Techniques of "weak bisimulation up to"*

Davide Sangiorgi Robin Milner
Laboratory for Foundations of Computer Science
Department of Computer Science, University of Edinburgh
The King's Buildings, Edinburgh EH9 3JZ, U.K.

Abstract

"Bisimulation up to" is a technique for reducing the size of the relation needed to define a bisimulation. It works smoothly in the *strong* case, where it was first introduced [5]. But this does not directly generalise to the *weak* case, as erroneously reported in [5]. To overcome this problem, two new "up-to" techniques are proposed: They are respectively based on the use of *expansion* [1] and of *almost-weak bisimulation*. The second solution is more general than the first one, but expansion enjoys a nicer mathematical treatment. The usefulness and generality of the solutions is motivated with non-trivial examples: Two different implementations of a sorting machine.

1 Introduction

Bisimulation has emerged as one of the most stable and mathematically natural concepts formulated in the study of concurrency over the past decade. Both in the *strong* case and in the *weak* case — distinguished by whether or not we want to abstract from internal details of systems — bisimulation seems the finest extensional equivalence one would want to impose.

By definition, two processes are bisimilar if there exists a bisimulation relation containing them as a pair. However, in practice this definition is hardly ever followed plainly; instead, to reduce the size of the relations exhibited one prefers to define relations which are bisimulations only when closed up under some specific and privileged relation, so to relieve the proof work needed. We call this an *"up-to" technique*. It is a pretty general

---

[*] Work partially supported by the ESPRIT BRA Project 6454 "CONFER". A preliminary version of this article appeared in the Proc. CONCUR '92, LNCS 630, Springer Verlag, 1992.

device which allows a great variety of possibilities, but which so far has not received the attention which it deserves. For instance, one could define a strong bisimulation up to strong bisimilarity [5], a weak bisimulation up to a strong equivalence, a bisimulation up to a preorder, a preorder up to an equivalence, and so on... One does not even have to limit oneself to a *single* closure relation, but different equivalences/preorders may be employed in different positions. Some (simple) up-to facilities are also implemented in the *Concurrency Workbench* [3], a software tool which supports mechanical reasoning with processes. For instance, the Workbench can systematically remove the **0** process (i.e., the inactive process), which may reduce an infinite state transition system to a finite one and permit the termination of the Workbench algorithms.

The variety of the up-to techniques is useful because it allows us each time to make the most convenient choice, depending upon the equilibrium we want between the size of the relation to exhibit and the fineness of the closure relation(s). However there are limits on the range of legitimate choices, and the border between what is legitimate and what is not is uncertain. In particular, care is needed when both the relation to prove and the closure relation(s) are weak, i.e. they abstract from internal behaviour. For instance, the technique of weak bisimulation up to weak bisimilarity ($\approx$), as originally proposed in [5] is not sound[1], in that it allows us to define relations which are not contained in $\approx$.

This raises an important problem, which is also the main concern of this paper. Weak bisimilarity is one of the most studied equivalences in process algebra and is the one where the up-to technique is most needed. We investigate here two new up-to techniques for $\approx$ based on the use of *expansion* and *almost-weak bisimulation*.

Expansion is a preorder derived from $\approx$ by, essentially, comparing the number of silent actions performed by two processes. Expansion enjoys an elegant mathematical theory, explored in [1]. We show here that it also enjoys a nice up-to technique. This can be exploited for $\approx$ because expansion implies bisimilarity and because in most practical cases where $\approx$ holds, P and Q are indeed ordered by expansion.

We introduce almost-weak bisimulation not for its intrinsic merit, but purely as supporting relation for $\approx$. With almost-weak bisimulation, our purpose is to define a relation as coarse as possible but still capable of providing us with a sound up-to technique for $\approx$. Almost-weak bisimulation is not very appealing on its own; for instance,

---

[1] This has been amended in recent printing of the book.

it is not preserved by parallel composition. But a supporting relation does not need to be substitutive; its usefulness may come from the fact that:

1. It implies the supported substitutive relation: Thus the former, if more convenient to prove and although by itself not substitutive, can even be used to derive compositional proofs of the latter, by employing the former on subsystems and then lifting it up to the latter.
2. It gives rise to a powerful up-to technique for the supported relation.

The generality and usefulness of the techniques presented is tested with examples involving different implementations of a sorting machine. Applications can also be found in [2] and [8]. The work in [8] suggests that not only can these techniques reduce the space and time resources needed for a proof, but they may even be *necessary* to make some proof possible: Certain proofs in [8] proceed by induction on the length of a derivation and the control on the production of silent actions offered by our techniques seems essential to be able to close the inductions.

NOTATION. The results we present hold in general for any transition system equipped with a special action called *silent action*. However examples/counterexamples and comparison results among relations will be presented in the setting of CCS [5], with which we assume some familiarity. If $\mathcal{R}$ and $\mathcal{S}$ are relations, then $\mathcal{R}\,\mathcal{S}$ is their composition; that is, $P\,\mathcal{R}\,\mathcal{S}\,Q$ holds if there is R s.t. $P\,\mathcal{R}\,R$ and $R\,\mathcal{S}\,Q$.

Let $(Pr, Act, \rightarrow)$ represent our transition system, where Pr is its domain, Act is the set of actions (labels) and $\rightarrow$ describes the possible transitions. We use P, Q, R and T to range over Pr; α to range over Act; ℓ to range over $Act - \{\tau\}$, where τ represents the silent action. Actions in $Act - \{\tau\}$ are usually called *visible actions.* It is important to fix a convenient notation for the arrows. We write $P \xrightarrow{\alpha} Q$ when $(P, \alpha, Q) \in \rightarrow$, to be interpreted as "$P$ may become Q by performing an action labelled α". We also use $P \xrightarrow{\widehat{\alpha}} Q$ to mean:

- $P \xrightarrow{\alpha} Q$, if $\alpha \neq \tau$,
- $P = Q$ or $P \xrightarrow{\tau} Q$, if $\alpha = \tau$.

We shall often abbreviate $P \xrightarrow{\tau} Q$ and $P \xrightarrow{\widehat{\tau}} Q$ with $P \rightarrow Q$ and $P \xrightarrow{\wedge} Q$, respectively. As usual, the "weak" arrow $\Rightarrow$ is the reflexive and transitive closure of $\rightarrow$, and then $\overset{\alpha}{\Rightarrow}$ stands for $\Rightarrow \xrightarrow{\alpha} \Rightarrow$. Finally, similarly to $P \xrightarrow{\widehat{\alpha}} Q$, we use $P \overset{\widehat{\alpha}}{\Rightarrow} Q$ to mean:

- $P \overset{\alpha}{\Rightarrow} Q$, if $\alpha \neq \tau$,
- $P \Rightarrow Q$, if $\alpha = \tau$.

2 The problem of weak bisimulation up to

Let us begin by mentioning the strong case, where the "bisimulation up to" technique was first introduced and where its theory works smoothly. We refer to [5] for details. *Strong bisimulation*, written $\sim$, is defined as the largest symmetric relation such that whenever $P \sim Q$ and $P \xrightarrow{\alpha} P'$, then for some Q' also $Q \xrightarrow{\alpha} Q'$ with $P' \sim Q'$. Then, a symmetric relation $\mathcal{S}$ is a *strong bisimulation up to* $\sim$, if whenever $P\,\mathcal{S}\,Q$ and $P \xrightarrow{\alpha} P'$, there exists Q' such that $Q \xrightarrow{\alpha} Q'$ and $P' \sim \mathcal{S} \sim Q'$. With a simple diagram-chasing argument one can prove that if $\mathcal{S}$ is a bisimulation up to $\sim$, then $\mathcal{S} \subset \sim$.

Unfortunately, the "bisimulation up to" technique for the strong case cannot be directly generalised to the weak case. Consider the definition of *weak bisimulation*, also called *observational equivalence*:

Definition 2.1 *$\mathcal{S}$ is a* weak bisimulation *if $P\,\mathcal{S}\,Q$ implies, for all α:*

1. *whenever $P \xrightarrow{\alpha} P'$, then Q' exists s.t. $Q \overset{\widehat{\alpha}}{\Rightarrow} Q'$ and $P'\,\mathcal{S}\,Q'$;*
2. *the converse, i.e. whenever $Q \xrightarrow{\alpha} Q'$, then P' exists s.t. $P \overset{\widehat{\alpha}}{\Rightarrow} P'$ and $P'\,\mathcal{S}\,Q'$.*

Two processes P and Q are weakly bisimilar, *written $P \approx Q$, if $P\,\mathcal{S}\,Q$, for some weak bisimulation $\mathcal{S}$.* $\square$

If we now want to define bisimulation up to $\approx$, our experience from the strong case would suggest the following definition:

Definition 2.2 *$\mathcal{S}$ is a* weak bisimulation up to $\approx$ *if $P\,\mathcal{S}\,Q$ implies, for all α:*

1. *whenever $P \xrightarrow{\alpha} P'$, then Q' exists s.t. $Q \overset{\widehat{\alpha}}{\Rightarrow} Q'$ and $P' \approx \mathcal{S} \approx Q'$;*
2. *the converse, i.e. whenever $Q \xrightarrow{\alpha} Q'$, then P' exists s.t. $P \overset{\widehat{\alpha}}{\Rightarrow} P'$ and $P' \approx \mathcal{S} \approx Q'$.*

$\square$

This is in fact the definition originally proposed in [5]. Unfortunately, it is not true in general that if $\mathcal{S}$ is a bisimulation up to $\approx$ as by Definition 2.2, then $\mathcal{S} \subseteq \approx$. This was proved, independently, by the first author of this paper and by Gunnar Sjödin and Bengt Jonsson using more or less the same counterexample, namely $\mathcal{S} = \{(\tau.a.\mathbf{0}, \mathbf{0})\}$. The processes $\tau.a.\mathbf{0}$ and $\mathbf{0}$ are not weakly bisimilar, but $\mathcal{S}$ does satisfy the requirements of Definition 2.2 (see the diagram chasing at the beginning of Section 4).

One could adjust things by replacing all $\xrightarrow{\alpha}$ in Definition 2.2 with $\overset{\alpha}{\Rightarrow}$: Then the resulting conditions are indeed necessary and sufficient for $\approx\mathcal{S}\approx$ to be a bisimulation, but they are expensive to check in general. The set of derivatives of P under $\overset{\alpha}{\Rightarrow}$ can be much bigger — even sometimes infinite — than the set under $\xrightarrow{\alpha}$; moreover, certain derivatives would be considered more than once (for instance, if $P \xrightarrow{\tau} P' \overset{\alpha}{\Rightarrow} P''$, then P'' would be examined both as an $\overset{\alpha}{\Rightarrow}$ derivative of P and as an $\overset{\alpha}{\Rightarrow}$ derivative of P'). For this reason, it is important to keep the "strong" arrow $P \xrightarrow{\alpha} P'$, and look for something else. The solution which was used to correct the mistake in [5] replaces some of the occurrences of $\approx$ in Definition 2.2 with $\sim$.

Theorem 2.3 *Let $\mathcal{S}$ be such that $P\,\mathcal{S}\,Q$ implies, for all α:*

1. *whenever $P \xrightarrow{\alpha} P'$, then Q' exists s.t. $Q \overset{\widehat{\alpha}}{\Rightarrow} Q'$ and $P' \sim \mathcal{S} \approx Q'$;*
2. *the converse, i.e. whenever $Q \xrightarrow{\alpha} Q'$, then P' exists s.t. $P \overset{\widehat{\alpha}}{\Rightarrow} P'$ and $P' \approx \mathcal{S} \sim Q'$.*

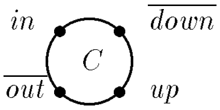
Then $\mathcal{S} \subseteq \approx$. □

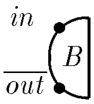
However in this solution the presence of $\sim$ may represent a too heavy restriction, as the following example shows.

Example 2.4 For this example and the following ones which continue it, we shall use the value-passing version of CCS. (The extension to value-passing of the results examined in this paper can be done in the usual fashion). Consider the sorting machine $Sorter_n$ described on page 136 of [5], with $n = 2$ fixed. $Sorter_2$ must accept exactly two integers one by one at port *in*. Then it must deliver them one by one in descending order at port $\overline{out}$, terminated by a zero. For simplicity here we also assume that $Sorter_2$ does not resume its initial state after completing the job. The specification of $Sorter_2$ is given by $Spec_{2,0}$, which is described as follows:

$$\begin{aligned}
Spec_{2,0} &\overset{def}{=} in\,(x_1).Spec_{2,1}(x_1) && (*)\\
Spec_{2,1}(x_1) &\overset{def}{=} in\,(x_2).Spec_{2,2}(x_1,x_2)\\
Spec_{2,2}(x_1,x_2) &\overset{def}{=} \overline{out}\,\big(max\{x_1,x_2\}\big).Spec_{2,-1}\big(min\{x_1,x_2\}\big)\\
Spec_{2,-1}(x_1) &\overset{def}{=} \overline{out}\,(x_1).Spec_{2,-2}\\
Spec_{2,-2} &\overset{def}{=} \overline{out}\,(0).\mathbf{0}
\end{aligned}$$

Sorter_2 is build using 2 identical cells C and a single barrier B:





The cell C has storage capacity for two numbers, and is able to compare them. Its behaviour has two phases. In the first phase — described by the equations for C and $C'(x)$ — it receives inputs at in and puts them out at $\overline{\mathit{down}}$; but since it does not know the size of the sorter, it is ready to change at any moment to its second phase — described by the equation for $C''(x,y)$ — in which it receives inputs at up and (using comparisons) puts them out at $\overline{\mathit{out}}$.

$$\begin{aligned}
C &\overset{def}{=} \mathit{in}\,(x).C'(x)\\
C'(x) &\overset{def}{=} \overline{\mathit{down}}\,(x).C + \mathit{up}\,(y).C''(x,y)\\
C''(x,y) &\overset{def}{=} \overline{\mathit{out}}\,\big(\mathit{max}\{x,y\}\big).\ \text{if } y=0 \quad \text{then } \overline{\mathit{out}}\,(0).\mathbf{0}\\
&\qquad\qquad\qquad\qquad\qquad\qquad \text{else } \mathit{up}\,(z).C''\big(z,\mathit{min}\{x,y\}\big)
\end{aligned}$$

The barrier cell B is fixed by the equation $B \overset{def}{=} \mathit{Spec}_{2,-2}$. Cells are then linked together through the chaining combinator $\frown$ which is defined using renaming and restriction as follows:

$$P \frown Q \overset{def}{=} \big(P\big[a/\mathit{down}, b/\mathit{up}\big] \mid Q\big[a/\mathit{in}, b/\mathit{out}\big]\big) \setminus \{a,b\}$$

Finally, we have $\mathit{Sorter}_2 \overset{def}{=} C \frown C \frown B$.

Let us consider how one could proceed in proving $\mathit{Spec}_{2,0} \approx \mathit{Sorter}_2$.

1. One could prove that Sorter_2 satisfies equations $(*)$ too, up to weak congruence (by the principle of unique solutions of equations, [5, section 7.3]);
2. one could try to simplify Sorter_2; following the inductive structure of Sorter_n, prove first that $\mathit{Spec}_{1,0} \approx \mathit{Sorter}_1$ and then prove that $\mathit{Spec}_{2,0} \approx C \frown \mathit{Spec}_{1,0}$;
3. one could directly find the bisimulation containing $(\mathit{Spec}_{2,0}, \mathit{Sorter}_2)$.

Now, (3) is the least cunning method, but in many examples we would perhaps not be cunning enough to do something like (1) or (2). Then our weak bisimulation should include at least (replacing processes of the form $P \frown \mathbf{0}$ with P)

$$\begin{array}{llll} \mathcal{S} = \{ & (Spec_{2,0}\ ,\ C \frown C \frown B), & (Spec_{2,1}(x)\ ,\ C \frown C''(x,0)), & \\ & (Spec_{2,2}(x,y)\ ,\ C''(x,y) \frown \overline{out}\,(0).\mathbf{0}), & (Spec_{2,-1}(x)\ ,\ C''(x,0)), & \\ & (Spec_{2,-2}\ ,\ \overline{out}\,0.\mathbf{0}), & (\mathbf{0}\ ,\ \mathbf{0}) & \} \end{array}$$

and in fact, preferably we would like to consider *only* $\mathcal{S}$. (The full bisimulation would have at least 11 pairs). However, we cannot apply Theorem 2.3: It is ok in one direction (i.e., clause (1)), but if you take for instance $C \frown C''(x,0)$, then you have

$$C \frown C''(x,0) \overset{in\,(y)}{\rightarrow} C'(y) \frown C''(x,0)$$

and the right hand-side is not strongly bisimilar to anything in the codomain of $\mathcal{S}$. It is the strong demand $\sim$ which fails us. □

3 Expansion

The expansion relation, denoted by $\lesssim$, is an asymmetric version of $\approx$ where $P \lesssim Q$ means that $P \approx Q$, but also that P achieves the same as Q with no more work, i.e. with no more τ actions. Intuitively, if $P \lesssim Q$ holds, then we can think of P as being at least as fast as Q, or more generally, we can think that Q uses at least as many resources as P. The interest of $\lesssim$ derives from the fact that, in practice, most of the uses of $\approx$ are indeed instances of $\lesssim$. For example, we are rightly interested in the question $Spec \lesssim Imp$, and $Spec$ is as usual a sequential agent with no internal τ transition possible.

Definition 3.1 *$\mathcal{E}$ is an* expansion *if $P \mathcal{E} Q$ implies, for all α:*

1. *whenever $P \overset{\alpha}{\rightarrow} P'$, then Q' exists s.t. $Q \overset{\alpha}{\Rightarrow} Q'$ and $P' \mathcal{E} Q'$;*
2. *whenever $Q \overset{\alpha}{\rightarrow} Q'$, then P' exists s.t. $P \overset{\widehat{\alpha}}{\rightarrow} P'$ and $P' \mathcal{E} Q'$;*

We say that Q expands *P, written $P \lesssim Q$, if $P \mathcal{E} Q$, for some expansion $\mathcal{E}$.* □

Relation $\lesssim$ is studied — using a different terminology — by Arun-Kumar and Hennessy: In [1] they show that $\lesssim$ is a mathematically tractable preorder, in that it is preserved by all CCS operators but sum, and that it has a complete proof system for finite terms based on a modification of the standard τ laws for CCS. A preorder called *contraction*, very close indeed to expansion, has been implemented in the Concurrency Workbench [3]. The following two theorems are from [1]:

Theorem 3.2 *$\lesssim$ is preserved by all CCS operators but sum.* $\Box$

Theorem 3.3 *It holds that $\sim \subset \lesssim$ and $\lesssim \subset \approx$; moreover each inclusion is strict.*

Proof: The inclusions are obvious. For the strictness, we have that $P \not\sim \tau.P$, $P \lesssim \tau.P$, and $\tau.P \not\lesssim P$, $\tau.P \approx P$. $\Box$

We show now that there is a simple and efficient way to define "expansions up to $\lesssim$", which in turn, since $\lesssim \subset \approx$, becomes an important instrument to prove results for $\approx$.

Definition 3.4 *$\mathcal{S}$ is an* expansion up to $\lesssim$ *if $P \,\mathcal{S}\, Q$ implies, for all α:*

1. *whenever $P \overset{\alpha}{\rightarrow} P'$, then Q' exists s.t. $Q \overset{\alpha}{\Rightarrow} Q'$ and $P' \sim \mathcal{S} \lesssim Q'$;*
2. *whenever $Q \overset{\alpha}{\rightarrow} Q'$, then P' exists s.t. $P \overset{\hat{\alpha}}{\rightarrow} P'$ and $P' \lesssim \mathcal{S} \lesssim Q'$.* $\Box$

Theorem 3.5 *If $\mathcal{S}$ is an expansion up to $\lesssim$, then $\mathcal{S} \subseteq \lesssim$.*

Proof: Simple diagram-chasing $\Box$

Comparing Theorems 2.3 and 3.5: Since all inclusions in Theorem 3.3 are strict, it is not true that Theorem 2.3 is more general than Theorem 3.5, nor is it true the other way round.

Example 3.6 (continues Example 2.4) Let $\mathcal{S}$ be the relation defined in Example 2.4. We show $\mathcal{S} \subseteq \approx$ by proving that $\mathcal{S}$ is an expansion up to $\lesssim$. We only examine the transitions for $C \frown C \frown B$; the other cases are easier and can be treated similarly. Process $C \frown C \frown B$ can only perform the transition

$$C \frown C \frown B \overset{in\,(x)}{\rightarrow} C'(x) \frown C \frown B$$

This can be matched by $\mathit{Spec}_{2,0} \overset{in\,(x)}{\rightarrow} \mathit{Spec}_{2,1}$ if we can show that

$$C \frown C''(x,0) \lesssim C'(x) \frown C \frown B \tag{1}$$

since $(\mathit{Spec}_{2,1}(x),\ C \frown C''(x,0)) \in \mathcal{S}$. By simple use of the expansion law and the obvious law $P \lesssim \tau.P$, we get

$$C'(x) \frown C \quad \gtrsim \quad C \frown C'(x) \tag{2}$$

$$C'(x) \frown B \quad \gtrsim \quad C''(x,0) \tag{3}$$

Now (1) can be derived applying (2) and (3) in sequence and using the congruence properties of $\lesssim$ from Theorem 3.2. $\square$

4 Almost-weak Bisimulation

We study in this section how close to Definition 2.2 it is possible to go without losing soundness w.r.t $\approx$. The solution we derive is based on the notion of *almost-weak bisimulation*, denoted by $\leqq$, and subsumes both Theorem 2.3 and Theorem 3.5. This however does not cancel the interest in Theorems 2.3 and 3.5, since $\leqq$ does not have the same nice mathematical theory as $\sim$ and $\lesssim$. To introduce $\leqq$, let us consider again the counterexample $\mathcal{S} = \{(\tau.a.\mathbf{0}, \mathbf{0})\}$ of Section 2, and the faulty diagram-chasing associated with it:

$$\begin{tikzcd}
\tau.a.\mathbf{0} \arrow[d] & \mathcal{S} & \mathbf{0} \arrow[d, Rightarrow] \\
a.\mathbf{0} & \approx \ \mathcal{S} & \mathbf{0}
\end{tikzcd}$$

This diagram makes $\mathcal{S}$ a bisimulation up to $\approx$ according to Definition 2.2, despite the fact that $\tau.a.\mathbf{0} \not\approx \mathbf{0}$. What is wrong here is that no "evolution" of $\tau.a.\mathbf{0}$ can be tested in $\mathcal{S}$. We consider the transition $\tau.a.\mathbf{0} \rightarrow a.\mathbf{0}$, but then we go back to $\tau.a.\mathbf{0}$ without testing $a.\mathbf{0}$. The idea for $\leqq$ is to recognise an evolution when a visible action has been produced. Once such an evolution has been met, ties can be relaxed; but as long as only τ-action are produced, we need to be restrictive and avoid any expansion.

Definition 4.1 *A relation $\mathcal{S}$ is an* almost-weak bisimulation *if $P \,\mathcal{S}\, Q$ implies, for all α, ℓ:*

1. *whenever $P \xrightarrow{\alpha} P'$, then Q' exists s.t. $Q \overset{\widehat{\alpha}}{\Rightarrow} Q'$ and $P' \approx Q'$;*
2. *whenever $Q \rightarrow Q'$, then P' exists s.t. $P \xrightarrow{\wedge} P'$ and $P' \,\mathcal{S}\, Q'$;*
3. *whenever $Q \xrightarrow{\ell} Q'$, then P' exists s.t. $P \xrightarrow{\ell}\Rightarrow P'$ and $P' \approx Q'$.*

Two processes P and Q are almost-weak bisimilar, *written $P \leqq Q$, if $P \,\mathcal{S}\, Q$, for some almost-weak bisimulation $\mathcal{S}$.* $\square$

Notice that the only restrictions w.r.t. the definition of $\approx$ are in clause 3, the use of $\xrightarrow{\ell}\Rightarrow$ instead of $\overset{\ell}{\Rightarrow}$, and in clause 2, the bound on the silent actions which P can perform together with the subsequent appearance of $\mathcal{S}$. Indeed, there are interesting cases in which $\approx$ implies $\leqq$; we mention here informally, and omitting the proof, two of them:

1. Suppose P is *stable*, i.e. it cannot perform any τ-action; then $P \approx Q$ implies $P \leqq Q$.
2. Consider the contexts $a.[\,]$ and $a.a.[\,] \,|\, \overline{a}$: They have the property that some visible action has to be produced before a process in the hole can become active, i.e. perform some action. Let us call this kind of context *strongly guarded*. Then if $C[\,]$ is a strongly guarded context, for every Q_1 and Q_2, it holds that $C[Q_1] \approx C[Q_2]$ implies $C[Q_1] \leqq C[Q_2]$. (An example of a context which is not strongly guarded is $a.[\,] \,|\, \overline{a}$)

As for $\lesssim$, it is straightforward to verify that $\leqq$ is also a preorder. What makes $\leqq$ mathematically less tractable is that $\leqq$ is not preserved by parallel composition. Consider in fact $P = \tau.a + \tau.\tau.b$ and $Q = \tau.\tau.a + \tau.b$. Then $P \approx Q$ and hence also $a.P \leqq a.Q$, since $a.P$ is stable — fact (1) above; however $P \,|\, \overline{a} \nleqq Q \,|\, \overline{a}$. This example also proves that inclusions $\lesssim \,\subset\, \leqq$ and $\leqq \,\subset\, \approx$ are strict, since it holds that $a.P \not\lesssim a.Q$ and $a.P \,|\, \overline{a} \approx a.Q \,|\, \overline{a}$.

Theorem 4.2 *The following is a chain of strict inclusions:* $\sim \,\subset\, \lesssim \,\subset\, \leqq \,\subset\, \approx$. ☐

We define now bisimulation up to $\leqq$, following the same idea for $\lesssim$.

Definition 4.3 $\mathcal{S}$ *is a* bisimulation up to $\leqq$ *if* $P \,\mathcal{S}\, Q$ *implies, for all* ℓ*:*

1. *whenever* $P \rightarrow P'$, *then* Q' *exists s.t.* $Q \Rightarrow Q'$ *and* $P' \geqq \mathcal{S} \approx Q'$*;*
2. *whenever* $P \xrightarrow{\ell} P'$, *then* Q' *exists s.t.* $Q \stackrel{\ell}{\Rightarrow} Q'$ *and* $P' \approx \mathcal{S} \approx Q'$.
3. *the converse of (1), i.e.*

   whenever $Q \rightarrow Q'$, *then* P' *exists s.t.* $P \Rightarrow P'$ *and* $P' \approx \mathcal{S} \leqq Q'$*;*
4. *the converse of (2), i.e.*

   whenever $Q \xrightarrow{\ell} Q'$, *then* P' *exists s.t.* $P \stackrel{\ell}{\Rightarrow} P'$ *and* $P' \approx \mathcal{S} \approx Q'$. ☐

If we compare it with Definition 2.2, the only restrictions in Definition 4.3 are in clauses (1) and (3), the occurrence of $\leqq$ instead of $\approx$. In fact in some cases, such as in Example 4.6 below, the verification of the clauses of Definition 4.3 reduces exactly to the verification of the clauses of Definition 2.2.

We now prove that if $\mathcal{S}$ is a bisimulation up to $\leqq$, then $\mathcal{S} \subseteq \approx$. For this, we need the following lemma.

Lemma 4.4 *Let $\mathcal{S}$ be a bisimulation up to $\leqq$. We have:*

1. *if $P \, \mathcal{S} \, Q$ and $P \Rightarrow P'$, then Q' exists s.t. $Q \Rightarrow Q'$ and $P' \geqq \mathcal{S} \approx Q'$;*
2. *if $P \approx \mathcal{S} \approx Q$ and $P \Rightarrow P'$, then Q' exists s.t. $Q \Rightarrow Q'$ and $P' \approx \mathcal{S} \approx Q'$;*
3. *if $P \geqq \mathcal{S} \approx Q$ and $P \xrightarrow{\ell} P'$, then Q' exists s.t. $Q \overset{\ell}{\Rightarrow} Q'$ and $P' \approx \mathcal{S} \approx Q'$*
4. *if $P \approx \mathcal{S} \approx Q$ and $P \xrightarrow{\alpha} P'$, then Q' exists s.t. $Q \overset{\widehat{\alpha}}{\Rightarrow} Q'$ and $P' \approx \mathcal{S} \approx Q'$*

PROOF: We consider each of the cases, in order.

1. By the induction on the length of the transition $P \Rightarrow P'$. In the basic case, this length is zero and there is nothing to prove. For the inductive case, let us write $P \Rightarrow P'$ as $P \Rightarrow R \rightarrow P'$. By the inductive hypothesis, there exist R', T', T, such that

$$Q \Rightarrow T \quad \text{and} \quad R \geqq R' \, \mathcal{S} \, T' \approx T.$$

   Then, since $R \rightarrow P'$, by definitions of $\leqq$, bisimulation up to $\leqq$ and $\approx$, respectively, there exist R'', T'', Q' such that $R \overset{\wedge}{\rightarrow} R''$, $T' \Rightarrow T''$, $Q \Rightarrow Q'$ and:

$$P' \geqq R'' \geqq \mathcal{S} \approx T'' \approx Q$$

   Now the assertion of the lemma follows from transitivity of $\leqq$ and $\approx$.

2. A simple diagram-chasing which uses (1) and the inclusion $\leqq \subseteq \approx$.

3. Let R and T be processes such that $P \geqq R \, \mathcal{S} \, T \approx Q$ holds. Then, since $P \xrightarrow{\ell} P'$ and $P \geqq R$, we have

$$R \xrightarrow{\ell} R' \Rightarrow R''.$$

Since $R\,\mathcal{S}\,T$, by definition of bisimulation up to $\leqq$ we get that:

$$T \overset{\ell}{\Rightarrow} T' \quad \text{with} \quad R' \approx \mathcal{S} \approx T'.$$

Hence, since $R' \Rightarrow R''$, using (2) also

$$T' \Rightarrow T'' \quad \text{with} \quad R'' \approx \mathcal{S} \approx T''$$

Moreover, since $T \approx Q$, also $Q \overset{\ell}{\Rightarrow} Q' \approx T''$. Finally, $P' \approx \mathcal{S} \approx Q'$ follows from the transitivity of $\approx$.

4. Suppose $\alpha \neq \tau$ (the case $\alpha = \tau$ is simpler) and let $P \approx R\,\mathcal{S}\,T \approx Q$, for some processes R and T. By definition of $\approx$, if $P \xrightarrow{\alpha} P'$, then

   $$R \Rightarrow R' \xrightarrow{\alpha} R'' \Rightarrow R''' \approx P'.$$

   Now, using (1), (2) and (3), we show that T can match these moves of R: We get

   $$T \Rightarrow T' \quad \text{with} \quad R' \geqq \mathcal{S} \approx T'$$

   by (1),

   $$T' \overset{\alpha}{\Rightarrow} T'' \quad \text{with} \quad R'' \approx \mathcal{S} \approx T''$$

   by (3), and

   $$T'' \Rightarrow T''' \quad \text{with} \quad R''' \approx \mathcal{S} \approx T'''$$

   by (2). Summarising, we have found T''' such that $T \overset{\alpha}{\Rightarrow} T'''$ and $R''' \approx \mathcal{S} \approx T'''$. Since $Q \approx T$, also $Q \overset{\alpha}{\Rightarrow} Q' \approx T'''$. Finally, by transitivity of $\approx$, we get $P' \approx \mathcal{S} \approx Q'$. □

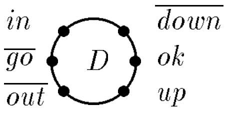
Theorem 4.5 *If $\mathcal{S}$ is a bisimulation up to $\leqq$, then $\mathcal{S} \subseteq \approx$.*

PROOF: Take $\mathcal{S}' = \{(P,Q) \mid P \approx \mathcal{S} \approx Q\}$. Now, use Lemma 4.4(4) to prove that $\mathcal{S}'$ is a weak bisimulation. □

Due to the strict inclusions in Theorem 4.2, Theorem 4.5 is indeed more general both of Theorem 2.3 and of Theorem 3.5. Moreover, it is possible to show that if any of the restrictions in Definition 4.1 or 4.3 were removed, Theorem 4.5 would fail. Therefore, it really seems that with bisimulations up to $\leqq$ we have come as close as possible to Definition 2.2.

The following is an example where Theorem 4.5 is useful. It also shows that the verification of the clauses of Definition 4.3 sometimes reduces to the verification of the clauses of Definition 2.2.

Example 4.6 (continues Examples 2.4 and 3.6) In [5], page 137, exercise 7, a different implementation of $Sorter_n$ is suggested, using a new cell D which performs comparisons and exchanges between integers during the first phase (rather than during the second phase as the cell C of $Sorter_n$):



The ports go and ok implement control synchronisations between consecutive cells, which are needed if we want to limit ourselves to cells with storage capacity of two numbers. In our solution, communications along ports go and ok guarantee that $Sorter_n$ does not input more than n numbers. The behaviour of D is defined by (constants D_1, D_2 are introduced to give a name to states which will later be referenced):

$$\begin{array}{rcl}
D & \overset{def}{=} & \overline{go}\,.D_1 \qquad\qquad D_1 \;\overset{def}{=}\; in\,(x).D'(x) \\
D'(x) & \overset{def}{=} & ok\;.\;\overline{go}\,.D_2(x) + \; up\,(y).D''(x,y) \\
D_2(x) & \overset{def}{=} & in\,(y).\,\overline{down}\,\big(min\{x,y\}\big).D'\big(max\{x,y\}\big) \\
D''(x,y) & \overset{def}{=} & \overline{out}\,(x).\text{ if } y=0 \text{ then } \overline{out}\,(0).\mathbf{0} \\
 & & \qquad\qquad\qquad \text{else } up\,(z).D''(y,z)
\end{array}$$

The same barrier cell B as in Example 2.4 is used and cells are linked together with the obvious chain operator. Also, the hiding operator '$/\,go$' [5, page 122] is needed to hide go actions produced by the first cell of $Sorter_n$ to the external environment. This operator acts in P/go as an absorber for the actions at go performed by P; that is, P/go behaves as P except that actions at go are replaced by τ's. Then we set

$$Sorter'_n \overset{def}{=} (D_1 \frown \underbrace{D \frown \ldots\ldots \frown D}_{n-1} \frown B)/\,go$$

Let us take Sorter'_2 and consider how we could prove $\mathit{Sorter}_2 \approx \mathit{Sorter}'_2$ by exhibiting a bisimulation containing the two. The following are pairs which quite reasonably should go in the bisimulation:

$$\begin{aligned}
\mathcal{S} = \{ \;& \big(C \frown C \frown B \;,\; (D_1 \frown D \frown B)/\,\mathit{go}\big) \;, \\
& \big(C \frown C''(x,0) \;,\; (D_2(x) \frown D_1 \frown B)/\,\mathit{go}\big) \;, \\
& \big(C''(x,y) \frown \overline{\mathit{out}}\,(0).\mathbf{0} \;,\; (D''(x,y) \frown \overline{\mathit{out}}\,(0).\mathbf{0})/\,\mathit{go}\big) \;, \\
& \big(C''(x,0) \;,\; D''(x,0)/\,\mathit{go}\big) \;,\; \big(\overline{\mathit{out}}\,(0).\mathbf{0} \;,\; \overline{\mathit{out}}\,(0).\mathbf{0}\big) \;,\; \big(\mathbf{0} \;,\; \mathbf{0}\big) \;\; \}
\end{aligned}$$

If we tried to prove $\mathcal{S} \subseteq \approx$ by using Theorem 2.3 or 3.5, we would fail. The reason we cannot use expansions is that Sorter_2 and Sorter'_2 are not refinements or implementations of each other. Instead, we can use Theorem 4.5. As all the processes in $\mathcal{S}$ are stable, verification of clauses of Definition 4.3 reduces to the verification of clauses in Definition 2.2. Hence, consider for instance $\big(C \frown C \frown B \;,\; (D_1 \frown D \frown B)/\,\mathit{go}\big)$ and the action

$$(D_1 \frown D \frown B)/\,\mathit{go} \;\stackrel{\mathit{in}\,(x)}{\rightarrow}\; (D'(x) \frown D \frown B)/\,\mathit{go}$$

To match it with

$$C \frown C \frown B \;\stackrel{\mathit{in}\,(x)}{\rightarrow}\; C'(x) \frown C \frown B$$

it is enough to show

$$\begin{aligned}
& (D'(x) \frown D \frown B)/\,\mathit{go} \;\approx\; (D_2(x) \frown D_1 \frown B)/\,\mathit{go} \\
\text{and} \qquad & C'(x) \frown C \frown B \;\approx\; C \frown C''(x,0)
\end{aligned}$$

This can be done similarly to what we did in Example 3.6 to derive equation (1). All remaining transitions of processes in $\mathcal{S}$ can be treated in analogous way. ☐

5 Conclusions

We have presented two new techniques for proving "weak bisimulations up to", the first based on the relation of expansion, the second one on the relation of almost-weak bisimulation. The second solution is more general than the first one, but expansions enjoy a nicer mathematical treatment and are easier to prove. We have demonstrated the usefulness of the two techniques with examples involving different implementations of a sorting machine. Further examples of applications can be found in [2], where expansion up to $\lesssim$ is used in combination with certain refinements of modal process logics, and in [8], where up-to techniques are employed to prove the full abstraction of certain encodings between behavioural equivalences.

We introduced almost-weak bisimulation with the purpose of defining a relation as coarse as possible but still providing a powerful "bisimulation up to" technique for $\approx$. A plausible question is whether $\leqq$ is the best possible relation for this. One way to make $\leqq$ more general (i.e., coarser) is by following the idea of *branching bisimulation* [4] of discriminating among τ transitions. That is, in Definition 4.1, we could aske to recognise an "evolution" by Q not only when the action performed is a visible one, but also when it represents a silent transition between non-equivalent states. This would add more flexibility in the treatment of the silent transitions by Q, which is the point (clause 2 of Definition 4.1) where the definition of $\leqq$ is most rigid. We preferred to avoid this in order to keep the resulting relation and the proof of Lemma 4.4 more manageable.

Instead, if we agree to abandon some of the generality of $\leqq$ and try to get the best of Definitions 3.4 and 4.3, a third solution is possible.

> Let $\mathcal{S}$ be such that $P \,\mathcal{S}\, Q$ implies, for all α:
>
> 1. if $P \xrightarrow{\alpha} P'$, then then Q' exists s.t. $Q \overset{\widehat{\alpha}}{\Rightarrow} Q'$ and $P' \gtrsim \mathcal{S} \approx Q'$;
> 2. if $Q \xrightarrow{\alpha} Q'$, then then P' exists s.t. $P \overset{\widehat{\alpha}}{\Rightarrow} P'$ and $P' \approx \mathcal{S} \lesssim Q'$;
>
> Then $\mathcal{S} \subset \approx$.

We think that this solution would often suffice. For instance, it is enough indeed to handle Example 3. This solution subsumes Theorems 2.3 and 3.5; it is subsumed by Theorem 4.5 but, again, it is easier to prove than Theorem 4.5 because it uses $\lesssim$ instead of $\leqq$.

Having mentioned branching bisimulation, let us point out that it would not help to replace $\approx$ with branching bisimulation in Definition 2.2: The counterexample presented would still hold.

We believe that the search for techniques reducing the size of relations which represent bisimulations may lead to interesting and useful developments. A different direction from the one we pursue in this paper is taken in [7], with the technique of *bisimulation up to context*. The idea is to factorise out common contexts and to replace the ordinary clause in the definition of bisimulation with the clause (we consider here the strong case):

- if $P \xrightarrow{\alpha} P'$, then Q', P'' and a context $C[\,]$ exist s.t.

$$P' = C[P''],\ Q \xrightarrow{\alpha} C[Q'] \text{ and } P''\mathcal{S}Q'$$

This idea gives rise to some nice theory and in some cases it allows us to reduce substantially the size of $\mathcal{S}$. It would be interesting then, and we leave it for future work, to examine if and what extent the techniques of "bisimulation up to context" and "bisimulation up to" can be integrated with each other. Some experimentation in this direction is conducted in [8], where some proofs exploit a combination of "up to context" and "up to expansion". Actually, in [8] the appeal to the up-to techniques is not just a matter of efficiency: It is not at all clear to us how the same results could be proved otherwise.

Acknowledgement

We thank Faron Moller for comments and suggestions.

References

[1] Arun-Kumar, S., and Hennessy, M., *An efficiency preorder for Processes*, Acta Informatica, vol. 29, pp 737–760, 1992.

[2] Bruns, G. *Applying process refinement to a safety-relevant system*, submitted for publication, Dept of Computer Science, University of Edinburgh, 1993.

[3] Moller, F. *The Edinburgh Concurrency Workbench (Version 6.12)* Report No. LFCS-TN-34, Dept of Computer Science, University of Edinburgh, April 1993.

[4] Glabbeek, R.J. van, and Weijland, W. P., *Branching Time and Abstraction in Bisimulation Semantics*, in Information Processing '89 (G.X. Ritter ed.), Elsevier Science, pp 613-618, 1989.

[5] Milner, R., **A Calculus of Communicating Systems**, Prentice Hall, 1989.

[6] Park, D.M., *Concurrency on Automata and infinite sequences*, in Conf. on Theoretical Computer Science, (P. Deussen ed.) Springer Verlag, LNCS vol. 104, 1981.

[7] Sangiorgi, D., *Bisimulations up to context* (provisory title), in preparation, Dept of Computer Science, University of Edinburgh, 1993.

[8] Sangiorgi, D., *Locality and true-concurrency in calculi for mobile processes*, to appear as Technical Report, Dept of Computer Science, University of Edinburgh, 1993.