elimination theorem of $APTC$ (see Theorem 4.16), we know that the normal form of $APTC$ does not contain ∂_H, and for each closed $APTC$ term p, there exists a closed basic $APTC$ term p', such that $APTC \vdash p = p'$, so, we only need to consider closed basic $APTC$ terms.

Similarly to Theorem 4.18, we can prove that for normal forms n and n', if $n \sim_p n'$ then $n =_{AC} n'$.

Finally, let s and t be basic $APTC$ terms, and $s \sim_p t$, there are normal forms n and n', such that $s = n$ and $t = n'$. The soundness theorem of $APTC$ modulo pomset bisimulation equivalence (see Theorem 4.19) yields $s \sim_p n$ and $t \sim_p n'$, so $n \sim_p s \sim_p t \sim_p n'$. Since if $n \sim_p n'$ then $n =_{AC} n'$, $s = n =_{AC} n' = t$, as desired. □

Theorem 4.21 (Soundness of $APTC$ modulo hp-bisimulation equivalence). Let x and y be $APTC$ terms including encapsulation operator ∂_H. If $APTC \vdash x = y$, then $x \sim_{hp} y$.

Proof. Since hp-bisimulation $\sim_{hp}$ is both an equivalent and a congruent relation with respect to the operator ∂_H, we only need to check if each axiom in Table 13 is sound modulo hp-bisimulation equivalence.

From the definition of hp-bisimulation (see Definition 2.21), we know that hp-bisimulation is defined on the posetal product $(C_1, f, C_2), f : C_1 \to C_2$ isomorphism. Two process terms s related to C_1 and t related to C_2, and $f : C_1 \to C_2$ isomorphism. Initially, $(C_1, f, C_2) = (\varnothing, \varnothing, \varnothing)$, and $(\varnothing, \varnothing, \varnothing) \in \sim_{hp}$. When $s \xrightarrow{e} s'$ ($C_1 \xrightarrow{e} C_1'$), there will be $t \xrightarrow{e} t'$ ($C_2 \xrightarrow{e} C_2'$), and we define $f' = f[e \mapsto e]$. Then, if $(C_1, f, C_2) \in \sim_{hp}$, then $(C_1', f', C_2') \in \sim_{hp}$.

Similarly to the proof of soundness of $APTC$ modulo pomset bisimulation equivalence (see Theorem 4.19), we can prove that each axiom in Table 13 is sound modulo hp-bisimulation equivalence, we just need additionally to check the above conditions on hp-bisimulation, we omit them. □

Theorem 4.22 (Completeness of $APTC$ modulo hp-bisimulation equivalence). Let p and q be closed $APTC$ terms including encapsulation operator ∂_H, if $p \sim_{hp} q$ then $p = q$.

Proof. Firstly, by the elimination theorem of $APTC$ (see Theorem 4.16), we know that the normal form of

$APTC$ does not contain ∂_H, and for each closed $APTC$ term p, there exists a closed basic $APTC$ term p', such that $APTC \vdash p = p'$, so, we only need to consider closed basic $APTC$ terms.

Similarly to Theorem 4.20, we can prove that for normal forms n and n', if $n \sim_{hp} n'$ then $n =_{AC} n'$.

Finally, let s and t be basic $APTC$ terms, and $s \sim_{hp} t$, there are normal forms n and n', such that $s = n$ and $t = n'$. The soundness theorem of $APTC$ modulo hp-bisimulation equivalence (see Theorem 4.21) yields $s \sim_{hp} n$ and $t \sim_{hp} n'$, so $n \sim_{hp} s \sim_{hp} t \sim_{hp} n'$. Since if $n \sim_{hp} n'$ then $n =_{AC} n'$, $s = n =_{AC} n' = t$, as desired. □

5. Recursion

In this section, we introduce recursion to capture infinite processes based on $APTC$. Since in $APTC$, there are three basic operators $\cdot$, $+$ and $\|$, the recursion must adapted this situation to include $\|$.

In the following, E, F, G are recursion specifications, X, Y, Z are recursive variables.

5.1. Guarded Recursive Specifications

Definition 5.1 (Recursive specification). A recursive specification is a finite set of recursive equations

$$X_1 = t_1(X_1, \cdots, X_n)$$
$$\cdots$$
$$X_n = t_n(X_1, \cdots, X_n)$$

where the left-hand sides of X_i are called recursion variables, and the right-hand sides $t_i(X_1, \cdots, X_n)$ are reversible process terms in ARCP with possible occurrences of the recursion variables $X_1, \cdots, X_n$.

Definition 5.2 (Solution). Processes $p_1, \cdots, p_n$ are a solution for a recursive specification $\{X_i = t_i(X_1, \cdots, X_n) | i \in \{1, \cdots, n\}\}$ (with respect to truly concurrent bisimulation equivalences $\sim_s (\sim_p, \sim_{hp})$) if $p_i \sim_s (\sim_p, \sim_{hp}) t_i(p_1, \cdots, p_n)$ for $i \in \{1, \cdots, n\}$.

Definition 5.3 (Guarded recursive specification). A recursive specification

$$X_1 = t_1(X_1, \cdots, X_n)$$
$$\cdots$$
$$X_n = t_n(X_1, \cdots, X_n)$$

is guarded if the right-hand sides of its recursive equations can be adapted to the form by applications of the axioms in $APTC$ and replacing recursion variables by the right-hand sides of their recursive equations,

$$(a_{11} \| \cdots \| a_{1i_1}) \cdot s_1(X_1, \cdots, X_n) + \cdots + (a_{k1} \| \cdots \| a_{ki_k}) \cdot s_k(X_1, \cdots, X_n) + (b_{11} \| \cdots \| b_{1j_1}) + \cdots + (b_{1j_1} \| \cdots \| b_{lj_l})$$

where $a_{11}, \cdots, a_{1i_1}, a_{k1}, \cdots, a_{ki_k}, b_{11}, \cdots, b_{1j_1}, b_{1j_1}, \cdots, b_{lj_l} \in \mathbb{E}$, and the sum above is allowed to be empty, in which case it represents the deadlock δ.

Definition 5.4 (Linear recursive specification). A recursive specification is linear if its recursive equations are of the form

$$(a_{11} \| \cdots \| a_{1i_1}) X_1 + \cdots + (a_{k1} \| \cdots \| a_{ki_k}) X_k + (b_{11} \| \cdots \| b_{1j_1}) + \cdots + (b_{1j_1} \| \cdots \| b_{lj_l})$$

where $a_{11}, \cdots, a_{1i_1}, a_{k1}, \cdots, a_{ki_k}, b_{11}, \cdots, b_{1j_1}, b_{1j_1}, \cdots, b_{lj_l} \in \mathbb{E}$, and the sum above is allowed to be empty, in which case it represents the deadlock δ.

For a guarded recursive specifications E with the form

$$X_1 = t_1(X_1, \cdots, X_n)$$
$$\cdots$$

$$\frac{t_i(\langle X_1|E\rangle,\cdots,\langle X_n|E\rangle)\xrightarrow{e}\surd}{\langle X_i|E\rangle\xrightarrow{e}\surd}$$

$$\frac{t_i(\langle X_1|E\rangle,\cdots,\langle X_n|E\rangle)\xrightarrow{e}y}{\langle X_i|E\rangle\xrightarrow{e}y}$$

Table 15. Transition rules of guarded recursion

No.	Axiom
RDP	$\langle X_i\|E\rangle = t_i(\langle X_1\|E,\cdots,X_n\|E\rangle)\quad(i\in\{1,\cdots,n\})$
RSP	if $y_i = t_i(y_1,\cdots,y_n)$ for $i\in\{1,\cdots,n\}$, then $y_i = \langle X_i\|E\rangle\quad(i\in\{1,\cdots,n\})$

Table 16. Recursive definition and specification principle

$$X_n = t_n(X_1,\cdots,X_n)$$

the behavior of the solution $\langle X_i|E\rangle$ for the recursion variable X_i in E, where $i\in\{1,\cdots,n\}$, is exactly the behavior of their right-hand sides $t_i(X_1,\cdots,X_n)$, which is captured by the two transition rules in Table 15.

Theorem 5.5 (Conservitivity of $APTC$ with guarded recursion). $APTC$ with guarded recursion is a conservative extension of $APTC$.

Proof. Since the transition rules of $APTC$ are source-dependent, and the transition rules for guarded recursion in Table 15 contain only a fresh constant in their source, so the transition rules of $APTC$ with guarded recursion are a conservative extension of those of $APTC$. □

Theorem 5.6 (Congruence theorem of $APTC$ with guarded recursion). Truly concurrent bisimulation equivalences $\sim_p$, $\sim_s$ and $\sim_{hp}$ are all congruences with respect to $APTC$ with guarded recursion.

Proof. It follows the following two facts:

1. in a guarded recursive specification, right-hand sides of its recursive equations can be adapted to the form by applications of the axioms in $APTC$ and replacing recursion variables by the right-hand sides of their recursive equations;
2. truly concurrent bisimulation equivalences $\sim_p$, $\sim_s$ and $\sim_{hp}$ are all congruences with respect to all operators of $APTC$.

□

5.2. Recursive Definition and Specification Principles

The RDP (Recursive Definition Principle) and the RSP (Recursive Specification Principle) are shown in Table 16.

RDP follows immediately from the two transition rules for guarded recursion, which express that $\langle X_i|E\rangle$ and $t_i(\langle X_1|E\rangle,\cdots,\langle X_n|E\rangle)$ have the same initial transitions for $i\in\{1,\cdots,n\}$. RSP follows from the fact that guarded recursive specifications have only one solution.

Theorem 5.7 (Elimination theorem of $APTC$ with linear recursion). Each process term in $APTC$ with linear recursion is equal to a process term $\langle X_1|E\rangle$ with E a linear recursive specification.

Proof. By applying structural induction with respect to term size, each process term t_1 in $APTC$ with linear recursion generates a process can be expressed in the form of equations

$$t_i = (a_{i11}\parallel\cdots\parallel a_{i1i_1})t_{i1}+\cdots+(a_{ik_i1}\parallel\cdots\parallel a_{ik_ii_k})t_{ik_i}+(b_{i11}\parallel\cdots\parallel b_{i1i_1})+\cdots+(b_{il_i1}\parallel\cdots\parallel b_{il_ii_l})$$

for $i\in\{1,\cdots,n\}$. Let the linear recursive specification E consist of the recursive equations

$$X_i = (a_{i11} \parallel \cdots \parallel a_{i1i_1})X_{i1} + \cdots + (a_{ik_i1} \parallel \cdots \parallel a_{ik_ii_k})X_{ik_i} + (b_{i11} \parallel \cdots \parallel b_{i1i_1}) + \cdots + (b_{il_i1} \parallel \cdots \parallel b_{il_ii_l})$$

for $i \in \{1, \cdots, n\}$. Replacing X_i by t_i for $i \in \{1, \cdots, n\}$ is a solution for E, RSP yields $t_1 = \langle X_1|E\rangle$. □

Theorem 5.8 (Soundness of $APTC$ with guarded recursion). Let x and y be $APTC$ with guarded recursion terms. If $APTC$ with guarded recursion $\vdash x = y$, then

1. $x \backsim_s y$;
2. $x \backsim_p y$;
3. $x \backsim_{hp} y$.

Proof. (1) Soundness of $APTC$ with guarded recursion with respect to step bisimulation $\backsim_s$.

Since step bisimulation $\backsim_s$ is both an equivalent and a congruent relation with respect to $APTC$ with guarded recursion, we only need to check if each axiom in Table 16 is sound modulo step bisimulation equivalence.

Though transition rules in Table 15 are defined in the flavor of single event, they can be modified into a step (a set of events within which each event is pairwise concurrent), we omit them. If we treat a single event as a step containing just one event, the proof of this soundness theorem does not exist any problem, so we use this way and still use the transition rules in Table 15.

- RDP. $\langle X_i|E\rangle = t_i(\langle X_1|E, \cdots, X_n|E\rangle) \quad (i \in \{1, \cdots, n\})$, it is sufficient to prove that $\langle X_i|E\rangle \backsim_s t_i(\langle X_1|E, \cdots, X_n|E\rangle) \quad (i \in \{1, \cdots, n\})$. By the transition rules for guarded recursion in Table 15, we get

$$\frac{t_i(\langle X_1|E\rangle, \cdots, \langle X_n|E\rangle) \xrightarrow{e} \surd}{\langle X_i|E\rangle \xrightarrow{e} \surd}$$

$$\frac{t_i(\langle X_1|E\rangle, \cdots, \langle X_n|E\rangle) \xrightarrow{e} y}{\langle X_i|E\rangle \xrightarrow{e} y}$$

  So, $\langle X_i|E\rangle \backsim_s t_i(\langle X_1|E, \cdots, X_n|E\rangle) \quad (i \in \{1, \cdots, n\})$, as desired.
- RSP. if $y_i = t_i(y_1, \cdots, y_n)$ for $i \in \{1, \cdots, n\}$, then $y_i = \langle X_i|E\rangle \quad (i \in \{1, \cdots, n\})$, it is sufficient to prove that if $y_i = t_i(y_1, \cdots, y_n)$ for $i \in \{1, \cdots, n\}$, then $y_i \backsim_s \langle X_i|E\rangle \quad (i \in \{1, \cdots, n\})$. By the transition rules for guarded recursion in Table 15, we get

$$\frac{t_i(\langle X_1|E\rangle, \cdots, \langle X_n|E\rangle) \xrightarrow{e} \surd}{\langle X_i|E\rangle \xrightarrow{e} \surd}$$

$$\frac{t_i(\langle X_1|E\rangle, \cdots, \langle X_n|E\rangle) \xrightarrow{e} \surd}{y_i \xrightarrow{e} \surd}$$

$$\frac{t_i(\langle X_1|E\rangle, \cdots, \langle X_n|E\rangle) \xrightarrow{e} y}{\langle X_i|E\rangle \xrightarrow{e} y}$$

$$\frac{t_i(\langle X_1|E\rangle, \cdots, \langle X_n|E\rangle) \xrightarrow{e} y}{y_i \xrightarrow{e} y}$$

  So, if $y_i = t_i(y_1, \cdots, y_n)$ for $i \in \{1, \cdots, n\}$, then $y_i \backsim_s \langle X_i|E\rangle \quad (i \in \{1, \cdots, n\})$, as desired.

(2) Soundness of $APTC$ with guarded recursion with respect to pomset bisimulation $\backsim_p$.

Since pomset bisimulation $\backsim_p$ is both an equivalent and a congruent relation with respect to the guarded recursion, we only need to check if each axiom in Table 16 is sound modulo pomset bisimulation equivalence.

From the definition of pomset bisimulation (see Definition 2.17), we know that pomset bisimulation is defined by pomset transitions, which are labeled by pomsets. In a pomset transition, the events in the pomset are either within causality relations (defined by $\cdot$) or in concurrency (implicitly defined by $\cdot$ and $+$, and explicitly defined by $\between$), of course, they are pairwise consistent (without conflicts). In (1), we have

already proven the case that all events are pairwise concurrent, so, we only need to prove the case of events in causality. Without loss of generality, we take a pomset of $P = \{e_1, e_2 : e_1 \cdot e_2\}$. Then the pomset transition labeled by the above P is just composed of one single event transition labeled by e_1 succeeded by another single event transition labeled by e_2, that is, $\xrightarrow{P} = \xrightarrow{e_1}\xrightarrow{e_2}$.

Similarly to the proof of soundness of $APTC$ with guarded recursion modulo step bisimulation equivalence (1), we can prove that each axiom in Table 16 is sound modulo pomset bisimulation equivalence, we omit them.

(3) Soundness of $APTC$ with guarded recursion with respect to hp-bisimulation $\sim_{hp}$.

Since hp-bisimulation $\sim_{hp}$ is both an equivalent and a congruent relation with respect to guarded recursion, we only need to check if each axiom in Table 16 is sound modulo hp-bisimulation equivalence.

From the definition of hp-bisimulation (see Definition 2.21), we know that hp-bisimulation is defined on the posetal product $(C_1, f, C_2), f : C_1 \to C_2$ isomorphism. Two process terms s related to C_1 and t related to C_2, and $f : C_1 \to C_2$ isomorphism. Initially, $(C_1, f, C_2) = (\varnothing, \varnothing, \varnothing)$, and $(\varnothing, \varnothing, \varnothing) \in \sim_{hp}$. When $s \xrightarrow{e} s'$ ($C_1 \xrightarrow{e} C_1'$), there will be $t \xrightarrow{e} t'$ ($C_2 \xrightarrow{e} C_2'$), and we define $f' = f[e \mapsto e]$. Then, if $(C_1, f, C_2) \in \sim_{hp}$, then $(C_1', f', C_2') \in \sim_{hp}$.

Similarly to the proof of soundness of $APTC$ with guarded recursion modulo pomset bisimulation equivalence (2), we can prove that each axiom in Table 16 is sound modulo hp-bisimulation equivalence, we just need additionally to check the above conditions on hp-bisimulation, we omit them. □

Theorem 5.9 (Completeness of $APTC$ with linear recursion). Let p and q be closed $APTC$ with linear recursion terms, then,

1. if $p \sim_s q$ then $p = q$;
2. if $p \sim_p q$ then $p = q$;
3. if $p \sim_{hp} q$ then $p = q$.

Proof. Firstly, by the elimination theorem of $APTC$ with guarded recursion (see Theorem 5.7), we know that each process term in $APTC$ with linear recursion is equal to a process term $\langle X_1|E\rangle$ with E a linear recursive specification.

It remains to prove the following cases.

(1) If $\langle X_1|E_1\rangle \sim_s \langle Y_1|E_2\rangle$ for linear recursive specification E_1 and E_2, then $\langle X_1|E_1\rangle = \langle Y_1|E_2\rangle$.

Let E_1 consist of recursive equations $X = t_X$ for $X \in \mathcal{X}$ and E_2 consists of recursion equations $Y = t_Y$ for $Y \in \mathcal{Y}$. Let the linear recursive specification E consist of recursion equations $Z_{XY} = t_{XY}$, and $\langle X|E_1\rangle \sim_s \langle Y|E_2\rangle$, and t_{XY} consists of the following summands:

1. t_{XY} contains a summand $(a_1 \parallel \cdots \parallel a_m)Z_{X'Y'}$ iff t_X contains the summand $(a_1 \parallel \cdots \parallel a_m)X'$ and t_Y contains the summand $(a_1 \parallel \cdots \parallel a_m)Y'$ such that $\langle X'|E_1\rangle \sim_s \langle Y'|E_2\rangle$;
2. t_{XY} contains a summand $b_1 \parallel \cdots \parallel b_n$ iff t_X contains the summand $b_1 \parallel \cdots \parallel b_n$ and t_Y contains the summand $b_1 \parallel \cdots \parallel b_n$.

Let σ map recursion variable X in E_1 to $\langle X|E_1\rangle$, and let ψ map recursion variable Z_{XY} in E to $\langle X|E_1\rangle$. So, $\sigma((a_1 \parallel \cdots \parallel a_m)X') \equiv (a_1 \parallel \cdots \parallel a_m)\langle X'|E_1\rangle \equiv \psi((a_1 \parallel \cdots \parallel a_m)Z_{X'Y'})$, so by RDP, we get $\langle X|E_1\rangle = \sigma(t_X) = \psi(t_{XY})$. Then by RSP, $\langle X|E_1\rangle = \langle Z_{XY}|E\rangle$, particularly, $\langle X_1|E_1\rangle = \langle Z_{X_1Y_1}|E\rangle$. Similarly, we can obtain $\langle Y_1|E_2\rangle = \langle Z_{X_1Y_1}|E\rangle$. Finally, $\langle X_1|E_1\rangle = \langle Z_{X_1Y_1}|E\rangle = \langle Y_1|E_2\rangle$, as desired.

(2) If $\langle X_1|E_1\rangle \sim_p \langle Y_1|E_2\rangle$ for linear recursive specification E_1 and E_2, then $\langle X_1|E_1\rangle = \langle Y_1|E_2\rangle$.

It can be proven similarly to (1), we omit it.

(3) If $\langle X_1|E_1\rangle \sim_{hp} \langle Y_1|E_2\rangle$ for linear recursive specification E_1 and E_2, then $\langle X_1|E_1\rangle = \langle Y_1|E_2\rangle$.

It can be proven similarly to (1), we omit it. □

5.3. Approximation Induction Principle

In this subsection, we introduce approximation induction principle (AIP) and try to explain that AIP is still valid in true concurrency. AIP can be used to try and equate truly concurrent bisimilar guarded recursive specifications. AIP says that if two process terms are truly concurrent bisimilar up to any finite depth, then they are truly concurrent bisimilar.

$$\frac{x \xrightarrow{e} \surd}{\pi_{n+1}(x) \xrightarrow{e} \surd} \quad \frac{x \xrightarrow{e} x'}{\pi_{n+1}(x) \xrightarrow{e} \pi_n(x')}$$

Table 17. Transition rules of encapsulation operator ∂_H

No.	Axiom
$PR1$	$\pi_n(x+y) = \pi_n(x) + \pi_n(y)$
$PR2$	$\pi_n(x \parallel y) = \pi_n(x) \parallel \pi_n(y)$
$PR3$	$\pi_{n+1}(e) = e$
$PR4$	$\pi_{n+1}(e \cdot x) = e \cdot \pi_n(x)$
$PR5$	$\pi_0(x) = \delta$
$PR6$	$\pi_n(\delta) = \delta$

Table 18. Axioms of projection operator

Also, we need the auxiliary unary projection operator π_n for $n \in \mathbb{N}$ and $\mathbb{N} \triangleq \{0, 1, 2, \cdots\}$. The transition rules of π_n are expressed in Table 17.

Based on the transition rules for projection operator π_n in Table 17, we design the axioms as Table 18 shows.

The axioms $PR1 - PR2$ say that $\pi_n(s+t)$ and $\pi_n(s \parallel t)$ can execute transitions of s and t up to depth n. $PR3$ says that $\pi_{n+1}(e)$ executes e and terminates successfully. $PR4$ says that $\pi_{n+1}(e \cdots t)$ executes e and then executes transitions of t up to depth n. $PR5$ and $PR6$ say that $\pi_0(t)$ and $\pi_n(\delta)$ exhibit no actions.

Theorem 5.10 (Conservativity of $APTC$ with projection operator and guarded recursion). $APTC$ with projection operator and guarded recursion is a conservative extension of $APTC$ with guarded recursion.

Proof. It follows from the following two facts (see Theorem 2.8).

1. The transition rules of $APTC$ with guarded recursion are all source-dependent;
2. The sources of the transition rules for the projection operator contain an occurrence of π_n.

□

Theorem 5.11 (Congruence theorem of projection operator π_n). Truly concurrent bisimulation equivalences $\sim_p$, $\sim_s$, $\sim_{hp}$ and $\sim_{hhp}$ are all congruences with respect to projection operator π_n.

Proof. (1) Case pomset bisimulation equivalence $\sim_p$.

Let x and y be $APTC$ with projection operator and guarded recursion processes, and $x \sim_p y$, it is sufficient to prove that $\pi_{n+1}(x) \sim_p \pi_{n+1}(y)$.

By the definition of pomset bisimulation $\sim_p$ (Definition 2.17), $x \sim_p y$ means that

$$x \xrightarrow{X} x' \quad y \xrightarrow{Y} y'$$

with $X \subseteq x$, $Y \subseteq y$, $X \sim Y$ and $x' \sim_p y'$.

By the pomset transition rules for projection operator π_n in Table 17, we can get

$$\pi_{n+1}(x) \xrightarrow{X} \surd \quad \pi_{n+1}(y) \xrightarrow{Y} \surd$$

with $X \subseteq x$, $Y \subseteq y$, and $X \sim Y$, so, we get $\pi_{n+1}(x) \sim_p \pi_{n+1}(y)$, as desired.

Or, we can get

$$\pi_{n+1}(x) \xrightarrow{X} \pi_n(x') \quad \pi_{n+1}(y) \xrightarrow{Y} \pi_n(y')$$

with $X \subseteq x$, $Y \subseteq y$, $X \sim Y$, $x' \sim_p y'$ and the assumption $\pi_n(x') \sim_p \pi_n(y')$, so, we get $\pi_{n+1}(x) \sim_p \pi_{n+1}(y)$, as desired.

(2) The cases of step bisimulation $\sim_s$, hp-bisimulation $\sim_{hp}$ and hhp-bisimulation $\sim_{hhp}$ can be proven similarly, we omit them. □

Theorem 5.12 (Elimination theorem of $APTC$ with linear recursion and projection operator). Each process term in $APTC$ with linear recursion and projection operator is equal to a process term $\langle X_1|E\rangle$ with E a linear recursive specification.

Proof. By applying structural induction with respect to term size, each process term t_1 in $APTC$ with linear recursion and projection operator π_n generates a process can be expressed in the form of equations

$$t_i = (a_{i11} \parallel \cdots \parallel a_{i1i_1})t_{i1} + \cdots + (a_{ik_i1} \parallel \cdots \parallel a_{ik_ii_k})t_{ik_i} + (b_{i11} \parallel \cdots \parallel b_{i1i_1}) + \cdots + (b_{il_i1} \parallel \cdots \parallel b_{il_ii_l})$$

for $i \in \{1, \cdots, n\}$. Let the linear recursive specification E consist of the recursive equations

$$X_i = (a_{i11} \parallel \cdots \parallel a_{i1i_1})X_{i1} + \cdots + (a_{ik_i1} \parallel \cdots \parallel a_{ik_ii_k})X_{ik_i} + (b_{i11} \parallel \cdots \parallel b_{i1i_1}) + \cdots + (b_{il_i1} \parallel \cdots \parallel b_{il_ii_l})$$

for $i \in \{1, \cdots, n\}$. Replacing X_i by t_i for $i \in \{1, \cdots, n\}$ is a solution for E, RSP yields $t_1 = \langle X_1|E\rangle$. That is, in E, there is not the occurrence of projection operator π_n. □

Theorem 5.13 (Soundness of $APTC$ with projection operator and guarded recursion). Let x and y be $APTC$ with projection operator and guarded recursion terms. If $APTC$ with projection operator and guarded recursion $\vdash x = y$, then

1. $x \sim_s y$;
2. $x \sim_p y$;
3. $x \sim_{hp} y$.

Proof. (1) Soundness of $APTC$ with projection operator and guarded recursion with respect to step bisimulation $\sim_s$.

Since step bisimulation $\sim_s$ is both an equivalent and a congruent relation with respect to $APTC$ with projection operator and guarded recursion, we only need to check if each axiom in Table 18 is sound modulo step bisimulation equivalence.

Though transition rules in Table 17 are defined in the flavor of single event, they can be modified into a step (a set of events within which each event is pairwise concurrent), we omit them. If we treat a single event as a step containing just one event, the proof of this soundness theorem does not exist any problem, so we use this way and still use the transition rules in Table 17.

We only prove the soundness of the non-trivial axioms $PR1$, $PR2$ and $PR4$.

- **Axiom** $PR1$. Let p and q be $APTC$ with guarded recursion and projection operator processes. $\pi_n(p+q) = \pi_n(p)+\pi_n(q)$, it is sufficient to prove that $\pi_n(p+q) \sim_s \pi_n(p)+\pi_n(q)$. By the transition rules for projection operator π_n in Table 17 and $+$ in Table 5, we get

$$\frac{p \xrightarrow{e_1} \surd}{\pi_{n+1}(p+q) \xrightarrow{e_1} \surd} \quad \frac{p \xrightarrow{e_1} \surd}{\pi_{n+1}(p) + \pi_{n+1}(q) \xrightarrow{e_1} \surd}$$

$$\frac{q \xrightarrow{e_2} \surd}{\pi_{n+1}(p+q) \xrightarrow{e_2} \surd} \quad \frac{q \xrightarrow{e_2} \surd}{\pi_{n+1}(p) + \pi_{n+1}(q) \xrightarrow{e_2} \surd}$$

$$\frac{p \xrightarrow{e_1} p'}{\pi_{n+1}(p+q) \xrightarrow{e_1} \pi_n(p')} \quad \frac{p \xrightarrow{e_1} p'}{\pi_{n+1}(p) + \pi_{n+1}(q) \xrightarrow{e_1} \pi_n(p')}$$

$$\frac{q \xrightarrow{e_2} q'}{\pi_{n+1}(p+q) \xrightarrow{e_2} \pi_n(q')} \quad \frac{q \xrightarrow{e_2} q'}{\pi_{n+1}(p) + \pi_{n+1}(q) \xrightarrow{e_2} \pi_n(q')}$$

, we get

So, $\pi_n(p+q) \sim_s \pi_n(p) + \pi_n(q)$, as desired.

- **Axiom** $PR2$. Let p, q be $APTC$ with guarded recursion and projection operator processes, and $\pi_n(p \parallel q) = \pi_n(p) \parallel \pi_n(q)$, it is sufficient to prove that $\pi_n(p \parallel q) \sim_s \pi_n(p) \parallel \pi_n(q)$. By the transition rules for operator $\parallel$ in Table 7 and π_n in Table 17, we get

$$\frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} \surd}{\pi_{n+1}(p \parallel q) \xrightarrow{\{e_1,e_2\}} \surd} \quad \frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} \surd}{\pi_{n+1}(p) \parallel \pi_{n+1}(q) \xrightarrow{\{e_1,e_2\}} \surd}$$

$$\frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} \surd}{\pi_{n+1}(p \parallel q) \xrightarrow{\{e_1,e_2\}} \pi_n(p')} \quad \frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} \surd}{\pi_{n+1}(p) \parallel \pi_{n+1}(q) \xrightarrow{\{e_1,e_2\}} \pi_n(p')}$$

$$\frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} q'}{\pi_{n+1}(p \parallel q) \xrightarrow{\{e_1,e_2\}} \pi_n(q')} \quad \frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} q'}{\pi_{n+1}(p) \parallel \pi_{n+1}(q) \xrightarrow{\{e_1,e_2\}} \pi_n(q')}$$

$$\frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} q'}{\pi_{n+1}(p \parallel q) \xrightarrow{\{e_1,e_2\}} \pi_n(p' \between q')} \quad \frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} q'}{\pi_{n+1}(p) \parallel \pi_{n+1}(q) \xrightarrow{\{e_1,e_2\}} \pi_n(p') \between \pi_n(q')}$$

  So, with the assumption $\pi_n(p' \between q') = \pi_n(p') \between \pi_n(q')$, $\pi_n(p \parallel q) \sim_s \pi_n(p) \parallel \pi_n(q)$, as desired.
- **Axiom** $PR4$. Let p be an $APTC$ with guarded recursion and projection operator process, and $\pi_{n+1}(e \cdot p) = e \cdot \pi_n(p)$, it is sufficient to prove that $\pi_{n+1}(e \cdot p) \sim_s e \cdot \pi_n(p)$. By the transition rules for operator $\cdot$ in Table 5 and π_n in Table 17, we get

$$\frac{e \xrightarrow{e} \surd}{\pi_{n+1}(e \cdot p) \xrightarrow{e} \pi_n(p)} \quad \frac{e \xrightarrow{e} \surd}{e \cdot \pi_n(p) \xrightarrow{e} \pi_n(p)}$$

  So, $\pi_{n+1}(e \cdot p) \sim_s e \cdot \pi_n(p)$, as desired.

(2) Soundness of $APTC$ with guarded recursion and projection operator with respect to pomset bisimulation $\sim_p$.

Since pomset bisimulation $\sim_p$ is both an equivalent and a congruent relation with respect to $APTC$ with guarded recursion and projection operator, we only need to check if each axiom in Table 18 is sound modulo pomset bisimulation equivalence.

From the definition of pomset bisimulation (see Definition 2.17), we know that pomset bisimulation is defined by pomset transitions, which are labeled by pomsets. In a pomset transition, the events in the pomset are either within causality relations (defined by $\cdot$) or in concurrency (implicitly defined by $\cdot$ and $+$, and explicitly defined by $\between$), of course, they are pairwise consistent (without conflicts). In (1), we have already proven the case that all events are pairwise concurrent, so, we only need to prove the case of events in causality. Without loss of generality, we take a pomset of $P = \{e_1, e_2 : e_1 \cdot e_2\}$. Then the pomset transition labeled by the above P is just composed of one single event transition labeled by e_1 succeeded by another single event transition labeled by e_2, that is, $\xrightarrow{P} = \xrightarrow{e_1} \xrightarrow{e_2}$.

Similarly to the proof of soundness of $APTC$ with guarded recursion and projection operator modulo step bisimulation equivalence (1), we can prove that each axiom in Table 18 is sound modulo pomset bisimulation equivalence, we omit them.

(3) Soundness of $APTC$ with guarded recursion and projection operator with respect to hp-bisimulation $\sim_{hp}$.

Since hp-bisimulation $\sim_{hp}$ is both an equivalent and a congruent relation with respect to $APTC$ with guarded recursion and projection operator, we only need to check if each axiom in Table 18 is sound modulo hp-bisimulation equivalence.

From the definition of hp-bisimulation (see Definition 2.21), we know that hp-bisimulation is defined on the posetal product $(C_1, f, C_2), f : C_1 \to C_2$ isomorphism. Two process terms s related to C_1 and t related to C_2, and $f : C_1 \to C_2$ isomorphism. Initially, $(C_1, f, C_2) = (\varnothing, \varnothing, \varnothing)$, and $(\varnothing, \varnothing, \varnothing) \in \sim_{hp}$. When $s \xrightarrow{e} s'$ ($C_1 \xrightarrow{e} C_1'$), there will be $t \xrightarrow{e} t'$ ($C_2 \xrightarrow{e} C_2'$), and we define $f' = f[e \mapsto e]$. Then, if $(C_1, f, C_2) \in \sim_{hp}$, then $(C_1', f', C_2') \in \sim_{hp}$.

No.	Axiom
AIP	if $\pi_n(x)=\pi_n(y)$ for $n\in\mathbb{N}$, then $x=y$

Table 19. AIP

Similarly to the proof of soundness of $APTC$ with guarded recursion and projection operator modulo pomset bisimulation equivalence (2), we can prove that each axiom in Table 18 is sound modulo hp-bisimulation equivalence, we just need additionally to check the above conditions on hp-bisimulation, we omit them. □

Then AIP is given in Table 19.

Theorem 5.14 (Soundness of AIP). Let x and y be $APTC$ with projection operator and guarded recursion terms.

1. If $\pi_n(x)\sim_s\pi_n(y)$ for $n\in\mathbb{N}$, then $x\sim_s y$;
2. If $\pi_n(x)\sim_p\pi_n(y)$ for $n\in\mathbb{N}$, then $x\sim_p y$;
3. If $\pi_n(x)\sim_{hp}\pi_n(y)$ for $n\in\mathbb{N}$, then $x\sim_{hp} y$.

Proof. (1) If $\pi_n(x)\sim_s\pi_n(y)$ for $n\in\mathbb{N}$, then $x\sim_s y$.

Since step bisimulation $\sim_s$ is both an equivalent and a congruent relation with respect to $APTC$ with guarded recursion and projection operator, we only need to check if AIP in Table 19 is sound modulo step bisimulation equivalence.

Let p,p_0 and q,q_0 be closed $APTC$ with projection operator and guarded recursion terms such that $\pi_n(p_0)\sim_s\pi_n(q_0)$ for $n\in\mathbb{N}$. We define a relation R such that pRq iff $\pi_n(p)\sim_s\pi_n(q)$. Obviously, p_0Rq_0, next, we prove that $R\in\sim_s$.

Let pRq and $p\xrightarrow{\{e_1,\cdots,e_k\}}\surd$, then $\pi_1(p)\xrightarrow{\{e_1,\cdots,e_k\}}\surd$, $\pi_1(p)\sim_s\pi_1(q)$ yields $\pi_1(q)\xrightarrow{\{e_1,\cdots,e_k\}}\surd$. Similarly, $q\xrightarrow{\{e_1,\cdots,e_k\}}\surd$ implies $p\xrightarrow{\{e_1,\cdots,e_k\}}\surd$.

Let pRq and $p\xrightarrow{\{e_1,\cdots,e_k\}}p'$. We define the set of process terms

$$S_n\triangleq\{q'|q\xrightarrow{\{e_1,\cdots,e_k\}}q'\text{ and }\pi_n(p')\sim_s\pi_n(q')\}$$

1. Since $\pi_{n+1}(p)\sim_s\pi_{n+1}(q)$ and $\pi_{n+1}(p)\xrightarrow{\{e_1,\cdots,e_k\}}\pi_n(p')$, there exist q' such that $\pi_{n+1}(q)\xrightarrow{\{e_1,\cdots,e_k\}}\pi_n(q')$ and $\pi_n(p')\sim_s\pi_n(q')$. So, S_n is not empty.
2. There are only finitely many q' such that $q\xrightarrow{\{e_1,\cdots,e_k\}}q'$, so, S_n is finite.
3. $\pi_{n+1}(p)\sim_s\pi_{n+1}(q)$ implies $\pi_n(p')\sim_s\pi_n(q')$, so $S_n\supseteq S_{n+1}$.

So, S_n has a non-empty intersection, and let q' be in this intersection, then $q\xrightarrow{\{e_1,\cdots,e_k\}}q'$ and $\pi_n(p')\sim_s\pi_n(q')$, so $p'Rq'$. Similarly, let pq, we can obtain $q\xrightarrow{\{e_1,\cdots,e_k\}}q'$ implies $p\xrightarrow{\{e_1,\cdots,e_k\}}p'$ such that $p'Rq'$.

Finally, $R\in\sim_s$ and $p_0\sim_s q_0$, as desired.

(2) If $\pi_n(x)\sim_p\pi_n(y)$ for $n\in\mathbb{N}$, then $x\sim_p y$.

Since pomset bisimulation $\sim_p$ is both an equivalent and a congruent relation with respect to $APTC$ with guarded recursion and projection operator, we only need to check if AIP in Table 19 is sound modulo pomset bisimulation equivalence.

From the definition of pomset bisimulation (see Definition 2.17), we know that pomset bisimulation is defined by pomset transitions, which are labeled by pomsets. In a pomset transition, the events in the pomset are either within causality relations (defined by $\cdot$) or in concurrency (implicitly defined by $\cdot$ and $+$, and explicitly defined by $\between$), of course, they are pairwise consistent (without conflicts). In (1), we have already proven the case that all events are pairwise concurrent, so, we only need to prove the case of events in causality. Without loss of generality, we take a pomset of $P=\{e_1,e_2:e_1\cdot e_2\}$. Then the pomset transition labeled by the above P is just composed of one single event transition labeled by e_1 succeeded by another single event transition labeled by e_2, that is, $\xrightarrow{P}=\xrightarrow{e_1}\xrightarrow{e_2}$.

Similarly to the proof of soundness of AIP modulo step bisimulation equivalence (1), we can prove that AIP in Table 19 is sound modulo pomset bisimulation equivalence, we omit them.

(3) If $\pi_n(x) \leftrightarrows_{hp} \pi_n(y)$ for $n \in \mathbb{N}$, then $x \leftrightarrows_{hp} y$.

Since hp-bisimulation $\leftrightarrows_{hp}$ is both an equivalent and a congruent relation with respect to $APTC$ with guarded recursion and projection operator, we only need to check if AIP in Table 19 is sound modulo hp-bisimulation equivalence.

From the definition of hp-bisimulation (see Definition 2.21), we know that hp-bisimulation is defined on the posetal product $(C_1, f, C_2), f : C_1 \to C_2$ isomorphism. Two process terms s related to C_1 and t related to C_2, and $f : C_1 \to C_2$ isomorphism. Initially, $(C_1, f, C_2) = (\varnothing, \varnothing, \varnothing)$, and $(\varnothing, \varnothing, \varnothing) \in \leftrightarrows_{hp}$. When $s \xrightarrow{e} s'$ ($C_1 \xrightarrow{e} C_1'$), there will be $t \xrightarrow{e} t'$ ($C_2 \xrightarrow{e} C_2'$), and we define $f' = f[e \mapsto e]$. Then, if $(C_1, f, C_2) \in \leftrightarrows_{hp}$, then $(C_1', f', C_2') \in \leftrightarrows_{hp}$.

Similarly to the proof of soundness of AIP modulo pomset bisimulation equivalence (2), we can prove that AIP in Table 19 is sound modulo hp-bisimulation equivalence, we just need additionally to check the above conditions on hp-bisimulation, we omit them. □

Theorem 5.15 (Completeness of AIP). Let p and q be closed $APTC$ with linear recursion and projection operator terms, then,

1. if $p \leftrightarrows_s q$ then $\pi_n(p) = \pi_n(q)$;
2. if $p \leftrightarrows_p q$ then $\pi_n(p) = \pi_n(q)$;
3. if $p \leftrightarrows_{hp} q$ then $\pi_n(p) = \pi_n(q)$.

Proof. Firstly, by the elimination theorem of $APTC$ with guarded recursion and projection operator (see Theorem 5.12), we know that each process term in $APTC$ with linear recursion and projection operator is equal to a process term $\langle X_1 | E \rangle$ with E a linear recursive specification:

$$X_i = (a_{i11} \parallel \cdots \parallel a_{i1i_1})X_{i1} + \cdots + (a_{ik_i1} \parallel \cdots \parallel a_{ik_ii_k})X_{ik_i} + (b_{i11} \parallel \cdots \parallel b_{i1i_1}) + \cdots + (b_{il_i1} \parallel \cdots \parallel b_{il_ii_l})$$

for $i \in \{1, \cdots, n\}$.

It remains to prove the following cases.

(1) if $p \leftrightarrows_s q$ then $\pi_n(p) = \pi_n(q)$.

Let $p \leftrightarrows_s q$, and fix an $n \in \mathbb{N}$, there are p', q' in basic $APTC$ terms such that $p' = \pi_n(p)$ and $q' = \pi_n(q)$. Since $\leftrightarrows_s$ is a congruence with respect to $APTC$, if $p \leftrightarrows_s q$ then $\pi_n(p) \leftrightarrows_s \pi_n(q)$. The soundness theorem yields $p' \leftrightarrows_s \pi_n(p) \leftrightarrows_s \pi_n(q) \leftrightarrows_s q'$. Finally, the completeness of $APTC$ modulo $\leftrightarrows_s$ (see Theorem 4.17) ensures $p' = q'$, and $\pi_n(p) = p' = q' = \pi_n(q)$, as desired.

(2) if $p \leftrightarrows_p q$ then $\pi_n(p) = \pi_n(q)$.

Let $p \leftrightarrows_p q$, and fix an $n \in \mathbb{N}$, there are p', q' in basic $APTC$ terms such that $p' = \pi_n(p)$ and $q' = \pi_n(q)$. Since $\leftrightarrows_p$ is a congruence with respect to $APTC$, if $p \leftrightarrows_p q$ then $\pi_n(p) \leftrightarrows_p \pi_n(q)$. The soundness theorem yields $p' \leftrightarrows_p \pi_n(p) \leftrightarrows_p \pi_n(q) \leftrightarrows_p q'$. Finally, the completeness of $APTC$ modulo $\leftrightarrows_p$ (see Theorem 4.19) ensures $p' = q'$, and $\pi_n(p) = p' = q' = \pi_n(q)$, as desired.

(3) if $p \leftrightarrows_{hp} q$ then $\pi_n(p) = \pi_n(q)$.

Let $p \leftrightarrows_{hp} q$, and fix an $n \in \mathbb{N}$, there are p', q' in basic $APTC$ terms such that $p' = \pi_n(p)$ and $q' = \pi_n(q)$. Since $\leftrightarrows_{hp}$ is a congruence with respect to $APTC$, if $p \leftrightarrows_{hp} q$ then $\pi_n(p) \leftrightarrows_{hp} \pi_n(q)$. The soundness theorem yields $p' \leftrightarrows_{hp} \pi_n(p) \leftrightarrows_{hp} \pi_n(q) \leftrightarrows_{hp} q'$. Finally, the completeness of $APTC$ modulo $\leftrightarrows_{hp}$ (see Theorem 4.21) ensures $p' = q'$, and $\pi_n(p) = p' = q' = \pi_n(q)$, as desired. □

6. Abstraction

To abstract away from the internal implementations of a program, and verify that the program exhibits the desired external behaviors, the silent step τ and abstraction operator τ_I are introduced, where $I \subseteq \mathbb{E}$ denotes the internal events. The silent step τ represents the internal events, when we consider the external behaviors of a process, τ events can be removed, that is, τ events must keep silent. The transition rule of τ is shown in Table 20. In the following, let the atomic event e range over $\mathbb{E} \cup \{\delta\} \cup \{\tau\}$, and let the communication function $\gamma : \mathbb{E} \cup \{\tau\} \times \mathbb{E} \cup \{\tau\} \to \mathbb{E} \cup \{\delta\}$, with each communication involved τ resulting in δ.

$$\overline{\tau \xrightarrow{\tau} \surd}$$

Table 20. Transition rule of the silent step

The silent step τ was firstly introduced by Milner in his CCS [3], the algebraic laws about τ were introduced in [1], and finally the axiomatization of τ and τ_I formed a part of ACP [4]. Though τ has been discussed in the interleaving bisimulation background, several years ago, we introduced τ into true concurrency, called weakly true concurrency [16], and also designed its logic based on a uniform logic for true concurrency [14] [15].

In this section, we try to find the algebraic laws of τ and τ_I in true concurrency, or, exactly, to what extent the laws of τ and τ_I in interleaving bisimulation fit the situation of true concurrency.

6.1. Rooted Branching Truly Concurrent Bisimulation Equivalence

In section 2.2, we introduce τ into event structure, and also give the concept of weakly true concurrency. In this subsection, we give the concepts of rooted branching truly concurrent bisimulation equivalences, based on these concepts, we can design the axiom system of the silent step τ and the abstraction operator τ_I. Similarly to rooted branching bisimulation equivalence, rooted branching truly concurrent bisimulation equivalences are following.

Definition 6.1 (Branching pomset, step bisimulation). Assume a special termination predicate $\downarrow$, and let $\surd$ represent a state with $\surd\downarrow$. Let $\mathcal{E}_1$, $\mathcal{E}_2$ be PESs. A branching pomset bisimulation is a relation $R \subseteq \mathcal{C}(\mathcal{E}_1) \times \mathcal{C}(\mathcal{E}_2)$, such that:

1. if $(C_1, C_2) \in R$, and $C_1 \xrightarrow{X} C_1'$ then
   - either $X \equiv \tau^*$, and $(C_1', C_2) \in R$;
   - or there is a sequence of (zero or more) τ-transitions $C_2 \xrightarrow{\tau^*} C_2^0$, such that $(C_1, C_2^0) \in R$ and $C_2^0 \xRightarrow{X} C_2'$ with $(C_1', C_2') \in R$;
2. if $(C_1, C_2) \in R$, and $C_2 \xrightarrow{X} C_2'$ then
   - either $X \equiv \tau^*$, and $(C_1, C_2') \in R$;
   - or there is a sequence of (zero or more) τ-transitions $C_1 \xrightarrow{\tau^*} C_1^0$, such that $(C_1^0, C_2) \in R$ and $C_1^0 \xRightarrow{X} C_1'$ with $(C_1', C_2') \in R$;
3. if $(C_1, C_2) \in R$ and $C_1 \downarrow$, then there is a sequence of (zero or more) τ-transitions $C_2 \xrightarrow{\tau^*} C_2^0$ such that $(C_1, C_2^0) \in R$ and $C_2^0 \downarrow$;
4. if $(C_1, C_2) \in R$ and $C_2 \downarrow$, then there is a sequence of (zero or more) τ-transitions $C_1 \xrightarrow{\tau^*} C_1^0$ such that $(C_1^0, C_2) \in R$ and $C_1^0 \downarrow$.

We say that $\mathcal{E}_1$, $\mathcal{E}_2$ are branching pomset bisimilar, written $\mathcal{E}_1 \approx_{bp} \mathcal{E}_2$, if there exists a branching pomset bisimulation R, such that $(\varnothing, \varnothing) \in R$.

By replacing pomset transitions with steps, we can get the definition of branching step bisimulation. When PESs $\mathcal{E}_1$ and $\mathcal{E}_2$ are branching step bisimilar, we write $\mathcal{E}_1 \approx_{bs} \mathcal{E}_2$.

Definition 6.2 (Rooted branching pomset, step bisimulation). Assume a special termination predicate $\downarrow$, and let $\surd$ represent a state with $\surd\downarrow$. Let $\mathcal{E}_1$, $\mathcal{E}_2$ be PESs. A branching pomset bisimulation is a relation $R \subseteq \mathcal{C}(\mathcal{E}_1) \times \mathcal{C}(\mathcal{E}_2)$, such that:

1. if $(C_1, C_2) \in R$, and $C_1 \xrightarrow{X} C_1'$ then $C_2 \xrightarrow{X} C_2'$ with $C_1' \approx_{bp} C_2'$;
2. if $(C_1, C_2) \in R$, and $C_2 \xrightarrow{X} C_2'$ then $C_1 \xrightarrow{X} C_1'$ with $C_1' \approx_{bp} C_2'$;
3. if $(C_1, C_2) \in R$ and $C_1 \downarrow$, then $C_2 \downarrow$;

4. if $(C_1, C_2) \in R$ and $C_2 \downarrow$, then $C_1 \downarrow$.

We say that $\mathcal{E}_1$, $\mathcal{E}_2$ are rooted branching pomset bisimilar, written $\mathcal{E}_1 \approx_{rbp} \mathcal{E}_2$, if there exists a rooted branching pomset bisimulation R, such that $(\varnothing, \varnothing) \in R$.

By replacing pomset transitions with steps, we can get the definition of rooted branching step bisimulation. When PESs $\mathcal{E}_1$ and $\mathcal{E}_2$ are rooted branching step bisimilar, we write $\mathcal{E}_1 \approx_{rbs} \mathcal{E}_2$.

Definition 6.3 (Branching (hereditary) history-preserving bisimulation). Assume a special termination predicate $\downarrow$, and let $\surd$ represent a state with $\surd \downarrow$. A branching history-preserving (hp-) bisimulation is a weakly posetal relation $R \subseteq \mathcal{C}(\mathcal{E}_1)\overline{\times}\mathcal{C}(\mathcal{E}_2)$ such that:

1. if $(C_1, f, C_2) \in R$, and $C_1 \xrightarrow{e_1} C_1'$ then
   - either $e_1 \equiv \tau$, and $(C_1', f[e_1 \mapsto \tau], C_2) \in R$;
   - or there is a sequence of (zero or more) τ-transitions $C_2 \xrightarrow{\tau^*} C_2^0$, such that $(C_1, f, C_2^0) \in R$ and $C_2^0 \xrightarrow{e_2} C_2'$ with $(C_1', f[e_1 \mapsto e_2], C_2') \in R$;
2. if $(C_1, f, C_2) \in R$, and $C_2 \xrightarrow{e_2} C_2'$ then
   - either $X \equiv \tau$, and $(C_1, f[e_2 \mapsto \tau], C_2') \in R$;
   - or there is a sequence of (zero or more) τ-transitions $C_1 \xrightarrow{\tau^*} C_1^0$, such that $(C_1^0, f, C_2) \in R$ and $C_1^0 \xrightarrow{e_1} C_1'$ with $(C_1', f[e_2 \mapsto e_1], C_2') \in R$;
3. if $(C_1, f, C_2) \in R$ and $C_1 \downarrow$, then there is a sequence of (zero or more) τ-transitions $C_2 \xrightarrow{\tau^*} C_2^0$ such that $(C_1, f, C_2^0) \in R$ and $C_2^0 \downarrow$;
4. if $(C_1, f, C_2) \in R$ and $C_2 \downarrow$, then there is a sequence of (zero or more) τ-transitions $C_1 \xrightarrow{\tau^*} C_1^0$ such that $(C_1^0, f, C_2) \in R$ and $C_1^0 \downarrow$.

$\mathcal{E}_1, \mathcal{E}_2$ are branching history-preserving (hp-)bisimilar and are written $\mathcal{E}_1 \approx_{bhp} \mathcal{E}_2$ if there exists a branching hp-bisimulation R such that $(\varnothing, \varnothing, \varnothing) \in R$.

A branching hereditary history-preserving (hhp-)bisimulation is a downward closed branching hhp-bisimulation. $\mathcal{E}_1, \mathcal{E}_2$ are branching hereditary history-preserving (hhp-)bisimilar and are written $\mathcal{E}_1 \approx_{bhhp} \mathcal{E}_2$.

Definition 6.4 (Rooted branching (hereditary) history-preserving bisimulation). Assume a special termination predicate $\downarrow$, and let $\surd$ represent a state with $\surd \downarrow$. A rooted branching history-preserving (hp-) bisimulation is a weakly posetal relation $R \subseteq \mathcal{C}(\mathcal{E}_1)\overline{\times}\mathcal{C}(\mathcal{E}_2)$ such that:

1. if $(C_1, f, C_2) \in R$, and $C_1 \xrightarrow{e_1} C_1'$, then $C_2 \xrightarrow{e_2} C_2'$ with $C_1' \approx_{bhp} C_2'$;
2. if $(C_1, f, C_2) \in R$, and $C_2 \xrightarrow{e_2} C_1'$, then $C_1 \xrightarrow{e_1} C_2'$ with $C_1' \approx_{bhp} C_2'$;
3. if $(C_1, f, C_2) \in R$ and $C_1 \downarrow$, then $C_2 \downarrow$;
4. if $(C_1, f, C_2) \in R$ and $C_2 \downarrow$, then $C_1 \downarrow$.

$\mathcal{E}_1, \mathcal{E}_2$ are rooted branching history-preserving (hp-)bisimilar and are written $\mathcal{E}_1 \approx_{rbhp} \mathcal{E}_2$ if there exists rooted a branching hp-bisimulation R such that $(\varnothing, \varnothing, \varnothing) \in R$.

A rooted branching hereditary history-preserving (hhp-)bisimulation is a downward closed rooted branching hhp-bisimulation. $\mathcal{E}_1, \mathcal{E}_2$ are rooted branching hereditary history-preserving (hhp-)bisimilar and are written $\mathcal{E}_1 \approx_{rbhhp} \mathcal{E}_2$.

6.2. Guarded Linear Recursion

The silent step τ as an atomic event, is introduced into E. Considering the recursive specification $X = \tau X$, τs, $\tau\tau s$, and $\tau \cdots s$ are all its solutions, that is, the solutions make the existence of τ-loops which cause unfairness. To prevent τ-loops, we extend the definition of linear recursive specification (Definition 5.4) to guarded one.

Definition 6.5 (Guarded linear recursive specification). A recursive specification is linear if its recursive equations are of the form

No.	Axiom
$B1$	$e \cdot \tau = e$
$B2$	$e \cdot (\tau \cdot (x + y) + x) = e \cdot (x + y)$
$B3$	$x \parallel \tau = x$

Table 21. Axioms of silent step

$$(a_{11} \parallel \cdots \parallel a_{1i_1})X_1 + \cdots + (a_{k1} \parallel \cdots \parallel a_{ki_k})X_k + (b_{11} \parallel \cdots \parallel b_{1j_1}) + \cdots + (b_{1j_1} \parallel \cdots \parallel b_{lj_l})$$

where $a_{11}, \cdots, a_{1i_1}, a_{k1}, \cdots, a_{ki_k}, b_{11}, \cdots, b_{1j_1}, b_{1j_1}, \cdots, b_{lj_l} \in \mathbb{E} \cup \{\tau\}$, and the sum above is allowed to be empty, in which case it represents the deadlock δ.

A linear recursive specification E is guarded if there does not exist an infinite sequence of τ-transitions $\langle X|E\rangle \xrightarrow{\tau} \langle X'|E\rangle \xrightarrow{\tau} \langle X''|E\rangle \xrightarrow{\tau} \cdots$.

Theorem 6.6 (Conservitivity of $APTC$ with silent step and guarded linear recursion). $APTC$ with silent step and guarded linear recursion is a conservative extension of $APTC$ with linear recursion.

Proof. Since the transition rules of $APTC$ with linear recursion are source-dependent, and the transition rules for silent step in Table 20 guarded linear recursion contain only a fresh constant τ in their source, so the transition rules of $APTC$ with silent step and guarded linear recursion is a conservative extension of those of $APTC$ with linear recursion. □

Theorem 6.7 (Congruence theorem of $APTC$ with silent step and guarded linear recursion). Rooted branching truly concurrent bisimulation equivalences $\approx_{rbp}$, $\approx_{rbs}$ and $\approx_{rbhp}$ are all congruences with respect to $APTC$ with silent step and guarded linear recursion.

Proof. It follows the following three facts:

1. in a guarded linear recursive specification, right-hand sides of its recursive equations can be adapted to the form by applications of the axioms in $APTC$ and replacing recursion variables by the right-hand sides of their recursive equations;
2. truly concurrent bisimulation equivalences $\sim_p$, $\sim_s$ and $\sim_{hp}$ are all congruences with respect to all operators of $APTC$, while truly concurrent bisimulation equivalences $\sim_p$, $\sim_s$ and $\sim_{hp}$ imply the corresponding rooted branching truly concurrent bisimulation $\approx rbp$, $\approx_{rbs}$ and $\approx_{rbhp}$ (see Proposition 2.23), so rooted branching truly concurrent bisimulation $\approx rbp$, $\approx_{rbs}$ and $\approx_{rbhp}$ are all congruences with respect to all operators of $APTC$;
3. While $\mathbb{E}$ is extended to $\mathbb{E} \cup \{\tau\}$, it can be proved that rooted branching truly concurrent bisimulation $\approx rbp$, $\approx_{rbs}$ and $\approx_{rbhp}$ are all congruences with respect to all operators of $APTC$, we omit it.

□

6.3. Algebraic Laws for the Silent Step

We design the axioms for the silent step τ in Table 21.

The axioms $B1$, $B2$ and $B3$ are the conditions in which τ really keeps silent to act with the operators $\cdot$, $+$ and $\parallel$.

Theorem 6.8 (Elimination theorem of $APTC$ with silent step and guarded linear recursion). Each process term in $APTC$ with silent step and guarded linear recursion is equal to a process term $\langle X_1|E\rangle$ with E a guarded linear recursive specification.

Proof. By applying structural induction with respect to term size, each process term t_1 in $APTC$ with silent step and guarded linear recursion generates a process can be expressed in the form of equations

$$t_i = (a_{i11} \parallel \cdots \parallel a_{i1i_1})t_{i1} + \cdots + (a_{ik_i1} \parallel \cdots \parallel a_{ik_ii_k})t_{ik_i} + (b_{i11} \parallel \cdots \parallel b_{i1i_1}) + \cdots + (b_{il_i1} \parallel \cdots \parallel b_{il_ii_l})$$

for $i \in \{1,\cdots,n\}$. Let the linear recursive specification E consist of the recursive equations

$$X_i = (a_{i11} \parallel \cdots \parallel a_{i1i_1})X_{i1} + \cdots + (a_{ik_i1} \parallel \cdots \parallel a_{ik_ii_k})X_{ik_i} + (b_{i11} \parallel \cdots \parallel b_{i1i_1}) + \cdots + (b_{il_i1} \parallel \cdots \parallel b_{il_ii_l})$$

for $i \in \{1,\cdots,n\}$. Replacing X_i by t_i for $i \in \{1,\cdots,n\}$ is a solution for E, RSP yields $t_1 = \langle X_1|E\rangle$. □

Theorem 6.9 (Soundness of $APTC$ with silent step and guarded linear recursion). Let x and y be $APTC$ with silent step and guarded linear recursion terms. If $APTC$ with silent step and guarded linear recursion $\vdash x = y$, then

1. $x \approx_{rbs} y$;
2. $x \approx_{rbp} y$;
3. $x \approx_{rbhp} y$.

Proof. (1) Soundness of $APTC$ with silent step and guarded linear recursion with respect to rooted branching step bisimulation $\approx_{rbs}$.

Since rooted branching step bisimulation $\approx_{rbs}$ is both an equivalent and a congruent relation with respect to $APTC$ with silent step and guarded linear recursion, we only need to check if each axiom in Table 21 is sound modulo rooted branching step bisimulation equivalence.

Though transition rules in Table 20 are defined in the flavor of single event, they can be modified into a step (a set of events within which each event is pairwise concurrent), we omit them. If we treat a single event as a step containing just one event, the proof of this soundness theorem does not exist any problem, so we use this way and still use the transition rules in Table 20.

- **Axiom** $B1$. Assume that $e \cdot \tau = e$, it is sufficient to prove that $e \cdot \tau \approx_{rbs} e$. By the transition rules for operator $\cdot$ in Table 5 and τ in Table 20, we get

$$\frac{e \xrightarrow{e} \surd}{e \cdot \tau \xrightarrow{e}\xrightarrow{\tau} \surd}$$

$$\frac{e \xrightarrow{e} \surd}{e \xrightarrow{e} \surd}$$

  So, $e \cdot \tau \approx_{rbs} e$, as desired.
- **Axiom** $B2$. Let p and q be $APTC$ with silent step and guarded linear recursion processes, and assume that $e \cdot (\tau \cdot (p + q) + p) = e \cdot (p + q)$, it is sufficient to prove that $e \cdot (\tau \cdot (p + q) + p) \approx_{rbs} e \cdot (p + q)$. By the transition rules for operators $\cdot$ and $+$ in Table 5 and τ in Table 20, we get

$$\frac{e \xrightarrow{e} \surd \quad p \xrightarrow{e_1} \surd}{e \cdot (\tau \cdot (p + q) + p) \xrightarrow{e}\xrightarrow{e_1} \surd}$$

$$\frac{e \xrightarrow{e} \surd \quad p \xrightarrow{e_1} \surd}{e \cdot (p + q) \xrightarrow{e}\xrightarrow{e_1} \surd}$$

$$\frac{e \xrightarrow{e} \surd \quad p \xrightarrow{e_1} p'}{e \cdot (\tau \cdot (p + q) + p) \xrightarrow{e}\xrightarrow{e_1} p'}$$

$$\frac{e \xrightarrow{e} \surd \quad p \xrightarrow{e_1} p'}{e \cdot (p + q) \xrightarrow{e}\xrightarrow{e_1} p'}$$

$$\frac{e \xrightarrow{e} \surd \quad q \xrightarrow{e_2} \surd}{e \cdot (\tau \cdot (p + q) + p) \xrightarrow{e}\xrightarrow{\tau}\xrightarrow{e_2} \surd}$$

$$\frac{e \xrightarrow{e} \surd \quad q \xrightarrow{e_2} \surd}{e \cdot (p+q) \xrightarrow{e} \xrightarrow{e_2} \surd}$$

$$\frac{e \xrightarrow{e} \surd \quad q \xrightarrow{e_2} q'}{e \cdot (\tau \cdot (p+q)+p) \xrightarrow{e} \xrightarrow{\tau} \xrightarrow{e_2} q'}$$

$$\frac{e \xrightarrow{e} \surd \quad q \xrightarrow{e_2} q'}{e \cdot (p+q) \xrightarrow{e} \xrightarrow{e_2} q'}$$

So, $e \cdot (\tau \cdot (p+q)+p) \approx_{rbs} e \cdot (p+q)$, as desired.

- **Axiom** $B3$. Let p be an $APTC$ with silent step and guarded linear recursion process, and assume that $p \parallel \tau = p$, it is sufficient to prove that $p \parallel \tau \approx_{rbs} p$. By the transition rules for operator $\parallel$ in Table 7 and τ in Table 20, we get

$$\frac{p \xrightarrow{e} \surd}{p \parallel \tau \stackrel{e}{\Rightarrow} \surd}$$

$$\frac{p \xrightarrow{e} p'}{p \parallel \tau \stackrel{e}{\Rightarrow} p'}$$

So, $p \parallel \tau \approx_{rbs} p$, as desired.

(2) Soundness of $APTC$ with silent step and guarded linear recursion with respect to rooted branching pomset bisimulation $\approx_{rbp}$.

Since rooted branching pomset bisimulation $\approx_{rbp}$ is both an equivalent and a congruent relation with respect to $APTC$ with silent step and guarded linear recursion, we only need to check if each axiom in Table 21 is sound modulo rooted branching pomset bisimulation $\approx_{rbp}$.

From the definition of rooted branching pomset bisimulation $\approx_{rbp}$ (see Definition 6.2), we know that rooted branching pomset bisimulation $\approx_{rbp}$ is defined by weak pomset transitions, which are labeled by pomsets with τ. In a weak pomset transition, the events in the pomset are either within causality relations (defined by $\cdot$) or in concurrency (implicitly defined by $\cdot$ and $+$, and explicitly defined by $\between$), of course, they are pairwise consistent (without conflicts). In (1), we have already proven the case that all events are pairwise concurrent, so, we only need to prove the case of events in causality. Without loss of generality, we take a pomset of $P = \{e_1, e_2 : e_1 \cdot e_2\}$. Then the weak pomset transition labeled by the above P is just composed of one single event transition labeled by e_1 succeeded by another single event transition labeled by e_2, that is, $\stackrel{P}{\Longrightarrow} = \stackrel{e_1}{\Longrightarrow} \stackrel{e_2}{\Longrightarrow}$.

Similarly to the proof of soundness of $APTC$ with silent step and guarded linear recursion modulo rooted branching step bisimulation $\approx_{rbs}$ (1), we can prove that each axiom in Table 21 is sound modulo rooted branching pomset bisimulation $\approx_{rbp}$, we omit them.

(3) Soundness of $APTC$ with silent step and guarded linear recursion with respect to rooted branching hp-bisimulation $\approx_{rbhp}$.

Since rooted branching hp-bisimulation $\approx_{rbhp}$ is both an equivalent and a congruent relation with respect to $APTC$ with silent step and guarded linear recursion, we only need to check if each axiom in Table 21 is sound modulo rooted branching hp-bisimulation $\approx_{rbhp}$.

From the definition of rooted branching hp-bisimulation $\approx_{rbhp}$ (see Definition 6.4), we know that rooted branching hp-bisimulation $\approx_{rbhp}$ is defined on the weakly posetal product (C_1, f, C_2), $f : \hat{C_1} \to \hat{C_2}$ isomorphism. Two process terms s related to C_1 and t related to C_2, and $f : \hat{C_1} \to \hat{C_2}$ isomorphism. Initially, $(C_1, f, C_2) = (\varnothing, \varnothing, \varnothing)$, and $(\varnothing, \varnothing, \varnothing) \in \approx_{rbhp}$. When $s \xrightarrow{e} s'$ ($C_1 \xrightarrow{e} C_1'$), there will be $t \stackrel{e}{\Rightarrow} t'$ ($C_2 \stackrel{e}{\Rightarrow} C_2'$), and we define $f' = f[e \mapsto e]$. Then, if $(C_1, f, C_2) \in \approx_{rbhp}$, then $(C_1', f', C_2') \in \approx_{rbhp}$.

Similarly to the proof of soundness of $APTC$ with silent step and guarded linear recursion modulo rooted branching pomset bisimulation equivalence (2), we can prove that each axiom in Table 21 is sound modulo

rooted branching hp-bisimulation equivalence, we just need additionally to check the above conditions on rooted branching hp-bisimulation, we omit them. □

Theorem 6.10 (Completeness of $APTC$ with silent step and guarded linear recursion). Let p and q be closed $APTC$ with silent step and guarded linear recursion terms, then,

1. if $p \approx_{rbs} q$ then $p = q$;
2. if $p \approx_{rbp} q$ then $p = q$;
3. if $p \approx_{rbhp} q$ then $p = q$.

Proof. Firstly, by the elimination theorem of $APTC$ with silent step and guarded linear recursion (see Theorem 6.8), we know that each process term in $APTC$ with silent step and guarded linear recursion is equal to a process term $\langle X_1|E\rangle$ with E a guarded linear recursive specification.

It remains to prove the following cases.

(1) If $\langle X_1|E_1\rangle \approx_{rbs} \langle Y_1|E_2\rangle$ for guarded linear recursive specification E_1 and E_2, then $\langle X_1|E_1\rangle = \langle Y_1|E_2\rangle$.

Firstly, the recursive equation $W = \tau + \cdots + \tau$ with $W \notin X_1$ in E_1 and E_2, can be removed, and the corresponding summands aW are replaced by a, to get E_1' and E_2', by use of the axioms RDP, $A3$ and $B1$, and $\langle X|E_1\rangle = \langle X|E_1'\rangle$, $\langle Y|E_2\rangle = \langle Y|E_2'\rangle$.

Let E_1 consists of recursive equations $X = t_X$ for $X \in \mathcal{X}$ and E_2 consists of recursion equations $Y = t_Y$ for $Y \in \mathcal{Y}$, and are not the form $\tau + \cdots + \tau$. Let the guarded linear recursive specification E consists of recursion equations $Z_{XY} = t_{XY}$, and $\langle X|E_1\rangle \approx_{rbs} \langle Y|E_2\rangle$, and t_{XY} consists of the following summands:

1. t_{XY} contains a summand $(a_1 \parallel \cdots \parallel a_m)Z_{X'Y'}$ iff t_X contains the summand $(a_1 \parallel \cdots \parallel a_m)X'$ and t_Y contains the summand $(a_1 \parallel \cdots \parallel a_m)Y'$ such that $\langle X'|E_1\rangle \approx_{rbs} \langle Y'|E_2\rangle$;
2. t_{XY} contains a summand $b_1 \parallel \cdots \parallel b_n$ iff t_X contains the summand $b_1 \parallel \cdots \parallel b_n$ and t_Y contains the summand $b_1 \parallel \cdots \parallel b_n$;
3. t_{XY} contains a summand $\tau Z_{X'Y}$ iff $XY \notin X_1Y_1$, t_X contains the summand $\tau X'$, and $\langle X'|E_1\rangle \approx_{rbs} \langle Y|E_2\rangle$;
4. t_{XY} contains a summand $\tau Z_{XY'}$ iff $XY \notin X_1Y_1$, t_Y contains the summand $\tau Y'$, and $\langle X|E_1\rangle \approx_{rbs} \langle Y'|E_2\rangle$.

Since E_1 and E_2 are guarded, E is guarded. Constructing the process term u_{XY} consist of the following summands:

1. u_{XY} contains a summand $(a_1 \parallel \cdots \parallel a_m)\langle X'|E_1\rangle$ iff t_X contains the summand $(a_1 \parallel \cdots \parallel a_m)X'$ and t_Y contains the summand $(a_1 \parallel \cdots \parallel a_m)Y'$ such that $\langle X'|E_1\rangle \approx_{rbs} \langle Y'|E_2\rangle$;
2. u_{XY} contains a summand $b_1 \parallel \cdots \parallel b_n$ iff t_X contains the summand $b_1 \parallel \cdots \parallel b_n$ and t_Y contains the summand $b_1 \parallel \cdots \parallel b_n$;
3. u_{XY} contains a summand $\tau\langle X'|E_1\rangle$ iff $XY \notin X_1Y_1$, t_X contains the summand $\tau X'$, and $\langle X'|E_1\rangle \approx_{rbs} \langle Y|E_2\rangle$.

Let the process term s_{XY} be defined as follows:

1. $s_{XY} \triangleq \tau\langle X|E_1\rangle + u_{XY}$ iff $XY \notin X_1Y_1$, t_Y contains the summand $\tau Y'$, and $\langle X|E_1\rangle \approx_{rbs} \langle Y'|E_2\rangle$;
2. $s_{XY} \triangleq \langle X|E_1\rangle$, otherwise.

So, $\langle X|E_1\rangle = \langle X|E_1\rangle + u_{XY}$, and $(a_1 \parallel \cdots \parallel a_m)(\tau\langle X|E_1\rangle + u_{XY}) = (a_1 \parallel \cdots \parallel a_m)((\tau\langle X|E_1\rangle + u_{XY}) + u_{XY}) = (a_1 \parallel \cdots \parallel a_m)(\langle X|E_1\rangle + u_{XY}) = (a_1 \parallel \cdots \parallel a_m)\langle X|E_1\rangle$, hence, $(a_1 \parallel \cdots \parallel a_m)s_{XY} = (a_1 \parallel \cdots \parallel a_m)\langle X|E_1\rangle$.

Let σ map recursion variable X in E_1 to $\langle X|E_1\rangle$, and let ψ map recursion variable Z_{XY} in E to s_{XY}. It is sufficient to prove $s_{XY} = \psi(t_{XY})$ for recursion variables Z_{XY} in E. Either $XY \equiv X_1Y_1$ or $XY \notin X_1Y_1$, we all can get $s_{XY} = \psi(t_{XY})$. So, $s_{XY} = \langle Z_{XY}|E\rangle$ for recursive variables Z_{XY} in E is a solution for E. Then by RSP, particularly, $\langle X_1|E_1\rangle = \langle Z_{X_1Y_1}|E\rangle$. Similarly, we can obtain $\langle Y_1|E_2\rangle = \langle Z_{X_1Y_1}|E\rangle$. Finally, $\langle X_1|E_1\rangle = \langle Z_{X_1Y_1}|E\rangle = \langle Y_1|E_2\rangle$, as desired.

(2) If $\langle X_1|E_1\rangle \approx_{rbp} \langle Y_1|E_2\rangle$ for guarded linear recursive specification E_1 and E_2, then $\langle X_1|E_1\rangle = \langle Y_1|E_2\rangle$.

It can be proven similarly to (1), we omit it.

(3) If $\langle X_1|E_1\rangle \approx_{rbhb} \langle Y_1|E_2\rangle$ for guarded linear recursive specification E_1 and E_2, then $\langle X_1|E_1\rangle = \langle Y_1|E_2\rangle$.

It can be proven similarly to (1), we omit it. □

$$\frac{x \xrightarrow{e} \surd}{\tau_I(x) \xrightarrow{e} \surd} \quad e \notin I \qquad \frac{x \xrightarrow{e} x'}{\tau_I(x) \xrightarrow{e} \tau_I(x')} \quad e \notin I$$

$$\frac{x \xrightarrow{e} \surd}{\tau_I(x) \xrightarrow{\tau} \surd} \quad e \in I \qquad \frac{x \xrightarrow{e} x'}{\tau_I(x) \xrightarrow{\tau} \tau_I(x')} \quad e \in I$$

Table 22. Transition rule of the abstraction operator

6.4. Abstraction

The unary abstraction operator τ_I ($I \subseteq \mathbb{E}$) renames all atomic events in I into τ. $APTC$ with silent step and abstraction operator is called $APTC_\tau$. The transition rules of operator τ_I are shown in Table 22.

Theorem 6.11 (Conservitivity of $APTC_\tau$ with guarded linear recursion). $APTC_\tau$ with guarded linear recursion is a conservative extension of $APTC$ with silent step and guarded linear recursion.

Proof. Since the transition rules of $APTC$ with silent step and guarded linear recursion are source-dependent, and the transition rules for abstraction operator in Table 22 guarded linear recursion contain only a fresh operator τ_I in their source, so the transition rules of $APTC_\tau$ with guarded linear recursion is a conservative extension of those of $APTC$ with silent step and guarded linear recursion. □

Theorem 6.12 (Congruence theorem of $APTC_\tau$ with guarded linear recursion). Rooted branching truly concurrent bisimulation equivalences $\approx_{rbp}$, $\approx_{rbs}$ and $\approx_{rbhp}$ are all congruences with respect to $APTC_\tau$ with guarded linear recursion.

Proof. (1) Case rooted branching pomset bisimulation equivalence $\approx_{rbp}$.

Let x and y be $APTC_\tau$ with guarded linear recursion processes, and $x \approx_{rbp} y$, it is sufficient to prove that $\tau_I(x) \approx_{rbp} \tau_I(y)$.

By the transition rules for operator τ_I in Table 22, we can get

$$\tau_I(x) \xrightarrow{X} \surd(X \notin I) \quad \tau_I(y) \xrightarrow{Y} \surd(Y \notin I)$$

with $X \subseteq x$, $Y \subseteq y$, and $X \sim Y$.

Or, we can get

$$\tau_I(x) \xrightarrow{X} \tau_I(x')(X \notin I) \quad \tau_I(y) \xrightarrow{Y} \tau_I(y')(Y \notin I)$$

with $X \subseteq x$, $Y \subseteq y$, and $X \sim Y$ and the hypothesis $\tau_I(x') \approx_{rbp} \tau_I(y')$.

Or, we can get

$$\tau_I(x) \xrightarrow{\tau^*} \surd(X \subseteq I) \quad \tau_I(y) \xrightarrow{\tau^*} \surd(Y \subseteq I)$$

with $X \subseteq x$, $Y \subseteq y$, and $X \sim Y$.

Or, we can get

$$\tau_I(x) \xrightarrow{\tau^*} \tau_I(x')(X \subseteq I) \quad \tau_I(y) \xrightarrow{\tau^*} \tau_I(y')(Y \subseteq I)$$

with $X \subseteq x$, $Y \subseteq y$, and $X \sim Y$ and the hypothesis $\tau_I(x') \approx_{rbp} \tau_I(y')$.

So, we get $\tau_I(x) \approx_{rbp} \tau_I(y)$, as desired

(2) The cases of rooted branching step bisimulation $\approx_{rbs}$, rooted branching hp-bisimulation $\approx_{rbhp}$ can be proven similarly, we omit them. □

We design the axioms for the abstraction operator τ_I in Table 23.

The axioms $TI1-TI3$ are the defining laws for the abstraction operator τ_I; $TI4-TI6$ say that in process term $\tau_I(t)$, all transitions of t labeled with atomic events from I are renamed into τ.

No.	Axiom
$TI1$	$e \notin I \quad \tau_I(e) = e$
$TI2$	$e \in I \quad \tau_I(e) = \tau$
$TI3$	$\tau_I(\delta) = \delta$
$TI4$	$\tau_I(x + y) = \tau_I(x) + \tau_I(y)$
$TI5$	$\tau_I(x \cdot y) = \tau_I(x) \cdot \tau_I(y)$
$TI6$	$\tau_I(x \parallel y) = \tau_I(x) \parallel \tau_I(y)$

Table 23. Axioms of abstraction operator

Theorem 6.13 (Soundness of $APTC_\tau$ with guarded linear recursion). Let x and y be $APTC_\tau$ with guarded linear recursion terms. If $APTC_\tau$ with guarded linear recursion $\vdash x = y$, then

1. $x \approx_{rbs} y$;
2. $x \approx_{rbp} y$;
3. $x \approx_{rbhp} y$.

Proof. (1) Soundness of $APTC_\tau$ with guarded linear recursion with respect to rooted branching step bisimulation $\approx_{rbs}$.

Since rooted branching step bisimulation $\approx_{rbs}$ is both an equivalent and a congruent relation with respect to $APTC_\tau$ with guarded linear recursion, we only need to check if each axiom in Table 23 is sound modulo rooted branching step bisimulation equivalence.

Though transition rules in Table 22 are defined in the flavor of single event, they can be modified into a step (a set of events within which each event is pairwise concurrent), we omit them. If we treat a single event as a step containing just one event, the proof of this soundness theorem does not exist any problem, so we use this way and still use the transition rules in Table 23.

We only prove soundness of the non-trivial axioms $TI4 - TI6$, and omit the defining axioms $TI1 - TI3$.

- **Axiom** $TI4$. Let p, q be $APTC_\tau$ with guarded linear recursion processes, and $\tau_I(p + q) = \tau_I(p) + \tau_I(q)$, it is sufficient to prove that $\tau_I(p + q) \approx_{rbs} \tau_I(p) + \tau_I(q)$. By the transition rules for operator + in Table 5 and τ_I in Table 22, we get

$$\frac{p \xrightarrow{e_1} \surd \quad (e_1 \notin I)}{\tau_I(p + q) \xrightarrow{e_1} \surd} \qquad \frac{p \xrightarrow{e_1} \surd \quad (e_1 \notin I)}{\tau_I(p) + \tau_I(q) \xrightarrow{e_1} \surd}$$

$$\frac{q \xrightarrow{e_2} \surd \quad (e_2 \notin I)}{\tau_I(p + q) \xrightarrow{e_2} \surd} \qquad \frac{q \xrightarrow{e_2} \surd \quad (e_2 \notin I)}{\tau_I(p) + \tau_I(q) \xrightarrow{e_2} \surd}$$

$$\frac{p \xrightarrow{e_1} p' \quad (e_1 \notin I)}{\tau_I(p + q) \xrightarrow{e_1} \tau_I(p')} \qquad \frac{p \xrightarrow{e_1} p' \quad (e_1 \notin I)}{\tau_I(p) + \tau_I(q) \xrightarrow{e_1} \tau_I(p')}$$

$$\frac{q \xrightarrow{e_2} q' \quad (e_2 \notin I)}{\tau_I(p + q) \xrightarrow{e_2} \tau_I(q')} \qquad \frac{q \xrightarrow{e_2} q' \quad (e_2 \notin I)}{\tau_I(p) + \tau_I(q) \xrightarrow{e_2} \tau_I(q')}$$

$$\frac{p \xrightarrow{e_1} \surd \quad (e_1 \in I)}{\tau_I(p + q) \xrightarrow{\tau} \surd} \qquad \frac{p \xrightarrow{e_1} \surd \quad (e_1 \in I)}{\tau_I(p) + \tau_I(q) \xrightarrow{\tau} \surd}$$

$$\frac{q \xrightarrow{e_2} \surd \quad (e_2 \in I)}{\tau_I(p + q) \xrightarrow{\tau} \surd} \qquad \frac{q \xrightarrow{e_2} \surd \quad (e_2 \in I)}{\tau_I(p) + \tau_I(q) \xrightarrow{\tau} \surd}$$

$$\frac{p \xrightarrow{e_1} p' \quad (e_1 \in I)}{\tau_I(p + q) \xrightarrow{\tau} \tau_I(p')} \qquad \frac{p \xrightarrow{e_1} p' \quad (e_1 \in I)}{\tau_I(p) + \tau_I(q) \xrightarrow{\tau} \tau_I(p')}$$

$$\frac{q \xrightarrow{e_2} q' \quad (e_2 \in I)}{\tau_I(p+q) \xrightarrow{\tau} \tau_I(q')} \qquad \frac{q \xrightarrow{e_2} q' \quad (e_2 \in I)}{\tau_I(p)+\tau_I(q) \xrightarrow{\tau} \tau_I(q')}$$

So, $\tau_I(p+q) \approx_{rbs} \tau_I(p)+\tau_I(q)$, as desired.

- **Axiom** $TI5$. Let p,q be $APTC_\tau$ with guarded linear recursion processes, and $\tau_I(p\cdot q)=\tau_I(p)\cdot\tau_I(q)$, it is sufficient to prove that $\tau_I(p\cdot q) \approx_{rbs} \tau_I(p)\cdot\tau_I(q)$. By the transition rules for operator $\cdot$ in Table 5 and τ_I in Table 22, we get

$$\frac{p \xrightarrow{e_1} \surd \quad (e_1 \notin I)}{\tau_I(p\cdot q) \xrightarrow{e_1} \tau_I(q)} \qquad \frac{p \xrightarrow{e_1} \surd \quad (e_1 \notin I)}{\tau_I(p)\cdot\tau_I(q) \xrightarrow{e_1} \tau_I(q)}$$

$$\frac{p \xrightarrow{e_1} p' \quad (e_1 \notin I)}{\tau_I(p\cdot q) \xrightarrow{e_1} \tau_I(p'\cdot q)} \qquad \frac{p \xrightarrow{e_1} p' \quad (e_1 \notin I)}{\tau_I(p)\cdot\tau_I(q) \xrightarrow{e_1} \tau_I(p')\cdot\tau_I(q)}$$

$$\frac{p \xrightarrow{e_1} \surd \quad (e_1 \in I)}{\tau_I(p\cdot q) \xrightarrow{\tau} \tau_I(q)} \qquad \frac{p \xrightarrow{e_1} \surd \quad (e_1 \in I)}{\tau_I(p)\cdot\tau_I(q) \xrightarrow{\tau} \tau_I(q)}$$

$$\frac{p \xrightarrow{e_1} p' \quad (e_1 \in I)}{\tau_I(p\cdot q) \xrightarrow{\tau} \tau_I(p'\cdot q)} \qquad \frac{p \xrightarrow{e_1} p' \quad (e_1 \in I)}{\tau_I(p)\cdot\tau_I(q) \xrightarrow{\tau} \tau_I(p')\cdot\tau_I(q)}$$

So, with the assumption $\tau_I(p'\cdot q)=\tau_I(p')\cdot\tau_I(q)$, $\tau_I(p\cdot q) \approx_{rbs} \tau_I(p)\cdot\tau_I(q)$, as desired.

- **Axiom** $TI6$. Let p,q be $APTC_\tau$ with guarded linear recursion processes, and $\tau_I(p \parallel q)=\tau_I(p) \parallel \tau_I(q)$, it is sufficient to prove that $\tau_I(p \parallel q) \approx_{rbs} \tau_I(p) \parallel \tau_I(q)$. By the transition rules for operator $\parallel$ in Table 7 and τ_I in Table 22, we get

$$\frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} \surd \quad (e_1,e_2 \notin I)}{\tau_I(p \parallel q) \xrightarrow{\{e_1,e_2\}} \surd} \qquad \frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} \surd \quad (e_1,e_2 \notin I)}{\tau_I(p) \parallel \tau_I(q) \xrightarrow{\{e_1,e_2\}} \surd}$$

$$\frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} \surd \quad (e_1,e_2 \notin I)}{\tau_I(p \parallel q) \xrightarrow{\{e_1,e_2\}} \tau_I(p')} \qquad \frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} \surd \quad (e_1,e_2 \notin I)}{\tau_I(p) \parallel \tau_I(q) \xrightarrow{\{e_1,e_2\}} \tau_I(p')}$$

$$\frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} q' \quad (e_1,e_2 \notin I)}{\tau_I(p \parallel q) \xrightarrow{\{e_1,e_2\}} \tau_I(q')} \qquad \frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} q' \quad (e_1,e_2 \notin I)}{\tau_I(p) \parallel \tau_I(q) \xrightarrow{\{e_1,e_2\}} \tau_I(q')}$$

$$\frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} q' \quad (e_1,e_2 \notin I)}{\tau_I(p \parallel q) \xrightarrow{\{e_1,e_2\}} \tau_I(p' \between q')} \qquad \frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} q' \quad (e_1,e_2 \notin I)}{\tau_I(p) \parallel \tau_I(q) \xrightarrow{\{e_1,e_2\}} \tau_I(p') \between \tau_I(q')}$$

$$\frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} \surd \quad (e_1 \notin I, e_2 \in I)}{\tau_I(p \parallel q) \overset{e_1}{\Rightarrow} \surd} \qquad \frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} \surd \quad (e_1 \notin I, e_2 \in I)}{\tau_I(p) \parallel \tau_I(q) \overset{e_1}{\Rightarrow} \surd}$$

$$\frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} \surd \quad (e_1 \notin I, e_2 \in I)}{\tau_I(p \parallel q) \overset{e_1}{\Rightarrow} \tau_I(p')} \qquad \frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} \surd \quad (e_1 \notin I, e_2 \in I)}{\tau_I(p) \parallel \tau_I(q) \overset{e_1}{\Rightarrow} \tau_I(p')}$$

$$\frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} q' \quad (e_1 \notin I, e_2 \in I)}{\tau_I(p \parallel q) \overset{e_1}{\Rightarrow} \tau_I(q')} \qquad \frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} q' \quad (e_1 \notin I, e_2 \in I)}{\tau_I(p) \parallel \tau_I(q) \overset{e_1}{\Rightarrow} \tau_I(q')}$$

$$\frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} q' \quad (e_1 \notin I, e_2 \in I)}{\tau_I(p \parallel q) \overset{e_1}{\Rightarrow} \tau_I(p' \between q')} \qquad \frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} q' \quad (e_1 \notin I, e_2 \in I)}{\tau_I(p) \parallel \tau_I(q) \overset{e_1}{\Rightarrow} \tau_I(p') \between \tau_I(q')}$$

$$\frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} \surd \quad (e_1 \in I, e_2 \notin I)}{\tau_I(p \parallel q) \overset{e_2}{\Rightarrow} \surd} \qquad \frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} \surd \quad (e_1 \in I, e_2 \notin I)}{\tau_I(p) \parallel \tau_I(q) \overset{e_2}{\Rightarrow} \surd}$$

$$\frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} \surd \quad (e_1 \in I, e_2 \notin I)}{\tau_I(p \parallel q) \overset{e_2}{\Rightarrow} \tau_I(p')} \qquad \frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} \surd \quad (e_1 \in I, e_2 \notin I)}{\tau_I(p) \parallel \tau_I(q) \overset{e_2}{\Rightarrow} \tau_I(p')}$$

$$\frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} q' \quad (e_1 \in I, e_2 \notin I)}{\tau_I(p \parallel q) \overset{e_2}{\Rightarrow} \tau_I(q')} \qquad \frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} q' \quad (e_1 \in I, e_2 \notin I)}{\tau_I(p) \parallel \tau_I(q) \overset{e_2}{\Rightarrow} \tau_I(q')}$$

$$\frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} q' \quad (e_1 \in I, e_2 \notin I)}{\tau_I(p \parallel q) \overset{e_2}{\Rightarrow} \tau_I(p') \between \tau_I(q')} \qquad \frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} q' \quad (e_1 \in I, e_2 \notin I)}{\tau_I(p) \parallel \tau_I(q) \overset{e_2}{\Rightarrow} \tau_I(p') \between \tau_I(q')}$$

$$\frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} \surd \quad (e_1, e_2 \in I)}{\tau_I(p \parallel q) \xrightarrow{\tau^*} \surd} \qquad \frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} \surd \quad (e_1, e_2 \in I)}{\tau_I(p) \parallel \tau_I(q) \xrightarrow{\tau^*} \surd}$$

$$\frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} \surd \quad (e_1, e_2 \in I)}{\tau_I(p \parallel q) \xrightarrow{\tau^*} \tau_I(p')} \qquad \frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} \surd \quad (e_1, e_2 \in I)}{\tau_I(p) \parallel \tau_I(q) \xrightarrow{\tau^*} \tau_I(p')}$$

$$\frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} q' \quad (e_1, e_2 \in I)}{\tau_I(p \parallel q) \xrightarrow{\tau^*} \tau_I(q')} \qquad \frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} q' \quad (e_1, e_2 \in I)}{\tau_I(p) \parallel \tau_I(q) \xrightarrow{\tau^*} \tau_I(q')}$$

$$\frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} q' \quad (e_1, e_2 \in I)}{\tau_I(p \parallel q) \xrightarrow{\tau^*} \tau_I(p') \between \tau_I(q')} \qquad \frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} q' \quad (e_1, e_2 \in I)}{\tau_I(p) \parallel \tau_I(q) \xrightarrow{\tau^*} \tau_I(p') \between \tau_I(q')}$$

So, with the assumption $\tau_I(p' \between q') = \tau_I(p') \between \tau_I(q')$, $\tau_I(p \parallel q) \approx_{rbs} \tau_I(p) \parallel \tau_I(q)$, as desired.

(2) Soundness of $APTC_\tau$ with guarded linear recursion with respect to rooted branching pomset bisimulation $\approx_{rbp}$.

Since rooted branching pomset bisimulation $\approx_{rbp}$ is both an equivalent and a congruent relation with respect to $APTC_\tau$ with guarded linear recursion, we only need to check if each axiom in Table 23 is sound modulo rooted branching pomset bisimulation $\approx_{rbp}$.

From the definition of rooted branching pomset bisimulation $\approx_{rbp}$ (see Definition 6.2), we know that rooted branching pomset bisimulation $\approx_{rbp}$ is defined by weak pomset transitions, which are labeled by pomsets with τ. In a weak pomset transition, the events in the pomset are either within causality relations (defined by $\cdot$) or in concurrency (implicitly defined by $\cdot$ and $+$, and explicitly defined by $\between$), of course, they are pairwise consistent (without conflicts). In (1), we have already proven the case that all events are pairwise concurrent, so, we only need to prove the case of events in causality. Without loss of generality, we take a pomset of $P = \{e_1, e_2 : e_1 \cdot e_2\}$. Then the weak pomset transition labeled by the above P is just composed of one single event transition labeled by e_1 succeeded by another single event transition labeled by e_2, that is, $\overset{P}{\Longrightarrow} = \overset{e_1}{\Longrightarrow} \overset{e_2}{\Longrightarrow}$.

Similarly to the proof of soundness of $APTC_\tau$ with guarded linear recursion modulo rooted branching step bisimulation $\approx_{rbs}$ (1), we can prove that each axiom in Table 23 is sound modulo rooted branching pomset bisimulation $\approx_{rbp}$, we omit them.

(3) Soundness of $APTC_\tau$ with guarded linear recursion with respect to rooted branching hp-bisimulation $\approx_{rbhp}$.

Since rooted branching hp-bisimulation $\approx_{rbhp}$ is both an equivalent and a congruent relation with respect to $APTC_\tau$ with guarded linear recursion, we only need to check if each axiom in Table 23 is sound modulo rooted branching hp-bisimulation $\approx_{rbhp}$.

From the definition of rooted branching hp-bisimulation $\approx_{rbhp}$ (see Definition 6.4), we know that rooted branching hp-bisimulation $\approx_{rbhp}$ is defined on the weakly posetal product (C_1, f, C_2), $f : \hat{C}_1 \to \hat{C}_2$ isomorphism. Two process terms s related to C_1 and t related to C_2, and $f : \hat{C}_1 \to \hat{C}_2$ isomorphism. Initially, $(C_1, f, C_2) =$

No.	Axiom
$CFAR$	If X is in a cluster for I with exits $\{(a_{11} \parallel \cdots \parallel a_{1i})Y_1, \cdots, (a_{m1} \parallel \cdots \parallel a_{mi})Y_m, b_{11} \parallel \cdots \parallel b_{1j}, \cdots, b_{n1} \parallel \cdots \parallel b_{nj}\}$, then $\tau \cdot \tau_I(\langle X\|E\rangle) =$ $\tau \cdot \tau_I((a_{11} \parallel \cdots \parallel a_{1i})\langle Y_1\|E\rangle + \cdots + (a_{m1} \parallel \cdots \parallel a_{mi})\langle Y_m\|E\rangle + b_{11} \parallel \cdots \parallel b_{1j} + \cdots + b_{n1} \parallel \cdots \parallel b_{nj})$

Table 24. Cluster fair abstraction rule

$(\varnothing, \varnothing, \varnothing)$, and $(\varnothing, \varnothing, \varnothing) \in \approx_{rbhp}$. When $s \xrightarrow{e} s'$ ($C_1 \xrightarrow{e} C_1'$), there will be $t \xRightarrow{e} t'$ ($C_2 \xRightarrow{e} C_2'$), and we define $f' = f[e \mapsto e]$. Then, if $(C_1, f, C_2) \in \approx_{rbhp}$, then $(C_1', f', C_2') \in \approx_{rbhp}$.

Similarly to the proof of soundness of $APTC_\tau$ with guarded linear recursion modulo rooted branching pomset bisimulation equivalence (2), we can prove that each axiom in Table 23 is sound modulo rooted branching hp-bisimulation equivalence, we just need additionally to check the above conditions on rooted branching hp-bisimulation, we omit them. □

Though τ-loops are prohibited in guarded linear recursive specifications (see Definition 6.5) in specifiable way, they can be constructed using the abstraction operator, for example, there exist τ-loops in the process term $\tau_{\{a\}}(\langle X|X = aX\rangle)$. To avoid τ-loops caused by τ_I and ensure fairness, the concept of cluster and $CFAR$ (Cluster Fair Abstraction Rule) [17] are still valid in true concurrency, we introduce them below.

Definition 6.14 (Cluster). Let E be a guarded linear recursive specification, and $I \subseteq \mathbb{E}$. Two recursion variable X and Y in E are in the same cluster for I iff there exist sequences of transitions $\langle X|E\rangle \xrightarrow{\{b_{11}, \cdots, b_{1i}\}} \cdots \xrightarrow{\{b_{m1}, \cdots, b_{mi}\}} \langle Y|E\rangle$ and $\langle Y|E\rangle \xrightarrow{\{c_{11}, \cdots, c_{1j}\}} \cdots \xrightarrow{\{c_{n1}, \cdots, c_{nj}\}} \langle X|E\rangle$, where $b_{11}, \cdots, b_{mi}, c_{11}, \cdots, c_{nj} \in I \cup \{\tau\}$.

$a_1 \parallel \cdots \parallel a_k$ or $(a_1 \parallel \cdots \parallel a_k)X$ is an exit for the cluster C iff: (1) $a_1 \parallel \cdots \parallel a_k$ or $(a_1 \parallel \cdots \parallel a_k)X$ is a summand at the right-hand side of the recursive equation for a recursion variable in C, and (2) in the case of $(a_1 \parallel \cdots \parallel a_k)X$, either $a_l \notin I \cup \{\tau\} (l \in \{1, 2, \cdots, k\})$ or $X \notin C$.

Theorem 6.15 (Soundness of $CFAR$). $CFAR$ is sound modulo rooted branching truly concurrent bisimulation equivalences $\approx_{rbs}$, $\approx_{rbp}$ and $\approx_{rbhp}$.

Proof. (1) Soundness of $CFAR$ with respect to rooted branching step bisimulation $\approx_{rbs}$.

Let X be in a cluster for I with exits $\{(a_{11} \parallel \cdots \parallel a_{1i})Y_1, \cdots, (a_{m1} \parallel \cdots \parallel a_{mi})Y_m, b_{11} \parallel \cdots \parallel b_{1j}, \cdots, b_{n1} \parallel \cdots \parallel b_{nj}\}$. Then $\langle X|E\rangle$ can execute a string of atomic events from $I \cup \{\tau\}$ inside the cluster of X, followed by an exit $(a_{i'1} \parallel \cdots \parallel a_{i'i})Y_{i'}$ for $i' \in \{1, \cdots, m\}$ or $b_{j'1} \parallel \cdots \parallel b_{j'j}$ for $j' \in \{1, \cdots, n\}$. Hence, $\tau_I(\langle X|E\rangle)$ can execute a string of τ^* inside the cluster of X, followed by an exit $\tau_I((a_{i'1} \parallel \cdots \parallel a_{i'i})\langle Y_{i'}|E\rangle)$ for $i' \in \{1, \cdots, m\}$ or $\tau_I(b_{j'1} \parallel \cdots \parallel b_{j'j})$ for $j' \in \{1, \cdots, n\}$. And these τ^* are non-initial in $\tau\tau_I(\langle X|E\rangle)$, so they are truly silent by the axiom $B1$, we obtain $\tau\tau_I(\langle X|E\rangle) \approx_{rbs} \tau \cdot \tau_I((a_{11} \parallel \cdots \parallel a_{1i})\langle Y_1|E\rangle + \cdots + (a_{m1} \parallel \cdots \parallel a_{mi})\langle Y_m|E\rangle + b_{11} \parallel \cdots \parallel b_{1j} + \cdots + b_{n1} \parallel \cdots \parallel b_{nj})$, as desired.

(2) Soundness of $CFAR$ with respect to rooted branching pomset bisimulation $\approx_{rbp}$.

From the definition of rooted branching pomset bisimulation $\approx_{rbp}$ (see Definition 6.2), we know that rooted branching pomset bisimulation $\approx_{rbp}$ is defined by weak pomset transitions, which are labeled by pomsets with τ. In a weak pomset transition, the events in the pomset are either within causality relations (defined by $\cdot$) or in concurrency (implicitly defined by $\cdot$ and $+$, and explicitly defined by $\between$), of course, they are pairwise consistent (without conflicts). In (1), we have already proven the case that all events are pairwise concurrent, so, we only need to prove the case of events in causality. Without loss of generality, we take a pomset of $P = \{e_1, e_2 : e_1 \cdot e_2\}$. Then the weak pomset transition labeled by the above P is just composed of one single event transition labeled by e_1 succeeded by another single event transition labeled by e_2, that is, $\stackrel{P}{\Longrightarrow} = \stackrel{e_1}{\Longrightarrow}\stackrel{e_2}{\Longrightarrow}$.

Similarly to the proof of soundness of $CFAR$ modulo rooted branching step bisimulation $\approx_{rbs}$ (1), we can prove that $CFAR$ in Table 24 is sound modulo rooted branching pomset bisimulation $\approx_{rbp}$, we omit them.

(3) Soundness of $CFAR$ with respect to rooted branching hp-bisimulation $\approx_{rbhp}$.

From the definition of rooted branching hp-bisimulation $\approx_{rbhp}$ (see Definition 6.4), we know that rooted branching hp-bisimulation $\approx_{rbhp}$ is defined on the weakly posetal product (C_1, f, C_2), $f : \hat{C}_1 \to \hat{C}_2$ isomorphism. Two process terms s related to C_1 and t related to C_2, and $f : \hat{C}_1 \to \hat{C}_2$ isomorphism. Initially, $(C_1, f, C_2) =$

$(\varnothing,\varnothing,\varnothing)$, and $(\varnothing,\varnothing,\varnothing) \in \approx_{rbhp}$. When $s \xrightarrow{e} s'$ ($C_1 \xrightarrow{e} C_1'$), there will be $t \xRightarrow{e} t'$ ($C_2 \xRightarrow{e} C_2'$), and we define $f' = f[e \mapsto e]$. Then, if $(C_1, f, C_2) \in \approx_{rbhp}$, then $(C_1', f', C_2') \in \approx_{rbhp}$.

Similarly to the proof of soundness of $CFAR$ modulo rooted branching pomset bisimulation equivalence (2), we can prove that $CFAR$ in Table 24 is sound modulo rooted branching hp-bisimulation equivalence, we just need additionally to check the above conditions on rooted branching hp-bisimulation, we omit them. □

Theorem 6.16 (Completeness of $APTC_\tau$ with guarded linear recursion and $CFAR$). Let p and q be closed $APTC_\tau$ with guarded linear recursion and $CFAR$ terms, then,

1. if $p \approx_{rbs} q$ then $p = q$;
2. if $p \approx_{rbp} q$ then $p = q$;
3. if $p \approx_{rbhp} q$ then $p = q$.

Proof. (1) For the case of rooted branching step bisimulation, the proof is following.

Firstly, in the proof the Theorem 6.10, we know that each process term p in $APTC$ with silent step and guarded linear recursion is equal to a process term $\langle X_1|E\rangle$ with E a guarded linear recursive specification. And we prove if $\langle X_1|E_1\rangle \approx_{rbs} \langle Y_1|E_2\rangle$, then $\langle X_1|E_1\rangle = \langle Y_1|E_2\rangle$

The only new case is $p \equiv \tau_I(q)$. Let $q = \langle X|E\rangle$ with E a guarded linear recursive specification, so $p = \tau_I(\langle X|E\rangle)$. Then the collection of recursive variables in E can be divided into its clusters $C_1, \cdots, C_N$ for I. Let

$$(a_{1i1} \parallel \cdots \parallel a_{k_{i1}i1})Y_{i1} + \cdots + (a_{1im_i} \parallel \cdots \parallel a_{k_{im_i}im_i})Y_{im_i} + b_{1i1} \parallel \cdots \parallel b_{l_{i1}i1} + \cdots + b_{1im_i} \parallel \cdots \parallel b_{l_{im_i}im_i}$$

be the conflict composition of exits for the cluster C_i, with $i \in \{1, \cdots, N\}$.

For $Z \in C_i$ with $i \in \{1, \cdots, N\}$, we define

$$s_Z \triangleq (a_{\hat{1}i1} \parallel \cdots \parallel a_{\hat{k_{i1}}i1})\tau_I(\langle Y_{i1}|E\rangle) + \cdots + (a_{1\hat{im}_i} \parallel \cdots \parallel a_{\hat{k_{im_i}}im_i})\tau_I(\langle Y_{im_i}|E\rangle) + \hat{b_{1i1}} \parallel \cdots \parallel \hat{b_{l_{i1}i1}} + \cdots + \hat{b_{1im_i}} \parallel \cdots \parallel \hat{b_{l_{im_i}im_i}}$$

For $Z \in C_i$ and $a_1, \cdots, a_j \in \mathbb{E} \cup \{\tau\}$ with $j \in \mathbb{N}$, we have

$(a_1 \parallel \cdots \parallel a_j)\tau_I(\langle Z|E\rangle)$

$= (a_1 \parallel \cdots \parallel a_j)\tau_I((a_{1i1} \parallel \cdots \parallel a_{k_{i1}i1})\langle Y_{i1}|E\rangle + \cdots + (a_{1im_i} \parallel \cdots \parallel a_{k_{im_i}im_i})\langle Y_{im_i}|E\rangle + b_{1i1} \parallel \cdots \parallel b_{l_{i1}i1} + \cdots + b_{1im_i} \parallel \cdots \parallel b_{l_{im_i}im_i})$

$= (a_1 \parallel \cdots \parallel a_j)s_Z$

Let the linear recursive specification F contain the same recursive variables as E, for $Z \in C_i$, F contains the following recursive equation

$$Z = (a_{\hat{1}i1} \parallel \cdots \parallel a_{\hat{k_{i1}}i1})Y_{i1} + \cdots + (a_{1\hat{im}_i} \parallel \cdots \parallel a_{\hat{k_{im_i}}im_i})Y_{im_i} + \hat{b_{1i1}} \parallel \cdots \parallel \hat{b_{l_{i1}i1}} + \cdots + \hat{b_{1im_i}} \parallel \cdots \parallel \hat{b_{l_{im_i}im_i}}$$

It is easy to see that there is no sequence of one or more τ-transitions from $\langle Z|F\rangle$ to itself, so F is guarded.

For

$$s_Z = (a_{\hat{1}i1} \parallel \cdots \parallel a_{\hat{k_{i1}}i1})Y_{i1} + \cdots + (a_{1\hat{im}_i} \parallel \cdots \parallel a_{\hat{k_{im_i}}im_i})Y_{im_i} + \hat{b_{1i1}} \parallel \cdots \parallel \hat{b_{l_{i1}i1}} + \cdots + \hat{b_{1im_i}} \parallel \cdots \parallel \hat{b_{l_{im_i}im_i}}$$

is a solution for F. So, $(a_1 \parallel \cdots \parallel a_j)\tau_I(\langle Z|E\rangle) = (a_1 \parallel \cdots \parallel a_j)s_Z = (a_1 \parallel \cdots \parallel a_j)\langle Z|F\rangle$.

So,

$$\langle Z|F\rangle = (a_{\hat{1}i1} \parallel \cdots \parallel a_{\hat{k_{i1}}i1})\langle Y_{i1}|F\rangle + \cdots + (a_{1\hat{im}_i} \parallel \cdots \parallel a_{\hat{k_{im_i}}im_i})\langle Y_{im_i}|F\rangle + \hat{b_{1i1}} \parallel \cdots \parallel \hat{b_{l_{i1}i1}} + \cdots + \hat{b_{1im_i}} \parallel \cdots \parallel \hat{b_{l_{im_i}im_i}}$$

Hence, $\tau_I(\langle X|E\rangle = \langle Z|F\rangle)$, as desired.

(2) For the case of rooted branching pomset bisimulation, it can be proven similarly to (1), we omit it.

(3) For the case of rooted branching hp-bisimulation, it can be proven similarly to (1), we omit it. □

Finally, in section 4, during conflict elimination, the axioms $U25$ and $U27$ are $(\sharp(e_1, e_2)) \quad e_1 \lhd e_2 = \tau$ and $(\sharp(e_1, e_2), e_2 \leq e_3) \quad e3 \lhd e_1 = \tau$. Their functions are like abstraction operator τ_I, their rigorous soundness can

be proven similarly to Theorem 6.9 and Theorem 6.13, really, they are based on weakly true concurrency. We just illustrate their intuition through an example.

$$\begin{aligned}
P &= \Theta((a\cdot b\cdot c)\parallel(d\cdot e\cdot f)\quad(\sharp(b,e)))\\
&\overset{CE23}{=} (\Theta(a\cdot b\cdot c)\lhd(d\cdot e\cdot f))\parallel(d\cdot e\cdot f)+(\Theta(d\cdot e\cdot f)\lhd(a\cdot b\cdot c))\parallel(a\cdot b\cdot c)\quad(\sharp(b,e))\\
&\overset{CE22}{=} (a\cdot b\cdot c)\lhd(d\cdot e\cdot f))\parallel(d\cdot e\cdot f)+((d\cdot e\cdot f)\lhd(a\cdot b\cdot c))\parallel(a\cdot b\cdot c)\quad(\sharp(b,e))\\
&\overset{U31,U35}{=} (a\cdot\tau\cdot\tau)\parallel(d\cdot e\cdot f)+(d\cdot\tau\cdot\tau)\parallel(a\cdot b\cdot c)\\
&\overset{B1}{=} a\parallel(d\cdot e\cdot f)+d\parallel(a\cdot b\cdot c)
\end{aligned}$$

We see that the conflict relation $\sharp(b,e)$ is eliminated.

7. Applications

APTC provides a formal framework based on truly concurrent behavioral semantics, which can be used to verify the correctness of system behaviors. In this section, we tend to choose one protocol verified by *ACP* [4] – alternating bit protocol (ABP) [19].

The ABP protocol is used to ensure successful transmission of data through a corrupted channel. This success is based on the assumption that data can be resent an unlimited number of times, which is illustrated in Fig.2, we alter it into the true concurrency situation.

1. Data elements $d_1,d_2,d_3,\cdots$ from a finite set Δ are communicated between a Sender and a Receiver.
2. If the Sender reads a datum from channel A_1, then this datum is sent to the Receiver in parallel through channel A_2.
3. The Sender processes the data in Δ, formes new data, and sends them to the Receiver through channel B.
4. And the Receiver sends the datum into channel C.
5. If channel B is corrupted, the message communicated through B can be turn into an error message $\bot$.
6. Every time the Receiver receives a message via channel B, it sends an acknowledgement to the Sender via channel D, which is also corrupted.
7. Finally, then Sender and the Receiver send out their outputs in parallel through channels C_1 and C_2.

In the truly concurrent ABP, the Sender sends its data to the Receiver; and the Receiver can also send its data to the Sender, for simplicity and without loss of generality, we assume that only the Sender sends its data and the Receiver only receives the data from the Sender. The Sender attaches a bit 0 to data elements d_{2k-1} and a bit 1 to data elements d_{2k}, when they are sent into channel B. When the Receiver reads a datum, it sends back the attached bit via channel D. If the Receiver receives a corrupted message, then it sends back the previous acknowledgement to the Sender.

Then the state transition of the Sender can be described by *APTC* as follows.

$$\begin{aligned}
S_b &= \sum_{d\in\Delta} r_{A_1}(d)\cdot T_{db}\\
T_{db} &= (\sum_{d'\in\Delta}(s_B(d',b)\cdot s_{C_1}(d'))+s_B(\bot))\cdot U_{db}\\
U_{db} &= r_D(b)\cdot S_{1-b}+(r_D(1-b)+r_D(\bot))\cdot T_{db}
\end{aligned}$$

where s_B denotes sending data through channel B, r_D denotes receiving data through channel D, similarly, r_{A_1} means receiving data via channel A_1, s_{C_1} denotes sending data via channel C_1, and $b\in\{0,1\}$.

And the state transition of the Receiver can be described by *APTC* as follows.

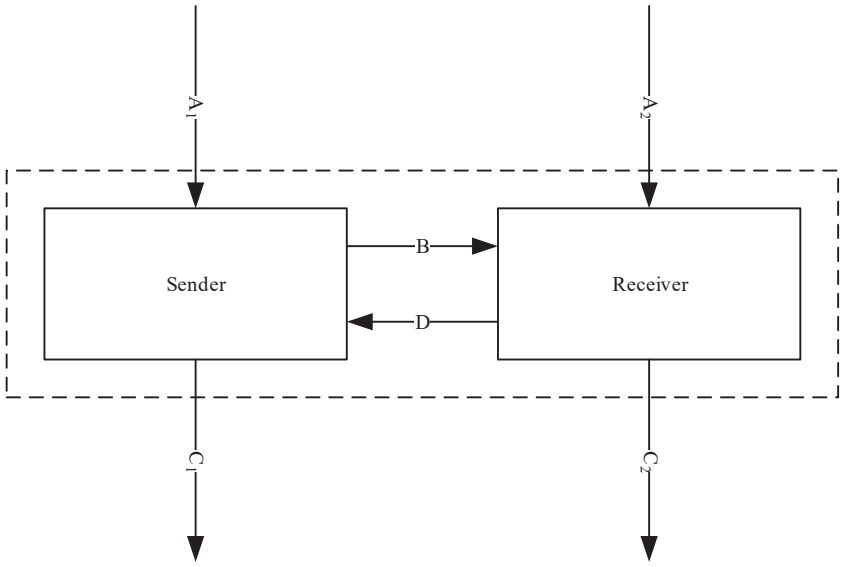

Fig. 2. Alternating bit protocol

$$R_b = \sum_{d\in\Delta} r_{A_2}(d)\cdot R'_b$$
$$R'_b = \sum_{d'\in\Delta} \{r_B(d',b)\cdot s_{C_2}(d')\cdot Q_b + r_B(d',1-b)\cdot Q_{1-b}\} + r_B(\bot)\cdot Q_{1-b}$$
$$Q_b = (s_D(b) + s_D(\bot))\cdot R_{1-b}$$

where r_{A_2} denotes receiving data via channel A_2, r_B denotes receiving data via channel B, s_{C_2} denotes sending data via channel C_2, s_D denotes sending data via channel D, and $b \in \{0,1\}$.

The send action and receive action of the same data through the same channel can communicate each other, otherwise, a deadlock δ will be caused. We define the following communication functions.

$$\gamma(s_B(d',b), r_B(d',b)) \triangleq c_B(d',b)$$
$$\gamma(s_B(\bot), r_B(\bot)) \triangleq c_B(\bot)$$
$$\gamma(s_D(b), r_D(b)) \triangleq c_D(b)$$
$$\gamma(s_D(\bot), r_D(\bot)) \triangleq c_D(\bot)$$

Let R_0 and S_0 be in parallel, then the system R_0S_0 can be represented by the following process term.

$$\tau_I(\partial_H(\Theta(R_0 \between S_0))) = \tau_I(\partial_H(R_0 \between S_0))$$

where $H = \{s_B(d',b), r_B(d',b), s_D(b), r_D(b) | d' \in \Delta, b \in \{0,1\}\}$
$\{s_B(\bot), r_B(\bot), s_D(\bot), r_D(\bot)\}$
$I = \{c_B(d',b), c_D(b) | d' \in \Delta, b \in \{0,1\}\} \cup \{c_B(\bot), c_D(\bot)\}$.
Then we get the following conclusion.

Theorem 7.1 (Correctness of the ABP protocol). The ABP protocol $\tau_I(\partial_H(R_0 \between S_0))$ exhibits desired external behaviors.

Proof. By use of the algebraic laws of $APTC$, we have the following expansions.

$$\begin{aligned}
R_0 \between S_0 &\overset{\text{P1}}{=} R_0 \parallel S_0 + R_0 \mid S_0\\
&\overset{\text{RDP}}{=} (\sum_{d\in\Delta} r_{A_2}(d)\cdot R'_0) \parallel (\sum_{d\in\Delta} r_{A_1}(d)T_{d0})\\
&\quad +(\sum_{d\in\Delta} r_{A_2}(d)\cdot R'_0) \mid (\sum_{d\in\Delta} r_{A_1}(d)T_{d0})\\
&\overset{\text{P6,C14}}{=} \sum_{d\in\Delta}(r_{A_2}(d) \parallel r_{A_1}(d))R'_0 \between T_{d0} + \delta\cdot R'_0 \between T_{d0}\\
&\overset{\text{A6,A7}}{=} \sum_{d\in\Delta}(r_{A_2}(d) \parallel r_{A_1}(d))R'_0 \between T_{d0}
\end{aligned}$$

$$\begin{aligned}
\partial_H(R_0 \between S_0) &= \partial_H(\sum_{d\in\Delta}(r_{A_2}(d) \parallel r_{A_1}(d))R'_0 \between T_{d0})\\
&= \sum_{d\in\Delta}(r_{A_2}(d) \parallel r_{A_1}(d))\partial_H(R'_0 \between T_{d0})
\end{aligned}$$

Similarly, we can get the following equations.

$$\begin{aligned}
\partial_H(R_0 \between S_0) &= \sum_{d\in\Delta}(r_{A_2}(d) \parallel r_{A_1}(d))\cdot\partial_H(T_{d0} \between R'_0)\\
\partial_H(T_{d0} \between R'_0) &= c_B(d',0)\cdot(s_{C_1}(d') \parallel s_{C_2}(d'))\cdot\partial_H(U_{d0} \between Q_0) + c_B(\bot)\cdot\partial_H(U_{d0} \between Q_1)\\
\partial_H(U_{d0} \between Q_1) &= (c_D(1)+c_D(\bot))\cdot\partial_H(T_{d0} \between R'_0)\\
\partial_H(Q_0 \between U_{d0}) &= c_D(0)\cdot\partial_H(R_1 \between S_1) + c_D(\bot)\cdot\partial_H(R'_1 \between T_{d0})\\
\partial_H(R'_1 \between T_{d0}) &= (c_B(d',0)+c_B(\bot))\cdot\partial_H(Q_0 \between U_{d0})\\
\partial_H(R_1 \between S_1) &= \sum_{d\in\Delta}(r_{A_2}(d) \parallel r_{A_1}(d))\cdot\partial_H(T_{d1} \between R'_1)\\
\partial_H(T_{d1} \between R'_1) &= c_B(d',1)\cdot(s_{C_1}(d') \parallel s_{C_2}(d'))\cdot\partial_H(U_{d1} \between Q_1) + c_B(\bot)\cdot\partial_H(U_{d1} \between Q'_0)\\
\partial_H(U_{d1} \between Q'_0) &= (c_D(0)+c_D(\bot))\cdot\partial_H(T_{d1} \between R'_1)\\
\partial_H(Q_1 \between U_{d1}) &= c_D(1)\cdot\partial_H(R_0 \between S_0) + c_D(\bot)\cdot\partial_H(R'_0 \between T_{d1})\\
\partial_H(R'_0 \between T_{d1}) &= (c_B(d',1)+c_B(\bot))\cdot\partial_H(Q_1 \between U_{d1})
\end{aligned}$$

Let $\partial_H(R_0 \between S_0) = \langle X_1|E\rangle$, where E is the following guarded linear recursion specification:

$$\begin{aligned}
&\{X_1 = \sum_{d\in\Delta}(r_{A_2}(d) \parallel r_{A_1}(d))\cdot X_{2d}, Y_1 = \sum_{d\in\Delta}(r_{A_2}(d) \parallel r_{A_1}(d))\cdot Y_{2d},\\
&X_{2d} = c_B(d',0)\cdot X_{4d} + c_B(\bot)\cdot X_{3d}, Y_{2d} = c_B(d',1)\cdot Y_{4d} + c_B(\bot)\cdot Y_{3d},\\
&X_{3d} = (c_D(1)+c_D(\bot))\cdot X_{2d}, Y_{3d} = (c_D(0)+c_D(\bot))\cdot Y_{2d},\\
&X_{4d} = (s_{C_1}(d') \parallel s_{C_2}(d'))\cdot X_{5d}, Y_{4d} = (s_{C_1}(d') \parallel s_{C_2}(d'))\cdot Y_{5d},\\
&X_{5d} = c_D(0)\cdot Y_1 + c_D(\bot)\cdot X_{6d}, Y_{5d} = c_D(1)\cdot X_1 + c_D(\bot)\cdot Y_{6d},\\
&X_{6d} = (c_B(d,0)+c_B(\bot))\cdot X_{5d}, Y_{6d} = (c_B(d,1)+c_B(\bot))\cdot Y_{5d}\\
&|d,d'\in\Delta\}
\end{aligned}$$

Then we apply abstraction operator τ_I into $\langle X_1|E\rangle$.

$$\frac{x \xrightarrow{e} \surd}{\rho_f(x) \xrightarrow{f(e)} \surd} \quad \frac{x \xrightarrow{e} x'}{\rho_f(x) \xrightarrow{f(e)} \rho_f(x')}$$

Table 25. Transition rule of the renaming operator

$$\begin{aligned}\tau_I(\langle X_1|E\rangle) &= \sum_{d\in\Delta}(r_{A_1}(d) \parallel r_{A_2}(d)) \cdot \tau_I(\langle X_{2d}|E\rangle)\\ &= \sum_{d\in\Delta}(r_{A_1}(d) \parallel r_{A_2}(d)) \cdot \tau_I(\langle X_{4d}|E\rangle)\\ &= \sum_{d,d'\in\Delta}(r_{A_1}(d) \parallel r_{A_2}(d)) \cdot (s_{C_1}(d') \parallel s_{C_2}(d')) \cdot \tau_I(\langle X_{5d}|E\rangle)\\ &= \sum_{d,d'\in\Delta}(r_{A_1}(d) \parallel r_{A_2}(d)) \cdot (s_{C_1}(d') \parallel s_{C_2}(d')) \cdot \tau_I(\langle Y_1|E\rangle)\end{aligned}$$

Similarly, we can get $\tau_I(\langle Y_1|E\rangle) = \sum_{d,d'\in\Delta}(r_{A_1}(d) \parallel r_{A_2}(d)) \cdot (s_{C_1}(d') \parallel s_{C_2}(d')) \cdot \tau_I(\langle X_1|E\rangle)$.

We get $\tau_I(\partial_H(R_0 \between S_0)) = \sum_{d,d'\in\Delta}(r_{A_1}(d) \parallel r_{A_2}(d)) \cdot (s_{C_1}(d') \parallel s_{C_2}(d')) \cdot \tau_I(\partial_H(R_0 \between S_0))$. So, the ABP protocol $\tau_I(\partial_H(R_0 \between S_0))$ exhibits desired external behaviors. □

8. Extensions

$APTC$ also has the modularity as ACP, so, $APTC$ can be extended easily. By introducing new operators or new constants, $APTC$ can have more properties, modularity provides $APTC$ an elegant fashion to express a new property. In this section, we take an example of renaming operator which is used to rename the atomic events and firstly introduced by Milner in his CCS [3].

8.1. Transition Rules of Renaming Operator

Renaming operator $\rho_f(t)$ renames all actions in process term t, and assumes a renaming function $f : \mathbb{E} \cup \{\tau\} \to \mathbb{E} \cup \{\tau\}$ with $f(\tau) \triangleq \tau$, which is expressed by the following two transition rules in Table 25.

Theorem 8.1 (Conservativity of $APTC$ with respect to the renaming operator). $APTC_\tau$ with guarded linear recursion and renaming operator is a conservative extension of $APTC_\tau$ with guarded linear recursion.

Proof. It follows from the following two facts (see Theorem 2.8).

1. The transition rules of $APTC_\tau$ with guarded linear recursion in section 6 are all source-dependent;
2. The sources of the transition rules for the renaming operator contain an occurrence of ρ_f.

□

Theorem 8.2 (Congruence theorem of the renaming operator). Rooted branching truly concurrent bisimulation equivalences $\approx_{rbp}$, $\approx_{rbs}$ and $\approx_{rbhp}$ are all congruences with respect to $APTC_\tau$ with guarded linear recursion and the renaming operator.

Proof. (1) Case rooted branching pomset bisimulation equivalence $\approx_{rbp}$.

Let x and y be $APTC_\tau$ with guarded linear recursion and the renaming operator processes, and $x \approx_{rbp} y$, it is sufficient to prove that $\rho_f(x) \approx_{rbp} \rho_f(y)$.

By the transition rules for operator ρ_f in Table 25, we can get

$$\rho_f(x) \xrightarrow{f(X)} \surd \quad \rho_f(y) \xrightarrow{f(Y)} \surd$$

with $X \subseteq x$, $Y \subseteq y$, and $X \sim Y$.

No.	Axiom
$RN1$	$\rho_f(e) = e$
$RN2$	$\rho_f(\delta) = \delta$
$RN3$	$\rho_f(x + y) = \rho_f(x) + \rho_f(y)$
$RN4$	$\rho_f(x \cdot y) = \rho_f(x) \cdot \rho_f(y)$
$RN5$	$\rho_f(x \parallel y) = \rho_f(x) \parallel \rho_f(y)$

Table 26. Axioms of renaming operator

Or, we can get

$$\rho_f(x) \xrightarrow{f(X)} \rho_f(x') \quad \rho_f(y) \xrightarrow{f(Y)} \rho_f(y')$$

with $X \subseteq x$, $Y \subseteq y$, and $X \sim Y$ and the hypothesis $\rho_f(x') \approx_{rbp} \rho_f(y')$.

So, we get $\rho_f(x) \approx_{rbp} \rho_f(y)$, as desired

(2) The cases of rooted branching step bisimulation $\approx_{rbs}$, rooted branching hp-bisimulation $\approx_{rbhp}$ can be proven similarly, we omit them. □

8.2. Axioms for Renaming Operators

We design the axioms for the renaming operator ρ_f in Table 26.

$RN1 - RN2$ are the defining equations for the renaming operator ρ_f; $RN3 - RN5$ say that in $\rho_f(t)$, the labels of all transitions of t are renamed by means of the mapping f.

Theorem 8.3 (Soundness of the renaming operator). Let x and y be $APTC_\tau$ with guarded linear recursion and the renaming operator terms. If $APTC_\tau$ with guarded linear recursion and the renaming operator $\vdash x = y$, then

1. $x \approx_{rbs} y$;
2. $x \approx_{rbp} y$;
3. $x \approx_{rbhp} y$.

Proof. (1) Soundness of $APTC_\tau$ with guarded linear recursion and the renaming operator with respect to rooted branching step bisimulation $\approx_{rbs}$.

Since rooted branching step bisimulation $\approx_{rbs}$ is both an equivalent and a congruent relation with respect to $APTC_\tau$ with guarded linear recursion and the renaming operator, we only need to check if each axiom in Table 26 is sound modulo rooted branching step bisimulation equivalence.

Though transition rules in Table 25 are defined in the flavor of single event, they can be modified into a step (a set of events within which each event is pairwise concurrent), we omit them. If we treat a single event as a step containing just one event, the proof of this soundness theorem does not exist any problem, so we use this way and still use the transition rules in Table 26.

We only prove soundness of the non-trivial axioms $RN3 - RN5$, and omit the defining axioms $RN1 - RN2$.

- **Axiom** $RN3$. Let p, q be $APTC_\tau$ with guarded linear recursion and the renaming operator processes, and $\rho_f(p + q) = \rho_f(p) + \rho_f(q)$, it is sufficient to prove that $\rho_f(p + q) \approx_{rbs} \rho_f(p) + \rho_f(q)$. By the transition rules for operator + in Table 5 and ρ_f in Table 25, we get

$$\frac{p \xrightarrow{e_1} \surd}{\rho_f(p + q) \xrightarrow{f(e_1)} \surd} \quad \frac{p \xrightarrow{e_1} \surd}{\rho_f(p) + \rho_f(q) \xrightarrow{f(e_1)} \surd}$$

$$\frac{q \xrightarrow{e_2} \surd}{\rho_f(p + q) \xrightarrow{f(e_2)} \surd} \quad \frac{q \xrightarrow{e_2} \surd}{\rho_f(p) + \rho_f(q) \xrightarrow{f(e_2)} \surd}$$

$$\frac{p \xrightarrow{e_1} p'}{\rho_f(p+q) \xrightarrow{f(e_1)} \rho_f(p')} \quad \frac{p \xrightarrow{e_1} p'}{\rho_f(p) + \rho_f(q) \xrightarrow{f(e_1)} \rho_f(p')}$$

$$\frac{q \xrightarrow{e_2} q'}{\rho_f(p+q) \xrightarrow{f(e_2)} \rho_f(q')} \quad \frac{q \xrightarrow{e_2} q'}{\rho_f(p) + \rho_f(q) \xrightarrow{f(e_2)} \rho_f(q')}$$

So, $\rho_f(p+q) \approx_{rbs} \rho_f(p) + \rho_f(q)$, as desired.

- **Axiom** $RN4$. Let p, q be $APTC_\tau$ with guarded linear recursion and the renaming operator processes, and $\rho_f(p \cdot q) = \rho_f(p) \cdot \rho_f(q)$, it is sufficient to prove that $\rho_f(p \cdot q) \approx_{rbs} \rho_f(p) \cdot \rho_f(q)$. By the transition rules for operator $\cdot$ in Table 5 and ρ_f in Table 25, we get

$$\frac{p \xrightarrow{e_1} \surd}{\rho_f(p \cdot q) \xrightarrow{f(e_1)} \rho_f(q)} \quad \frac{p \xrightarrow{e_1} \surd}{\rho_f(p) \cdot \rho_f(q) \xrightarrow{f(e_1)} \rho_f(q)}$$

$$\frac{p \xrightarrow{e_1} p'}{\rho_f(p \cdot q) \xrightarrow{f(e_1)} \rho_f(p' \cdot q)} \quad \frac{p \xrightarrow{e_1} p'}{\rho_f(p) \cdot \rho_f(q) \xrightarrow{f(e_1)} \rho_f(p') \cdot \rho_f(q)}$$

So, with the assumption $\rho_f(p' \cdot q) = \rho_f(p') \cdot \rho_f(q)$, $\rho_f(p \cdot q) \approx_{rbs} \rho_f(p) \cdot \rho_f(q)$, as desired.

- **Axiom** $RN5$. Let p, q be $APTC_\tau$ with guarded linear recursion and the renaming operator processes, and $\rho_f(p \parallel q) = \rho_f(p) \parallel \rho_f(q)$, it is sufficient to prove that $\rho_f(p \parallel q) \approx_{rbs} \rho_f(p) \parallel \rho_f(q)$. By the transition rules for operator $\parallel$ in Table 7 and ρ_f in Table 25, we get

$$\frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} \surd}{\rho_f(p \parallel q) \xrightarrow{\{f(e_1), f(e_2)\}} \surd} \quad \frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} \surd}{\rho_f(p) \parallel \rho_f(q) \xrightarrow{\{f(e_1), f(e_2)\}} \surd}$$

$$\frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} \surd}{\rho_f(p \parallel q) \xrightarrow{\{f(e_1), f(e_2)\}} \rho_f(p')} \quad \frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} \surd}{\rho_f(p) \parallel \rho_f(q) \xrightarrow{\{f(e_1), f(e_2)\}} \rho_f(p')}$$

$$\frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} q'}{\rho_f(p \parallel q) \xrightarrow{\{f(e_1), f(e_2)\}} \rho_f(q')} \quad \frac{p \xrightarrow{e_1} \surd \quad q \xrightarrow{e_2} q'}{\rho_f(p) \parallel \rho_f(q) \xrightarrow{\{f(e_1), f(e_2)\}} \rho_f(q')}$$

$$\frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} q'}{\rho_f(p \parallel q) \xrightarrow{\{f(e_1), f(e_2)\}} \rho_f(p') \between \rho_f(q')} \quad \frac{p \xrightarrow{e_1} p' \quad q \xrightarrow{e_2} q'}{\rho_f(p) \parallel \rho_f(q) \xrightarrow{\{f(e_1), f(e_2)\}} \rho_f(p') \between \rho_f(q')}$$

So, with the assumption $\rho_f(p' \between q') = \rho_f(p') \between \rho_f(q')$, $\rho_f(p \parallel q) \approx_{rbs} \rho_f(p) \parallel \rho_f(q)$, as desired.

(2) Soundness of $APTC_\tau$ with guarded linear recursion and the renaming operator with respect to rooted branching pomset bisimulation $\approx_{rbp}$.

Since rooted branching pomset bisimulation $\approx_{rbp}$ is both an equivalent and a congruent relation with respect to $APTC_\tau$ with guarded linear recursion and the renaming operator, we only need to check if each axiom in Table 26 is sound modulo rooted branching pomset bisimulation $\approx_{rbp}$.

From the definition of rooted branching pomset bisimulation $\approx_{rbp}$ (see Definition 6.2), we know that rooted branching pomset bisimulation $\approx_{rbp}$ is defined by weak pomset transitions, which are labeled by pomsets with τ. In a weak pomset transition, the events in the pomset are either within causality relations (defined by $\cdot$) or in concurrency (implicitly defined by $\cdot$ and $+$, and explicitly defined by $\between$), of course, they are pairwise consistent (without conflicts). In (1), we have already proven the case that all events are pairwise concurrent, so, we only need to prove the case of events in causality. Without loss of generality, we take a pomset of $P = \{e_1, e_2 : e_1 \cdot e_2\}$. Then the weak pomset transition labeled by the above P is just composed of one single event transition labeled by e_1 succeeded by another single event transition labeled by e_2, that is, $\overset{P}{\Longrightarrow} = \overset{e_1}{\Longrightarrow} \overset{e_2}{\Longrightarrow}$.

Similarly to the proof of soundness of $APTC_\tau$ with guarded linear recursion and the renaming operator modulo rooted branching step bisimulation $\approx_{rbs}$ (1), we can prove that each axiom in Table 26 is sound modulo rooted branching pomset bisimulation $\approx_{rbp}$, we omit them.

(3) Soundness of $APTC_\tau$ with guarded linear recursion and the renaming operator with respect to rooted branching hp-bisimulation $\approx_{rbhp}$.

Since rooted branching hp-bisimulation $\approx_{rbhp}$ is both an equivalent and a congruent relation with respect to $APTC_\tau$ with guarded linear recursion and the renaming operator, we only need to check if each axiom in Table 26 is sound modulo rooted branching hp-bisimulation $\approx_{rbhp}$.

From the definition of rooted branching hp-bisimulation $\approx_{rbhp}$ (see Definition 6.4), we know that rooted branching hp-bisimulation $\approx_{rbhp}$ is defined on the weakly posetal product $(C_1, f, C_2), f : \hat{C_1} \to \hat{C_2}$ isomorphism. Two process terms s related to C_1 and t related to C_2, and $f : \hat{C_1} \to \hat{C_2}$ isomorphism. Initially, $(C_1, f, C_2) = (\varnothing, \varnothing, \varnothing)$, and $(\varnothing, \varnothing, \varnothing) \in \approx_{rbhp}$. When $s \xrightarrow{e} s'$ ($C_1 \xrightarrow{e} C_1'$), there will be $t \overset{e}{\Longrightarrow} t'$ ($C_2 \overset{e}{\Longrightarrow} C_2'$), and we define $f' = f[e \mapsto e]$. Then, if $(C_1, f, C_2) \in \approx_{rbhp}$, then $(C_1', f', C_2') \in \approx_{rbhp}$.

Similarly to the proof of soundness of $APTC_\tau$ with guarded linear recursion and the renaming operator modulo rooted branching pomset bisimulation equivalence (2), we can prove that each axiom in Table 26 is sound modulo rooted branching hp-bisimulation equivalence, we just need additionally to check the above conditions on rooted branching hp-bisimulation, we omit them. □

Theorem 8.4 (Completeness of the renaming operator). Let p and q be closed $APTC_\tau$ with guarded linear recursion and $CFAR$ and the renaming operator terms, then,

1. if $p \approx_{rbs} q$ then $p = q$;
2. if $p \approx_{rbp} q$ then $p = q$;
3. if $p \approx_{rbhp} q$ then $p = q$.

Proof. (1) For the case of rooted branching step bisimulation, the proof is following.

Firstly, in the proof the Theorem 6.16, we know that each process term p in $APTC_\tau$ with guarded linear recursion is equal to a process term $\langle X_1|E\rangle$ with E a guarded linear recursive specification. And we prove if $\langle X_1|E_1\rangle \approx_{rbs} \langle Y_1|E_2\rangle$, then $\langle X_1|E_1\rangle = \langle Y_1|E_2\rangle$

Structural induction with respect to process term p can be applied. The only new case (where $RN1-RN5$ are needed) is $p \equiv \rho_f(q)$. First assuming $q = \langle X_1|E\rangle$ with a guarded linear recursive specification E, we prove the case of $p = \rho_f(\langle X_1|E\rangle)$. Let E consist of guarded linear recursive equations

$$X_i = (a_{1i1} \parallel \cdots \parallel a_{k_{i1}i1})X_{i1} + ... + (a_{1im_i} \parallel \cdots \parallel a_{k_{im_i}im_i})X_{im_i} + b_{1i1} \parallel \cdots \parallel b_{l_{i1}i1} + ... + b_{1im_i} \parallel \cdots \parallel b_{l_{im_i}im_i}$$

for $i \in 1, ..., n$. Let F consist of guarded linear recursive equations
$Y_i = (f(a_{1i1}) \parallel \cdots \parallel f(a_{k_{i1}i1}))Y_{i1} + ... + (f(a_{1im_i}) \parallel \cdots \parallel f(a_{k_{im_i}im_i}))Y_{im_i}$
$+f(b_{1i1}) \parallel \cdots \parallel f(b_{l_{i1}i1}) + ... + f(b_{1im_i}) \parallel \cdots \parallel f(b_{l_{im_i}im_i})$
for $i \in 1, ..., n$.

$$\begin{aligned}
&\rho_f(\langle X_i|E\rangle)\\
\overset{\text{RDP}}{=}\ &\rho_f((a_{1i1} \parallel \cdots \parallel a_{k_{i1}i1})X_{i1} + ... + (a_{1im_i} \parallel \cdots \parallel a_{k_{im_i}im_i})X_{im_i} + b_{1i1} \parallel \cdots \parallel b_{l_{i1}i1} + ... + b_{1im_i} \parallel \cdots \parallel b_{l_{im_i}im_i})\\
\overset{\text{RN1-RN5}}{=}\ &(f(a_{1i1}) \parallel \cdots \parallel f(a_{k_{i1}i1}))\rho_f(X_{i1}) + ... + (f(a_{1im_i}) \parallel \cdots \parallel f(a_{k_{im_i}im_i}))\rho_f(X_{im_i})\\
&+f(b_{1i1}) \parallel \cdots \parallel f(b_{l_{i1}i1}) + ... + f(b_{1im_i}) \parallel \cdots \parallel f(b_{l_{im_i}im_i})
\end{aligned}$$

Replacing Y_i by $\rho_f(\langle X_i|E\rangle)$ for $i \in \{1, ..., n\}$ is a solution for F. So by RSP, $\rho_f(\langle X_1|E\rangle) = \langle Y_1|F\rangle$, as desired.

(2) For the case of rooted branching pomset bisimulation, it can be proven similarly to (1), we omit it.

(3) For the case of rooted branching hp-bisimulation, it can be proven similarly to (1), we omit it. □

9. Conclusions

Now, let us conclude this paper. We try to find the algebraic laws for true concurrency, as a uniform logic for true concurrency [14] [15] already existed. There are simple comparisons between Hennessy and Milner (HM) logic and bisimulation equivalence as the uniform logic and truly concurrent bisimulation equivalences, the algebraic laws [1], ACP [4] and bisimulation equivalence, as truly concurrent bisimulation equivalences and *what*, which is still missing.

Following the above idea, we find the algebraic laws for true concurrency, which is called $APTC$, an algebra for true concurrency. Like ACP, $APTC$ also has four modules: $BATC$ (Basic Algebra for True Concurrency), $APTC$ (Algebra for Parallelism in True Concurrency), recursion and abstraction, and we prove the soundness and completeness of their algebraic laws modulo truly concurrent bisimulation equivalences. And we show its applications in verification of behaviors of system in a truly concurrent flavor, and its modularity by extending a new renaming operator into it.

Unlike ACP, in $APTC$, the parallelism is a fundamental computational pattern, and cannot be steadied by other computational patterns. We establish a whole theory which has correspondences to ACP.

But, a theory for hereditary history-preserving (hhp-) bisimulation equivalence is still missing (we only prove the soundness and completeness with respect to $BATC$ modulo hhp-bisimulation equivalence), it is one of our future directions. In future, we also pursue the wide applications of $APTC$ in verifications of the behavioral correctness of concurrent systems.

References

[1] M. Hennessy and R. Milner. Algebraic laws for nondeterminism and concurrency. J. ACM, 1985, 32, 137-161.

[2] R. Milner. Communication and concurrency. Printice Hall, 1989.

[3] R. Milner. A calculus of communicating systems. LNCS 92, Springer, 1980.

[4] W. Fokkink. Introduction to process algebra 2nd ed. Springer-Verlag, 2007.

[5] M. Nielsen, G. D. Plotkin, and G. Winskel. Petri nets, event structures and domains, Part I. Theoret. Comput. Sci. 1981, 13, 85-108.

[6] G. Winskel. Event structures. In Petri Nets: Applications and Relationships to Other Models of Concurrency, Wilfried Brauer, Wolfgang Reisig, and Grzegorz Rozenberg, Eds., Lecture Notes in Computer Science, 1987, vol. 255, Springer, Berlin, 325-392.

[7] G. Winskel and M. Nielsen. Models for concurrency. In Samson Abramsky, Dov M. Gabbay,and Thomas S. E. Maibaum, Eds., Handbook of logic in Computer Science, 1995, vol. 4, Clarendon Press, Oxford, UK.

[8] M. A. Bednarczyk. Hereditary history preserving bisimulations or what is the power of the future perfect in program logics. Tech. Rep. Polish Academy of Sciences. 1991.

[9] S. B. Fröschle and T. T. Hildebrandt. On plain and hereditary history-preserving bisimulation. In Proceedings of MFCS'99, Miroslaw Kutylowski, Leszek Pacholski, and Tomasz Wierzbicki, Eds., Lecture Notes in Computer Science, 1999, vol. 1672, Springer, Berlin, 354-365.

[10] J. Bradfield and C. Stirling. Modal mu-calculi. In Handbook of Modal Logic, Patrick Blackburn, Johan van Benthem, and Franck Wolter, Eds., Elsevier, Amsterdam, The Netherlands, 2006, 721-756.

[11] I. Phillips and I. Ulidowski. Reverse bisimulations on stable configuration structures. In Proceedings of SOS'09, B. Klin and P. Sobociǹski, Eds., Electronic Proceedings in Theoretical Computer Science, 2010, vol. 18. Elsevier, Amsterdam, The Netherlands, 62-76.

[12] I. Phillips and I. Ulidowski. A logic with reverse modalities for history-preserving bisimulations. In Proceedings of EXPRESS'11, Bas Luttik and Frank Valencia, Eds., Electronic Proceedings in Theoretical Computer Science, 2011, vol. 64, Elsevier, Amsterdam, The Netherlands, 104-118.

[13] J. Gutierrez. On bisimulation and model-checking for concurrent systems with partial order semantics. Ph.D. dissertation. LFCS- University of Edinburgh, 2011.

[14] P. Baldan and S. Crafa. A logic for true concurrency. In Proceedings of CONCUR'10, Paul Gastin and François Laroussinie, Eds., Lecture Notes in Computer Science, 2010, vol. 6269, Springer, Berlin, 147-161.

[15] P. Baldan and S. Crafa. A logic for true concurrency. J.ACM, 2014, 61(4): 36 pages.

[16] Y. Wang. Weakly true concurrency and its logic. 2016, Manuscript, arXiv:1606.06422.

[17] F.W. Vaandrager. Verification of two communication protocols by means of process algebra. Report CS-R8608, CWI, Amsterdam, 1986.

[18] R. Glabbeek and U. Goltz. Refinement of actions and equivalence notions for concurrent systems. Acta Inf. 2001, 37, 4/5, 229-327.

[19] K.A. Bartlett, R.A. Scantlebury, and P.T. Wilkinson. A note on reliable full-duplex transmission over half-duplex links. Communications of the ACM, 12(5):260-261, 1969.