A Complete Axiomatization for Branching Bisimulation Congruence of Finite-State Behaviours

R.J. van Glabbeek*
Computer Science Department, Stanford University
Stanford, CA 94305, USA.
`rvg@cs.stanford.edu`

This paper offers a complete inference system for branching bisimulation congruence on a basic sublanguage of CCS for representing regular processes with silent moves. Moreover, complete axiomatizations are provided for the guarded expressions in this language, representing the divergence-free processes, and for the recursion-free expressions, representing the finite processes. Furthermore it is argued that in abstract interleaving semantics (at least for finite processes) branching bisimulation congruence is the finest reasonable congruence possible. The argument is that for closed recursion-free process expressions, in the presence of some standard process algebra operations like partially synchronous parallel composition and relabelling, branching bisimulation congruence is completely axiomatized by the usual axioms for strong congruence together with Milner's first τ-law $a\tau X = aX$.

Contents

1 Introduction

An important class of mathematical models for concurrent systems are the *term models*, in which a *process* or behaviour (of a system) is represented as a congruence class of expressions in a *system description language*. The best known system description language is MILNER's *Calculus of Communicating Sytems* (CCS), and the best known congruence on CCS expressions[1] is *bisimulation congruence* [7]. The choice of bisimulation congruence was originally motivated by a notion of *observability*: "processes are equal iff they are indistinguishable by any experiment based on observation" [7]. However, since the appearance of bisimulation congruence, many alternative notions

*This work was supported by ONR under grant number N00014-92-J-1974.
[1]In this first paragraph I restrict myself to expressions that model behaviours without hidden moves (τ-actions).

of observability, or *testing scenarios*, have been proposed, all leading to different—and invariably coarser—congruences. See VAN GLABBEEK [3] for an overview. What makes bisimulation congruence special among all these alternatives is not so much the underlying notion of observation, but the fact that it is the *finest* reasonable congruence. To be precise, this is the case in *interleaving semantics*, where the "concurrent occurrence of two observable actions is not distinguished from their occurrence in arbitrary sequence" [7]. In non-interleaving semantics one finds finer congruences, but the finest ones are just variations of bisimulation congruence that take *causal dependence* between action occurrences explicitly into account. What makes bisimulation congruence the finest reasonable congruence are two properties:

- Any two bisimilar process expressions have the same internal structure, to be precise the same *branching structure*. As the observable behaviour of processes according to any alternative (interleaving based) testing scenario is completely determined by their branching structure, it follows that other observable congruences must be coarser.
- Finer equivalences than bisimulation congruence (such as *tree equivalence* or *graph isomorphism*) suffer from serious drawbacks such as a higher complexity (to decide the equivalence of finite-state behaviours) and the inequivalence of standard operational and denotational interpretations of CCS-like system description languages.

A crucial tool in practical applications of system description languages like CCS, especially for verification purposes, is an *abstraction* mechanism. Abstraction is usually performed by turning actions that are considered unimportant into the *invisible* action τ. Then, a system that after some activity reaches a state from which only an invisible action is possible, leading to another state, is considered equivalent to an otherwise identical system, that after said activity immediately reaches the other state. Thus the mechanism of abstraction by hiding of irrelevant or unobservable actions needs support from the congruence notion employed.

There are many ways to extend bisimulation congruence to processes with hidden moves. The simplest generalization is *strong (bisimulation) equivalence*, in which τ-actions are treated no different than visible actions. For this reason strong congruence is not *abstract* in the sense stipulated above. Another option is to take the testing scenario underlying bisimulation equivalence as primary, incorporating the unobservable nature of hidden moves. This yields MILNER's notion of *weak (bisimulation) congruence* [7], also called *observation congruence*, in spite of the rather far-going assumptions about the capabilities of observers that need to be made for weak congruence to be truly observable. In VAN GLABBEEK & WEIJLAND [4] another generalization of bisimulation congruence was proposed. *Branching (bisimulation) congruence* is not so much motivated in terms of its testing scenario (although it has one that is arguably only twice as contrived as that of weak bisimulation congruence), but generalizes the property of bisimulation congruence of being the finest reasonable interleaving congruence to an abstract setting. To be precise: it preserves the branching structure of processes (unlike weak congruence) [4], and (at least for finite processes) any finer or incomparable abstract version of bisimulation congruence violates the *expansion theorem* [7], that is characteristic for interleaving semantics.

Besides a substantiation of the last claim, this paper offers a complete axiomatization of branching bisimulation congruence for a sublanguage BCCS$^\omega$ of CCS, only containing operators for action, inaction, choice and recursion. The B stands for *Basic* and ω is a strict upper bound for the number of arguments of the choice and recursion operators. This language represents all and only the *regular processes* or *finite-state behaviours*. Moreover, complete axiomatizations for two sublanguages are given: the language of *recursion-free* BCCS$^\omega$ expressions, representing the *finite* processes, and the language of *guarded* BCCS$^\omega$ expressions, representing the *divergence-free* processes, where a processes is *divergent* if it has a state from which an infinite sequence of hidden moves is possible.

A complete axiomatization for strong congruence on BCCS$^\omega$ was provided in MILNER [5]. It consisted of the axioms E1-4, A0-3 and R1-3 of Section 4 (as well as α-conversion, which is derivable). A complete axiomatization for weak congruence on a slightly different language was first provided in BERGSTRA & KLOP [2]. A more aesthetic axiomatization (on BCCS$^\omega$), partly inspired by the one in [2], was given in MILNER [6]. It consisted of the axioms for strong congruence, 3 so-called τ-laws, and 2 extra axioms for unguarded recursion (besides R3). The present axiomatization counts, besides the axioms for strong congruence, only one τ-law, but 3 extra axioms for unguarded recursion (all weaker than the axioms for weak congruence). In all three cases the axioms for unguarded recursion can be dropped to obtain complete axiomatizations for guarded expressions, and on top of that R1 and R2 can be dropped to obtain complete axiomatizations for finite processes.

Milner's completeness proof was delivered in five steps:

(a) Any expression can be converted into a guarded one.

(b) Any guarded expression provably satisfies a standard guarded set of equations.

(c) Any standard guarded set of equations can be converted into a saturated one (preserving the property of being provably satisfied by an expression).

(d) Two congruent processes that each provably satisfy a saturated standard guarded set of equations, provably satisfy a common guarded set of equations.

(e) If two guarded expressions satisfy the same guarded set of equations, they are provably equal.

Steps (b) and (e) only use the axioms for strong congruence, and can thus be applied in the setting of branching bisimulation as well. Step (a) can be made completely analogous, even though the present axioms for unguarded recursion are much more complicated (in particular, the side-condition of R4 can not be eliminated, as could be done for the corresponding axiom R5 in [6]). Step (c) must be skipped as saturation is unsound in branching bisimulation semantics, and therefore step (d) needs to be made more subtle. But the absence of step (c) makes it possible to incorporate step (b) into step (d) at no extra cost.

The completeness theorem for branching congruence on recursion-free process expressions, at least the closed ones, was already proven in [4] by the method of graph transformations, due to BERGSTRA & KLOP [1]. The present proof is distinctly shorter. On the other hand, the method of graph transformations, once mastered, tends to deliver completeness proofs on finite closed terms for arbitrary interleaving equivalences almost instantaneously, whereas the method used here seems rather bisimulation oriented and requires more thought.

2 A language for finite-state behaviours

Let the nonempty set A of *visible actions* and the disjoint infinite set V of variables be given. Let $\tau \notin A$ be the *invisible action* or *hidden move* and write $A_\tau = A \cup \{\tau\}$.

Definition 1 The set $\mathcal{E}$ of *process expressions* over BCCS$^\omega$ is given by

X	$\in \mathcal{E}$	for $X \in V$	(variable)
0	$\in \mathcal{E}$		(inaction)
aE	$\in \mathcal{E}$	for $a \in A_\tau$ and $E \in \mathcal{E}$	(action)
$E+F$	$\in \mathcal{E}$	for $E, F \in \mathcal{E}$	(choice)
μXE	$\in \mathcal{E}$	for $X \in V_S$ and $E \in \mathcal{E}$	(recursion)

The expression 0 represents a process that is unable to perform any action. aE represents a process that first performs the action a and then proceeds as E. $E+F$ represents a process that will behave as either E or F, and μXE represents a solution of the equation $X = E$.

Definition 2 An occurrence of a variable X in an expression $E \in \mathcal{E}$ is *bound* if it occurs in a subexpression of the form μXF. Otherwise it is *free*. E is *open* if it contains a free occurrence of a variable, and *closed* otherwise. $E\{F/X\}$ denotes the result of substituting F for all free occurrences of X in E, if necessary[2] renaming bound variables in E in order to ensure that no free occurrence of a variable in F becomes bound in $E\{F/X\}$. Likewise $E\{E_X/X\}_{X\in V'}$, for $V' \subseteq V$, denotes the result of simultaneously substituting E_X for X in the same fashion.

Definition 3 The transition relation $\longrightarrow \subseteq \mathcal{E} \times (A_\tau \cup V) \times \mathcal{E}$ is the smallest relation satisfying

- $X \xrightarrow{X} 0$ for $X \in V$
- $aE \xrightarrow{a} E$ for $a \in A_\tau$
- if $E \xrightarrow{x} G$ or $F \xrightarrow{x} G$ then $E + F \xrightarrow{x} G$
- if $E\{\mu XE/X\} \xrightarrow{x} F$ then $\mu XE \xrightarrow{x} F$

Here $E \xrightarrow{a} F$ for $a \in A_\tau$ means that the system represented by E can perform the action a, thereby evolving into F, and $E \xrightarrow{X} 0$ means that the system represented by E has the possibility to continue as whatever system is substituted for the variable X.

Definition 4 Let $E \in \mathcal{E}$. The set $\mathcal{E}_E$ of process expressions *reachable* from E is defined as the smallest subset of $\mathcal{E}$ satisfying $E \in \mathcal{E}_E$ and if $F \xrightarrow{a} G$ with $a \in A_\tau$ and $F \in \mathcal{E}_E$ then $G \in \mathcal{E}_E$.

Proposition 1 $\mathcal{E}_E$ is finite for $E \in \mathcal{E}$.

Proof: Consider the transition relation $\rightarrow \subseteq \mathcal{E}' \times (A_\tau \dot\cup \{+, \mu\}) \times \mathcal{E}'$, given by

- $aE \stackrel{a}{\rightarrow} E$
- $E + F \stackrel{+}{\rightarrow} E$ and $E + F \stackrel{+}{\rightarrow} F$
- $\mu XE \stackrel{\mu}{\rightarrow} E\{\mu XE/X\}$

[2] Renaming is necessary if a free occurrence of X appears in a subterm μYG of E with Y occurring free in F.

Here $\mathcal{E}'$ is defined as $\mathcal{E}$, except that every operator symbol $(X, 0, a, +, \mu X)$ in an expression $E \in \mathcal{E}'$ is coloured either red or black. Furthermore, if in a subexpression aF, $F+G$ or μXF of E the leading operator a, $+$ or μX is coloured black, the entire subexpression must be black. Whether an occurrence of a variable is free or bound does not depend on its colour. Substitution on $\mathcal{E}'$ is defined such that $E\{F/X\}$ means $E\{black(F)/X\}$ (i.e. a black version of F is substituted for any free red or black occurrence of X), and renaming of bound variables doesn't change their colour. Furthermore colours are preserved under transitions.

Choose $E \in \mathcal{E}$ and let $\mathcal{E}'_E$ be the set of coloured expressions in $\mathcal{E}'$ that are reachable by $\rightarrow$ from $red(E)$. If $F \xrightarrow{a} F'$ for $F, F' \in \mathcal{E}$, and $F_0 \in \mathcal{E}'$ is a coloured version of F, then there must be $F_1, \ldots, F_{n+1} \in \mathcal{E}'$ with $n \in \mathbb{N}$ such that $F_{i-1} \xrightarrow{+} F_i$ or $F_{i-1} \xrightarrow{\mu} F_i$ for $i = 1, \ldots, n$, $F_n \xrightarrow{a} F_{n+1}$ and F_{n+1} is a coloured version of F'. Thus for any $F \in \mathcal{E}_E$ a coloured version appears in $\mathcal{E}'_E$, and it suffices to proof that $\mathcal{E}'_E$ is finite, or becomes finite after forgetting the colours.

Observe that if an expression F is partly red and $F \rightarrow F'$ then the red part of F' is smaller than the red part of F. Thus there are only finitely many expressions in $\mathcal{E}'_E$ that are partly red.

Furthermore observe that for any $F \in \mathcal{E}'_E$, if F contains a subexpression μYG with μY red, then no black subexpression of G contains a free occurrence of Y. This property is trivially true for $red(E)$, trivially preserved under $\xrightarrow{a}$ and $\xrightarrow{+}$, and preserved under $\xrightarrow{\mu}$ by the renaming-of-bound-variables convention of Definition 2. It follows that if $F \in \mathcal{E}'_E$ is partly red and $F \rightarrow F'$, then the black subexpressions of F that are inherited by F'—unlike the red ones—are unchanged in F'. Thus if $H \in \mathcal{E}'_E$ is partly red, $H \rightarrow H'$ and H' is completely black, then H' has the form μXG and has been generated by a derivation $\mu XG \xrightarrow{\mu} G\{\mu XG/X\}$. Hence the black term $H = \mu XG \in \mathcal{E}'_E$ also occurs as a partly red term $\mu XG \in \mathcal{E}'_E$.

It follows that $\mathcal{E}_E$ is finite. In fact $\mathcal{E}_E$ contains at most one element more than it has subexpressions of the form aF. □

Definition 5 A free occurrence of a variable X in an expression $E \in \mathcal{E}$ is *guarded* if it occurs in a subexpression of the form aF with $a \in A$ (i.e. $a \neq \tau$). X is *(un)guarded* in E if (not) every free occurrence of X in E is guarded. A process expression $E \in \mathcal{E}$ is *guarded* if for every subexpression μXF, X is guarded in F. Let $\mathcal{E}^g \subseteq \mathcal{E}$ be the set of guarded process expressions over BCCS$^\omega$.

Definition 6 A process expression $E \in \mathcal{E}$ is called *finite* or, more accurately, *recursion-free* if it has no subexpression of the form μXF. Let $\mathcal{E}^f \subseteq \mathcal{E}^g \subseteq \mathcal{E}$ be the set of finite process expressions over BCCS$^\omega$.

Lemma 1 If $E \in \mathcal{E}^f$, then the relation $\longrightarrow$ is well-founded in $\mathcal{E}_E$. This means that there are no $F_i \in \mathcal{E}_E$ and $a_i \in A_\tau$ for $i \in \mathbb{N}$ with $F_i \xrightarrow{a_i} F_{i+1}$ for $i \in \mathbb{N}$.

Proof: If $F \in \mathcal{E}^f$ and $F \xrightarrow{a} F'$ then $F' \in \mathcal{E}^f$ and F' is smaller than F. □

Lemma 2 If $E \in \mathcal{E}^g$, then the relation $\xrightarrow{\tau}$ is well-founded in $\mathcal{E}_E$.

Proof: First note that if E is guarded and $F \in \mathcal{E}_E$ then F is guarded. This follows with a straightforward induction on derivations. For $F \in \mathcal{E}$, let F^* be F, in which every occurrence of a subterm aG with $a \in A$ is replaced by 0. Note that if F is guarded then F^* is guarded, and if $F \xrightarrow{\tau} G$ then $F^* \xrightarrow{\tau} G^*$. Now suppose there is an infinite path $F_0 \xrightarrow{\tau} F_1 \xrightarrow{\tau} F_2 \xrightarrow{\tau} \cdots$ as denied in the lemma. Then there must be an infinite path $F_0^* \xrightarrow{\tau} F_1^* \xrightarrow{\tau} F_2^* \xrightarrow{\tau} \cdots$, only passing through guarded process expressions without subexpressions of the form aG for $a \in A$. But if H is such an expression and $H \xrightarrow{\tau} H'$, then H' is smaller than H, yielding a contradiction. □

Write $E \Longrightarrow E'$ if there are $E_0, \ldots, E_n \in \mathcal{E}$ with $E = E_0 \stackrel{\tau}{\longrightarrow} E_1 \stackrel{\tau}{\longrightarrow} \cdots \stackrel{\tau}{\longrightarrow} E_n = E'$.

Lemma 3 $X \in V$ is unguarded in $E \in \mathcal{E}$ iff $E \Longrightarrow E' \stackrel{X}{\longrightarrow} 0$.
Proof: Straightforward. $\square$

Definition 7 Renaming of bound variables is called *α-conversion*. Write $E =_\alpha F$ if $E, F \in \mathcal{E}$ only differ by α-conversion.

Lemma 4 Let $x \in A_\tau \cup V$.

1. $H \stackrel{X}{\longrightarrow} 0 \wedge E \stackrel{x}{\longrightarrow} F \quad \Rightarrow \quad H\{E/X\} \stackrel{x}{\longrightarrow} F$
2. $H \stackrel{x}{\longrightarrow} H' \wedge x \neq X \quad \Rightarrow \quad H\{E/X\} \stackrel{x}{\longrightarrow} H''\{E/X\}$ with $H' =_\alpha H''$
3. $H\{E/X\} \stackrel{x}{\longrightarrow} F \quad \Rightarrow \quad (H \stackrel{X}{\longrightarrow} 0 \wedge E \stackrel{x}{\longrightarrow} F) \vee (x \neq X \wedge H \stackrel{x}{\longrightarrow} H' =_\alpha H'' \wedge F = H''\{E/X\})$

Proof: 1. and 2. are straightforward by induction on inference. I will prove 3. by induction on the inference of $H\{E/X\} \stackrel{x}{\longrightarrow} F$. In case $H = X$ the first alternative applies: $H \stackrel{X}{\longrightarrow} 0 \wedge E \stackrel{x}{\longrightarrow} F$. The cases $F = Y \neq X$, $H = aG$, $H = H_1 + H_2$ and $H = \mu XG$ are straightforward, so assume $H = \mu YG$ with $Y \neq X$. Let $\tilde{H} = \mu\tilde{Y}\tilde{G}$ be the result of renaming bound variables in H, as described in Definition 2. Now, by a shorter inference $\tilde{G}\{E/X\}\{\tilde{H}\{E/X\}/\tilde{Y}\} = \tilde{G}\{\tilde{H}/\tilde{Y}\}\{E/X\} \stackrel{x}{\longrightarrow} F$, so by induction $(\tilde{H} \stackrel{X}{\longrightarrow} 0 \wedge E \stackrel{x}{\longrightarrow} F) \vee (x \neq X \wedge \tilde{H} \stackrel{x}{\longrightarrow} \tilde{H}' =_\alpha H'' \wedge F = H''\{E/X\})$, from which the desired conclusion follows. $\square$

3 Branching bisimulation congruence

Definition 8 A *branching bisimulation* is a symmetric relation $\mathcal{R} \subseteq \mathcal{E} \times \mathcal{E}$ such that $\forall x \in A_\tau \cup V$:

$$\text{if } (E,F) \in \mathcal{R} \wedge E \stackrel{x}{\longrightarrow} E' \text{ then } \begin{array}{l} x = \tau \text{ and } (E', F) \in \mathcal{R} \\ \text{or } \exists F'', F' : F \Longrightarrow F'' \stackrel{x}{\longrightarrow} F' \wedge (E, F'') \in \mathcal{R} \wedge (E', F') \in \mathcal{R}. \end{array}$$

Two expressions E and F are *branching (bisimulation) equivalent*—notation $E \leftrightarrow_b F$—if there exists a branching bisimulation $\mathcal{R}$ with $(E, F) \in \mathcal{R}$.

For further motivation of branching bisimulation equivalence see VAN GLABBEEK & WEIJLAND [4]. The consise definition above is possible thanks to the following lemma.

Lemma 5 If $E \leftrightarrow_b F$, $E \leftrightarrow_b F''$ and $F \Longrightarrow F''$, then $E \leftrightarrow_b F'$ for any F' with $F \Longrightarrow F' \Longrightarrow F''$.
Proof: In [4]. $\square$

It is more common to use Definition 8 for closed process expressions only, thereby avoiding the use of the transitions $\stackrel{X}{\longrightarrow}$, and to extend the definition to open process expressions by

$$E \leftrightarrow_b F \text{ iff for all closed process expressions } G,\ E\{G/X\} \leftrightarrow_b F\{G/X\}$$

By Propositions 2 and 3 below both approaches yield the same equivalence relation. The way of defining $\leftrightarrow_b$ on open process expressions employed here is a mild variation of the way weak equivalence was defined in MILNER [6]. It does not carry over to full CCS.

Proposition 2 $\underline{\leftrightarrow}_b \subseteq \mathcal{E} \times \mathcal{E}$ is a bisimulation and an equivalence, satisfying, for $E, F, G \in \mathcal{E}$

$$E \underline{\leftrightarrow}_b F \Rightarrow E\{G/X\} \underline{\leftrightarrow}_b F\{G/X\}.$$

Proof: The identity relation $\mathrm{Id}_{\mathcal{E}}$ is a branching bisimulation and if $\mathcal{R}$ and $\mathcal{S}$ are branching bisimulations, then so are $\mathcal{R}^{-1}$ and $\mathcal{R} \circ \mathcal{S} = \{(E, F) \mid \exists G \in \mathcal{E} \text{ with } (E, G) \in \mathcal{R} \text{ and } (G, F) \in \mathcal{S}\}$. Hence $\underline{\leftrightarrow}_b$ is an equivalence.

If $\mathcal{R}_i$ $(i \in I)$ are branching bisimulations, so is $\bigcup_{i \in I} \mathcal{R}_i$. Thus $\underline{\leftrightarrow}_b = \bigcup\{\mathcal{R} \mid \mathcal{R} \text{ is a bisimulation}\}$ is a branching bisimulation.

$\{(E\{G/X\}, F\{G/X\}) \mid E \underline{\leftrightarrow}_b F,\ G \in \mathcal{E}\} \cup \mathrm{Id}_{\mathcal{E}}$ is a bisimulation by Lemma 4 (using $=_\alpha \subseteq \underline{\leftrightarrow}_b$).□

Proposition 3 If $E\{G/X\} \underline{\leftrightarrow}_b F\{G/X\}$ for all closed process expressions G, then $E \underline{\leftrightarrow}_b F$.

Proof: As A is nonempty, there is an $a \in A$. It is easy to see that $a^m \not\underline{\leftrightarrow}_b a^n$ for $m \neq n$, where $a^n = aa \cdots a0$ with n a's. Thus, by Proposition 1, for given E and F it is possible to choose $n \in \mathbb{N}$ such that $a^{n-1} \not\underline{\leftrightarrow}_b H$ and thus $a^{n-1} \not\underline{\leftrightarrow}_b H\{a^n/X\}$ for $H \in \mathcal{E}_E \cup \mathcal{E}_F$. By assumption $E\{a^n/X\} \underline{\leftrightarrow}_b F\{a^n/X\}$. It suffices to prove that $\{(E', F') \subseteq \mathcal{E}_E \times \mathcal{E}_F \mid E'\{a^n/X\} \underline{\leftrightarrow}_b F'\{a^n/X\}\}$ is a branching bisimulation, which is a straightforward application of Lemma 4. □

The following is a powerful tool for establishing statements $E \underline{\leftrightarrow}_b F$. It is analogous to MILNER's notions of *strong bisimulation up to* $\underline{\leftrightarrow}_s$ and *weak bisimulation up to* $\underline{\leftrightarrow}_w$. As for weak bisimulation up to $\underline{\leftrightarrow}_w$, versions of the notion below without the double arrow in the premises are easily seen to be unsound [7].

Definition 9 A *branching bisimulation up to* $\underline{\leftrightarrow}_b$ is a symmetric relation $\mathcal{R} \subseteq \mathcal{E} \times \mathcal{E}$ such that if $E\mathcal{R}F$ and $E \Longrightarrow E' \xrightarrow{x} E''$ with $E \underline{\leftrightarrow}_b E'$ and $x \neq \tau \vee E' \not\underline{\leftrightarrow}_b E''$ then $\exists E_1', E_1'', F_1', F_1'', F', F''$ such that

$$\begin{array}{ccccccc}
E & & & \mathcal{R} & & & F \\
\Downarrow & & & & & & \Downarrow \\
E' & \underline{\leftrightarrow}_b & E_1' & \mathcal{R} & F_1' & \underline{\leftrightarrow}_b & F' \\
\downarrow x & & & & & & \downarrow x \\
E'' & \underline{\leftrightarrow}_b & E_1'' & \mathcal{R} & F_1'' & \underline{\leftrightarrow}_b & F''
\end{array}$$

Proposition 4 If R is a branching bisimulation up to $\underline{\leftrightarrow}_b$ and $E\mathcal{R}F$, then $E \underline{\leftrightarrow}_b F$.

Proof: It suffices to prove that the relation $\underline{\leftrightarrow}_b \mathcal{R} \underline{\leftrightarrow}_b = \{(E_0, F_0) \mid \exists E, F : E_0 \underline{\leftrightarrow}_b E\mathcal{R}F \underline{\leftrightarrow}_b F_0\}$ is a branching bisimulation. So suppose E_0, E, F and F_0 are as indicated, and $E_0 \xrightarrow{x} E_0''$. Then either $x = \tau$ and $E_0'' \underline{\leftrightarrow}_b E$, which completes the proof, or there are E' and E'' with $E \Longrightarrow E' \xrightarrow{x} E''$, $E' \underline{\leftrightarrow}_b E_0 \underline{\leftrightarrow}_b E$ and $E'' \underline{\leftrightarrow}_b E_0''(\not\underline{\leftrightarrow}_b E$ if $x = \tau)$. In the latter case apply Definition 9, and use that $F_0 \underline{\leftrightarrow}_b F \Longrightarrow F' \xrightarrow{x} F''$ implies $x = \tau \wedge F'' \underline{\leftrightarrow}_b F_0$ or $F_0 \Longrightarrow F_0' \xrightarrow{x} F_0''$ with $F' \underline{\leftrightarrow}_b F_0'$ and $F'' \underline{\leftrightarrow}_b F_0''$ by Definition 8 (and in one case Lemma 5 to find F_0'). □

Just like weak bisimulation equivalence, branching equivalence is not a congruence on BCCS$^\omega$. Also the simplest counterexample is the same: $a \underline{\leftrightarrow}_b \tau a$ but, for $b \neq a$, $a + b \not\underline{\leftrightarrow}_b \tau a + b$. Here, as usual, $a0$ is abbreviated by a and action prefixing binds stronger than choice. Milner selected weak bisimulation congruence to be the largest (= coarsest) congruence contained in weak equivalence, and the same solution is applied here. Just like weak congruence, branching congruence has a nice characterization, showing that it is close to the original equivalence.

Definition 10 Two expressions E and F are *rooted branching bisimulation equivalent* or *branching (bisimulation) congruent*—notation $E \leftrightarroweq_{rb} F$—if $\forall x \in A_\tau \cup V$:

$$E \xrightarrow{x} E' \text{ implies } \exists F' : F \xrightarrow{x} F' \wedge E' \leftrightarroweq_b F'$$

$$F \xrightarrow{x} F' \text{ implies } \exists E' : E \xrightarrow{x} E' \wedge E' \leftrightarroweq_b F'.$$

Proposition 5 (Congruence) $\leftrightarroweq_{rb}$ is an equivalence relation such that

$$\text{if } E = F \text{ then } aE = aF, \quad E + G = F + G, \quad G + E = G + F \text{ and } \mu X E = \mu X F.$$

Moreover it is the coarsest relation with these properties contained in $\leftrightarroweq_b$.

Proof: Similar to the congruence proofs for strong and weak bisimulation congruence in [7]. □

The following shows that the definition of $\leftrightarroweq_{rb}$ for open expressions yields the same notion as the standard approach based on substitution of closed terms.

Proposition 6 Let $E, F \in \mathcal{E}$. Then $E \leftrightarroweq_b F$ implies $E\{G/X\} \leftrightarroweq_b F\{G/X\}$ for $G \in \mathcal{E}$, and if $E\{G/X\} \leftrightarroweq_b F\{G/X\}$ for closed $G \in \mathcal{E}$, then $E \leftrightarroweq_b F$.

Proof: Straightforward with Lemma 4, using Propositions 2 and 3 and the same G as before. □

Milner [7] listed two results that show how close weak equivalence and congruence are to each other. The first was that for *stable* processes (processes without outgoing τ-transitions) the equivalence and congruence coincide. This result carries over to branching bisimulation, as follows immediately from the definitions. The second result says that in each weak bisimulation equivalence class there are at most two congruence classes, with representatives E and τE for some $E \in \mathcal{E}$. This is not true for branching bisimulation, indicating that branching equivalence and congruence are less close than weak equivalence and congruence. However, a corollary of this property does hold, showing that the distance is still reasonable.

Proposition 7 $E \leftrightarroweq_b F \;\Leftrightarrow\; \tau E \leftrightarroweq_{rb} \tau F$.
Proof: Immediate from Definition 10. □

This proposition effectively turns any complete axiomatization for $\leftrightarroweq_{rb}$ into one for $\leftrightarroweq_b$.

4 The axioms

The following set of axioms will be proven to be sound and complete for $\leftrightarroweq_{rb}$. The entries below are actually axiom schemes, in metavariables $E, F, G \in \mathcal{E}$, $X \in V$ and (in the axiom B) $a \in A_\tau$. This means that there is an axiom for every choice of E, F, G, X and a. The axiom schemes E1-3 and A1-4 could be replaced by single axioms, by using real variables X, Y and Z instead of the metavariables E, F and G, and adding the law of substitution: if $E = F$ then $E\{G/X\} = F\{G/X\}$, which is sound by Proposition 6. However, this would not work for R1-6, since the bound variable X is allowed to occur in E, F and G. The axioms $\mu X E = \mu Y(E\{Y/X\})$ (α-*conversion*) are derivable from R1-6, using Theorem 3 and R2.

E1	$E = E$
E2	if $E = F$ then $F = E$
E3	if $E = F$ and $F = G$ then $E = G$
E4	if $E = F$ then $aE = aF$, $E + G = F + G$, $G + E = G + F$, and $\mu X E = \mu X F$
A0	$E + 0 = E$
A1	$E + F = F + E$
A2	$E + (F + G) = (E + F) + G$
A3	$E + E = E$
B	$a(\tau(E + F) + E) = a(E + F)$ for $a \in A_\tau$
R1	$\mu X E = E\{\mu X E/X\}$
R2	if $F = E\{F/X\}$ then $F = \mu X E$, provided X is guarded in E
R3	$\mu X(X + E) = \mu X E$
R4	$\mu X(\tau(\tau E + F) + G) = \mu X(\tau(E + F) + G)$, provided X is unguarded in E
R5	$\mu X(\tau(X + E) + \tau(X + F) + G) = \mu X(\tau(X + E + F) + G)$
R6	$\mu X(\tau(X + E) + F) = \mu X(\tau(E + F) + F)$

One writes $T \vdash E = F$, with T a list of axiom names, if the equation $E = F$ is derivable from the axioms in T. Moreover, in this paper the convention is adopted that the axioms E1-4 and A0-3 are always in T, even if not explicitly listed. In the next 3 sections I will establish the following completeness theorems.

- For $E, F \in \mathcal{E}^g$: $E \leftrightarrow_{rb} F \;\Leftrightarrow\;$ BR1-2 $\vdash E = F$
- For $E, F \in \mathcal{E}^f$: $E \leftrightarrow_{rb} F \;\Leftrightarrow\;$ B $\vdash E = F$
- For $E, F \in \mathcal{E}$: $E \leftrightarrow_{rb} F \;\Leftrightarrow\;$ BR $\vdash E = F$

The rest of this section will be devoted to the soundness of the axioms.

Soundness: The soundness of E1-4 is established in Proposition 5. As far as R1 concerns, one has $\mu X E \xrightarrow{x} F \;\Leftrightarrow\; E\{\mu X E/X\} \xrightarrow{x} F$ from which it follows that $\mu X E \leftrightarrow_{rb} E\{\mu X E/X\}$ (the terms are even *strongly* bisimilar). In the same way the soundness of A0-4 is established. By inspection of their outgoing transitions, it follows that $\{(\tau(E+F)+E, E+F)\} \cup \mathrm{Id}_{\mathcal{E}}$ is a branching bisimulation and hence $a(\tau(E + F) + E) \leftrightarrow_{rb} a(E + F)$.

Proposition 8 If $F \leftrightarrow_{rb} E\{F/X\}$ then $F \leftrightarrow_{rb} \mu X E$, provided X is guarded in E.

Proof: For $G, H \in \mathcal{E}$ write $H(G)$ for $H\{G/X\}$. Let $E, F, G \in \mathcal{E}$, such that X is guarded in E, $F \leftrightarrow_{rb} E(F)$ and $G \leftrightarrow_{rb} E(G)$. I will show that the symmetric closure of $\{(H(E(F)), H(E(G))) \mid H \in \mathcal{E}\}$ is a bisimulation up to $\leftrightarrow_b$. So suppose that $H(E(F)) \Longrightarrow K' \xrightarrow{x} K''$ (in this proof one doesn't even need to assume that $H(E(F)) \leftrightarrow_b K'$ and $x \neq \tau \vee K' \not\leftrightarrow_b K''$). As X is guarded in E and hence in $H(E)$, it cannot be that $H(E) \Longrightarrow\xrightarrow{X} 0$, by Lemma 3. Thus K' and K'' are of the form $H'(F)$ and $H''(F)$ by Lemma 4.3, and by Lemma 4.2 $H(E(G)) \Longrightarrow H'''(G) \xrightarrow{x} H''''(G)$ with $H''' =_\alpha H'$ and $H'''' =_\alpha H''$. Furthermore, by Proposition 5, $H'(E(F)) \leftrightarrow_b H'(F)$,

$H'''(G) \leftrightarroweq_b H'(E(G))$, $H''(E(F)) \leftrightarroweq_b H''(F)$ and $H''''(G) \leftrightarroweq_b H''(E(G))$. The requirement starting with $H(E(G))$ follows by symmetry, so the relation is a branching bisimulation up to $\leftrightarroweq_b$ and by Proposition 4 $H(E(F)) \leftrightarroweq_b H(E(G))$ for $H \in \mathcal{E}$. Using this, a repeat of the argument above with $K' = H(E(F))$ gives $H(E(F)) \leftrightarroweq_{rb} H(E(G))$, so in particular $E(F) \leftrightarroweq_{rb} E(G)$, and hence $F \leftrightarroweq_{rb} G$. Finally take $G = \mu X E$. □

Proposition 9 $\mu X(\tau(\tau E + F) + G) \leftrightarroweq_{rb} \mu X(\tau(E + F) + G)$, provided X is unguarded in E.

Proof: By Lemma 3 there are $E_0, \ldots, E_n$ such that $\tau E + F \xrightarrow{\tau} E_0 \xrightarrow{\tau} E_1 \xrightarrow{\tau} \cdots E_n \xrightarrow{X}$ with $E_0 = E$ and $n \in \mathbb{N}$. Write E'_{-1} for $\tau E + F$ and E''_0 for $E + F$. Then by Lemma 4

$$L \xrightarrow{\tau} E'_{-1}\{L/X\} \xrightarrow{\tau} E'_0\{L/X\} \xrightarrow{\tau} E'_1\{L/X\} \xrightarrow{\tau} \cdots E'_n\{L/X\} \xrightarrow{\tau} E'_{-1}\{L/X\}$$

for certain $E'_i =_\alpha E_i$ $(i = 0, ..., n)$ and

$$R \xrightarrow{\tau} E''_0\{R/X\} \xrightarrow{\tau} E''_1\{R/X\} \xrightarrow{\tau} \cdots E''_n\{R/X\} \xrightarrow{\tau} E''_0\{R/X\}$$

for certain $E''_j =_\alpha E_j$ $(j = 1, ..., n)$. Let $\mathcal{R} \subseteq \mathcal{E} \times \mathcal{E}$ be the symmetric closure of

$$\{(H\{L/X\}, H'\{R/X\}) \mid H =_\alpha H'\} \cup \{(E'_i\{L/X\}, E''_j\{R/X\}) \mid -1 \leq i \leq n,\ 0 \leq j \leq n\}$$

Then $\mathcal{R}$ is a branching bisimulation and $L \leftrightarroweq_{rb} R$ by Lemma 4. □

Proposition 10 $\mu X(\tau(X + E) + \tau(X + F) + G) \leftrightarroweq_{rb} \mu X(\tau(X + E + F) + G)$.

Proof: The closure under symmetry and α-recursion of $\{(H\{L/X\}, H\{R/X\}) \mid H \in \mathcal{E}\} \cup$

$$\{(\tau(X + E)\{L/X\}, \tau(X + E + F)\{R/X\}) \cup \{(\tau(X + F)\{L/X\}, \tau(X + E + F)\{R/X\})\}$$

is a branching bisimulation. □

In the same way one proves the soundness of R3 and R6.

Proposition 11 $\mu X(X + E) \leftrightarroweq_{rb} \mu X E$. □

Proposition 12 $\mu X(\tau(X + E) + F) \leftrightarroweq_{rb} \mu X(\tau(E + F) + F)$. □

Corollary 1 (**Soundness**) For $E, F \in \mathcal{E}$: $\text{BR} \vdash E = F \ \Rightarrow\ E \leftrightarroweq_{rb} F$.

5 Completeness for guarded process expressions

Let, for $S = \{E_1, \ldots, E_n\}$, $\sum S$ be an abbreviation for $E_1 + \cdots + E_n$. This notation is justified by the axioms A0-3.

Lemma 6 For $E \in \mathcal{E}^g$, $\text{R1} \vdash E = \sum\{aE' \mid E \xrightarrow{a} E'\} + \sum\{W \mid E \xrightarrow{W} 0\}$.

Proof: By induction on the number of recursion operators in E, not counting the ones that occur in a subterm aG. If this number is 0, then E has the form $\sum_{i\in I} a_iE_i + \sum_{j\in J} W_j$ with $a_i \in A_\tau$ and $W_j \in V$ (the so-called *head normal form*) and the statement holds trivially. Otherwise E has a summand μXF, which can be replaced by $F\{\mu XF/X\}$ using R1, yielding E''. As E is guarded, E' has less recursion operators that don't occur in a subterm aG, so by induction $\text{R1} \vdash E'' = \sum\{aE' \mid E'' \xrightarrow{a} E'\} + \sum\{W \mid E'' \xrightarrow{W} 0\}$. As $E'' \xrightarrow{x} E' \Leftrightarrow E \xrightarrow{x} E'$ for $x \in A_\tau \cup V$, the statement follows. □

Definition 11 A *recursive specification* S is a set of equations $\{X = S_X \mid X \in V_S\}$ with $V_S \subseteq V$ and $S_X \in \mathcal{E}$ for $X \in V_S$. $E \in \mathcal{E}$ *T-provably satisfies* the recursive specification S *in* the variable $X_0 \in V_S$ if there are expressions E_X for $X \in V_S$ with $E = E_{X_0}$, such that for $X \in V_S$

$$T \vdash E_X = S_X\{E_Y/Y\}_{Y\in V_S}.$$

Definition 12 Let S be a recursive specification. The relations $\xrightarrow{o} \subseteq V_S \times V_S$ and $\xrightarrow{u} \subseteq V_S \times V_S$ are defined by

- $X \xrightarrow{o} Y$ if Y occurs free in S_X
- $X \xrightarrow{u} Y$ if Y occurs free and unguarded in S_X

Now S is called *well-founded* if $\xrightarrow{o}$ is well-founded on V_S, and *guarded* if $\xrightarrow{u}$ is well-founded on V_S.

Proposition 13 (**Unique solutions**) If S is a finite guarded recursive specification and $X_0 \in V_S$, then there is an expression E which R1-provably satisfies S in X_0. Moreover if there are two such expressions E and F, then $\text{R2} \vdash E = F$.

Proof: In MILNER [6]. □

Theorem 1 Let $E_0, F_0 \in \mathcal{E}^g$ with $E_0 \leftrightarroweq_{rb} F_0$. Then there is a finite guarded recursive specification S BR1-provably satisfied in the same variable $X_0 \in V_S$ by both E_0 and F_0.

Proof: Take a fresh set of variables $V_S = \{X_{EF} \mid E \in \mathcal{E}_{E_0}, F \in \mathcal{E}_{F_0}, E \leftrightarroweq_b F\}$. $X_0 = X_{E_0F_0}$. Now for $X_{EF} \in V_S$, S contains the equation

$$X_{EF} = \sum\{W \mid E \xrightarrow{W} 0 \text{ and } F \xrightarrow{W} 0\} + \sum\{aX_{E'F'} \mid E \xrightarrow{a} E',\ F \xrightarrow{a} F' \text{ and } E' \leftrightarroweq_b F'\} +$$

$$\sum\{\tau X_{E'F} \mid X_{EF} \neq X_0,\ E \xrightarrow{\tau} E' \text{ and } E' \leftrightarroweq_b F\} + \sum\{\tau X_{EF'} \mid X_{EF} \neq X_0,\ F \xrightarrow{\tau} F' \text{ and } E \leftrightarroweq_b F'\}.$$

Using that $X_{EF} \xrightarrow{u} X_{E'F'}$ iff $S_{X_{EF}}$ has a summand $\tau X_{E'F'}$, it is easy to show that any infinite u-path $X_{EF} \xrightarrow{u} X_{E'F'} \xrightarrow{u} \cdots$ implies an infinite τ-path $E \xrightarrow{\tau} E' \xrightarrow{\tau} \cdots$ or $F \xrightarrow{\tau} F' \xrightarrow{\tau} \cdots$, which cannot exist by Lemma 2 since E_0 and F_0 are guarded. Hence S is a guarded recursive specification. Moreover S is finite by Proposition 1. It remains to be established that E_0 BR1-provably satisfies S in X_0. The same statement for F_0 then follows by symmetry.

For $X_{EF} \in V_S$, let H_{EF} be the expression $\sum\{W \mid E \xrightarrow{W} 0 \text{ and } F \xrightarrow{W} 0\}+$

$$+\sum\{aE' \mid E \xrightarrow{a} E',\ F \xrightarrow{a} F' \text{ and } E' \leftrightarroweq_b F'\} + \sum\{\tau E' \mid X_{EF} \neq X_0,\ E \xrightarrow{\tau} E' \text{ and } E' \leftrightarroweq_b F\}$$

and define the expression G_{EF} by

$$G_{EF} = \begin{cases} H_{EF} + \tau E & \text{if } X_{EF} \neq X_0 \text{ and } \exists F' \text{ with } F \xrightarrow{\tau} F' \text{ and } E \leftrightarrow_b F' \\ E & \text{otherwise.} \end{cases}$$

It follows from Lemma 6 that R1 $\vdash E = E + H_{EF}$ and hence BR1 $\vdash a(H_{EF} + \tau E) = aE$. Thus

$$\text{BR1} \vdash aG_{EF} = aE \quad \text{for } a \in A_\tau. \tag{1}$$

It suffices to prove that for $X_{EF} \in V_S$

$$\text{BR1} \vdash G_{EF} = \sum\{W \mid E \xrightarrow{W} 0 \text{ and } F \xrightarrow{W} 0\} + \sum\{aG_{E'F'} \mid E \xrightarrow{a} E',\ F \xrightarrow{a} F' \text{ and } E' \leftrightarrow_b F'\} +$$

$$\sum\{\tau G_{E'F} \mid X_{EF} \neq X_0,\ E \xrightarrow{\tau} E' \text{ and } E' \leftrightarrow_b F\} + \sum\{\tau G_{EF'} \mid X_{EF} \neq X_0,\ F \xrightarrow{\tau} F' \text{ and } E \leftrightarrow_b F'\}.$$

By (1) this is equivalent to

$$\text{BR1} \vdash G_{EF} = H_{EF} + \sum\{\tau E \mid X_{EF} \neq X_0,\ F \xrightarrow{\tau} F' \text{ and } E \leftrightarrow_b F'\}. \tag{2}$$

In case $X_{EF} \neq X_0$ and $\exists F'$ with $F \xrightarrow{\tau} F'$ and $E \leftrightarrow_b F'$, this follows from the definition of G_{EF}. In case $X_{EF} \neq X_0$ and $\nexists F'$ with $F \xrightarrow{\tau} F'$ and $E \leftrightarrow_b F'$, (2) reduces to BR1 $\vdash E = H_{EF}$, and by Lemma 6 if suffices to establish, for $x \in A_\tau \cup V$, that

$$\text{if } E \xrightarrow{x} E' \text{ then } \begin{array}{l} x = \tau \text{ and } E' \leftrightarrow_b F \\ \text{or } \exists F' : F \xrightarrow{x} F' \wedge E' \leftrightarrow_b F'. \end{array}$$

But this follows from $E \leftrightarrow_b F$, using that if $F \xrightarrow{\tau} F_1 \Longrightarrow F''$ with $F'' \leftrightarrow_b E \leftrightarrow_b F$, then $E \leftrightarrow_b F_1$ by Lemma 5, violating the assumptions. Finally, in case $X_{EF} = X_0$, (2) also reduces to BR1 $\vdash E = H_{EF}$, and this time I have to establish, for $x \in A_\tau \cup V$, that

$$\text{if } E \xrightarrow{x} E' \text{ then } \exists F' : F \xrightarrow{x} F' \wedge E' \leftrightarrow_b F',$$

which follows immediately from $E \leftrightarrow_{rb} F$. □

Corollary 2 (**Completeness**) For $E, F \in \mathcal{E}^g$: $E \leftrightarrow_{rb} F \Leftrightarrow$ BR1-2 $\vdash E = F$.

6 Completeness for finite process expressions

Theorem 2 Let $E_0, F_0 \in \mathcal{E}^f$ with $E_0 \leftrightarrow_{rb} F_0$. Then there is a finite well-founded recursive specification S B-provably satisfied in the same variable $X_0 \in V_S$ by both E_0 and F_0.

Proof: The construction of S is exactly as in the proof of Theorem 1. Using that $X_{EF} \xrightarrow{o} X_{E'F'}$ iff $S_{X_{EF}}$ has a summand $aX_{E'F'}$ with $a \in A_\tau$, it is easy to show that any infinite o-path $X_{EF} \xrightarrow{o} X_{E'F'} \xrightarrow{o} \cdots$ implies an infinite path $E \xrightarrow{a_1} E' \xrightarrow{a_2} \cdots$ or $F \xrightarrow{b_1} F' \xrightarrow{b_2} \cdots$, which cannot exist by Lemma 1 since E_0 and F_0 are finite. Hence S is a well-founded recursive specification. The proof that S is finite and provably satisfies both E_0 and F_0 in X_0 is exactly as before, except that Lemma 6 is not needed, as recursion-free process expression are already in head normal form and therefore satisfy

$$\vdash E = \sum\{aE' \mid E \xrightarrow{a} E'\} + \sum\{W \mid E \xrightarrow{W} 0\}$$

without using axiom R1. As this was the only call for this axiom in the proof of Theorem 1 it follows that S is B-provably satisfied in $X_0 \in V_S$ by both E_0 and F_0. □

Proposition 14 (**Unique solutions**) If S is a finite well-founded recursive specification and $X_0 \in V_S$, then there is an expression E which provably satisfies S in X_0. Moreover if there are two such expressions E and F, then $\vdash E = F$.

Proof: By induction on the number of equations in S I find expressions E_X for $X \in V_S$, such that

$$\vdash E_X = S_X\{E_Y/Y\}_{Y \in V_S}$$

and if there are $F_X \in \mathcal{E}$ for $X \in V_S$ such that $\vdash F_X = S_X\{F_Y/Y\}_{Y \in V_S}$ then $\vdash E_X = F_Y$ for $X \in V_S$.

If S has only one equation $X = S_X$ then X does not occur free in S_X by the well-foundedness of $\xrightarrow{o}$. Hence S_X provably satisfies S, and for any other expression F satisfying S one has $\vdash F = S_X$.

Now suppose that S has more than one equation. By the well-foundedness of $\xrightarrow{o}$ there must be a variable $Z \in V_S$ such that $Y \xrightarrow{o} Z$ for no $Y \in V_S$. Obtain T from S by deleting the equation $Z = S_Z$. By induction there are $E_X \in \mathcal{E}$ for $X \in V_T$, such that $\vdash E_X = S_X\{E_Y/Y\}_{Y \in V_T}$ and if there are $F_X \in \mathcal{E}$ for $X \in V_T$ such that $\vdash F_X = S_X\{F_Y/Y\}_{Y \in V_T}$ then $\vdash E_X = F_Y$ for $X \in V_T$. Let $E_Z = S_X\{E_Y/Y\}_{Y \in V_T}$. Then, for $X \in V_S$,

$$\vdash E_X = S_X\{E_Y/Y\}_{Y \in V_T} = S_X\{E_Y/Y\}_{Y \in V_S}$$

and if there are $F_X \in \mathcal{E}$ for $X \in V_S$ such that $\vdash F_X = S_X\{F_Y/Y\}_{Y \in V_S}$ then $\vdash E_X = F_Y$ for $X \in V_T$ and hence $\vdash F_Z = S_Z\{F_Y/Y\}_{Y \in V_S} = S_Z\{F_Y/Y\}_{Y \in V_T} \overset{\text{E4}}{=} S_Z\{E_Y/Y\}_{Y \in V_T} = E_Z$. □

Corollary 3 (**Completeness**) For $E, F \in \mathcal{E}^f$: $E \leftrightarrow_{rb} F \;\Leftrightarrow\; \text{B} \vdash E = F$.

7 Completeness for all process expressions

Theorem 3 For every $E \in \mathcal{E}$ there exists a guarded expression E' with R1,3-6 $\vdash E = E'$.

Proof: It suffices to prove this for expressions of the form $E = \mu X F$. Following Milner, I prove a stronger result by induction on the depth of nesting of recursions in F, namely

For every $F \in \mathcal{E}$, there exists a guarded expression F' for which
- X is guarded in F'
- No free unguarded occurrence of any variable in F' lies within a recursion in F'
- R1,3-6 $\vdash \mu X F = \mu X F'$.

Assume that this property holds for every G whose recursion depth is less than that of F. Then for each recursion $\mu Y G$ in F that lies within no other recursion in F, there must be a guarded expression G' such that Y is guarded in G', no free unguarded occurrence of any variable in G' lies within a recursion in G', and R1,3-6 $\vdash \mu Y G = \mu Y G'$. These conditions ensure that no free unguarded occurrence of any variable in $G'\{\mu Y G'/Y\}$ lies within a recursion in this expression.

Let F_1 be the result of simultaneously replacing every such top-level recursion $\mu Y G$ in F by $G'\{\mu Y G'/Y\}$. Clearly F_1 is guarded, R1,3-6 $\vdash F = F_1$, and no free unguarded occurrence of any variable in F_1 lies within a recursion in F_1. In converting F_1 to F' such that R1,3-6 $\vdash \mu X F_1 = \mu X F'$, it remains only to remove all free unguarded occurrences of X from F_1, knowing that they do not lie within recursions. Here the axioms R3-6 are applied.

First any free unguarded occurrence of X that is not in the scope of a τ prefixing operator can be removed by R3. Next for any free unguarded occurrence of X that is in the scope of 2 or more τ's, this number can be lowered by application of R4. Applying R4 from left to right does not change the number of free unguarded occurrences of X, and does not raise the number of τ's scoping any particular such occurrence. So after finitely many applications all free unguarded occurrences of X are in the scope of exactly one τ operator, and applying R5 makes that they are all in the scope of the same τ. Finally by A3 at most one such occurrence remains, and this one is eliminated by R6. □

Corollary 4 For $E, F \in \mathcal{E}$: $E \leftrightarroweq_{rb} F \;\Leftrightarrow\; \mathrm{BR} \vdash E = F$.

8 Concluding remarks

The notion of branching bisimulation congruence employed here

- equates livelock and deadlock: $\mu X(\tau X) = \tau 0$
- does not equate divergence and livelock: $\mu X(\tau X + E) \neq \mu X(\tau X) + E$
- abstracts from divergence: $\mu X(\tau X + \tau E) = \tau E$
- and chooses minimal solutions in case of underspecification: $\mu X X = 0$,

just as Milner's standard version of weak bisimulation congruence. As in the case of weak congruence, there are alternative versions of branching congruence where these choices are made differently [4]. Complete axiomatizations for these notions remain to be provided. For weak bisimulation, such work has been done in Walker [8].

For arbitrary cardinals κ, one could define the language BCCS^κ by allowing sets of expressions as argument of a choice operator $\sum$, and functions $V_S \rightarrow \mathcal{E}$ for $V_S \subseteq V$ as argument of a recursion operator μ, as long as the size of these sets and functions is less than κ. Such a language would represent all and only the behavious with less than κ states. In generalizing the completeness theorem for guarded BCCS^κ expressions, one has to reformulate most axioms in an obvious way to deal with the new operators, slightly adapt the proof of Lemma 6 and make sure that there are at least κ variables in order for the first act in the proof of Theorem 1 to be possible. But nothing in my proof essentially depends on finiteness, and the result generalizes smoothly to guarded infinite-state behaviours. One could even take V to be a proper class and do away with all cardinality restrictions. Of course these axiomatizations are not effective, as some axioms have infinitely many premisis. The case for unguared expressions does not generalize in this way, as not every unguarded BCCS^κ expression is branching congruent with a guarded one.

By combining the axioms presented here with the complete axiomatizations for strong bisimulation that allow closed CCS, CSP and ACP expressions to be converted into head normal form, one obtains complete axiomatizations for closed terms in the language BCCS^ω to which the ACCSP operators have been added, provided that they do not occur in the scope of recursion operators (cf. Milner [6]). Remarkably, in this setting the axiom B can be simplified to $a\tau X = aX$.

Theorem 4 Every closed instance of B is derivable from $a\tau X = aX$.

Proof: (sketch) $a(bc+cb)+cab = ab\|c = a\tau b\|c = a(\tau(bc+cb)+c\tau b)+ca\tau b$. Placing both sides in CSP's synchronous composition with $a(b+c)$ yields $a(b+c) = a(\tau(b+c)+c)$. In this proof b can be replaced by $\sum_{i\in I} b_i$ and similarly for c. Now a parallel composition with $a(\sum_{i\in I} b_iE_i + \sum_{j\in J} c_jF_j)$, in which synchronization is required (only) for a, b_i and c_j yields

$$a(\sum_{i\in I} b_iE_i + \sum_{j\in J} c_jF_j) = a(\tau(\sum_{i\in I} b_iE_i + \sum_{j\in J} c_jF_j) + \sum_{j\in J} c_jF_j).$$

(In fact, one needs to assume here that the b_i and c_j are pairwise distinct, and do not occur in E_h and F_k, but this restriction can be removed with a relabelling.) □

If one would now require an abstract congruence to satisfy $a\tau X = aX$, and an interleaving congruence to be a congruence for all the operators needed above and to satisfy the equations needed above (which are standard and already satisfied by strong congruence), and if one agrees that any finite process is representable by an expression $\sum a_iE_i$, then it follows that for finite processes branching bisimulation is the finest abstract interleaving congruence that generalizes τ-less bisimulation.

References

[1] J.A. Bergstra & J.W. Klop (1985): *Algebra of communicating processes with abstraction. Theoretical Computer Science* 37(1), pp. 77–121.

[2] J.A. Bergstra & J.W. Klop (1988): *A complete inference system for regular processes with silent moves.* In F.R. Drake & J.K. Truss, editors: *Proceedings Logic Colloquium 1986,* Hull, North-Holland, pp. 21–81. First appeared as: Report CS-R8420, CWI, Amsterdam, 1984.

[3] R.J. van Glabbeek (1990): *The linear time – branching time spectrum.* In J.C.M. Baeten & J.W. Klop, editors: *Proceedings CONCUR 90,* Amsterdam, LNCS 458, Springer-Verlag, pp. 278–297.

[4] R.J. van Glabbeek & W.P. Weijland (1990): *Branching time and abstraction in bisimulation semantics.* Technical Report TUM-I9052, SFB-Bericht Nr. 342/29/90 A, Institut für Informatik, Technische Universität München, Munich, Germany. Extended abstract in G.X. Ritter, editor: *Information Processing 89*, Proceedings of the IFIP 11th World Computer Congress, San Fransisco, USA 1989, Elsevier Science Publishers B.V. (North-Holland), 1989, pp. 613-618.

[5] R. Milner (1984): *A complete inference system for a class of regular behaviours. Journal of Computer and System Sciences* 28, pp. 439–466.

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

[7] R. Milner (1990): *Operational and algebraic semantics of concurrent processes.* In J. van Leeuwen, editor: *Handbook of Theoretical Computer Science*, chapter 19, Elsevier Science Publishers B.V. (North-Holland), pp. 1201–1242. Alternatively see *Communication and Concurrency*, Prentice-Hall International, Englewood Cliffs, 1989, of which an earlier version appeared as *A Calculus of Communicating Systems*, LNCS 92, Springer-Verlag, 1980.

[8] D.J. Walker (1990): *Bisimulation and divergence. Information and Computation* 85(2), pp. 202–241.