A Characterisation of Weak Bisimulation Congruence

Rob van Glabbeek

National ICT Australia
and School of Computer Science and Engineering
The University of New South Wales
rvg@cs.stanford.edu

Abstract. This paper shows that weak bisimulation congruence can be characterised as rooted weak bisimulation equivalence, even without making assumptions on the cardinality of the sets of states or actions of the processes under consideration.

Introduction

Weak bisimulation equivalence, also known as observation equivalence [Mil90], is a fundamental semantic equivalence used in system verification, and one of the first proposed in the literature. It upgrades strong bisimulation equivalence by featuring abstraction from internal actions.

In order to allow compositional system verification, semantic equivalence relations need to be *congruences* for the operators under consideration, meaning that the equivalence class of an n-ary operator f applied to arguments $p_1, \ldots, p_n$ is completely determined by the equivalence classes of these arguments. Although strong bisimulation equivalence is a congruence for the operators of CCS, ACP_τ and many other languages found in the literature, weak bisimulation equivalence fails to be a congruence for the *choice* or *alternative composition* operator $+$ of CCS, as well as for the left-merge $\mathbin{\|\!\!\_}$ of ACP_τ. To bypass this problem, one uses the coarsest congruence relation for $+$ that is finer than weak bisimulation equivalence, called *weak bisimulation congruence*, and characterised as *rooted weak bisimulation equivalence* in [BK85]. This equivalence turns out to be a minor variant of weak bisimulation equivalence, and a congruence for all of CCS, ACP_τ and many other languages.

Classical proof sketches arguing that rooted weak bisimulation equivalence is indeed weak bisimulation congruence typically make some cardinally assumptions, such as that there is an infinite alphabet of actions of which each process uses only a finite subset. The current contribution establishes the validity of this characterisation without making such assumptions. It also argues that the *root condition* that turns weak bisimulation into rooted weak bisimulation embodies two properties, one of which is needed to obtain a congruence for the $+$, and one to obtain a congruence for the left-merge.

1 Process Graphs

Definition 1 ([BK86]). A *process graph* over an alphabet of actions *Act* is a rooted, directed graph whose edges are labelled by elements of *Act*. Formally, a process graph g is a triple $(\text{NODES}(g), \text{ROOT}(g), \text{EDGES}(g))$, where

- $\text{NODES}(g)$ is a set, of which the elements are called the *nodes* or *states* of g,
- $\text{ROOT}(g) \in \text{NODES}(g)$ is a special node: the *root* or *initial state* of g,
- and $\text{EDGES}(g) \subseteq \text{NODES}(g) \times Act \times \text{NODES}(g)$ is a set of triples (s, a, t) with $s, t \in \text{NODES}(g)$ and $a \in Act$: the *edges* or *transitions* of g.

Normally, one is not interested in the names of the nodes in a process graph. For this reason, process graphs are considered up to isomorphism.

Definition 2. Let g and h be process graphs. A *graph isomorphism* between g and h is a bijective function $f : \text{NODES}(g) \rightarrow \text{NODES}(h)$ satisfying

- $f(\text{ROOT}(g)) = \text{ROOT}(h)$ and
- $(s, a, t) \in \text{EDGES}(g) \;\Leftrightarrow\; (f(s), a, f(t)) \in \text{EDGES}(h)$.

Graphs g and h are *isomorphic*, notation $g \cong h$, if there exists a graph isomorphism between them.

If $g \cong h$ then g and h differ only in the identity of their nodes. Graph isomorphism is an equivalence relation on the class of process graphs.

Further on, process graphs are pictured by using open dots ($\circ$) to denote nodes, and labelled arrows to denote edges. The root is represented by an incoming arrow, not originating from another node. These drawings present process graphs only up to isomorphism.

Let $\mathbb{G}(Act)$ be the class of process graphs over the alphabet of actions *Act* *up to isomorphism*. This means that I am satisfied with a level of precision in describing elements of $\mathbb{G}(Act)$ that fails to distinguish isomorphic process graphs. In the digression below I will indicate how to raise the precision of my definitions to a fully formal level; the digression should also make clear that it is not really worthwhile to maintain this level throughout the paper.

Next I define the most basic process algebraic operations on $\mathbb{G}(Act)$: a constant 0 for *inaction*, a binary infix written operator $+$ for *alternative composition* or *choice*, and unary operators $a.$ for *action prefixing* for each $a \in Act$. For the sake of convenience, in the definition below I will only consider *root-acyclic* process graphs. In Sect. 3 I will extend the definition to arbitrary process graphs.

Definition 3 ([BK86]). A process graph is *root-acyclic* if it has no incoming edges at the root. Let $\mathbb{G}^\rho(Act)$ be the class of root-acyclic process graphs over *Act* up to isomorphism. The constant 0 and the operators $a.$ and $+_\rho$ are defined on $\mathbb{G}^\rho(Act)$ as follows. (The subscript ρ serves to distinguish this alternative composition from the more general one that will be defined in Sect. 3.)

- 0 is interpreted as the trivial graph, having one node (the root) and no edges;
- $a.g$ is obtained from g by adding a new node, which will be the root of $a.g$, and a new a-labelled edge from the root of $a.g$ to the root of g;
- $g +_\rho h$ is obtained by identifying the root nodes of disjoint copies of g and h.

Digression: Distinguishing Isomorphic Process Graphs

In Def. 3 I have not bothered to tell which node exactly will be the only node of the process graph 0 and which new node will be added in the construction of $a.g$. Moreover, in taking the disjoint union of g and h, no explicit solution is offered for what to do when g and h have nodes in common. Here I provide two possible answers. As it doesn't matter at all which one is chosen, the reader may pick himself, or make up a third.

Making arbitrary choices to resolve ambiguity.

The definitions of 0 and $a.g$ could for instance be given as follows:

- $\textsc{nodes}(0) = \{*\}$,
- $\textsc{root}(0) = *$,
- $\textsc{edges}(0) = \emptyset$.

- $\textsc{nodes}(a.g) = \{*\} \cup \{s' \mid s \in \textsc{nodes}(g)\}$,
- $\textsc{root}(a.g) = *$,
- $\textsc{edges}(a.g) = \{(*, a, \textsc{root}(g)')\} \cup \{(s', a, t') \mid (s, a, t) \in \textsc{edges}(g)\}$.

Here the nodes of g are renamed from s into s', so as to make sure that none of them happens to be the symbol $*$ that is used to name the new root node.

Working modulo isomorphism.

In this approach $\mathbb{G}(Act)$ is the class of process graph *modulo isomorphism*, meaning that the elements of $\mathbb{G}(Act)$ are isomorphism classes of process graphs.

Now $0 \in \mathbb{G}^\rho(Act)$ is defined as the isomorphism class of all trivial process graphs, after observing that all trivial graphs are isomorphic. To obtain the isomorphism class $a.G$, for G an isomorphism class of process graphs, I first pick a representative $g \in G$ and a fresh object $r \notin \textsc{nodes}(g)$. Then $a.g$ is defined by

- $\textsc{nodes}(a.g) = \textsc{nodes}(g) \cup \{r\}$,
- $\textsc{root}(a.g) = r$ and
- $\textsc{edges}(a.g) = \textsc{edges}(g) \cup \{(r, a, \textsc{root}(g))\}$.

Finally, $a.G$ is defined to be the isomorphism class containing $a.g$, and the exercise is concluded by showing that the result is independent of the choice of $g \in G$. Likewise, $G +_\rho H$, for $G, H \in \mathbb{G}^\rho(Act)$, is obtained as the isomorphism class of $g +_\rho h$, where $g \in G$ and $h \in H$ are chosen in such a way that $\textsc{nodes}(g) \cap \textsc{nodes}(h) = \textsc{root}(g) = \textsc{root}(h)$, and $g +_\rho h$ is defined by

- $\textsc{nodes}(g +_\rho h) = \textsc{nodes}(g) \cup \textsc{nodes}(h)$,
- $\textsc{root}(g +_\rho h) = \textsc{root}(g) = \textsc{root}(h)$ and
- $\textsc{edges}(g +_\rho h) = \textsc{edges}(g) \cup \textsc{edges}(h)$.

Again, it must be shown that the result is independent of the choice of g and h.

2 Bisimulation Semantics

To make process graphs into a useful semantic model of calculi that enable system verification, a semantic equivalence coarser than isomorphism needs to be defined. The most popular choices are strong and weak bisimulation equivalence, and some of their variants. Such an equivalence can be used fruitfully when it is a *congruence* for the process algebraic operators that are considered in a particular application. These almost always include the operators 0, $a.$ and $+$. A semantic equivalence $\sim$ is a congruence for these operators if $g \sim h$ implies $a.g \sim a.h$ and $g_1 \sim h_1 \wedge g_2 \sim h_2$ implies $g_1 + g_2 \sim h_1 + h_2$.

2.1 Strong Bisimulation

Definition 4. Let $g, h \in \mathbb{G}(Act)$. The graphs g and h are *(strong) bisimulation equivalent*, notation $g \leftrightarrows h$, if there exists a binary relation $R \subseteq \textsc{nodes}(g) \times \textsc{nodes}(h)$, called a *bisimulation* between g and h, satisfying, for all $a \in Act$:

- $\textsc{root}(g)\ R\ \textsc{root}(h)$.
- If sRt and $(s, a, s') \in \textsc{edges}(g)$, then $\exists (t, a, t') \in \textsc{edges}(h)$ such that $s'Rt'$.
- If sRt and $(t, a, t') \in \textsc{edges}(h)$, then $\exists (s, a, s') \in \textsc{edges}(g)$ such that $s'Rt'$.

It is well-known and easy to check that $\leftrightarrows$ is an equivalence relation indeed. I will now show that it is a congruence relation for $a.$ and $+_\rho$. Because these operators have so-far not been defined outside $\mathbb{G}^\rho(Act)$, for now this result will pertain to root-acyclic process graphs only.

Definition 5. A relation between the nodes of two root-acyclic process graphs is called *rooted* if it relates root nodes with root nodes only.

Lemma 1. *Let $g, h \in \mathbb{G}^\rho(Act)$. If $g \leftrightarrows h$ then there exist a rooted bisimulation between g and h.*

Proof. Let R be a bisimulation between g and h. A rooted bisimulation is obtained from R by omitting all liaisons between root nodes and non-root nodes.

Proposition 1. *On $\mathbb{G}^\rho(Act)$, bisimulation equivalence is a congruence for $a.$ and $+_\rho$.*

Proof. Suppose R is a bisimulation between g and h. Then

$$R \cup \{(\textsc{root}(a.g), \textsc{root}(a.h))\}$$

is a bisimulation between $a.g$ and $a.h$. Moreover, invoking Lemma 1, let R_i be a rooted bisimulation between g_i and h_i for $g_i, h_i \in \mathbb{G}^\rho(Act)$ and $i = 1, 2$ then $R_1 \cup R_2$ is a bisimulation between $g_1 +_\rho g_2$ and $h_1 +_\rho h_2$. $\square$

2.2 Weak Bisimulation

Let $\tau \in Act$ be the *invisible action* or *silent step*. Henceforth, write $s \xrightarrow{a}_g s'$ for $(s,a,s') \in \text{EDGES}(g)$ and $s \Longrightarrow_g s'$ when there are $s_0,\ldots,s_n$ in $\text{NODES}(g)$ such that $s = s_0 \xrightarrow{\tau}_g s_1 \xrightarrow{\tau}_g \cdots \xrightarrow{\tau}_g s_n = s'$. Moreover, $s \stackrel{a}{\Longrightarrow}_g s'$ denotes that there are nodes s_1 and s_2 in g such that $s \Longrightarrow_g s_1 \xrightarrow{a}_g s_2 \Longrightarrow_g s'$ and $s \stackrel{(a)}{\Longrightarrow}_g s'$ is a shorthand for $s \Longrightarrow_g s'$ when $a = \tau$, and $s \stackrel{a}{\Longrightarrow}_g s'$ when $a \neq \tau$. Thus, $s \stackrel{(\tau)}{\Longrightarrow}_g s'$ says that in g one can travel from s to s' by performing a sequence of zero or more τ-steps, whereas $s \stackrel{\tau}{\Longrightarrow}_g s'$ requires at least one τ-step. For $a \neq \tau$ there is no difference between $\stackrel{(a)}{\Longrightarrow}_g$ and $\stackrel{a}{\Longrightarrow}_g$.

Definition 6. Let $g,h \in \mathbb{G}(Act)$. The graphs g and h are *weak bisimulation equivalent*, notation $g \leftrightarroweq_w h$, if there exists a binary relation $R \subseteq \text{NODES}(g) \times \text{NODES}(h)$, called a *weak bisimulation* between g and h, satisfying, for all $a \in Act$:

- $\text{ROOT}(g)\ R\ \text{ROOT}(h)$.
- If sRt and $s \xrightarrow{a}_g s'$, then there is a t' such that $t \stackrel{(a)}{\Longrightarrow}_h t'$ and $s'Rt'$.
- If sRt and $t \xrightarrow{a}_g t'$, then there is an s' such that $s \stackrel{(a)}{\Longrightarrow}_h s'$ and $s'Rt'$.

It is well-known and easy to check that $\leftrightarroweq_w$ is an equivalence relation indeed. However, $\leftrightarroweq_w$ fails to be a congruence for the $+$. Namely, $\tau.a.0 \leftrightarroweq_w a.0$ but $\tau.a.0 +_\rho b.0 \not\leftrightarroweq_w a.0 +_\rho b.0$. The proof of Prop. 1 does not generalise to $\leftrightarroweq_w$ because Lemma 1 does not hold for $\leftrightarroweq_w$: there is no rooted weak bisimulation between $\tau.a.0$ and $a.0$.

2.3 Rooted Weak Bisimulation

Although weak bisimulation equivalence captures the invisible nature of the silent step rather well, in order to obtain a congruence, a finer equivalence relation is needed. Such an equivalence was proposed by Bergstra & Klop in [BK85]. In fact it is the obvious “fix” in the definition of weak bisimulation equivalence needed to inherit the proof of Prop. 1.

Definition 7 ([BK85]). Two graphs $g,h \in \mathbb{G}^\rho(Act)$ are *rooted weak bisimulation equivalent*, notation $g \leftrightarroweq_{rw} h$, if there exists a rooted weak bisimulation between them (recall Def. 5).

Again, it is easy to check that $\leftrightarroweq_{rw}$ is an equivalence relation. By definition it is finer than $\leftrightarroweq_w$. It is strictly finer because $\tau.0 \leftrightarroweq_w 0$ but $\tau.0 \not\leftrightarroweq_{rw} 0$ (and likewise $\tau.a.0 \leftrightarroweq_w a.0$ but $\tau.a.0 \not\leftrightarroweq_{rw} a.0$). Moreover, Lemma 1 implies that $\leftrightarroweq_{rw}$ is coarser than $\leftrightarroweq$. It is strictly coarser because $\tau.\tau.0 \leftrightarroweq_{rw} \tau.0$ but $\tau.\tau.0 \not\leftrightarroweq \tau.0$.

Proposition 2. *On $\mathbb{G}^\rho(Act)$, rooted weak bisimulation equivalence is a congruence for $a.$ and $+_\rho$.*

Proof. Suppose R is a rooted weak bisimulation between g and h. Then

$$R \cup \{(\text{ROOT}(a.g), \text{ROOT}(a.h))\}$$

is a rooted weak bisimulation between $a.g$ and $a.h$. Moreover, let R_i be a rooted weak bisimulation between g_i and h_i for $g_i, h_i \in \mathbb{G}^\rho(Act)$ and $i = 1,2$ then $R_1 \cup R_2$ is a rooted weak bisimulation between $g_1 +_\rho g_2$ and $h_1 +_\rho h_2$. □

3 Root Unwinding

In this section I will generalise the definitions and results of Sections 1 and 2 from root-acyclic to general process graphs.

3.1 The Definition of Alternative Composition

The definition of 0 and $a.$ on $\mathbb{G}(Act)$ is exactly as on $\mathbb{G}^\rho(Act)$ (see Def. 3). However, defining the $+$ on $\mathbb{G}(Act)$ as in Def. 3 would yield counterintuitive results. Namely we would have $g+h=m$, as in the top row of Fig. 1. However, m is able to first do a number of a-actions from g and then a b from h; this is inconsistent with the idea that $g+h$ should from the initial state onwards behave either as g or as h. In fact, strong bisimulation equivalence would fail to be a congruence for this definition of $+$, for $g \mathrel{\underline{\leftrightarrow}} g^\rho$, yet $m \not\mathrel{\underline{\leftrightarrow}} k$.

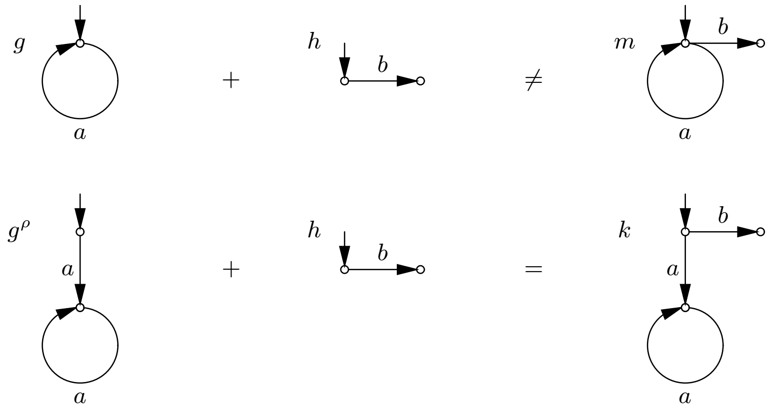

Fig. 1. Alternative composition of process graphs

For most applications we are interested in process graphs only up to strong bisimulation equivalence. Up to strong bisimilarity, the definition of the $+$ on $\mathbb{G}(Act)$ is completely determined by its definition on $\mathbb{G}^\rho(Act)$, because every process graph is strongly bisimilar with a root-acyclic process graph. The following construction is used to establish this.

Definition 8 ([BK86]). *Root unwinding* is the operator ρ on $\mathbb{G}(Act)$ given by

- $\textsc{nodes}(\rho(g)) = \textsc{nodes}(g) \stackrel{\bullet}{\cup} \{*\}$,
- $\textsc{root}(\rho(g)) = \{*\}$ and
- $\textsc{edges}(\rho(g)) = \textsc{edges}(g) \cup \{(*, a, s) \mid (\textsc{root}(g), a, s) \in \textsc{edges}(g)\}$.

Note that $\rho(g) \in \mathbb{G}^\rho(Act)$ for all $g \in \mathbb{G}(Act)$.

Proposition 3 ([BK86]). *$g \mathrel{\underline{\leftrightarrow}} \rho(g)$ for every process graph $g \in \mathbb{G}(Act)$.*

Proof. The relation $\{(s,s) \mid s \in \textsc{nodes}(g)\} \cup \{(\textsc{root}(g), *)\}$ is a bisimulation between g and $\rho(g)$. $\square$

Definition 9 ([BK86]). The definition of the + is extended from $\mathbb{G}^\rho(Act)$ to $\mathbb{G}(Act)$ by $g + h := \rho(g) +_\rho \rho(h)$, where $+_\rho$ is given by Def. 3.

This construction automatically entails that $\leftrightarrows$ is a congruence for the $+$.

Proposition 4. *On $\mathbb{G}(Act)$, $\leftrightarrows$ is a congruence for $a.$ and $+$.*

Proof. The case of $a.$ goes exactly as in Prop. 1. Let $g \leftrightarrows g'$ and $h \leftrightarrows h'$.
Then $\rho(g) \leftrightarrows g \leftrightarrows g' \leftrightarrows \rho(g')$ and likewise $\rho(h) \leftrightarrows \rho(h')$.
Hence $g + h = \rho(g) +_\rho \rho(h) \leftrightarrows \rho(g') +_\rho \rho(h') = g' + h'$ by Prop. 1. □

On $\mathbb{G}^\rho(Act)$ it is in general not the case that $g + h \cong g +_\rho h$. However, up to strong bisimilarity, both versions of the + agree.

Proposition 5. *Let $g, h \in \mathbb{G}^\rho(Act)$. Then $g + h \leftrightarrows g +_\rho h$.*

Proof. $g + h = \rho(g) +_\rho \rho(h) \leftrightarrows g +_\rho h$, using Propositions 3 and 1. □

It is also possible to merge the definitions of $+$, $+_\rho$ and root unwinding:

Definition 10. Let $g, h \in \mathbb{G}^\rho(Act)$. Then $g + h$ can alternatively be defined by

- $\textsc{nodes}(g + h) = \textsc{nodes}(g) \mathbin{\dot\cup} \textsc{nodes}(h) \mathbin{\dot\cup} \{*\}$,
- $\textsc{root}(g + h) = \{*\}$ and
- $\textsc{edges}(g + h) = \textsc{edges}(g) \cup \textsc{edges}(h) \cup$
  $\{(*, a, s) \mid (\textsc{root}(g), a, s) \in \textsc{nodes}(g) \vee (\textsc{root}(h), a, s) \in \textsc{nodes}(h)\}$.

It is trivial to check that up to isomorphism this definition yields the same alternative composition operator as Def. 9.

3.2 Rooted Weak Bisimulation

Postulating that rooted weak bisimulation has to be a coarser equivalence than strong bisimulation, Prop. 3 implies that there is a unique extension of $\leftrightarrows_{rw}$ from $\mathbb{G}^\rho(Act)$ to $\mathbb{G}(Act)$.

Definition 11. Let $g, h \in \mathbb{G}(Act)$. Then $g \leftrightarrows_{rw} h$ iff $\rho(g) \leftrightarrows_{rw} \rho(h)$ as per Def. 7.

Proposition 6. *On $\mathbb{G}(Act)$, $\leftrightarrows_{rw}$ is a congruence for $a.$ and $+$.*

Proof. The case of $a.$ goes exactly as in Prop. 2. Let $g \leftrightarrows_{rw} g'$ and $h \leftrightarrows_{rw} h'$.
Then $\rho(g) \leftrightarrows g \leftrightarrows_{rw} g' \leftrightarrows \rho(g')$, so $\rho(g) \leftrightarrows_{rw} \rho(g')$, and likewise $\rho(h) \leftrightarrows_{rw} \rho(h')$.
Hence $g + h = \rho(g) +_\rho \rho(h) \leftrightarrows_{rw} \rho(g') +_\rho \rho(h') = g' + h'$ by Prop. 2. □

The following characterisation of rooted weak bisimulation equivalence was taken as definition in [Mil90]. Let, for $g \in \mathbb{G}(Act)$ and $s \in \textsc{nodes}(g)$, g_s denote the process graph obtained from g by appointing s as its root.

Proposition 7. *Let $g, h \in \mathbb{G}(Act)$. Then $g \leftrightarrows_{rw} h$ iff*

- *if* $\textsc{root}(g) \xrightarrow{a}_g s$ *then there is a t such that* $\textsc{root}(h) \overset{a}{\Longrightarrow}_h t$ *and* $g_s \leftrightarrows_w h_t$*;*
- *if* $\textsc{root}(h) \xrightarrow{a}_h t$ *then there is an s such that* $\textsc{root}(g) \overset{a}{\Longrightarrow}_g s$ *and* $g_s \leftrightarrows_w h_t$*.*

Proof. "If": Let $B = \{(s,t) \in \textsc{nodes}(g) \times \textsc{nodes}(h) \mid g_s \leftrightarrows_w h_t\}$ and

$$B^\rho = B \,\dot{\cup}\, \{(\textsc{root}(\rho(g)), \textsc{root}(\rho(h)))\} \subseteq \textsc{nodes}(\rho(g)) \times \textsc{nodes}(\rho(h)),$$

recalling that $\textsc{nodes}(\rho(g)) = \textsc{nodes}(g) \,\dot{\cup}\, \{\textsc{root}(\rho(g))\}$, and likewise for $\rho(h)$. Assume that both clause above hold. It suffices to show that B^ρ is a rooted weak bisimulation between $\rho(g)$ and $\rho(h)$, for by Def. 11 this implies $g \leftrightarrows_{rw} h$. The relation B^ρ is rooted by construction. So I need to show it is a weak bisimulation.

Let $sB^\rho t$ and $s \xrightarrow{a}_{\rho(g)} s'$. It suffices to show that there is a t' such that $t \overset{(a)}{\Longrightarrow}_{\rho(h)} t'$ and $s' B^\rho t'$. The other requirement then follows by symmetry.

First assume sBt. In that case, $s \xrightarrow{a}_g s'$ and $g_s \leftrightarrows_w h_t$, so there is a weak bisimulation R between g_s and h_s. Thus sRt, and there must be a t' such that $t \overset{(a)}{\Longrightarrow}_h t'$ and $s'Rt'$. Hence $g_{s'} \leftrightarrows_w h_{t'}$, $s'Bt'$ and $s'B^\rho t'$. Def. 8 yields $t \overset{(a)}{\Longrightarrow}_{\rho(h)} t'$.

Now assume $s = \textsc{root}(\rho(g))$ and $t = \textsc{root}(\rho(h))$, so $\textsc{root}(\rho(g)) \xrightarrow{a}_{\rho(g)} s'$. Then, by Def. 8, $\textsc{root}(g) \xrightarrow{a}_g s'$, so by the first clause of Prop. 7 there is a t' such that $\textsc{root}(h) \overset{a}{\Longrightarrow}_h t'$ and $g_{s'} \leftrightarrows_w h_{t'}$. Hence $s'Bt'$ and $s'B^\rho t'$. Def. 8 yields $\textsc{root}(\rho(h)) \overset{a}{\Longrightarrow}_{\rho(h)} t'$, hence $t = \textsc{root}(\rho(h)) \overset{(a)}{\Longrightarrow}_{\rho(h)} t'$.

"Only if": Let R be a rooted weak bisimulation between $\rho(g)$ and $\rho(h)$. Suppose $\textsc{root}(g) \xrightarrow{a}_g s$. Def. 8 yields $\textsc{root}(\rho(g)) \xrightarrow{a}_{\rho(g)} s$, so there must be a t such that $\textsc{root}(\rho(h)) \overset{(a)}{\Longrightarrow}_{\rho(h)} t$ and sRt. As R is rooted, the case that $a = \tau$ and $t = \textsc{root}(\rho(h))$ cannot apply, so $\textsc{root}(\rho(h)) \overset{a}{\Longrightarrow}_{\rho(h)} t$, hence $\textsc{root}(h) \overset{a}{\Longrightarrow}_h t$ by Def. 8. Furthermore, sRt implies $g_s \leftrightarrows_w h_t$. The other clause follows by symmetry.

4 Weak Bisimulation Congruence

A different modification of weak bisimulation equivalence into a congruence for the + was proposed in [HM85].

Proposition 8. *For every equivalence relation $\sim$ on $\mathbb{G}(Act)$ and every set L of operators on $\mathbb{G}$ there is a coarsest congruence relation $\sim^c$ that is finer than $\sim$.*

Proof. Let a *semantic context* $C[\cdot]$ be an expression build from process graphs $g \in \mathbb{G}(Act)$ and the *hole* $[\cdot]$ through application of operators from L, and in which the hole occurs exactly once. If $C[\cdot]$ is a semantic context and $g \in \mathbb{G}(Act)$, then $C[g]$ denotes the process graph obtained by evaluating the expression $C[\cdot]$ in which g is substituted for the hole $[\cdot]$.

Alternatively, a semantic context can be regarded as a unary operator on process graphs: a *primitive context* is obtained from an operator in L by instantiating all but one of its arguments by process graphs; and a general semantic context is the composition of any number (possibly 0) of primitive contexts.

An equivalence relation $\approx$ on $\mathbb{G}(Act)$ is a congruence for L iff for every n-ary operator f in L one has $g_1 \approx h_1 \wedge \cdots \wedge g_n \approx h_n \Rightarrow f(g_1, \ldots, g_n) \approx f(h_1, \ldots, h_n)$. This is the case iff for every semantic context $C[\cdot]$ one has $g \approx h \Rightarrow C[g] \approx C[h]$.

Given an equivalence relation $\sim$ on $\mathbb{G}(Act)$, define $\sim^c$ by

$$g \sim^c h \text{ iff } C[g] \sim C[h] \text{ for every semantic context } C[\cdot].$$

By construction, $\sim^c$ is a congruence on $\mathbb{G}(Act)$. For if $g \sim^c h$ and $D[\cdot]$ is a semantic context, then for every semantic context $C[\cdot]$ also $C[D[\cdot]]$ is a semantic context, so $\forall C[\cdot](C[D[g]] \sim C[D[h]])$ and hence $D[g] \sim^c D[h]$.

The trivial context guarantees that $g \sim^c h \Rightarrow g \sim h$, so $\sim^c$ is finer than $\sim$.

Finally, $\sim^c$ is the coarsest congruence finer than $\sim$, because if $\approx$ is any congruence finer than $\sim$, then

$$g \approx h \;\Rightarrow\; \forall C[\cdot](C[g] \approx C[h]) \;\Rightarrow\; \forall C[\cdot](C[g] \sim C[h]) \;\Rightarrow\; g \sim^c h. \qquad \square$$

Definition 12. The coarsest congruence w.r.t. the $+$ that is finer than weak bisimulation equivalence is called *weak bisimulation congruence*, notation $\leftrightarrows_w^c$.

Here I address the question whether both approaches coincide, i.e. whether $\leftrightarrows_{rw} = \leftrightarrows_w^c$. Prop. 6 immediately yields $\leftrightarrows_{rw} \subseteq \leftrightarrows_w^c$. The following proof sketch, due to Jan Willem Klop [personal communication], is a first attempt to establish the reverse.

Proof Sketch. It suffices to restrict attention to the graphs in $\mathbb{G}^\rho(Act)$, for when $g \leftrightarrows_w^c h \Rightarrow g \leftrightarrows_{rw} h$ for $g, h \in \mathbb{G}^\rho(Act)$, then by Prop. 3 the same holds for $g, h \in \mathbb{G}(Act)$. [Namely $g \leftrightarrows_w^c h \Rightarrow \rho(g) \leftrightarrows g \leftrightarrows_w^c h \leftrightarrows \rho(h) \Rightarrow \rho(g) \leftrightarrows_w^c \rho(h) \Rightarrow \rho(g) \leftrightarrows_{rw} \rho(h) \Rightarrow g \leftrightarrows \rho(g) \leftrightarrows_{rw} \rho(h) \leftrightarrows h \Rightarrow g \leftrightarrows_{rw} h$.]

So let $g, h \in \mathbb{G}^\rho(Act)$ and assume $g \leftrightarrows_w^c h$. Let $a \neq \tau$ be an action that does not occur in either g or h. Then $g + a.0 \leftrightarrows_w h + a.0$ by the definition of $\leftrightarrows_w^c$. Let R be a weak bisimulation between $g + a.0$ and $h + a.0$.

Claim. R must be rooted.

Proof of Claim. $\textsc{root}(g + a.0) \xrightarrow{a}_{g+a.0} \textsc{root}(0)$, so if $\textsc{root}(g + a.0)\, R\, t$ then $t \overset{a}{\Longrightarrow}_{h+a.0} t'$ for some $t' \in \textsc{nodes}(h+a.0)$ with $\textsc{root}(0)\, R\, t'$. This is only possible if $t = \textsc{root}(h + a.0)$. By symmetry, $s\, R\, \textsc{root}(h + a.0)$ is only possible if $s = \textsc{root}(g + a.0)$.

Application of Claim. The restriction of R to the nodes of g and h is a rooted weak bisimulation between g and h, showing that $g \leftrightarrows_{rw} h$. $\square$

The only weak point in this proof sketch is in the choice of an action $a \neq \tau$ that does not occur in g or h. What if such an action does not exists? Below I present two solutions to this problem.

4.1 The Fresh Atom Principle

The *Fresh Atom Principle* (FAP) allows us to use fresh actions in proofs. It was invented and named by Jan Willem Klop [personal communication]. In order to justify the use of FAP, one needs to realise that for any choice of a set of actions *Act* containing τ there exists a class $\mathbb{G}(Act)$ of process graphs over *Act*. Likewise a *parametrised* semantic equivalence $\sim$ has an incarnation in each of these classes, and I write $\sim^{Act}$ to denote the equivalence $\sim$ as it exists in $\mathbb{G}(Act)$. Obviously, $\mathbb{G}(A) \subseteq \mathbb{G}(B)$ for sets of actions $A \subseteq B$. Now say that a parametrised equivalence $\sim$ *satisfies* FAP, if, whenever $A \subseteq B$, $\sim^A$ is the restriction of $\sim^B$ to $\mathbb{G}(A)$, i.e. if for $g, h \in \mathbb{G}(A) \subseteq \mathbb{G}(B)$ one has $g \sim^A h \Leftrightarrow g \sim^B h$. It is immediate from their definitions that $\leftrightarrows$, $\leftrightarrows_w$, $\leftrightarrows_{rw}$ and most other semantic equivalences defined in the literature satisfy FAP. In fact, satisfying FAP looks like a good sanity check for any meaningful equivalence.

In this spirit, one may want to use, instead of $\leftrightarrows_w^c$, the coarsest congruence finer than $\leftrightarrows_w$ that satisfies FAP, notation $\leftrightarrows_w^{fc}$. This congruence is obtained by allowing, in the proof of Prop. 8, contexts C that may involve fresh actions.

Theorem 1. $\leftrightarrows_w^{fc}$ *coincides with* $\leftrightarrows_{rw}$.

Proof. The proof sketch above applies here. After assuming $g \leftrightarrows_w^{fc} h$ in $\mathbb{G}(Act)$, apply FAP to obtain $g \leftrightarrows_w^{fc} h$ in $\mathbb{G}(Act \mathbin{\dot\cup} \{a\})$. As in the proof sketch, conclude that $g \leftrightarrows_{rw} h$ in $\mathbb{G}(Act \mathbin{\dot\cup} \{a\})$. Since $\leftrightarrows_{rw}$ satisfies FAP this implies that $g \leftrightarrows_{rw} h$ in $\mathbb{G}(Act)$. □

4.2 Arbitrary Many Non-bisimilar Processes

Even though we now know that $\leftrightarrows_{rw}$ coincides with $\leftrightarrows_w^{fc}$ and thus is the coarsest sane equivalence contained in $\leftrightarrows_w$ that is a congruence for the +, the question still remains if also $\leftrightarrows_w^c$ coincides with $\leftrightarrows_{rw}$. In case $Act = \{\tau\}$, this is not the case. For in that case $\leftrightarrows_w$, and hence also $\leftrightarrows_w^c$, is the universal relation, but $\tau.0 \not\leftrightarrows_{rw} 0$. However, as I will show in the following, in all other cases we have in fact that $\leftrightarrows_w^c = \leftrightarrows_{rw}$.

Proposition 9 (Jan Willem Klop). *Provided that there is at least one action* $a \in Act - \{\tau\}$*, for each infinite cardinal* κ *there are at least* κ *bisimulation equivalence classes of* τ*-free process graphs in* $\mathbb{G}(Act)$ *with less than* κ *nodes.*

Proof. For each ordinal λ define $g_\lambda \in \mathbb{G}(Act)$ as follows.

- $g_0 := 0$,
- $g_{\lambda+1} := g_\lambda + a.g_\lambda$ and
- for λ a limit ordinal, $g_\lambda := \sum_{\mu<\lambda} g_\mu$, meaning that g_λ is constructed from all graphs g_μ for $\mu < \lambda$ by identifying their root.

Claim 1. $\textsc{root}(g_\lambda) \xrightarrow{a}_{g_\lambda} s \quad \Leftrightarrow \quad s = \textsc{root}(g_\mu)$ for some $\mu < \lambda$.

Proof of Claim 1. A straightforward transfinite induction on λ.

Claim 2. If $\mu < \lambda$ then $g_\lambda \not\leftrightarrows g_\mu$.

Proof of Claim 2, by transfinite induction on λ: Let $\mu < \lambda$. By Claim 1 we have $\textsc{root}(g_\lambda) \xrightarrow{a}_{g_\lambda} \textsc{root}(g_\mu)$. However, when $\textsc{root}(g_\mu) \xrightarrow{a}_{g_\mu} s$ we have $s = \textsc{root}(g_\rho)$ for some $\rho < \mu$, and by induction $g_\mu \not\leftrightarrows g_\rho$. Thus $g_\lambda \not\leftrightarrows g_\mu$.

Claim 3. For infinite λ, $|\text{NODES}(g_\lambda)| = |\lambda|$.

Proof of Claim 3. A straightforward transfinite induction on λ, using that the cardinality of a disjoint union of $|\lambda|$ sets of cardinality $0 < \kappa \leq |\lambda|$ is $|\lambda|$.

Application of the claims. For each infinite cardinal κ there are κ ordinals λ smaller than κ. Now the proposition follows from Claims 2 and 3. $\square$

Theorem 2. *Provided that there is an $a \in Act - \{\tau\}$, on $\mathbb{G}(Act)$ weak bisimulation congruence coincides with rooted weak bisimulation equivalence.*

Proof. Following the idea from the earlier proof sketch, let $g \leftrightarrows_w^c h$ with $g, h \in \mathbb{G}^\rho(Act)$. Let κ be the smallest infinite cardinal, such that g and h have less than κ nodes. So there are less than κ process graphs g_s with $s \in \text{NODES}(g)$ and h_t with $t \in \text{NODES}(h)$. Hence by Prop. 9 there is a τ-free process graph k with $|\text{NODES}(k)| < \kappa$, such that $k \not\leftrightarrows g_s$ and $k \not\leftrightarrows h_t$ for any such g_s and h_t. Now $g + a.k \leftrightarrows_w h + a.k$ by the definition of $\leftrightarrows_w^c$. Let R be a weak bisimulation between $g + a.k$ and $h + a.k$.

Claim 1. The restriction of R to $\text{NODES}(g) \times \text{NODES}(h)$ must be rooted.

Proof of Claim 1. $\text{ROOT}(g + a.k) \xrightarrow{a}_{g+a.k} \text{ROOT}(k)$, so if $\text{ROOT}(g + a.k)\, R\, t$ for some $t \in \text{NODES}(h) - \{\text{ROOT}(h)\}$, then $t \overset{a}{\Longrightarrow}_{h+a.k} t'$ for some $t' \in \text{NODES}(h)$ with $\text{ROOT}(k)\, R\, t'$ and hence $k \leftrightarrows_w h_{t'}$. By the definition of k this is impossible. Likewise, it is impossible that $s\, R\, \text{ROOT}(h + a.k)$ for $s \in \text{NODES}(g) - \{\text{ROOT}(g)\}$.

Claim 2. The restriction of R to $\text{NODES}(g) \times \text{NODES}(h)$ is a weak bisimulation.

Proof of Claim 2. Let $\text{ROOT}(g) \xrightarrow{b}_g s$. Then $\text{ROOT}(g + a.k) \xrightarrow{b}_{g+a.k} s$, so there must be a $t \in \text{NODES}(h + a.k)$ such that $\text{ROOT}(h + a.k) \overset{(b)}{\Longrightarrow}_{h+a.k} t$ and $s\, R\, t$. The possibility that $b = a$ and $t = \text{ROOT}(k)$ can not occur, for this would imply $g_s \leftrightarrows_w k$. The possibility that $b = \tau$ and $t = \text{ROOT}(h + a.k)$ can not occur by Claim 1. Therefore $t \in \text{NODES}(h) - \{\text{ROOT}(h)\}$ and $\text{ROOT}(h) \overset{b}{\Longrightarrow}_h t$. Likewise, $\text{ROOT}(h) \xrightarrow{b}_h t$ implies that there is an $s \in \text{NODES}(g) - \{\text{ROOT}(g)\}$ with $\text{ROOT}(g) \overset{b}{\Longrightarrow}_g s$ and $s\, R\, t$. It follows that the restriction of R to $\text{NODES}(g) \times \text{NODES}(h)$ is a rooted weak bisimulation between g and h. Hence $g \leftrightarrows_{rw} h$. $\square$

Rather than working with arbitrary process graphs, some people prefer to set a bound on the number of nodes. This happens for instance when one insists that $\mathbb{G}(Act)$ should be set rather than a proper class. That can be achieved by first fixing a set $\mathcal{N}$ of potential nodes, and then allowing only process graphs g with $\text{NODES}(g) \subseteq \mathcal{N}$. For any infinite cardinal κ let $\mathbb{G}_\kappa(Act)$ be the class of process graphs over Act with less than κ nodes. In particular $\mathbb{G}_{\aleph_0}(Act)$ is the class of regular, or finite-state, process graphs. Theorem 2 transfers smoothly from $\mathbb{G}(Act)$ to $\mathbb{G}_\kappa(Act)$.

Theorem 3. *Provided that there is an $a \in Act - \{\tau\}$, on $\mathbb{G}_\kappa(Act)$ weak bisimulation congruence coincides with rooted weak bisimulation equivalence.*

Proof. The proof above goes through because the process k fits in $\mathbb{G}_\kappa(Act)$. $\square$

Another cardinality restriction that is sometimes imposed, is the requirement that each node should have less than κ outgoing edges. In case $\kappa > \aleph_0$, this class of *κ-branching* process graphs is not essentially different from $\mathbb{G}_\kappa(Act)$, and the same proof applies. More interesting is the case $\kappa = \aleph_0$. Up to strong bisimulation equivalence, the *finitely branching* process graphs form a proper superclass of $\mathbb{G}_{\aleph_0}(Act)$, the finite-state process graphs, and a proper subclass of $\mathbb{G}_{\aleph_1}(Act)$, the countable state process graphs. However, as shown in [BBK87], any countable state process graph is weakly bisimilar to a finitely branching process graph. Using this, the proof of Theorem 2 can be adapted to apply to the class of finitely branching process graphs as well.

5 The Left-merge and Rooted Weak Simulations

In [BK85] is has been shown that $\leftrightarroweq_{rw}$ is a congruence for all operators of ACP_τ. Just as for the $+$, the root condition comes to the rescue in the congruence proof for the left-merge $\lfloor\!\lfloor$. One has $\tau.a.0 \lfloor\!\lfloor b.0 \not\leftrightarroweq_w a.0 \lfloor\!\lfloor b.0$, because only the former process can do a b before an a, so $\leftrightarroweq_w$ fails to be a congruence for this operator. The reason that the root condition (Def. 5) helps here, is that in $g \lfloor\!\lfloor h$ steps from h can happen in any state of g except the initial one; thus only a weak bisimulation between g_1 and g_2 that does not relate roots with non-roots can be modified into a weak bisimulation between $g_1 \lfloor\!\lfloor h$ and $g_2 \lfloor\!\lfloor h$.

There is a directional difference in the way the root condition solves the congruence problems for $+$ and $\lfloor\!\lfloor$. This can be seen by applying it to weak simulation.

Definition 13. Let $g, h \in \mathbb{G}^\rho(Act)$. The graph g is *weakly simulated* by the graph h if there exists a binary relation $R \subseteq \textsc{nodes}(g) \times \textsc{nodes}(h)$, called a *weak simulation* from g to h, satisfying, for all $a \in Act$:

- $\textsc{root}(g)\ R\ \textsc{root}(h)$.
- If sRt and $s \xrightarrow{a}_g s'$, then there is a t' such that $t \stackrel{(a)}{\Longrightarrow}_h t'$ and $s'Rt'$.

A weak simulation from g to h is *source rooted* if $s\, R\, \textsc{root}(h) \;\Rightarrow\; s = \textsc{root}(g)$. It is *target rooted* if $\textsc{root}(g)\, R\, t \;\Rightarrow\; t = \textsc{root}(h)$.

The relation of "being weakly simulation by" is a *preorder* (transitive and reflexive), called the *weak simulation preorder.* Likewise one obtains the *source rooted weak simulation preorder* and the *target rooted weak simulation preorder.*

Proposition 10. *The target rooted weak simulation preorder is a precongruence for the $+$. This means that if there are target rooted weak simulations from g_1 to h_1 and from g_2 to h_2, then there is a target rooted weak simulation from $g_1 + g_2$ to $h_1 + h_2$.*

Proof. Suppose that R_i is a target rooted weak simulation from g_i to h_i for $g_i, h_i \in \mathbb{G}^\rho(Act)$ and $i = 1, 2$ then $R_1 \cup R_2$ is a target rooted weak simulation from $g_1 +_\rho g_2$ to $h_1 +_\rho h_2$. □

Lemma 2. *If there is a weak simulation from g to h then there is a target rooted weak simulation from g to h.*

Proof. Omit all liaisons of the form $\textsc{root}(g)\, R\, t$ with $t \neq \textsc{root}(h)$. $\square$

Corollary 1. *The weak simulation preorder is a precongruence for the $+$, as well as for action prefixing, even without upgrading it with root conditions.*

A complete axiomatisation of this preorder is given by the usual axioms for strong bisimulation, together with the axioms $x \sqsubseteq x + y$ and $\tau.x = x$, where the latter is a shorthand for $\tau.x \sqsubseteq x \wedge x \sqsubseteq \tau.x$.

Proposition 11. *The source rooted simulation preorder is a precongruence for the $\parallel\!\!\lfloor$.*

Proof Idea. Suppose that R is a weak simulation from g to h for $g, h \in \mathbb{G}^\rho(Act)$. When adapting R to a weak simulation from $g \parallel\!\!\lfloor k$ to $h \parallel\!\!\lfloor k$, one needs to make sure that when $g \parallel\!\!\lfloor k$ can do a step from k, then so can $h \parallel\!\!\lfloor k$. The only way this can fail is when a non-root state from g is related to the root of h. In that case $g \parallel\!\!\lfloor k$ can do a step from k, but $h \parallel\!\!\lfloor k$ cannot. $\square$

The weak simulation preorder fails to be a precongruence for the $\parallel\!\!\lfloor$, for $\tau.a.0$ is weakly simulated by $a.0$, but $\tau.a.0 \parallel\!\!\lfloor b.0$ is not weakly simulated by $a.0 \parallel\!\!\lfloor b.0$. This is because the weak simulation from $\tau.a.0$ to $a.0$ is not source rooted. It follows that the source rooted weak simulation preorder is strictly finer than the (target rooted) weak simulation preorder.

It is possible to upgrade the notion of weak simulation by various conditions, such as the following *stability* requirement [Gla93].

Definition 14. A weak simulation R from g to h is *stability respecting* if

– if sRt and $s \stackrel{\tau}{\not\longrightarrow}_g$ then there is a t' such that $t \Longrightarrow_h t' \stackrel{\tau}{\not\longrightarrow}_h$ and sRt'.

Again, target rootedness is needed to make the induced preorder into a precongruence for the $+$, and source rootedness to make it into a precongruence for the $\parallel\!\!\lfloor$. This time the two rooted variants are incomparable, because there is a target rooted, but not source rooted, stability respecting weak simulation from $\tau.a.0$ to $a.0$, and a source rooted, but not target rooted, stability respecting weak simulation from $a.0$ to $\tau.a.0$.

6 Concluding Remark

The method to turn simulation or bisimulation based equivalences into congruences by insisting on root conditions generalises smoothly to other notions of (bi-)simulation. In the case of branching, delay and η-bisimulation, this is elaborated in [GW96]. Interestingly, the alternative characterisation of rooted weak bisimulation presented in Prop. 7 takes a rather different shape when applied to rooted branching bisimulation. However, the characterisation with the root condition of Def. 5 remains the same.

Acknowledgement. Many crucial ideas incorporated in this paper originate from Jan Willem Klop, and stem from the mid eighties. Originally, Jan Willem and I planned to write a joint pamphlet on this matter, but as this plan never materialised, I finally collected the material in this paper, dedicated to Jan Willem at the occasion of his sixtieth birthday.

References

[BBK87] J.C.M. Baeten, J.A. Bergstra & J.W. Klop (1987): *On the consistency of Koomen's fair abstraction rule.* *Theoretical Computer Science* 51(1/2), pp. 129–176.

[BK85] J.A. Bergstra & J.W. Klop (1985): *Algebra of communicating processes with abstraction.* *Theoretical Computer Science* 37(1), pp. 77–121.

[BK86] J.A. Bergstra & J.W. Klop (1986): *Algebra of communicating processes.* In J.W. de Bakker, M. Hazewinkel & J.K. Lenstra, editors: *Mathematics and Computer Science*, CWI Monograph 1, North-Holland, pp. 89–138.

[Gla93] R.J. van Glabbeek (1993): *The linear time – branching time spectrum II; the semantics of sequential systems with silent moves (extended abstract).* In E. Best, editor: Proceedings *CONCUR'93,* 4^{th} International Conference on *Concurrency Theory*, Hildesheim, Germany, LNCS 715, Springer, pp. 66–81.

[GW96] R.J. van Glabbeek & W.P. Weijland (1996): *Branching time and abstraction in bisimulation semantics.* *Journal of the ACM* 43(3), pp. 555–600.

[HM85] M. Hennessy & R. Milner (1985): *Algebraic laws for nondeterminism and concurrency.* *Journal of the ACM* 32(1), pp. 137–161.

[Mil90] 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.