Universiteit
Leiden
The Netherlands

Enhanced Coinduction

Rot, J.C.

Citation
Rot, J. C. (2015, October 15). *Enhanced Coinduction*. *IPA Dissertation Series*. Retrieved from https://hdl.handle.net/1887/35814

Version:	Not Applicable (or Unknown)
License:	Leiden University Non-exclusive license
Downloaded from:	https://hdl.handle.net/1887/35814

Note: To cite this publication please use the final published version (if applicable).

Chapter 4

Bisimulation up-to

The theory of coalgebras provides bisimilarity as a fundamental notion of equivalence between systems. In this chapter, we introduce enhancements of the proof technique for bisimilarity at this abstract level, by providing a general account of bisimulation up-to techniques for arbitrary coalgebras. We show the use of these up-to techniques by instantiating them to (non)deterministic automata, weighted automata and stream systems.

The main challenge is to provide generic up-to techniques that are *sound*, meaning that they can safely be used for proving bisimilarity. One difficulty is that sound functions do not compose, thus obstructing a modular approach to proving the soundness of up-to techniques in terms of their basic constituents. This issue was addressed by Sangiorgi [San98] and Pous [Pou07, PS12], who introduced up-to techniques in the setting of coinduction in a lattice. The central feature in the framework of [Pou07] is the notion of *compatible* functions, defining a class of sound enhancements that is closed under composition. By instantiating this framework to coalgebraic bisimilarity, we obtain compatibility as a *modular* way of proving soundness.

The first up-to technique that appeared in the literature is Milner's *bisimulation up to bisimilarity* [Mil83]. We show that this is compatible whenever the behaviour functor under consideration preserves weak pullbacks. The *equivalence closure* is also useful as an up-to technique, and its compatibility depends on weak pullback preservation as well. In the presence of algebraic structure on the state space, the notion of *bisimulation up to context* becomes relevant; we show that this is compatible whenever the coalgebraic and algebraic structure together form a λ-bialgebra. This implies, for instance, that bisimulation up to context is sound on the supported model of any GSOS specification, which is more general than the De Simone format considered in [San98]. Moreover, our compatibility results can be combined; for instance, the compatibility of the *congruence closure* follows from that of the equivalence and contextual closure. The soundness of bisimulation up-to techniques for languages, as considered in Chapter 2, is an immediate consequence.

If the behaviour functor under consideration does not preserve weak pullbacks,

then one may be interested in behavioural equivalence rather than bisimilarity (if the functor preserves weak pullbacks then these two coincide, see Section 3.1). This is the case, for example, for certain weighted transition systems [GS01, Kli09, BBB$^+$12] and for neighbourhood structures used in modal logic [HKP09]. We conclude this chapter with a treatment of up-to techniques for behavioural equivalence, and show in particular the compatibility of the contextual closure and the equivalence closure.

Throughout this chapter we only consider coalgebras in the category Set of sets and functions. Most of the technical results are a special case of more general results on coinductive up-to techniques, presented in Chapter 5 of this thesis. The current chapter explains the essentials of up-to techniques for the fundamental coinductive predicate of coalgebraic bisimilarity, requiring only basic knowledge of category theory.

Outline. The next section contains the definition of bisimulation up-to. The main instances of up-to techniques as well as a number of example proofs are in Section 4.2. Section 4.3 is a short overview of Pous's framework. This is instantiated in Section 4.4 to prove the main soundness results. Section 4.5 treats behavioural equivalence up-to. In Section 4.6 a short summary of the soundness results is provided.

4.1 Progression and bisimulation up-to

The definition of bisimulation up-to on labelled transition systems can be stated conveniently in terms of *progression* [PS12], which we generalize to a coalgebraic setting as follows.

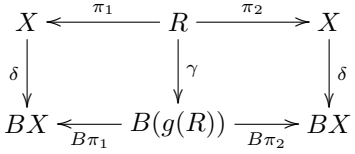
Definition 4.1.1. For a coalgebra $\delta\colon X \to BX$ and relations $R, S \subseteq X \times X$, we say R *progresses to* S if there exists a function $\gamma\colon R \to BS$ making the following diagram commute:

$$\begin{array}{ccccc} X & \xleftarrow{\pi_1} & R & \xrightarrow{\pi_2} & X \\ {\scriptstyle\delta}\downarrow & & \downarrow{\scriptstyle\gamma} & & \downarrow{\scriptstyle\delta} \\ BX & \xleftarrow[B\pi_1]{} & BS & \xrightarrow[B\pi_2]{} & BX \end{array}$$

We recover the standard definition of a bisimulation on a single coalgebra (Section 3.1) by taking $R = S$, i.e., a relation R that progresses to itself. Progression allows to define bisimulation up-to, and the crucial associated notion of soundness.

Definition 4.1.2. Let $\delta\colon X \to BX$ be a coalgebra and $g\colon \mathcal{P}(X \times X) \to \mathcal{P}(X \times X)$ be a function. A relation R is a *bisimulation up to* g if R progresses to $g(R)$, i.e., if

there is a function $\gamma\colon R \to B(g(R))$ making the following diagram commute:



We say that g is (δ)-*sound* if the following implication holds, for any $R \subseteq X \times X$:

$$\text{if } R \text{ is a bisimulation up to } g \text{ then } R \subseteq \sim_\delta ,$$

that is, g is sound if every bisimulation up to g is contained in bisimilarity.

Informally, to check that R is a bisimulation up to g, the derivatives or next states need not be related by R again, but by $g(R)$. Depending on $g(R)$, which in most examples is a bigger relation than R, this is a weaker requirement than the usual conditions for showing R to be a bisimulation. However, we only obtain a valid proof principle for bisimilarity if g is sound: then, to prove that two states are bisimilar, it suffices to relate them by a bisimulation up to g. Therefore, our main aim is to find useful functions g that are sound.

Not every function is sound; for a simple example, take the function g that maps every relation R on X to the Cartesian product $X \times X$. Then, a relation R on the states of a transition system is a bisimulation up to g if for each $(x, y) \in R$ and each label a: there is x' such that $x \xrightarrow{a} x'$ if and only if there is y' such that $x \xrightarrow{a} y'$. Clearly, this g is not sound.

4.2 Examples

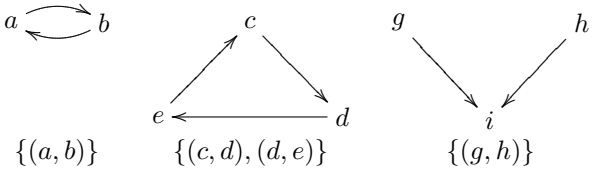
We introduce the most important instances of bisimulation up-to for a variety of systems. In each case, the up-to technique under consideration is sound (under certain assumptions), which follows from results in subsequent sections of this chapter. Thus, all of these examples can be seen as actual proofs of bisimilarity. More details on the types of coalgebras under consideration and their associated notions of bisimulation can be found in Example 3.1.1 and Example 3.1.2.

Bisimulation up to equivalence

Consider the function eq mapping a relation R to its equivalence closure $\mathsf{eq}(R)$. A bisimulation up to eq is also called a *bisimulation up to equivalence*.

Example 4.2.1. Given a coalgebra $\delta\colon X \to X + 1$, a relation R on X is a bisimulation up to equivalence if for all $(x, y) \in R$: either $\delta(x) = * = \delta(y)$, or $(\delta(x), \delta(y)) \in \mathsf{eq}(R)$. This is different than a bisimulation, which requires $(\delta(x), \delta(y)) \in R$ rather

than $(\delta(x), \delta(y)) \in \mathsf{eq}(R)$ (Example 3.1.2). Consider the following coalgebras and relations:



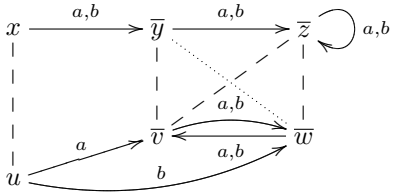
All three relations are bisimulations up to equivalence, whereas none of them are actual bisimulations. Consider, for example, the relation $\{(a, b)\}$: we have $\delta(a) = b$ and $\delta(b) = a$, but $(b, a) \notin \{(a, b)\}$. However, the pair (b, a) *is* in the least equivalence relation containing $\{(a, b)\}$.

The equivalence closure decomposes as

$$\mathsf{eq} = \mathsf{tra} \circ \mathsf{sym} \circ \mathsf{rfl}$$

where tra is transitive closure, sym is symmetric closure and rfl is reflexive closure. The relation $\{(a, b)\}$ from the above example is a bisimulation up to sym, $\{(g, h)\}$ is a bisimulation up to rfl and $\{(c, d), (d, e)\}$ is a bisimulation up to $\mathsf{tra} \circ \mathsf{sym}$.

Example 4.2.2. Consider the deterministic automaton below, with final states $\overline{y}, \overline{z}, \overline{v}, \overline{w}$ and transitions given by the solid arrows. The relation given by the four dashed lines together with the dotted line $(\overline{y}, \overline{w})$ is a bisimulation.



The relation R denoted by the four dashed lines is not a bisimulation, since $x \xrightarrow{b} y$ and $u \xrightarrow{b} w$ but $(y, w) \notin R$. However, R *is* a bisimulation up to equivalence, since the pair (y, w) is in $\mathsf{eq}(R)$. Hopcroft and Karp's algorithm [HK71] exploits this technique for checking equivalence of deterministic automata: rather than exploring n^2 pairs of states (where n is the number of states), the algorithm visits at most n pairs (that is the number of equivalence classes) (cf. [BP13]).

Bisimulation up to bisimilarity

Let $\sim$ be the bisimilarity relation of a given coalgebra $\delta\colon X \to BX$, and consider the *bisimilarity closure* function $\mathsf{bis}\colon \mathcal{P}(X \times X) \to \mathcal{P}(X \times X)$ defined by

$$\mathsf{bis}_\delta(R) = \sim \circ R \circ \sim .$$

The function bis_δ composes a relation with bisimilarity on both sides. In the sequel we sometimes drop the subscript δ and write bis, if the coalgebra under consideration is clear from the context.

A bisimulation up to bis is called a *bisimulation up to bisimilarity*. This is the very first up-to technique that appeared in the literature, in the context of labelled transition systems [Mil83].

Example 4.2.3. In this example, we prove that the stream $[1] = (1, 0, 0, \ldots)$ is the unit for the shuffle product $\otimes$, that is, $\sigma \otimes [1] \sim \sigma$. Let T be the set of terms given by the grammar $t ::= t \otimes t \mid t + t \mid [r]$, where $[r]$ ranges over $\{[r] \mid r \in \mathbb{R}\}$. As explained in Section 3.1.1, together with the appropriate behavioural differential equations, this induces a coalgebra $\langle (-)_0, (-)' \rangle \colon T \to \mathbb{R} \times T$.

We make use of the relation $R = \{(\sigma \otimes [1], \sigma) \mid \sigma \in T\}$. For any $\sigma \in T$, we have $(\sigma \otimes [1])_0 = \sigma_0 \cdot [1]_0 = \sigma_0$. Further $(\sigma \otimes [1])' = \sigma' \otimes [1] + \sigma \otimes [1]' = \sigma' \otimes [1] + \sigma \otimes [0]$; this element is not related to σ', so R is not a bisimulation. However given some basic laws of stream calculus, in particular $\sigma \otimes [0] \sim [0]$, $\sigma + [0] \sim \sigma$ and the fact that $\sim$ is a congruence, we obtain

$$((\sigma' \otimes [1]) + (\sigma \otimes [0])) \sim ((\sigma' \otimes [1]) + [0]) \sim (\sigma' \otimes [1]) \; R \; \sigma'$$

so R is a bisimulation up to bisimilarity (we use that $\sim$ is reflexive and transitive on stream systems), proving that $\sigma \otimes [1] \sim \sigma$.

On a final coalgebra, bisimilarity implies equality, so bisimulation up to bisimilarity is not interesting there.

Bisimulation up to union

Given a fixed relation S, we define $\mathsf{un}_S \colon \mathcal{P}(X \times X) \to \mathcal{P}(X \times X)$ by

$$\mathsf{un}_S(R) = R \cup S \,.$$

A bisimulation up to un_S, is called a *bisimulation up to union with* S or *bisimulation up to* S*-union*. If R is a bisimulation up to union with S, then next states are related either by R or by S. This technique is useful in combination with other ones, such as the equivalence closure eq. For instance, any bisimulation up to bisimilarity is also a bisimulation up to $\mathsf{eq} \circ \mathsf{un}_\sim$.

Bisimulation up to context

If the state space of the coalgebra under consideration has algebraic structure, then the notion of bisimulation up to context becomes relevant. Let $T \colon \mathsf{Set} \to \mathsf{Set}$ be a functor. For a T-algebra (X, α), the *contextual closure* function $\mathsf{ctx}_\alpha \colon \mathcal{P}(X \times X) \to \mathcal{P}(X \times X)$ is defined using relation lifting (Section 3.2.1):

$$\mathsf{ctx}_\alpha(R) = (\alpha \times \alpha)(\mathsf{Rel}(T)(R)) = \{(\alpha \circ T\pi_1(t), \alpha \circ T\pi_2(t)) \mid t \in TR\} \,. \tag{4.1}$$

We call $\mathsf{ctx}_\alpha(R)$ the *contextual closure* of R. Whenever α is clear from the context we simply write $\mathsf{ctx}(R)$. If R is a bisimulation up to ctx then we call R a *bisimulation up to context*. In many of the examples, T is the underlying functor of a monad, and α is an algebra for the monad. However, the above definition does not require this: α is simply an algebra for the functor T.

Example 4.2.4. Let Σ^* be the free monad for a polynomial functor representing a signature, with multiplication $\mu\colon \Sigma^*\Sigma^* \Rightarrow \Sigma^*$ (Section 3.4). Given a relation $R \subseteq \Sigma^* X \times \Sigma^* X$, the contextual closure $\mathsf{ctx}_{\mu_X}(R) \subseteq \Sigma^* X \times \Sigma^* X$ can be inductively characterized by the following rules:

$$\frac{s\ R\ t}{s\ \mathsf{ctx}(R)\ t} \qquad \frac{s_i\ \mathsf{ctx}(R)\ t_i \quad i = 1 \dots n}{\sigma(s_1, \dots, s_n)\ \mathsf{ctx}(R)\ \sigma(t_1, \dots, t_n)} \quad \text{for each } \sigma \in \Sigma, |\sigma| = n$$

This slightly differs from the definition in [PS12] where the contextual closure is defined as

$$\mathsf{ctx}'(R) = \{(C[s_1, \dots s_n], C[t_1, \dots t_n]) \mid C \text{ a context and for all } i\colon (s_i, t_i) \in R\}$$

(a context C is a term with $n \geq 0$ holes $[\cdot]_i$ in it). In our case, ctx' can be obtained as $\mathsf{ctx} \circ \mathsf{rfl}$, i.e., by precomposing ctx with the reflexive closure function rfl. To see the difference, consider, for instance, the signature which has only a binary operator $+$, and let $R = \{(x, y)\}$. Then the pair $\{(x + x, x + y)\}$ is in $\mathsf{ctx}'(R)$ but not in $\mathsf{ctx}(R)$.

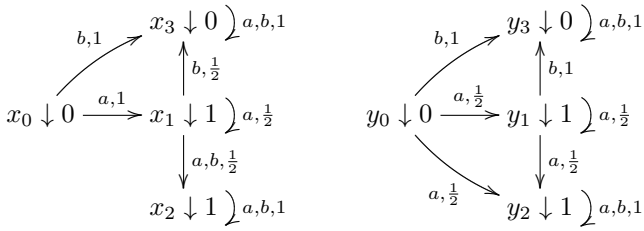
Example 4.2.5. Every weighted automaton $(X, \langle o, t\rangle)$ induces a coalgebra of the form $\langle o^\sharp, t^\sharp\rangle\colon \mathcal{M}X \to \mathbb{R} \times (\mathcal{M}X)^A$, where $\mathcal{M}X$ is the set of linear combinations with coefficients in $\mathbb{R}$. The coinductive extension of $\langle o^\sharp, t^\sharp\rangle$ maps a state x to the weighted language it accepts (Example 3.5.2). Therefore, we can prove weighted language equivalence between states x, y by proving that they are bisimilar on $\langle o^\sharp, t^\sharp\rangle$. In this example, we prove bisimilarity by constructing a bisimulation up to context, thus making use of the algebraic structure on $\mathcal{M}X$.

Given a relation $R \subseteq \mathcal{M}X \times \mathcal{M}X$, its contextual closure $\mathsf{ctx}(R) \subseteq \mathcal{M}X \times \mathcal{M}X$ (where the algebra is given by the multiplication of the monad $\mathcal{M}$, see Example 3.4.1) can be inductively characterized by the following rules:

$$\frac{v\ R\ w}{v\ \mathsf{ctx}(R)\ w} \qquad \frac{-}{0\ \mathsf{ctx}(R)\ 0} \qquad \frac{v_1\ \mathsf{ctx}(R)\ w_1 \quad v_2\ \mathsf{ctx}(R)\ w_2}{v_1 + v_2\ \mathsf{ctx}(R)\ w_1 + w_2} \qquad \frac{v\ \mathsf{ctx}(R)\ w \quad r \in \mathbb{R}}{r \cdot v\ \mathsf{ctx}(R)\ r \cdot w}$$

Now given a weighted automaton $\langle o, t\rangle\colon X \to \mathbb{R} \times (\mathcal{M}X)^A$, a bisimulation up to context is a relation $R \subseteq \mathcal{M}X \times \mathcal{M}X$ such that for all $(v, w) \in R$ we have $o_1^\sharp(v) = o_2^\sharp(w)$ and for all $a \in A$: $(t_1^\sharp(v)(a), t_2^\sharp(w)(a)) \in \mathsf{ctx}(R)$.

As an example, consider the following weighted automaton:



To prove that x_0 and y_0 are language equivalent, we need to prove that they are bisimilar on the induced $\mathbb{R} \times \mathsf{Id}^A$-coalgebra. But a bisimulation containing (x_0, y_0)

has to be infinite, since it needs to contain the pairs shown below by the dashed lines:

$$\begin{array}{ccccccccc}
x_0 \downarrow 0 & \xrightarrow{a} & x_1 \downarrow 1 & \xrightarrow{a} & \frac{1}{2}x_1 + \frac{1}{2}x_2 \downarrow 1 & \xrightarrow{a} & \frac{1}{4}x_1 + \frac{3}{4}x_2 \downarrow 1 & \xrightarrow{a} & \cdots \\
\vdots & & \vdots & & \vdots & & \vdots & & \vdots \\
y_0 \downarrow 0 & \xrightarrow[a]{} & \frac{1}{2}y_1 + \frac{1}{2}y_2 \downarrow 1 & \xrightarrow[a]{} & \frac{1}{4}y_1 + \frac{3}{4}y_2 \downarrow 1 & \xrightarrow[a]{} & \frac{1}{8}y_1 + \frac{7}{8}y_2 \downarrow 1 & \xrightarrow[a]{} & \cdots
\end{array}$$

However, the *finite* relation $R = \{(x_0, y_0), (x_2, y_2), (x_3, y_3), (x_1, \frac{1}{2}y_1 + \frac{1}{2}y_2)\}$ is a bisimulation up to context: consider $(x_1, \frac{1}{2}y_1 + \frac{1}{2}y_2)$ (the other pairs are trivial) and observe that we have the following related pairs:

$$\begin{array}{ccc}
x_1 & \xrightarrow{a} & \frac{1}{2}x_1 + \frac{1}{2}x_2 \\
{\scriptstyle R}\,\vdots & & \vdots\,{\scriptstyle \mathsf{ctx}(R)} \\
\frac{1}{2}y_1 + \frac{1}{2}y_2 & \xrightarrow[a]{} & \frac{1}{4}y_1 + \frac{3}{4}y_2
\end{array}
\qquad
\begin{array}{ccc}
x_1 & \xrightarrow{b} & \frac{1}{2}x_3 + \frac{1}{2}x_2 \\
{\scriptstyle R}\,\vdots & & \vdots\,{\scriptstyle \mathsf{ctx}(R)} \\
\frac{1}{2}y_1 + \frac{1}{2}y_2 & \xrightarrow[b]{} & \frac{1}{2}y_3 + \frac{1}{2}y_2
\end{array}$$

Thus, the finite relation R is a bisimulation up to context. Since this technique is sound (as we will see in Section 4.4), this suffices to prove that x_0 and y_0 are bisimilar, and hence accept the same weighted language.

In the above example we used a finite bisimulation up to context to show weighted language equivalence. Finite bisimulations up to context for weighted automata are used in [Win15] to obtain a decidability result for weighted language equivalence for a certain class of semirings.

Bisimulation up to congruence

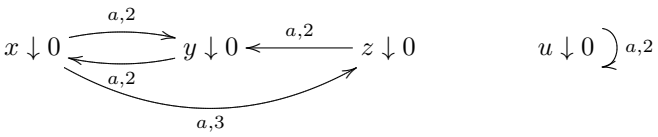
Given a T-algebra $\alpha \colon TX \to X$, the *congruence closure* function $\mathsf{cgr}_\alpha \colon \mathcal{P}(X \times X) \to \mathcal{P}(X \times X)$ is defined by

$$\mathsf{cgr}_\alpha = \bigcup_{i \geq 0} (\mathsf{tra} \cup \mathsf{sym} \cup \mathsf{ctx}_\alpha \cup \mathsf{rfl})^i$$

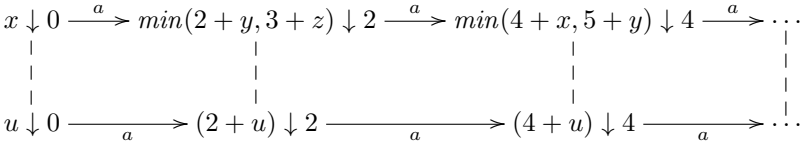
(cf. [BP13]) where $\cup$ is pointwise union. If R is a bisimulation up to cgr_α, then we call R a *bisimulation up to congruence*. The congruence closure and associated notion of bisimulation up to congruence given in Definition 2.3.1 are a special case of the above. (In fact, many of the examples in Chapter 2 do not use the equivalence closure, and therefore are also examples of bisimulations up to context.)

Example 4.2.6. We consider weighted automata for the tropical semiring $\mathbb{T} = (\mathbb{R} \cup \{\infty\}, min, \infty, +, 0)$. In this semiring, the addition operation is given by the function min having ∞ as neutral element. The multiplication is given by the function $+$ having 0 as neutral element.

The weighted automaton $(X, \langle o, t\rangle)$ given as follows:



induces the coalgebra $(\mathcal{M}X, \langle o^\sharp, t^\sharp\rangle)$ which is partially depicted below (the transitions are given by the solid arrows, the dashed lines represent a relation).



The states x and u are weighted language equivalent. To prove it we would need an infinite bisimulation, since it should relate all the pairs of states linked by the dashed lines in the above figure.

Given a relation $R \subseteq \mathcal{M}X \times \mathcal{M}X$, its congruence closure cgr (where the algebra is given by the multiplication of the monad $\mathcal{M}$, see Example 3.4.1) can be characterized inductively by the following rules:

$$\frac{v\ R\ w}{v\ \mathsf{cgr}(R)\ w} \qquad \frac{}{v\ \mathsf{cgr}(R)\ v} \qquad \frac{v\ \mathsf{cgr}(R)\ w}{w\ \mathsf{cgr}(R)\ v} \qquad \frac{u\ \mathsf{cgr}(R)\ v\ \mathsf{cgr}(R)\ w}{u\ \mathsf{cgr}(R)\ w}$$

$$\frac{v_1\ \mathsf{cgr}(R)\ w_1 \quad v_2\ \mathsf{cgr}(R)\ w_2}{min(v_1, v_2)\ \mathsf{cgr}(R)\ min(w_1, w_2)} \qquad \frac{v\ \mathsf{cgr}(R)\ w \quad r \in \mathbb{R} \cup \{\infty\}}{r + v\ \mathsf{cgr}(R)\ r + w}$$

Now consider the relation $R = \{(x, u),\ (min(2 + y, 3 + z), 2 + u)\}$. To prove that R is a bisimulation up to congruence we only have to show that $(min(4 + x, 5 + y), 4 + u) \in \mathsf{cgr}(R)$:

	$min(4 + x, 5 + y)$	
$\mathsf{cgr}(R)$	$min(4 + u, 5 + y)$	$((x, u) \in R)$
$\mathsf{cgr}(R)$	$min(2 + min(2 + y, 3 + z), 5 + y)$	$((min(2 + y, 3 + z), 2 + u) \in R)$
$=$	$2 + min(2 + y, 3 + z)$	
$\mathsf{cgr}(R)$	$4 + u$	$((min(2 + y, 3 + z), 2 + u) \in R)$

Note that R is not a bisimulation up to context, since $(min(4 + x, 5 + y), 4 + u) \notin \mathsf{ctx}(R)$. Here transitivity is really necessary.

Bisimulation up to union, context and equivalence

A bisimulation up to $\mathsf{eq} \circ \mathsf{ctx} \circ \mathsf{un}_S$ is called a *bisimulation up to S-union, context and equivalence*. This extension of bisimulation up to context allows to relate derivatives of R using $\mathsf{ctx}(R \cup S)$ in "multiple steps", similar to the case of up-to-congruence.

Example 4.2.7. Recall the operations of shuffle product and inverse from Section 3.1.1, and let T_{wf} be the set of well-formed terms over shuffle product and inverse introduced there. We prove that the inverse operation is really the inverse of shuffle product, that is, $\sigma \otimes \sigma^{-1} \sim [1]$ for all $\sigma \in T_{wf}(\mathbb{R}^\omega)$ such that $\sigma_0 \neq 0$. We use that $\otimes$ is associative and commutative (so $\sigma \otimes \tau \sim \tau \otimes \sigma$, etc.) and that $\sigma + (-\sigma) \sim [0]$ (see, e.g., [Rut03]). Let

$$R = \{(\sigma \otimes \sigma^{-1}, [1]) \mid \sigma \in T_{wf}(\mathbb{R}^\omega), \sigma_0 \neq 0\}\,.$$

We can now establish that R is a bisimulation up to $\sim$-union, context and equivalence. First we consider the initial values:

$$(\sigma \otimes \sigma^{-1})_0 = \sigma_0 \cdot (\sigma^{-1})_0 = \sigma_0 \cdot (\sigma_0)^{-1} = 1 = [1]_0\,.$$

Next, we relate the derivatives by $\mathsf{eq}(\mathsf{ctx}(R \cup \sim))$:

$$\begin{aligned}(\sigma \otimes \sigma^{-1})' &= \sigma' \otimes \sigma^{-1} + \sigma \otimes (\sigma^{-1})' \\ &= \sigma' \otimes \sigma^{-1} + \sigma \otimes (-\sigma' \otimes (\sigma^{-1} \otimes \sigma^{-1})) \\ &\quad \mathsf{tra}(\mathsf{ctx}(\sim))\ (\sigma' \otimes \sigma^{-1}) + (-(\sigma' \otimes \sigma^{-1}) \otimes (\sigma \otimes \sigma^{-1})) \\ &\quad \mathsf{ctx}(R \cup \sim)\ (\sigma' \otimes \sigma^{-1}) + (-(\sigma' \otimes \sigma^{-1}) \otimes 1) \\ &\quad \mathsf{tra}(\mathsf{ctx}(\sim))\ [0] = [1]'\end{aligned}$$

Since $\mathsf{tra}(\mathsf{ctx}(\sim)) \subseteq \mathsf{eq}(\mathsf{ctx}(R \cup \sim))$ and $\mathsf{ctx}(R \cup \sim) \subseteq \mathsf{eq}(\mathsf{ctx}(R \cup \sim))$ we may conclude that R is a bisimulation up to $\sim$-union, context, and equivalence. Notice that R is not a bisimulation; establishing that it is a bisimulation up-to is much easier than finding a bisimulation which contains R.

In the step above where we use $\mathsf{ctx}(R \cup \sim)$, we could have used $\mathsf{ctx}(\mathsf{rfl}(R))$ instead. Further, since in this example $\sim = \mathsf{tra}(\mathsf{ctx}(\sim))$, the above is also an example of *bisimulation up to context, reflexivity and bisimilarity*, that is, a bisimulation up to $\mathsf{bis} \circ \mathsf{ctx} \circ \mathsf{rfl}$. (Any bisimulation up to context, reflexivity and bisimilarity is also a bisimulation up to $\sim$-union, context and equivalence.)

4.3 Compatible functions

The above examples illustrate various up-to techniques available for bisimilarity. Many of these techniques are combinations of simpler ones; for instance, the equivalence closure is a composition of the transitive, symmetric and reflexive closure, and the congruence closure is a pointwise union of compositions of the transitive, symmetric, contextual and reflexive closure. Unfortunately, the soundness of a composed function does not follow from its basic constituents: the class of sound functions is not closed under composition. It is rather undesirable and sometimes difficult to reprove soundness of every suitable combination from scratch.

This calls for a theory of enhancements which allows one to freely compose them. Such a theory was developed in the setting of classical coinduction (Section 3.2), at the level of complete lattices [Pou07, PS12]. In the current section,

we recall the basic definitions and results of this theory. In the next section, we instantiate it to prove soundness of coalgebraic bisimulation up-to in a modular way. In Section 5.1, the framework is generalized to an abstract categorical setting.

Let f be a monotone function on a complete lattice L. Recall from Section 3.2 that the coinductive proof principle then asserts that, to prove that $x \leq \mathsf{gfp}(f)$, it suffices to prove that $x \leq f(x)$. Enhancements of the coinductive proof method allow one to weaken the requirement that x is an f-invariant: rather than checking $x \leq f(x)$, we would like to check $x \leq f(y)$ for some y which is possibly above x. The key idea consists in using a function g to obtain this larger y out of x: $y = g(x)$. For instance, in the lattice of relations on a fixed set, we often consider functions that add more pairs to the relation.

Definition 4.3.1. Let $f, g \colon L \to L$ be monotone functions.

- An *f-invariant up to g* is an $f \circ g$-invariant, i.e., a post-fixed point of $f \circ g$.
- g is *f-sound* if all f-invariants up to g are below $\mathsf{gfp}(f)$, that is, if $x \leq f(g(x))$ then $x \leq \mathsf{gfp}(f)$.
- g is *f-compatible* if $g \circ f \leq f \circ g$.

The notion of f-compatible function, which is the heart of the matter, is introduced to get around the fact that f-sound functions cannot easily be composed. Compatible functions satisfy two crucial properties: f-compatible functions are f-sound (Theorem 4.3.2) and the composition of two f-compatible functions is again an f-compatible function (Proposition 4.3.3).

Theorem 4.3.2. *All f-compatible functions are f-sound.*

Proof. Let $f, g \colon L \to L$ be monotone and suppose g is f-compatible, i.e., $g \circ f \leq f \circ g$. Let $x \leq f(g(x))$ be an f-invariant up to g; we need to prove that $x \leq \mathsf{gfp}(f)$.

We first show that $g^i(x) \leq f(g^{i+1}(x))$ for every $i \in \mathbb{N}$, by induction on i. The base case $x \leq f(g(x))$ holds by the assumption that x is an f-invariant up to g. Now suppose $g^i(x) \leq f(g^{i+1}(x))$. Since g is monotone, this means $g^{i+1}(x) \leq g(f(g^{i+1}(x)))$, and since g is f-compatible we get

$$g^{i+1}(x) \leq g(f(g^{i+1}(x))) \leq f(g(g^{i+1}(x))) = f(g^{i+2}(x))$$

as desired.

Monotonicity of f gives $g^i(x) \leq f(\bigvee_{i \in \mathbb{N}} g^i(x))$, so $\bigvee_{i \in \mathbb{N}} g^i(x) \leq f(\bigvee_{i \in \mathbb{N}} g^i(x))$, which means $\bigvee_{i \in \mathbb{N}} g^i(x) \leq \mathsf{gfp}(f)$, so $x \leq \bigvee_{i \in \mathbb{N}} g^i(x) \leq \mathsf{gfp}(f)$. □

The main reason for the introduction of compatible functions is that they can be constructed by combining other compatible functions, as stated by the next result.

Proposition 4.3.3. *The following functions on L are f-compatible:*

1. id—*the identity function;*
2. cst_x—*the constant-to-x function, for any f-invariant x;*

3. $g \circ h$ *for any* f*-compatible functions* g *and* h*;*

4. $\bigvee F$ *for any set* F *of* f*-compatible functions.*

In a lattice of relations, the last item states that compatible functions can also be combined using pointwise union. There is another way of combining two functions g and h on relations, using relational composition:

$$(g \bullet h)(R) = g(R) \circ h(R) \tag{4.2}$$

This composition operator does *not* always preserve f-compatibility, but the following lemma gives a sufficient condition.

Proposition 4.3.4. *If* $f \colon \mathcal{P}(X \times X) \to \mathcal{P}(X \times X)$ *satisfies the following condition:*

$$\text{for all relations } R, S \subseteq X \times X : \quad f(R) \circ f(S) \subseteq f(R \circ S) \tag{4.3}$$

then $g \bullet h$ *is* f*-compatible for all* f*-compatible functions* g *and* h*.*

This section is concluded with two lemmas that will be useful in the sequel. The first one gives an alternative characterization of f-compatible functions. The second lemma states that the coinductive predicate defined by f is closed under any f-compatible function.

Lemma 4.3.5. *A monotone function* g *is* f*-compatible iff for all* x, y*:* $x \leq f(y)$ *implies* $g(x) \leq f(g(y))$*.*

Lemma 4.3.6. *If* g *is* f*-compatible then* $g(\mathsf{gfp}(f)) \leq \mathsf{gfp}(f)$*.*

4.4 Compatibility results

We instantiate the framework of the previous section to prove soundness of bisimulation up-to techniques in a modular way, using the notion of compatible functions. To this end, recall from Section 3.2.1 that, given a coalgebra $\delta \colon X \to BX$, one can define the monotone function $\mathsf{b}_\delta(R) = (\delta \times \delta)^{-1}(\mathsf{Rel}(B)(R))$ on the complete lattice of relations on X ordered by inclusion, so that b_δ-invariants are precisely the bisimulations on δ. Progression and bisimulation up-to can also be stated in terms of this function, as an easy extension of Lemma 3.2.3.

Lemma 4.4.1. *For any coalgebra* $\delta \colon X \to BX$ *and for any relations* $R, S \subseteq X \times X$*:* $R \subseteq \mathsf{b}_\delta(S)$ *if and only if* R *progresses to* S*. As a consequence, given any monotone function* $g \colon \mathcal{P}(X \times X) \to \mathcal{P}(X \times X)$ *on the lattice of relations,*

$$R \text{ is a bisimulation up to } g \text{ if and only if it is a } \mathsf{b}_\delta\text{-invariant up to } g\,.$$

Bisimilarity on δ *coincides with the coinductive predicate defined by* b_δ *(i.e.,* $\mathsf{gfp}(\mathsf{b}_\delta)$*).*

Spelling out Definition 4.3.1, a monotone function g is b_δ-compatible if

$$g \circ \mathsf{b}_\delta \subseteq \mathsf{b}_\delta \circ g\,.$$

As a consequence of Lemma 4.4.1 and the fact that compatible functions are sound (Theorem 4.3.2), if g is b_δ-compatible then it is sound in the sense of Definition 4.1.2, i.e., bisimulation up to g is a sound proof technique for bisimilarity. Since compatible functions can be combined in various ways (Proposition 4.3.3), in particular by function composition, the advantage of proving compatibility rather than soundness is that it allows us to compositionally reason about the soundness of bisimulation up-to.

The instances of bisimulation up-to introduced in Section 4.2 can be roughly divided into three groups: (1) simple enhancements like up-to-union, (2) those that involve relational composition, such as up-to-transitivity and up-to-bisimilarity, and finally (3) up-to-context. Derived techniques such as up-to-congruence are just combinations of these basic enhancements, so their compatibility follows from proving the compatibility of their constituents.

In the remainder of this section, we show that functions (1) are compatible for any coalgebra, functions (2) are compatible under a mild condition on the behaviour functor, and functions (3) are compatible in the presence of a λ-bialgebra.

Theorem 4.4.2. *For any B-coalgebra (X, δ), the following are b_δ-compatible:*

1. un_S*—union with S, where S is a bisimulation on δ;*
2. rfl*—the reflexive closure;*
3. sym*—the symmetric closure.*

Proof. By definition, un_S is b_δ-compatible if $\mathsf{un}_S \circ \mathsf{b}_\delta \subseteq \mathsf{b}_\delta \circ \mathsf{un}_S$. Instead of proving this directly, we first decompose un_S as

$$\mathsf{un}_S(R) = R \cup S = \mathsf{id}(R) \cup \mathsf{cst}_S(R)\,.$$

By Proposition 4.3.3, id is b_δ-compatible, and the union of compatible functions is again compatible; so we only need to prove that the constant-to-S function cst_S is b_δ-compatible. Since S is a bisimulation, it is a b_δ-invariant, and thus by Proposition 4.3.3, the constant function cst_S is indeed b_δ-compatible.

For the compatibility of the reflexive closure, we use that the diagonal relation on any coalgebra is a bisimulation [Rut00]. Since $\mathsf{rfl} = \mathsf{un}_{\Delta_X}$, where Δ_X is the diagonal relation on X, rfl is b_δ-compatible by the first item.

Let $\mathsf{inv}(R) = R^{op}$. The symmetric closure sym is given by $\mathsf{sym}(R) = R \cup R^{op} = \mathsf{id}(R) \cup \mathsf{inv}(R)$. Thus, by Proposition 4.3.3, we obtain b_δ-compatibility of sym if we prove that inv is b_δ-compatible, i.e., that $\mathsf{inv} \circ \mathsf{b}_\delta \subseteq \mathsf{b}_\delta \circ \mathsf{inv}$. But this follows easily from the fact that $\mathsf{Rel}(B)(R^{op}) = (\mathsf{Rel}(B)(R))^{op}$ (Lemma 3.2.4). □

4.4.1 Relational composition

Bisimilarity on coalgebras is not a transitive relation, in general. However, the mild condition that the behaviour functor preserves weak pullbacks guarantees that it is [Rut00]. Similarly, up-to techniques that are based on composition, such as bisimulation up to transitivity, are not sound in general. In this section, we show that weak pullback preservation is equivalent to the property (4.3) of Section 4.3. This property implies that the composition operator $\bullet$ from Section 4.3 (Equation (4.2)) preserves compatibility. From this fact, compatibility of the transitive closure and the bisimilarity closure can be derived.

First, we adapt an example from [AM89] to show that bisimulation up to bisimilarity is not sound in general.

Example 4.4.3. Define the functor $B\colon \mathsf{Set} \to \mathsf{Set}$ as

$$\begin{aligned} BX &= \{(x_1, x_2, x_3) \in X^3 \mid |\{x_1, x_2, x_3\}| \leq 2\} \\ B(f)(x_1, x_2, x_3) &= (f(x_1), f(x_2), f(x_3)) \end{aligned}$$

Consider the B-coalgebra with states $X = \{0, 1, 2, \tilde{0}, \tilde{1}\}$ and transition structure

$$\begin{array}{lll} 0 \mapsto (0,1,0) & \tilde{0} \mapsto (0,0,0) & 2 \mapsto (2,2,2) \\ 1 \mapsto (0,0,1) & \tilde{1} \mapsto (1,1,1) & \end{array}$$

Then $0 \not\sim 1$. To see this, note that in order for the pair $(0,1)$ to be contained in a bisimulation R, there must be a transition structure on this relation which maps $(0,1)$ to $((0,0),(1,0),(0,1))$. But this triple can not be in BR, because it consists of three different elements. However, it is easy to show that $0 \sim 2$ and $1 \sim 2$: the relation $\{(0,2),(1,2)\}$ is a bisimulation.

The relation $S = \{(\tilde{0}, \tilde{1}), (2,2)\}$ is not a bisimulation, since for that there should be a function from S to BS mapping the elements as follows:

$$(\tilde{0}, \tilde{1}) \mapsto ((0,1),(0,1),(0,1)) \qquad (2,2) \mapsto ((2,2),(2,2),(2,2))$$

and $((0,1),(0,1),(0,1))$ is not contained in BS. However, since $0 \sim 2\ S\ 2 \sim 1$, the triple $((0,1),(0,1),(0,1))$ *is* contained in $B(\sim \circ S \circ \sim)$; so S is a bisimulation up to bisimilarity. Thus, if up-to-bisimilarity is sound, then $S \subseteq \sim$ and consequently $\tilde{0} \sim \tilde{1}$. It follows that $0 \sim 1$, which is a contradiction.

The key to obtaining b_δ-compatibility of functions that involve relational composition, is to assume that the behaviour functor B preserves weak pullbacks. Recall that a functor $B\colon \mathsf{Set} \to \mathsf{Set}$ preserves weak pullbacks if and only if $\mathsf{Rel}(B)$ preserves composition of relations (Theorem 3.2.5). A further equivalent condition is that bisimulations are closed under composition.

Theorem 4.4.4. *A functor $B\colon \mathsf{Set} \to \mathsf{Set}$ preserves weak pullbacks if and only if the composition of two B-bisimulations is again a B-bisimulation.*

Rutten [Rut00] established the implication from left to right, and the reverse implication is due to Gumm and Schröder [GS00]. Using Theorem 3.2.5 and Theorem 4.4.4 we show that preservation of weak pullbacks coincides with the property (4.3) of Section 4.3.

Proposition 4.4.5. *B preserves weak pullbacks iff for any B-coalgebra (X, δ), b_δ satisfies* (4.3)*, i.e., for all relations R, S: $\mathsf{b}_\delta(R) \circ \mathsf{b}_\delta(S) \subseteq \mathsf{b}_\delta(R \circ S)$.*

Proof. Suppose B preserves weak pullbacks. Let (X, δ) be an B-coalgebra, $R, S \subseteq X \times X$ relations, and $(x, z) \in \mathsf{b}_\delta(R) \circ \mathsf{b}_\delta(S)$, so there is some y such that $(x, y) \in \mathsf{b}_\delta(R)$ and $(y, z) \in \mathsf{b}_\delta(S)$. Then we have $(\delta(x), \delta(y)) \in \mathsf{Rel}(B)(R)$ and $(\delta(y), \delta(z)) \in \mathsf{Rel}(B)(S)$, so $(\delta(x), \delta(z)) \in \mathsf{Rel}(B)(R) \circ \mathsf{Rel}(B)(S)$. But by assumption and Theorem 3.2.5 $\mathsf{Rel}(B)$ preserves composition, so $\mathsf{Rel}(B)(R) \circ \mathsf{Rel}(B)(S) = \mathsf{Rel}(B)(R \circ S)$. Consequently $(x, z) \in \mathsf{b}_\delta(R \circ S)$ as desired.

Conversely, suppose that (4.3) holds; then by Proposition 4.3.4, b_δ-compatible functions are closed under $\bullet$. Let R, S be bisimulations, so the constant-to-R function cst_R and the constant-to-S function cst_S are both b_δ-compatible by Proposition 4.3.3. By assumption $\mathsf{cst}_R \bullet \mathsf{cst}_S$ is b_δ-compatible, so by Lemma 4.3.5 we have $R \circ S \subseteq \mathsf{b}_\delta(R \circ S)$, and thus $R \circ S$ is a bisimulation. From Theorem 4.4.4 we conclude that B preserves weak pullbacks. (In fact, we only considered bisimulations on a single coalgebra, whereas the condition 2 of the theorem mentions arbitrary bisimulations; however, it is easy to prove that, in Set, if bisimulations on a single coalgebra compose then bisimulations on different coalgebras compose as well [RBB+15]). □

As a consequence of Proposition 4.3.4 and the above result, b-compatible functions are closed under $\bullet$ if the behaviour functor preserves weak pullbacks.

Theorem 4.4.6. *Let (X, δ) be a coalgebra for a functor B that preserves weak pullbacks. The following functions are b_δ-compatible:*

1. tra*—the transitive closure;*
2. eq*—the equivalence closure;*
3. bis_δ*—the bisimilarity closure.*

Proof. If B preserves weak pullbacks, then b_δ-compatible functions are closed under $\bullet$, by Proposition 4.4.5 and Proposition 4.3.4.

For tra, inductively define the functions $(-)^{\bullet n}$ as $(-)^{\bullet 1} = \mathsf{id}$ and $(-)^{\bullet n+1} = \mathsf{id} \bullet (-)^{\bullet n}$. We thus have $(R)^{\bullet 1} = R$ and $(R)^{\bullet n+1} = R \circ R^{\bullet n}$. We prove by induction on n that $(-)^{\bullet n}$ is b_δ-compatible for any $n \in \mathbb{N}$. The base case is b_δ-compatibility of id, which follows from Proposition 4.3.3. Further, if $(-)^{\bullet n}$ is compatible then $(-)^{\bullet n+1} = \mathsf{id} \bullet (-)^{\bullet n}$ is also compatible. Thus

$$\mathsf{tra} = \bigcup_{n \geq 1} (-)^{\bullet n}$$

is a union of b_δ-compatible functions, so by Proposition 4.3.3 it is b_δ-compatible.

The equivalence closure is $\mathsf{eq} = \mathsf{tra} \circ \mathsf{sym} \circ \mathsf{rfl}$, which is a composition of b_δ-compatible functions and therefore b_δ-compatible.

For the bisimilarity closure bis_δ we have

$$\mathsf{bis}_\delta(R) = \sim \circ R \circ \sim = \mathsf{cst}_\sim \bullet \mathsf{id} \bullet \mathsf{cst}_\sim .$$

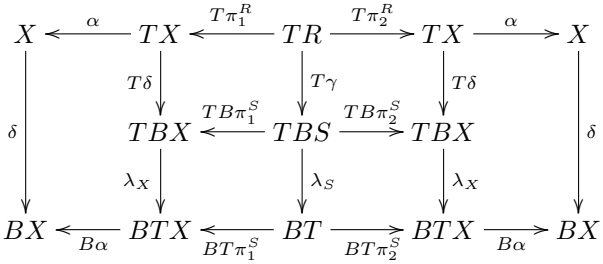
Since $\sim$ is a bisimulation, $\mathsf{cst}_\sim$ is b_δ-compatible. The b_δ-compatibility of bis_δ follows since b_δ-compatible functions are closed under $\bullet$, using the assumption. □

4.4.2 Contextual closure

The contextual closure ctx_α is defined with respect to a T-algebra $\alpha\colon TX \to X$ on the states of a coalgebra $\delta\colon X \to BX$, see (4.1) in Section 4.2. A first thought may be that for compatibility of the contextual closure, it suffices if bisimilarity is a congruence with respect to this algebra, i.e., that bisimilarity is closed under the algebra structure. However, this is not even enough for the soundness of bisimulation up to context [PS12]. As we show below, in order to prove that ctx is compatible, it is sufficient to assume that (X, α, δ) is a *λ-bialgebra* for a distributive law $\lambda\colon TB \Rightarrow BT$ of the functor T over the functor B (thus, λ is simply a natural transformation).

Theorem 4.4.7. *Let (X, α, δ) be a λ-bialgebra for a distributive law $\lambda\colon TB \Rightarrow BT$ of T over B. The contextual closure function ctx_α is b_δ-compatible.*

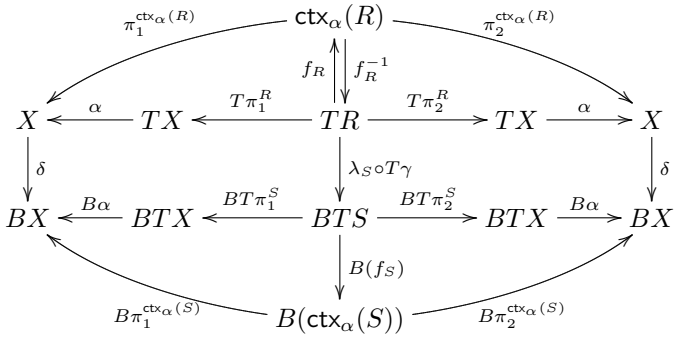
Proof. Suppose $R \subseteq \mathsf{b}_\delta(S)$ for some R and S. We prove that $\mathsf{ctx}_\alpha(R) \subseteq \mathsf{b}_\delta(\mathsf{ctx}_\alpha(S))$; by Lemma 4.3.5 this implies that ctx_α is b_δ-compatible. Consider the following diagram:



The existence of γ and commutativity of the upper squares follow since $R \subseteq \mathsf{b}_\delta(S)$, by Lemma 4.4.1. The lower squares commute by naturality. The (outer) rectangles commute since (X, α, δ) is a λ-bialgebra.

We show that the above argument implies that $\mathsf{ctx}_\alpha(R)$ progresses to $\mathsf{ctx}_\alpha(S)$. Let $f_R\colon TR \to \mathsf{ctx}_\alpha(R)$ be the corestriction of $\langle \alpha \circ T\pi_1^R, \alpha \circ T\pi_2^R \rangle\colon TR \to X \times X$ to its range, so that $f_R(TR) = \mathsf{ctx}_\alpha(R)$. Let $f_S\colon TS \to \mathsf{ctx}_\alpha(S)$ be defined analogously, and take f_R^{-1} to be any right inverse of f_R (so we use the axiom of choice). Then

the following diagram commutes:



This means that $\mathsf{ctx}_\alpha(R)$ progresses to $\mathsf{ctx}_\alpha(S)$, and thus $\mathsf{ctx}_\alpha(R) \subseteq \mathsf{b}_\alpha(\mathsf{ctx}_\alpha(S))$ by Lemma 4.4.1. □

Remark 4.4.8. The greatest bisimulation on a λ-bialgebra is closed under the algebraic operations. This was first shown by Turi and Plotkin [TP97] under the assumption that B preserves weak pullbacks; Bartels [Bar04] showed that this assumption is not necessary. We obtain the same result (for Set functors) as a direct consequence of the above Theorem and Lemma 4.3.6.

Under the assumption of a behaviour functor that preserves weak pullbacks and a λ-bialgebra, the *congruence closure* cgr_α is compatible as well, since it is a union of (compositions of) rfl, tra, sym and ctx_α, and each of these is compatible by Theorems 4.4.2, 4.4.6 and 4.4.7.

Coalgebras for copointed functors. There are many interesting examples of λ-bialgebras of the form $(X, \alpha, \langle \delta, \mathsf{id} \rangle)$, for some $\lambda\colon T(B \times \mathsf{Id}) \Rightarrow BT \times T$; in particular, this is relevant when λ arises from an abstract GSOS specification (Section 3.5). However, while Theorem 4.4.7 gives us $\mathsf{b}_{\langle \delta, \mathsf{id} \rangle}$-compatibility of the contextual closure ctx_α, it does *not* provide b_δ-compatibility. We recall a counterexample from [PS12].

Example 4.4.9 ([PS12])**.** Consider the following specification of the prefix and the replication operation on labelled transition systems:

$$\frac{}{a.x \xrightarrow{a} x} \qquad \frac{x \xrightarrow{a} x'}{!x \xrightarrow{a} !x|x'}$$

together with the standard definition of the parallel operator $x|y$ (Example 3.5.4), and the constant 0, which has no transitions. This specification is in the GSOS format. While this is arguably not the best way to specify replication in the context of CCS [PS12], it suffices for our purposes. This specification induces a coalgebra on closed terms. Now abbreviate $b.0$ and $c.0$ by b and c respectively, and consider the relations $R = \{(!a.b, !a.c)\}$ and $S = \{(!a.b|b, !a.c|c)\}$. Then R progresses to S,

but $\mathsf{ctx}(R)$ does not progress to $\mathsf{ctx}(S)$. For example, $(a.!a.b, a.!a.c) \in \mathsf{ctx}(R)$ but $!a.b$ is not related to $!a.c$ by $\mathsf{ctx}(S)$. Thus, by Lemma 4.3.5 the contextual closure ctx is not b_δ-compatible.

The solution of [PS12] is to consider invariants for a different function b'_δ, defined as $\mathsf{b}'_\delta(R) = \mathsf{b}_\delta(R) \cap R$. But $\mathsf{b}'_\delta = \mathsf{b}_{\langle \delta, \mathsf{id} \rangle}$ (an exercise in relation lifting), so in our framework this function arises naturally from the fact that one needs to consider a coalgebra for the cofree copointed functor in order to obtain compatibility.

In terms of progressions, we have $R \subseteq \mathsf{b}'_\delta(S)$ if and only if R progresses to S and $R \subseteq S$. Thus if R progresses to $g(R)$ for a function satisfying $R \subseteq g(R)$, then $R \subseteq \mathsf{b}'_\delta(g(R))$. But notice that for most functions g considered in Theorem 4.4.2 and Theorem 4.4.6 we have $R \subseteq g(R)$; an exception is the constant-to function. For the contextual closure function it suffices to assume that the functor T is *pointed*, i.e., there is a natural transformation $\eta \colon \mathsf{Id} \Rightarrow T$, and α is an algebra for this pointed functor, meaning that $\alpha \circ \eta = \mathsf{id}$. This holds in particular when α is an algebra for a monad (T, η, μ).

4.4.3 Bisimulation up-to modulo bisimilarity

We investigate the situation that there are two coalgebras on a common carrier, that behave the same up to bisimilarity. It turns out that in this case, if the functor preserves weak pullbacks, compatibility of a function g on one coalgebra can be transferred to compatibility of $\mathsf{bis} \circ g \circ \mathsf{bis}$ on the other. This rather technical result is only applied in Chapter 6, and does not play a further role in the current chapter. It was presented in [RB15].

Definition 4.4.10. Let δ, ϑ be B-coalgebras on a common carrier. We say δ and ϑ are *equal up to bisimilarity* if the bisimilarity relation $\sim_{\delta,\vartheta}$ between δ and ϑ is reflexive.

If B preserves weak pullbacks, then an equivalent definition is that the identity relation Δ is a bisimulation up to bisimilarity.

Lemma 4.4.11. *Let $\delta, \vartheta \colon X \to BX$ be coalgebras that are equal up to bisimilarity and assume that B preserves weak pullbacks. Then $\sim_\delta = \sim_{\delta,\vartheta} = \sim_\vartheta$.*

Proof. By assumption $\sim_{\delta,\vartheta}$ is reflexive, and by Theorem 4.4.4 the composition of two bisimulations is again a bisimulation. The desired equalities are now easy to prove; for example, $\sim_\delta \subseteq \sim_\delta \circ \sim_{\delta,\vartheta}$ by reflexivity of $\sim_{\delta,\vartheta}$, and $\sim_\delta \circ \sim_{\delta,\vartheta} \subseteq \sim_{\delta,\vartheta}$ since $\sim_\delta \circ \sim_{\delta,\vartheta}$ is a bisimulation between δ and ϑ and therefore contained in $\sim_{\delta,\vartheta}$, the greatest such bisimulation. Conversely, $\sim_{\delta,\vartheta} \subseteq \sim_{\delta,\vartheta} \circ \sim_{\vartheta,\delta} \subseteq \sim_\delta$ by a similar argument. □

Lemma 4.4.12. *Let B, δ and ϑ be as in Lemma 4.4.11.*

1. *If $R \subseteq \mathsf{b}_\delta(S)$ then $\mathsf{bis}(R) \subseteq \mathsf{b}_\vartheta(\mathsf{bis}(S))$.*
2. *If g is b_δ-compatible then $\mathsf{bis} \circ g \circ \mathsf{bis}$ is b_ϑ-compatible.*

where bis *is defined w.r.t. the bisimilarity relation* $\sim$ *(of both* δ *and* ϑ*).*

Proof. Suppose $R \subseteq \mathsf{b}_\delta(S)$, and let $(x, y) \in R$; then $\delta(x)\,\mathsf{Rel}(B)(S)\,\delta(y)$. Since δ and ϑ are equal up to bisimilarity, we have $\vartheta(x)\,\mathsf{Rel}(B)(\sim)\,\delta(x)$ and $\delta(y)\,\mathsf{Rel}(B)(\sim)\,\vartheta(y)$. Hence

$$\vartheta(x)\,\mathsf{Rel}(B)(\sim)\,\delta(x)\,\mathsf{Rel}(B)(S)\,\delta(y)\,\mathsf{Rel}(B)(\sim)\,\vartheta(y)$$

and since B preserves weak pullbacks, this implies $\vartheta(x)\,\mathsf{Rel}(B)(\sim \circ S \circ \sim)\,\vartheta(y)$ (Theorem 3.2.5). Thus $R \subseteq \mathsf{b}_\vartheta(\sim \circ S \circ \sim)$; by compatibility of bis and Lemma 4.3.5 this implies $\sim \circ R \circ \sim \subseteq \mathsf{b}_\vartheta(\sim \circ \sim \circ S \circ \sim \circ \sim)$, and by transitivity of $\sim$ (B preserves weak pullbacks) then $\mathsf{bis}(R) \subseteq \mathsf{b}_\vartheta(\mathsf{bis}(S))$.

For (2), suppose $R \subseteq \mathsf{b}_\vartheta(S)$. By (1) (replacing δ by ϑ and vice versa) then $\mathsf{bis}(R) \subseteq \mathsf{b}_\delta(\mathsf{bis}(S))$. We apply b_δ-compatibility of g to obtain $g \circ \mathsf{bis}(R) \subseteq \mathsf{b}_\delta(g \circ \mathsf{bis}(S))$. Finally, again apply (1) and get $\mathsf{bis} \circ g \circ \mathsf{bis}(R) \subseteq \mathsf{b}_\vartheta(\mathsf{bis} \circ g \circ \mathsf{bis}(S))$. □

4.5 Behavioural equivalence up-to

Whenever the functor B does not preserve weak pullbacks (as it is the case, for instance, with certain types of weighted transition systems [GS01, Kli09, BBB+12]) one can consider *behavioural equivalence*, rather than bisimilarity. In the current section, we instantiate the framework of Section 4.3 to develop up-to techniques for behavioural equivalence.

Recall from Section 3.1 that behavioural equivalence $\approx$ on a coalgebra $\delta\colon X \to BX$ is defined as follows: $x \approx y$ iff there is a homomorphism h from (X, δ) into some coalgebra such that $h(x) = h(y)$. As we see below (Lemma 4.5.1), behavioural equivalence $\approx$ can equivalently be characterized as the greatest fixed point of the monotone function $\mathsf{be}_\delta\colon \mathcal{P}(X \times X) \to \mathcal{P}(X \times X)$ on the lattice of relations on X, defined as follows [AM89]:

$$\mathsf{be}_\delta(R) = \{(x, y) \mid Bq_R \circ \delta(x) = Bq_R \circ \delta(y)\}$$

where $q_R\colon X \to X/\mathsf{eq}(R)$ is the quotient map of $\mathsf{eq}(R)$ (we sometimes drop the subscript δ from be_δ if it is clear from the context).

Lemma 4.5.1. *Let* $\approx$ *be behavioural equivalence on a coalgebra* $\delta\colon X \to BX$*. Then* $x \approx y$ *if and only if there is a relation* R *such that* $R \subseteq \mathsf{be}_\delta(R)$ *and* $(x, y) \in R$*.*

Proof. The quotient map q_R from the definition of $\mathsf{be}_\delta(R)$ is a coequalizer, and therefore a coalgebra morphism [Rut00, Theorem 4.2], which gives the implication from right to left. For the converse, we let h be a coalgebra morphism from δ and we prove that the kernel $\mathsf{ker}(h) = \{(x, y) \mid h(x) = h(y)\}$ of h is a be_δ-invariant, i.e., we show that the following inclusion holds:

$$\mathsf{ker}(h) \subseteq \mathsf{be}_\delta(\mathsf{ker}(h)) = \{(x, y) \mid Bq \circ \delta(x) = Bq \circ \delta(y)\}$$

where $q\colon X \to X/\mathsf{ker}(h)$ is the quotient map of (the equivalence relation) $\mathsf{ker}(h)$. By [Rut00, Theorem 7.1], h equals the composition of coalgebra homomorphisms

$h = m \circ q$ where q is as above and m is a monomorphism. This means that $q(x) = q(y)$ for any $(x, y) \in \mathsf{ker}(h)$, and since q is a coalgebra morphism from δ, we get $Bq \circ \delta(x) = Bq \circ \delta(y)$. Thus $\mathsf{ker}(h) \subseteq \mathsf{be}_\delta(\mathsf{ker}(h))$. □

The relation R of Example 4.2.2 is a be_δ-invariant. Note that, intuitively, be_δ-invariants are implicitly "up to equivalence", since the next states can be related by the equivalence closure $\mathsf{eq}(R)$.

We proceed to consider be-compatibility of the equivalence closure and contextual closure. In the previous section, we used the property (4.3) from Section 4.3 to prove b-compatibility of transitive and equivalence closure. However, this property does *not* hold for be, that is, in general it does not hold that $\mathsf{be}(R) \circ \mathsf{be}(S) \subseteq \mathsf{be}(R \circ S)$. This is shown by the following example.

Example 4.5.2. Consider the identity functor $BX = X$ and the B-coalgebra with states $\{x, y\}$ and transitions $x \mapsto x$ and $y \mapsto y$. Let $R = \{(x, y)\}$. Then $\mathsf{be}(R) = \{(x, x), (y, y), (x, y), (y, x)\}$ and $\mathsf{be}(\emptyset) = \{(x, x), (y, y)\}$. Now

$$\mathsf{be}(R) \circ \mathsf{be}(\emptyset) = \{(x, x), (y, y), (x, y), (y, x)\}\,, \text{whereas}$$
$$\mathsf{be}(R \circ \emptyset) = \mathsf{be}(\emptyset) = \{(x, x), (y, y)\}\,.$$

Indeed, $\mathsf{be}(R) \circ \mathsf{be}(\emptyset)$ is not included in $\mathsf{be}(R \circ \emptyset)$.

This motivates to prove be-compatibility of eq directly.

Theorem 4.5.3. *Let (X, δ) be any coalgebra. The following are be_δ-compatible:*

1. rfl*—the reflexive closure;*
2. eq*—the equivalence closure;*
3. un_S*—union with S (for a behavioural equivalence S).*

Proof. Items 1 and 3 are analogous to Theorem 4.4.2. We proceed with the compatibility of the equivalence closure. First, notice that $\mathsf{eq} \circ \mathsf{be} = \mathsf{be}$ since $\mathsf{be}(R)$ is an equivalence relation for any relation R. Second, since $\mathsf{eq}(R) = \mathsf{eq}(\mathsf{eq}(R))$ for any R, the quotient maps in the definition of $\mathsf{be}(R)$ and $\mathsf{be}(\mathsf{eq}(R))$ are equal, so $\mathsf{be}(R) = \mathsf{be}(\mathsf{eq}(R))$. Thus $\mathsf{eq} \circ \mathsf{be} = \mathsf{be} = \mathsf{be} \circ \mathsf{eq}$. □

Notice that the be-compatibility of the equivalence closure does not require any assumptions on the functor.

For the compatibility of contextual closure a λ-bialgebra is required, similar to the case of bisimulations in Theorem 4.4.7. However, in the case of behavioural equivalence, we require an algebra for a *monad*, although λ is still only required to be a distributive law between functors, that is, a plain natural transformation. Further, in the proof we need an additional assumption. A pair of functions $f, g\colon X \to Y$ is *reflexive* if it has a common section: a map $s\colon Y \to X$ such that $f \circ s = \mathsf{id} = g \circ s$. A *reflexive coequalizer* is a coequalizer of a reflexive pair. Reflexive coequalizers are important in the theory of monads, see, e.g., [BW05]. Below

we need the underlying functor T of the monad to preserve reflexive coequalizers, which is a non-trivial condition in Set; see [AKV00, Example 4.3] for an example of a functor that does not satisfy this property. We do not know if these additional assumptions can be dropped.

Theorem 4.5.4. *Let (T, η, μ) be a monad so that T preserves reflexive coequalizers, and let (X, α, δ) be a λ-bialgebra for a distributive law $\lambda\colon TB \Rightarrow BT$ (between functors), where α is an algebra for the monad (T, η, μ). Then $\mathsf{ctx}_\alpha \circ \mathsf{rfl}$ is be_δ-compatible.*

Proof. Suppose $R \subseteq \mathsf{be}_\delta(S)$ for some relations $R, S \subseteq X \times X$. By Theorem 4.5.3 rfl is be_δ-compatible, so $\mathsf{rfl}(R) \subseteq \mathsf{be}_\delta \circ \mathsf{rfl}(S)$. Further $\mathsf{rfl}(S) \subseteq \mathsf{ctx}_\alpha \circ \mathsf{rfl}(S)$, using the fact that α is an algebra for the monad (see the last part of Section 4.4.2). Therefore

$$\mathsf{rfl}(R) \subseteq \mathsf{be}_\delta \circ \mathsf{ctx}_\alpha \circ \mathsf{rfl}(S)\,. \tag{4.4}$$

Let $q\colon X \to X'$ be the quotient map of $\mathsf{eq} \circ \mathsf{ctx}_\alpha \circ \mathsf{rfl}(S)$ and its projections, or, equivalently, the coequalizer of the two composite arrows $\alpha \circ T\pi_1$, $\alpha \circ T\pi_2$ in the bottom of the diagram below:

$$\begin{array}{ccccccc}
TT(\mathsf{rfl}(S)) & \xrightarrow{TT\pi_1, TT\pi_2} & TTX & \xrightarrow{T\alpha} & TX & \xrightarrow{Tq} & TX' \\
\downarrow{\scriptstyle \mu_{\mathsf{rfl}(S)}} & & \downarrow{\scriptstyle \mu_X} & & \downarrow{\scriptstyle \alpha} & & \downarrow{\scriptstyle \alpha'} \\
T(\mathsf{rfl}(S)) & \xrightarrow[T\pi_1, T\pi_2]{} & TX & \xrightarrow[\alpha]{} & X & \xrightarrow[q]{} & X'
\end{array} \tag{4.5}$$

Define $d\colon X \to \mathsf{rfl}(S)$ by $d(x) = (x, x)$. Then the map $Td \circ \eta_X\colon X \to T(\mathsf{rfl}(S))$ is a section of the pair $\alpha \circ T\pi_1$, $\alpha \circ T\pi_2$, since $\alpha \circ T\pi_1 \circ Td \circ \eta_X = \alpha \circ \eta_X = \alpha \circ T\pi_2 \circ Td \circ \eta_X$ and $\alpha \circ \eta_X = \mathsf{id}$. Thus, $\alpha \circ T\pi_1$, $\alpha \circ T\pi_2$ is a reflexive pair, and q a reflexive coequalizer. The square on the left commutes (for $T\pi_1$ and $T\pi_2$ separately) by naturality, and the middle since α is an algebra for the monad. Since T preserves reflexive coequalizers, Tq is a coequalizer, and the map α' making the right-hand square commute arises by its universal property.

Now consider the following diagram:

$$\begin{array}{ccccccc}
T(\mathsf{rfl}(R)) & \overset{T\pi_1}{\underset{T\pi_2}{\rightrightarrows}} & TX & \xrightarrow{T\delta} & TBX & \xrightarrow{TBq} & TBX' \\
 & & \downarrow{\scriptstyle \alpha} & & \downarrow{\scriptstyle \lambda_X} & & \downarrow{\scriptstyle \lambda_{X'}} \\
 & & & & BTX & \xrightarrow{BTq} & BTX' \\
 & & & & \downarrow{\scriptstyle B\alpha} & & \downarrow{\scriptstyle B\alpha'} \\
 & & X & \xrightarrow[\delta]{} & BX & \xrightarrow[Bq]{} & BX'
\end{array}$$

The top horizontal paths commute by (4.4) and functoriality. The rectangle commutes by the assumption that (X, α, δ) is a λ-bialgebra. The upper square commutes by naturality of λ, and the lower square by (4.5) and functoriality. Thus we

have $Bq \circ \delta \circ \alpha \circ T\pi_1 = Bq \circ \delta \circ \alpha \circ T\pi_2$, and consequently

$$\mathsf{ctx}_\alpha(\mathsf{rfl}(R)) \overset{\pi_1}{\underset{\pi_2}{\rightrightarrows}} X \xrightarrow{\delta} BX \xrightarrow{Bq} BX'$$

commutes, which means $\mathsf{ctx}_\alpha \circ \mathsf{rfl}(R) \subseteq \mathsf{be}_\delta \circ \mathsf{ctx}_\alpha \circ \mathsf{rfl}(S)$. By Lemma 4.3.5, $\mathsf{ctx}_\alpha \circ \mathsf{rfl}$ is be_δ-compatible. □

The above result also applies to coalgebras of the form $\langle \delta, \mathsf{id} \rangle$, similar to the situation described for b_δ-compatibility in Section 4.4.2.

Example 4.5.5. For an example of behavioural equivalence up-to, we consider the *general process algebra with transition costs* (GPA) from [BK01]. GPA processes are defined for a given set of labels A and a semiring $\mathbb{S}$ which, for this example, we fix to be the semiring of reals $\mathbb{R}$ with the usual addition and multiplication. The operational semantics of GPA is expressed in terms of weighted transition systems, that is, coalgebras for the functor $(\mathcal{M}-)^A$ (Example 3.1.1).

As shown in Section 2.3 of [BBB$^+$12], the functor $(\mathcal{M}-)^A$ does not preserve weak pullbacks and therefore bisimulation up-to cannot be used in this context. However, thanks to Theorem 4.5.3 we can use behavioural equivalence up-to.

First observe that, by instantiating the definition of be above to a coalgebra $\delta\colon X \to (\mathcal{M}X)^A$, one obtains the function $\mathsf{be}_\delta\colon \mathcal{P}(X \times X) \to \mathcal{P}(X \times X)$ defined for a relation $R \subseteq X \times X$ as

$$\mathsf{be}_\delta(R) = \{(x_1, x_2) \mid \forall a \in A, y \in X : \sum_{y' \in [y]_R} \delta(x_1)(a)(y') = \sum_{y' \in [y]_R} \delta(x_2)(a)(y')\}$$

where $[y]_R$ denotes the equivalence class of y with respect to $\mathsf{eq}(R)$. Our notion of behavioural equivalence coincides with the notion of bisimilarity in [BK01] (which differs from coalgebraic bisimilarity).

To illustrate our example it suffices to consider a small fragment of GPA. The set P of *basic GPA processes* is defined by

$$p ::= 0 \mid p + p \mid (a, r).p$$

where $a \in A$, $r \in \mathbb{R}$. The operational semantics of basic GPA processes is given by the coalgebra $\delta\colon P \to (\mathcal{M}P)^A$ defined for all $a' \in A$ and $p' \in P$ as follows:

$$\begin{aligned} \delta(0)(a')(p') &= 0 \\ \delta((a,r).p)(a')(p') &= \begin{cases} r & \text{if } a = a', p = p' \\ 0 & \text{otherwise} \end{cases} \\ \delta(p_1 + p_2)(a')(p') &= \delta(p_1)(a')(p') + \delta(p_2)(a')(p') \end{aligned}$$

As an example, the operational semantics of $(a,1).0 + (a,-1).(a,0).0$ is as follows.

$$\begin{array}{ccc} & \overset{a,1}{\nearrow} & 0 \\ (a,1).0 + (a,-1).(a,0).0 & & \\ & \underset{a,-1}{\searrow} & \\ & & (a,0).0 \end{array} \tag{4.6}$$

Since $0 \approx (a,0).0$, we have that $(a,1).0 + (a,-1).(a,0).0 \approx 0$. More generally, it holds that for all $a \in A$, $r \in R$, p_1 and $p_2 \in P$:

$$\text{if } p_1 \approx p_2 \text{ then } 0 \approx (a,r).p_1 + (a,-r).p_2. \tag{4.7}$$

We prove (4.7) using behavioural equivalence up to union with $\approx$ (Theorem 4.5.3). To this end, consider the relation

$$R = \{(0, (a,r).p_1 + (a,-r).p_2) \mid p_1 \approx p_2\}.$$

Note that R is *not* a be_δ-invariant. For instance, 0 does not make any transitions whereas $(a,1).0 + (a,-1).(a,0).0$ makes two transitions, to processes that are not in the same equivalence class with respect to $\mathsf{eq}(R)$ (see (4.6)); thus $R \not\subseteq \mathsf{be}_\delta(R)$.

Instead, we prove that R is a be_δ-invariant up to $\mathsf{un}_\approx$, that is, $R \subseteq \mathsf{be}_\delta(R \cup \approx)$. We must show that for any $p = (a,r).p_1 + (a,-r).p_2$ and any process $q \in P$:

$$\sum_{y' \in [q]_{R \cup \approx}} \delta(0)(a)(y') = 0 = \sum_{y' \in [q]_{R \cup \approx}} \delta(p)(a)(y').$$

The left-hand equality comes from the semantics of the process 0. For the right-hand equality, if $p_1 \in [q]_{R \cup \approx}$ then also $p_2 \in [q]_{R \cup \approx}$ (and vice versa), which means that $\sum_{y' \in [q]_{R \cup \approx}} \delta(p)(a)(y') = r - r = 0$. If $p_1 \notin [q]_{R \cup \approx}$, then $p_2 \notin [q]_{R \cup \approx}$, so $\sum_{y' \in [q]_{R \cup \approx}} \delta(p)(a)(y') = 0$. We conclude that R is a be_δ-invariant up to $\mathsf{un}_\approx$.

4.6 Discussion and related work

In this chapter we have proved the soundness of a range of bisimulation up-to techniques by proving their *compatibility*. Compatible functions are sound, and are closed under composition. We conclude with a technical summary of the main compatibility results that are introduced in this chapter. In the table below we assume an arbitrary coalgebra $\delta\colon X \to BX$, an algebra $\alpha\colon TX \to X$ (for a functor T) and a distributive law λ of the functor T over the functor B. All functions in the table are defined in Section 4.2. Recall that if a function is b_δ-compatible, then bisimulation up to g is sound (Section 4.4).

Name	Notation	Condition b_δ-compatibility
Union with S	un_S	S is a bisimulation
Equivalence closure	eq	B preserves weak pullbacks
Bisimilarity closure	bis_δ	B preserves weak pullbacks
Contextual closure	ctx_α	(X, α, δ) a λ-bialgebra
Congruence closure	cgr_α	(X, α, δ) a λ-bialgebra, B pres. weak pullbacks

Further, we proved soundness of several up-to techniques for behavioural equivalence, by proving that they are be_δ-compatible (Section 4.5). The equivalence

closure eq is be_δ-compatible for *any* functor. The contextual closure ctx_α is be_δ-compatible if (X, α, δ) is a λ-bialgebra, α is an algebra for a *monad*, and T preserves reflexive coequalizers. It remains open whether the latter two assumptions are necessary.

A discussion of related work can be found in Chapter 5, which generalizes all of the above results on the soundness of bisimulation up-to.