A simple completeness proof for the axiomatisations of weak behavioural equivalences

Yuxin Deng
Shanghai Jiaotong University
yuxindeng@sjtu.edu.cn

Abstract

This paper presents a simple but uniform completeness proof for the axiomatisations of five weak behavioural equivalences: branching congruence, η-congruence, delay congruence, quasi-branching congruence, and weak congruence in the basic CCS without recursion. For the first three congruences, our result improves van Glabbeek and Weijland's completeness proof of using graph rewriting by a more direct proof that is fully equational. For quasi-branching congruence, our completeness result is, to the best of our knowledge, new.

1 Introduction

Strong bisimilarity, the widely used notion of equivalence for process algebra [7], provides a definition of equality that can capture similarities between processes without forcing them to be syntactically the same. The idea is to match any transition in one process with a transition, labelled by the same action, in the other process.

An important feature in process algebra is abstraction, which deems some of the actions in a process invisible or silent. Consequently, any consecutive execution of invisible transitions is not observable. It turns out that there exist many possibilities for extending strong bisimilarity with invisible transitions. The first extension is Milner's weak bisimilarity [7], which resembles strong bisimilarity, but allows arbitrary sequences of invisible τ-transitions to be inserted before or after an atomic transition. Van Glabbeek and Weijland have introduced branching bisimilarity [11] as an equivalence on processes that preserves the branching structure of processes. It distinguishes slightly more processes than weak bisimilarity. Situated between the two equivalences are two incomparable bisimilarities: η-bisimilarity and delay bisimilarity [11]. Cherief [3] has characterised quasi-branching bisimilarity (which is slightly coarser than branching bisimilarity) as the coarsest equivalence that is preserved under refinement and finer than η-bisimilarity and delay bisimilarity.

Unlike strong bisimilarity, all the five weak notions of bisimilarity are not closed under the summation operator, but for each of them one can define a corresponding congruence relation in a standard way. In the framework of basic CCS, where processes are built from inaction, prefixing, and summation operators, both strong bisimilarity and weak congruence can be completely axiomatised in an elegant way. The completeness property for strong bisimilarity is easy to prove. For weak congruence (that is called observation congruence in [7]), the completeness proof is also not difficult, thanks to a result attributed to Hennessy in [7], called the *Hennessy Lemma* which says that if $P \approx Q$ then either $\tau.P \simeq Q$ or $P \simeq Q$ or $P \simeq \tau.Q$, for weak bisimilarity $\approx$ and weak congruence $\simeq$. For all the weak notions of congruences mentioned above, except for quasi-branching congruence, complete axiomatisations are proposed in [11]. The completeness proof for branching congruence is achieved by using an advanced *graph rewriting technique* due to Bergstra and Klop [2]. The basic idea is to establish a graph rewriting system on finite process graphs, which is confluent and terminating. Then one proves that: (i) two normal forms of the graph rewriting system are bisimilar iff they are equal (i.e., isomorphic), (ii) every rewriting step in the system preserves bisimilarity, and (iii) every rewriting step corresponds to a proof step of the axiom system in question. The completeness proofs for delay, weak, and η-congruence are then derived from the proof for branching congruence in a uniform way. As far as finite processes are concerned, this technique seems a bit heavy and discouraging to non-experts in process algebra. One may wonder whether it is possible to give a simpler completeness proof which is uniform for all congruences but only involves equational reasoning similar to that in [7].

In this paper we give a positive answer to the above question and we are able to add quasi-branching congruence into the picture; we present a simple but uniform completeness proof that works for all the five weak notions of congruence. Our proof is very simple because it involves direct equational reasoning instead of graph rewriting. Our result is also very uniform in the sense that by mild modifications to the completeness proof for branching congruence we obtain the proofs for other four congruences. We also find a subtle difference between weak behavioural equivalences that are to some extent sensitive to the branching structure of processes (e.g. branching bisimilarity, quasi-branching bisimilarity and η-bisimilarity) and that are insensitive (e.g. delay bisimilarity and weak bisimilarity): the Hennessy Lemma is valid for delay bisimilarity and weak bisimilarity but not for branching bisimilarity, quasi-branching bisimilarity and η-bisimilarity.

Due to the difference, our proof schema for completeness deviates from that given in [7]: instead of using the Hennessy Lemma, we exploit a *Promotion Lemma* which says that if $P \approx Q$ then $\tau.P = \tau.Q$ is provably in some axiom system. The Promotion Lemma is less demanding than Hennessy Lemma and thus valid for all the five bisimilarities, based upon which we achieve a uniform proof.

This paper highlights the power of the Promotion Lemma because it shows another situation where the Hennessy Lemma fails but the Promotion Lemma leads to completeness. Similar phenomenon has already occurred in the π-calculus [6] and probabilistic process algebra [5], but none of the weak equivalences investigated in those papers is sensitive to the branching structure of processes.

2 Weak behavioural equivalences

Following [11], we consider a simple language, the basic CCS. But the result in this paper can be easily generalised (see the discussions in Section 5). Processes are built from inaction ($\mathbf{0}$), prefixing ($\alpha.P$), and summation ($P + Q$). The operational semantics of processes is standard. We write $P \Longrightarrow P'$ if there are processes $P_0, ..., P_n$ with $n \geq 0$ and $P \equiv P_0 \xrightarrow{\tau} P_1 \xrightarrow{\tau} ... \xrightarrow{\tau} P_n \equiv P'$. If in that sequence $n \geq 1$ then we write $P \overset{\tau}{\Longrightarrow} P'$.

We recall the definitions of several weak notions of bisimulation appeared in the literature (see e.g. [7, 11]).

Definition 2.1. *A binary relation $\mathcal{R}$ over processes is a* branching simulation *if $P\mathcal{R}Q$ implies that whenever $P \xrightarrow{\alpha} P'$ then*

(C_τ): either $\alpha = \tau$ and $P'\mathcal{R}Q$

(C_b): or there exist Q', Q'' such that $Q \Longrightarrow Q'' \xrightarrow{\alpha} Q'$ with $P\mathcal{R}Q''$ and $P'\mathcal{R}Q'$.

The relation $\mathcal{R}$ is a branching bisimulation *if both $\mathcal{R}$ and $\mathcal{R}^{-1}$ are branching simulations. Two processes P and Q are* branching bisimilar, *denoted $P \approx_b Q$, if there exists a branching bisimulation relating P and Q.*

There are some variants of the above matching conditions:

(C^s_τ): either $\alpha = \tau$ and there exists Q' such that $Q \Longrightarrow Q'$ with $P\mathcal{R}Q'$ and $P'\mathcal{R}Q'$

(C^q_τ): either $\alpha = \tau$ and there exists Q' such that $Q \Longrightarrow Q'$ with $P'\mathcal{R}Q'$

(C_η): or there exist Q', Q'' such that $Q \Longrightarrow Q'' \xrightarrow{\alpha}\Longrightarrow Q'$ with $P\mathcal{R}Q''$ and $P'\mathcal{R}Q'$

(C_d): or there exists Q' such that $Q \Longrightarrow\xrightarrow{\alpha} Q'$ with $P'\mathcal{R}Q'$

(C_w): or there exists Q' such that $Q \Longrightarrow\xrightarrow{\alpha}\Longrightarrow Q'$ with $P'\mathcal{R}Q'$

Semi-branching bisimilarity *($\approx_s$) is defined in terms of (C_τ^s) and (C_b).*
Quasi-branching bisimilarity *($\approx_q$) is defined in terms of (C_τ^q) and (C_b).*
η-bisimilarity *($\approx_\eta$) is defined in terms of (C_τ) and (C_η).*
Delay bisimilarity *($\approx_d$) is defined in terms of (C_τ) and (C_d).*
Weak bisimilarity *($\approx_w$) is defined in terms of (C_τ) and (C_w).*

It can be checked that all the bisimilarities defined above are indeed equivalence relations. In [11] it is shown that $\approx_s$ coincides with $\approx_b$, the inclusions $\approx_b \subseteq \approx_q \subseteq \approx_\eta \subseteq \approx_w$ and $\approx_b \subseteq \approx_q \subseteq \approx_d \subseteq \approx_w$ are strict, but $\approx_\eta$ and $\approx_d$ are incomparable.

It turns out that all the bisimilarities defined above are not congruences with respect to the operator $+$. The classical counterexample is that $\tau.a \approx_x a$ but $\tau.a + b \not\approx_x a + b$ for $x \in \{b, q, \eta, d, w\}$. A typical way of obtaining congruences from bisimilarities is to require that both processes make essential moves at the first step [7, 11]. For instance, we define quasi-branching congruence as follows.

Definition 2.2. *P and Q are* quasi-branching congruent*, written $P \simeq_q Q$, if*

1. *whenever $P \xrightarrow{\alpha} P'$ then*
   (a) *either $\alpha = \tau$ and there exists Q' such that $Q \overset{\tau}{\Longrightarrow} Q'$ and $P' \approx_q Q'$,*
   (b) *or there exists Q' such that $Q \xrightarrow{\alpha} Q'$ and $P' \approx_q Q'$;*
2. *symmetric to clause 1 by exchanging the roles of P and Q.*

The other four congruences can be defined in a similar way.

The next two lemmas report simple properties that hold for all the five bisimilarities studied in this paper. We shall exploit them to prove Lemma 3.3.

Lemma 2.3. *For $x \in \{b, q, \eta, d, w\}$, if $\tau.P + Q \approx_x P$ then $P + Q \approx_x P$.*

Proof. We make use of "bisimulation up to" techniques to construct appropriate bisimulations. See [4] for a detailed proof. □

Lemma 2.4. *For $x \in \{b, q, \eta, d, w\}$, if $P \approx_x Q$ then one of the three cases holds:*

1. *there exists some P' such that $P \xrightarrow{\tau} P'$ and $P' \approx_x Q$;*
2. *there exists some Q' such that $Q \xrightarrow{\tau} Q'$ and $P \approx_x Q'$;*
3. *$P \simeq_x Q$.*

Proof. See [4]. □

For $\approx_d$ and $\approx_w$, we have the following result, where the part on $\approx_w$ is known as the original Hennessy Lemma in CCS.

Lemma 2.5. *For $x \in \{d, w\}$, $P \approx_x Q$ iff ($\tau.P \simeq_x Q$ or $P \simeq_x Q$ or $P \simeq_x \tau.Q$).*

Proof. For $\approx_w$, a proof is given in [7]. It can be adapted for $\approx_d$ easily. □

Remark 2.6. *The above property does not hold for $\approx_b$, $\approx_q$ and $\approx_\eta$. For a counterexample, consider the two processes $\tau.(a + b) + a$ and $a + b$. Let $x \in \{b, q, \eta\}$, it is true that $\tau.(a + b) + a \approx_x a + b$. However,*

$$\begin{array}{rcll} \tau.(\tau.(a+b)+a) & \not\simeq_x & a+b & \qquad (i) \\ \tau.(a+b)+a & \not\simeq_x & a+b & \qquad (ii) \\ \tau.(a+b)+a & \not\simeq_x & \tau.(a+b) & \qquad (iii) \end{array}$$

In (i) an action b from the right hand side cannot be matched up by any action from the left hand side of the inequality. Similar for (ii). In (iii) an action a from the left hand side cannot be matched up by any action from the right hand side.

3 Axiomatisations

In this section we consider complete axiomatisations of branching congruence, quasi-branching congruence, η-congruence, delay congruence, and weak congruence. For all the axiomatisations, the soundness properties are quite easy to show, thus we omit them. So we focus on the completeness properties and provide a uniform but simple completeness proof that works for the five congruences.

All the axioms that we need are displayed in Figure 1. It is shown in [7] that **S1-4** form a complete axiom system for strong bisimilarity. By adding the three τ-laws **T1-3**, Milner has obtained a complete axiom system for weak congruence. In [11] van Glabbeek and Weijland have established a complete axiomatisation of branching congruence by adding **B** to **S1-4**. Two other congruences $\simeq_\eta$ and $\simeq_d$ are also axiomatised in [11], just by adding {**B,T3**} and **T1-2**, respectively, to **S1-4**. The axiom **T3′** is the special case of **T3** when $\alpha = \tau$, and it is derivable from **T2** and **S4**. We shall show that $\simeq_q$ can be axiomatised by adding {**B,T3′**} to **S1-4**. We use the following abbreviations for the axiom systems of the five congruences.

$$\begin{array}{rcl} \mathcal{A}_b & = & \{\textbf{S1-4},\textbf{B}\} \\ \mathcal{A}_q & = & \{\textbf{S1-4},\textbf{B},\textbf{T3}'\} \\ \mathcal{A}_\eta & = & \{\textbf{S1-4},\textbf{B},\textbf{T3}\} \\ \mathcal{A}_d & = & \{\textbf{S1-4},\textbf{T1-2}\} \\ \mathcal{A}_w & = & \{\textbf{S1-4},\textbf{T1-3}\} \end{array}$$

S1	$P + \mathbf{0}$	$=$	P
S2	$P + Q$	$=$	$Q + P$
S3	$P + (Q + R)$	$=$	$(P + Q) + R$
S4	$P + P$	$=$	P
B	$\alpha.(\tau.(P + Q) + Q)$	$=$	$\alpha.(P + Q)$
T1	$\alpha.\tau.P$	$=$	$\alpha.P$
T2	$\tau.P + P$	$=$	$\tau.P$
T3	$\alpha.(P + \tau.Q) + \alpha.Q$	$=$	$\alpha.(P + \tau.Q)$
T3′	$\tau.(P + \tau.Q) + \tau.Q$	$=$	$\tau.(P + \tau.Q)$

Table 1: All the axioms

Note that **B** implies **T1** and is derivable from **T1-2**. So we can also use **T1** when doing equational reasoning in $\mathcal{A}_b, \mathcal{A}_q, \mathcal{A}_\eta$, and use **B** in $\mathcal{A}_d, \mathcal{A}_w$. We write $\mathcal{A} \vdash P = Q$ if $P = Q$ can be derived from the equations in $\mathcal{A}$.

The following saturation properties, clauses 1 and 4 in particular, are well-known in CCS. Here we also consider two special cases of the transition relation $\Longrightarrow\xrightarrow{\alpha}\Longrightarrow$: $\xrightarrow{\alpha}\Longrightarrow$ and $\Longrightarrow\xrightarrow{\alpha}$, in clauses 2 and 3, respectively.

Lemma 3.1 (Saturation)**.** *1. if $P \overset{\tau}{\Longrightarrow} P'$ then {***S1-4,T3′***} $\vdash P = P + \tau.P'$;*

*2. if $P \xrightarrow{\alpha}\Longrightarrow P'$ then {***S1-4,T3***} $\vdash P = P + \alpha.P'$;*

*3. if $P \Longrightarrow\xrightarrow{\alpha} P'$ then {***S1-4,T2***} $\vdash P = P + \alpha.P'$;*

*4. if $P \Longrightarrow\xrightarrow{\alpha}\Longrightarrow P'$ then {***S1-4,T2-3***} $\vdash P = P + \alpha.P'$.*

Proof. By transition induction. As an example, we show the second clause.

Basis step: $P \xrightarrow{\alpha} P'$. Then the conclusion holds by **S4**.

Induction step: $P \xrightarrow{\alpha}\Longrightarrow P'' \xrightarrow{\tau} P'$. Then we infer

$$
\begin{array}{rlr}
\{\mathbf{S1\text{-}4,T3}\} \vdash P & = P + \alpha.P'' & \text{by induction} \\
& = P + \alpha.(P'' + \tau.P') & \text{by } \mathbf{S4} \\
& = P + \alpha.(P'' + \tau.P') + \alpha.P' & \text{by } \mathbf{T3} \\
& = P + \alpha.P' &
\end{array}
$$

□

As usual we use the notion of *normal form*. P is in normal form if it is of the form $\sum_{i=1}^{n} \alpha_i.P_i$, where each P_i is also in normal form.

Lemma 3.2. *For each P, there is a normal form P' such that $\{$**S1-4**$\} \vdash P = P'$.*

We now introduce the important promotion lemma. It relates operational semantics to equational rewriting. Its proof is achieved by induction on the sizes of processes. We define the size, $size(P)$, of process P as follows.

$$\begin{array}{rcl} size(\mathbf{0}) & = & 0 \\ size(\alpha.P) & = & 1 + size(P) \\ size(P+Q) & = & size(P) + size(Q) \end{array}$$

Lemma 3.3 (Promotion)**.** *1. If $P \approx_b Q$ then $\mathcal{A}_b \vdash \tau.P = \tau.Q$;*

2. If $P \approx_q Q$ then $\mathcal{A}_q \vdash \tau.P = \tau.Q$;

3. If $P \approx_\eta Q$ then $\mathcal{A}_\eta \vdash \tau.P = \tau.Q$;

4. If $P \approx_d Q$ then $\mathcal{A}_d \vdash \tau.P = \tau.Q$;

5. If $P \approx_w Q$ then $\mathcal{A}_w \vdash \tau.P = \tau.Q$.

Proof. By Lemma 3.2, we can assume that P and Q are in normal form.

1. We carry out the proof by induction on $size(P + Q)$.

   Basis step If $size(P + Q) = 0$, then it is straightforward to see that $\mathcal{A}_b \vdash P = Q = \mathbf{0}$, thus $\mathcal{A}_b \vdash \tau.P = \tau.Q$.

   Induction step Suppose $size(P + Q) > 0$. Since $P \approx_b Q$, by Lemma 2.4 we can distinguish three cases.

   (a) There exists some P' such that $P \xrightarrow{\tau} P'$ and $P' \approx_b Q$. To have the strong transition $P \xrightarrow{\tau} P'$, P must be of the form $\tau.P' + R$ for some process R. Since $\tau.P' + R \approx_b Q \approx_b P'$, it follows from Lemma 2.3 that $P' + R \approx_b P' \approx_b Q$. Note that $size(P' + R + Q) < size(\tau.P' + R + Q)$ and $size(P' + Q) < size(\tau.P' + R + Q)$. By induction hypothesis, it can be inferred that

   $$\mathcal{A}_b \vdash \tau.(P' + R) = \tau.Q = \tau.P'. \qquad (1)$$

   So we derive

   $$\begin{array}{rll} \mathcal{A}_b \vdash \tau.P & = \tau.(\tau.P' + R) & \\ & = \tau.(\tau.(P' + R) + R) & \text{by (1)} \\ & = \tau.(P' + R) & \text{by } \mathbf{B} \\ & = \tau.Q & \text{by (1)} \end{array}$$

(b) There exists some Q' such that $Q \xrightarrow{\tau} Q'$ and $P \approx_b Q'$. This case is symmetric to case 1 by exchanging the roles of P and Q.

(c) $P \simeq_b Q$. We aim to prove that each summand of P can be absorbed by Q. Let $\alpha.P'$ be a summand of P, which gives rise to a transition $P \xrightarrow{\alpha} P'$. Correspondingly, there exists some Q' such that $\underline{Q \xrightarrow{\alpha} Q'}$ and $P' \approx_b Q'$. Clearly we can derive

$$\mathcal{A}_b \vdash Q = Q + \alpha.Q' \qquad\qquad (*)$$

by <u>the axiom **S4**</u>. Note that $size(P'+Q') < size(P+Q)$. So by induction hypothesis we obtain

$$\mathcal{A}_b \vdash \tau.P' = \tau.Q'. \tag{2}$$

So we derive

$$\begin{aligned}
\mathcal{A}_b \vdash Q + \alpha.P' &= Q + \alpha.\tau.P' && \text{by } \mathbf{T1}\\
&= Q + \alpha.\tau.Q' && \text{by (2)}\\
&= Q + \alpha.Q' && \text{by } \mathbf{T1}\\
&= Q && \text{by (*)}
\end{aligned}$$

In summary, $\mathcal{A}_b \vdash Q + P = Q$ and symmetrically $\mathcal{A}_b \vdash P + Q = P$. Therefore $\mathcal{A}_b \vdash \tau.P = \tau.(P + Q) = \tau.Q$.

2. The arguments are the same as in the proof of clause 1 except that we change all the notations $\mathcal{A}_b$ and $\approx_b$ into $\mathcal{A}_q$ and $\approx_q$, and replace the two underlined parts with "either $Q \xrightarrow{\alpha} Q'$ or $\alpha = \tau$ and $Q \overset{\tau}{\Longrightarrow} Q'$" and "either the axiom **S4** or Lemma 3.1(1)" respectively.

3. The arguments are the same as in the proof of clause 1 except that we change all the notations $\mathcal{A}_b$ and $\approx_b$ into $\mathcal{A}_\eta$ and $\approx_\eta$, and replace the two underlined parts with $Q \xrightarrow{\alpha}\Longrightarrow Q'$ and Lemma 3.1(2) respectively.

4. The arguments are the same as in the proof of clause 1 except that we change all the notations $\mathcal{A}_b$ and $\approx_b$ into $\mathcal{A}_d$ and $\approx_d$, and replace the two underlined parts with $Q \Longrightarrow\xrightarrow{\alpha} Q'$ and Lemma 3.1(3) respectively.

5. The arguments are the same as in the proof of clause 1 except that we change all the notations $\mathcal{A}_b$ and $\approx_b$ into $\mathcal{A}_w$ and $\approx_w$, and replace the two underlined parts with $Q \Longrightarrow\xrightarrow{\alpha}\Longrightarrow Q'$ and Lemma 3.1(4) respectively.

□

Remark 3.4. *In the induction step of the above proof, we have distinguished three independent cases by Lemma 2.4, and in the first two cases Lemma 2.3 plays an important role. Nevertheless, for behavioural equivalences that are insensitive to the branching structure of processes such as $\approx_d$ and $\approx_w$, the proof of the above lemma can be simplified. For instance, in the case of $\approx_w$, one just needs to consider two possibilities: (i) $P \xrightarrow{\tau} P'$ with $P' \approx_w Q$, (ii) $P \xrightarrow{\alpha} P'$ with $Q \Longrightarrow\xrightarrow{\alpha}\Longrightarrow Q'$ and $P' \approx_w Q'$. In the particular case of (ii), one can prove the property (*) by Lemma 3.1(4). This proof schema is adopted in [6] to show the promotion property of all the behavioural equivalences considered in that paper. It is also adapted to a probabilistic setting in [5] where probabilistic weak bisimilarity is investigated. However, the proof schema does not apply to weak behavioural equivalences that are to some extent sensitive to the branching structure of processes, such as $\approx_b$, $\approx_q$ and $\approx_\eta$. The reason is that for these equivalences the τ-transitions before the α-transition in (ii) cannot be simply absorbed. Otherwise, the branching structure of processes would not be observable.*

With the saturation and promotion properties we are now ready to establish the following completeness theorem.

Theorem 3.5 (Completeness).

1. *If $P \simeq_b Q$ then $\mathcal{A}_b \vdash P = Q$;*
2. *If $P \simeq_q Q$ then $\mathcal{A}_q \vdash P = Q$;*
3. *If $P \simeq_\eta Q$ then $\mathcal{A}_\eta \vdash P = Q$;*
4. *If $P \simeq_d Q$ then $\mathcal{A}_d \vdash P = Q$;*
5. *If $P \simeq_w Q$ then $\mathcal{A}_w \vdash P = Q$.*

Proof. 1. Similar to the proof Lemma 3.3(1) we assume P, Q in normal form and proceed by induction on $size(P+Q)$. The basis step is trivial, so we only consider the induction step. Let $\alpha.P'$ be a summand of P. Then $P \xrightarrow{\alpha} P'$ must be matched up by $\underline{Q \xrightarrow{\alpha} Q'}$ for some Q' such that $P' \approx_b Q'$. Clearly we can derive

$$\mathcal{A}_b \vdash Q = Q + \alpha.Q' \qquad (**)$$

by <u>the axiom **S4**</u>. By Promotion Lemma,

$$\mathcal{A}_b \vdash \tau.P' = \tau.Q'. \qquad (3)$$

Therefore

$$\begin{aligned}\mathcal{A}_b \vdash Q + \alpha.P' &= Q + \alpha.Q' && \text{by } \mathbf{T1} \text{ and (3)}\\ &= Q && \text{by (**)}\end{aligned}$$

Hence we have $\mathcal{A}_b \vdash Q+P = Q$. Symmetrically $\mathcal{A}_b \vdash P+Q = P$. Therefore $\mathcal{A}_b \vdash P = Q$.

2. The arguments are the same as in the proof of clause 1 except that we change all the notations $\mathcal{A}_b$ and $\approx_b$ into $\mathcal{A}_q$ and $\approx_q$, and replace the two underlined parts with "either $Q \xrightarrow{\alpha} Q'$ or $\alpha = \tau$ and $Q \overset{\tau}{\Longrightarrow} Q'$" and "either the axiom **S4** or Lemma 3.1(1)" respectively.

3. The arguments are the same as in the proof of clause 1 except that we change all the notations $\mathcal{A}_b$ and $\approx_b$ into $\mathcal{A}_\eta$ and $\approx_\eta$, and replace the two underlined parts with $Q \xrightarrow{\alpha}\Longrightarrow Q'$ and Lemma 3.1(2) respectively.

4. The arguments are the same as in the proof of clause 1 except that we change all the notations $\mathcal{A}_b$ and $\approx_b$ into $\mathcal{A}_d$ and $\approx_d$, and replace the two underlined parts with $Q \Longrightarrow\xrightarrow{\alpha} Q'$ and Lemma 3.1(3) respectively.

5. The arguments are the same as in the proof of clause 1 except that we change all the notations $\mathcal{A}_b$ and $\approx_b$ into $\mathcal{A}_w$ and $\approx_w$, and replace the two underlined parts with $Q \Longrightarrow\xrightarrow{\alpha}\Longrightarrow Q'$ and Lemma 3.1(4) respectively.

□

Remark 3.6. *For $x \in \{d, w\}$, there exists a even simpler completeness proof that does not rely on the Promotion Lemma. The reason is that Lemma 2.5 helps to lift $P \approx_x Q$ to either $\tau.P \simeq_x Q$ or $P \simeq_x Q$ or $P \simeq_x \tau.Q$, thus allowing the induction hypothesis to apply when proving (3) in the last proof. For instance, this is the method adopted in [7] for showing that $\mathcal{A}_w$ constitutes a complete axiom system for $\simeq_w$. For $x \in \{b, q, \eta\}$, however, this method cannot be used because the Hennessy Lemma fails for them (cf. Remark 2.6).*

4 Related work

Van Glabbeek pointed out to us some related work, notably the paper [1] which uses an approach very close to ours. In that paper, Aceto *et al.* gave complete axiomatisations for branching, delay, weak, and η-congruences in the basic CCS augmented with prefix iteration. A careful comparison between [1] and this paper shows the following similarities and differences.

1. The completeness results in [1] are a bit stronger than ours because they are valid for open terms in the basic CCS with prefix iteration. However, to make our ideas as net as possible, in this paper we only consider closed terms in the basic CCS.

2. The main proof strategy of [1] is similar to that adopted in [11], i.e., one shows a completeness result for branching congruence, from which the

completeness results for delay, weak, and η-congruences are derived. However, since one of our aims is to obtain uniformity, we have adopted the strategy of setting up a general proof schema that gives us a direct completeness proof for all the congruences as well as quasi-branching congruence.

3. In [1] the completeness proof for branching congruence relies on a key result which states that (with a change of notations)

   $$\text{If } P \approx_b Q \text{ then } \mathcal{A}_b \vdash \alpha.P = \alpha.Q \text{ for all } \alpha \in A_\tau. \qquad (\dagger)$$

   This is very similar to our Promotion Lemma for branching bisimilarity (Lemma 3.3(1)), though the statement looks different. Moreover, in view of the axiom **T1**, the two results are essentially equivalent. The difference is that our statement appears more concise, and more importantly we have extended the result to all other congruences, which allows us to give direct (and mutually independent) proofs of completeness for all weak behavioural equivalences. In contrast, the completeness proofs for delay, weak, and η-congruences in [1] are indirect because they depend on the proof for branching congruence.

4. We treat delay, weak, and η-bisimilarity in a way different from [1]. We do saturation by means of some "preprocessing" jobs (Lemma 3.1) before showing the quite general Promotion Lemma, whereas Aceto *et al.* do it by means of some "postprocessing" jobs ([1] Proposition 4.7) that reduce the completeness proofs for delay, weak, and η-congruences to the proof for branching congruence.

5. We show the Promotion Lemma in a very simple way, but we use the property Lemma 2.3 whose proof is not trivial. So in fact we have transferred some complexity in proving the Promotion Lemma to the proof of Lemma 2.3. Nevertheless, [1] does not need a property like Lemma 2.3 in showing ($\dagger$), at the price of a less concise proof.

The notion of "branching bisimulation up to" first appears in [12], which axiomatises branching congruence over processes with finite-state behaviours (so the axiomatisation of branching congruence over finite processes is treated as a special case). The completeness proofs use, in the same style as [8], a unique solution theorem.

We may consider Theorem 3.7 in [11] as the semantic version of our Promotion Lemma. It states that $P \approx_x Q$ iff $\tau.P \simeq_x \tau.Q$ for $x \in \{b, \eta, d, w\}$. It is also remarked in [11] that the Hennessy Lemma holds for delay and weak bisimilarity, but not for branching and η-bisimilarity. Nevertheless, in [11] the usefulness of the above theorem in completeness proofs is not investigated.

5 Concluding remarks

This paper presents a simple but uniform completeness proof for the axiomatisations of branching congruence, quasi-branching congruence, η-congruence, delay congruence, and weak congruence over finite processes. Compared with the completeness proof of [11] that employs a graph rewriting technique, our proof by direct equational reasoning is more accessible to non-experts in process algebra. This paper also shows a setting where the Hennessy Lemma fails but the Promotion Lemma leads to completeness.

For convenience of presentation, we have focused on the basic CCS where processes are built from the operators of inaction, prefixing, and summation. Other operators, for example, restriction, relabelling, and parallel composition, can be incorporated while preserving the proof schema presented in this paper. It is also straightforward to extend our results from closed terms to open terms. However, since our completeness proof proceeds by induction on the size of processes, we are not able to handle recursion, which requires other machinery (e.g., the unique solution theorem investigated in [8]).

For future work, it is interesting to extend our results to the π-calculus [9, 10]. In that setting, one has the freedom to consider late vs. early style of weak behavioural equivalences. For example, we could define a *late branching bisimulation* and an *early branching bisimulation*. Here is a potential candidate:

Definition. A symmetric binary relation $\mathcal{R}$ is a late branching bisimulation if whenever $P\mathcal{R}Q$ and $P \xrightarrow{\alpha} P'$ then

1. either $\alpha = \tau$ and $P'\mathcal{R}Q$,
2. or the following property holds:
   (a) if α is not an input action then some Q', Q'' exist such that $Q \Longrightarrow Q'' \xrightarrow{\alpha} Q'$ with $P\mathcal{R}Q''$ and $P'\mathcal{R}Q'$;
   (b) if $\alpha = a(x)$ then some Q', Q'' exist such that $Q \Longrightarrow Q'' \xrightarrow{\alpha} Q'$ with $P\mathcal{R}Q''$ and $P'\{y/x\}\ \mathcal{R}\ Q'\{y/x\}$ for every name y.

To define early branching bisimulation, we just need to change clause 2(b) in the above definition into the following one:

if $\alpha = a(x)$ then for every name y some Q', Q'' exist such that $Q \Longrightarrow Q'' \xrightarrow{\alpha} Q'$ with $P\mathcal{R}Q''$ and $P'\{y/x\}\ \mathcal{R}\ Q'\{y/x\}$.

It is easy to check that the largest late/early branching bisimulation is an equivalence relation. For the corresponding congruences and their axiomatisations, the results in this paper and [6] would be useful.

Acknowledgements We thank Rob J. van Glabbeek for helpful comments on some related work. We also acknowledge the support of the National Nature Science Foundation of China (60703033).

References

[1] L. Aceto, R. J. van Glabbeek, W. Fokkink, and A. Ingólfsdóttir. Axiomatizing prefix iteration with silent steps. *Information and Computation*, 127(1):26–40, 1996.

[2] J. A. Bergstra and J. W. Klop. A complete inference system for regular processes with silent moves. In *Proceedings of Logic Colloquium 1986*, pages 21–81. North Holland, Amsterdam, 1988.

[3] F. Cherief. *Contributions à la sémantique du parallélisme: Bisimulations pour le raffinement et le vrai parallélisme*. PhD thesis, Institut National Polytechnique de Grenoble, 1992.

[4] Y. Deng. A simple completeness proof for the axiomatisations of weak behavioural equivalences, 2007. Full version of the current paper. Available at `http://basics.sjtu.edu.cn/~yuxin/publications/branch.ps`.

[5] Y. Deng and C. Palamidessi. Axiomatizations for probabilistic finite-state behaviors. *Theoretical Computer Science*, 373(1-2):92–114, 2007.

[6] Y. Fu and Z. Yang. Tau laws for pi calculus. *Theoretical Computer Science*, 308:55–130, 2003.

[7] R. Milner. *Communication and Concurrency*. Prentice-Hall, 1989.

[8] R. Milner. A complete axiomatisation for observational congruence of finite-state behaviours. *Information and Computation*, 81:227–247, 1989.

[9] R. Milner. *Communicating and Mobile Systems: the π-Calculus*. Cambridge University Press, 1999.

[10] D. Sangiorgi and D. Walker. *The π-calculus: a Theory of Mobile Processes*. Cambridge University Press, 2001.

[11] R. J. van Gabbeek and W. P. Weijland. Branching time and abstraction in bisimulation semantics. *Journal of the ACM*, 43(3):555–600, 1996.

[12] R. J. van Glabbeek. A complete axiomatization for branching bisimulation congruence of finite-state behaviours. In *Proceedings of the 18th International Symposium on Mathematical Foundations of Computer Science*, volume 711 of *Lecture Notes in Computer Science*, pages 473–484. Springer, 1993.