Operational congruences for reactive systems

James Judi Leifer

University of Cambridge Computer Laboratory

and

Trinity College

5 September 2001

This document consists of a slightly revised and corrected version of a dissertation submitted for the degree of Doctor of Philosophy

Declaration

Except where otherwise stated in the text, this dissertation is the result of my own work and is not the outcome of work done in collaboration.

This dissertation is not substantially the same as any work that I have submitted for a degree, diploma, or other qualification at any other university.

No part of this dissertation has already been or is being concurrently submitted for any such degree, diploma, or other qualification.

This dissertation does not exceed sixty-thousand words, including tables, footnotes, and bibliography.

Abstract

The dynamics of process calculi, e.g. CCS, have often been defined using a labelled transition system (LTS). More recently it has become common when defining dynamics to use *reaction rules* —i.e. unlabelled transition rules— together with a structural congruence. This form, which I call a *reactive system*, is highly expressive but is limited in an important way: LTSs lead more naturally to operational equivalences and preorders.

So one would like to derive from reaction rules a suitable LTS. This dissertation shows how to derive an LTS for a wide range of reactive systems. A label for an agent (process) a is defined to be any context F which intuitively is just large enough so that the agent Fa ("a in context F") is able to perform a reaction. The key contribution of my work is the precise definition of "just large enough", in terms of the categorical notion of *relative pushout* (RPO), which ensures that several operational equivalences and preorders (strong bisimulation, weak bisimulation, the traces preorder, and the failures preorder) are congruences when sufficient RPOs exist.

I present a substantial example of a family of reactive systems based on closed, shallow action calculi (those with no free names and no nesting). I prove that sufficient RPOs exist for a category of such contexts. The proof is carried out indirectly in terms of a category of action graphs and *embeddings* and gives precise (necessary and sufficient) conditions for the existence of RPOs. I conclude by arguing that these conditions are satisfied for a wide class of reaction rules. The thrust of this dissertation is, therefore, towards easing the burden of exploring new models of computation by providing a general method for achieving useful operational congruences.

Acknowledgements

I thank my supervisor Robin Milner for his inspiration, his support, and his warmth. I learned more about the ways of doing research and the style of presenting it from him than anyone else.

I discussed almost every idea in this dissertation with my friends Luca Cattani and Peter Sewell. Peter was unwavering in his clear-sightedness, forcing me to account for my tacitly held assumptions at every turn, dissecting every idea, and setting me straight. Luca opened the world of category theory for me, revealing its beauty and simplicity and its use as a discipline for abstraction.

I talked with many colleagues and learned much from their ideas. Here is a partial list: Marcelo Fiore, Philippa Gardner, Georges Gonthier, Andy Gordon, Tony Hoare, Martin Hyland, Alan Jeffrey, Ole Jensen, Jean-Jacques Lévy, Michael Norrish, Andy Pitts, John Power, Edmund Robinson, Glynn Winskel, Lucian Wischik

My office-mate David Richerby kept me sane in the final act of writing up with his wonderful humour.

C.T. Morley (my Tutor at Trinity College) was always welcoming and ready to advise. Hazel Felton (Side F Secretary at Trinity College) watched over me with great warmth and helped me with great efficiency.

I am indebted to John Roberts for his lucid and consistent thinking.

Natalie Tchernetska's influence touches all parts of this dissertation.

While carrying out this research I was supported by the following funding sources: EPSRC Research Grant GR/L62290; Trinity College Senior Rouse Ball Studentship; NSF Graduate Research Fellowship; Trinity College External Research Studentship; British Overseas Research Studentship; and Cambridge Overseas Trust Studentship.

Contents

Chapter 1

Introduction

1.1 Overview

This dissertation is concerned with *process calculi*, which are mathematical models of computation. Process calculi come in many forms, but share several common features. Much as the λ-calculus isolated the canonical features of sequential computation, process calculi do the same for concurrent phenomena, such as non-determinism, synchronisation, and communication. By concentrating on a few core syntactic primitives, process calculi provide a setting in which to reason about these phenomena without the added baggage associated with a full programming language.

There are three critical ingredients beyond syntax that a process calculus may possess. I summarise these now and discuss each in greater detail later.

The first consists of *unlabelled transitions*, or *reactions* as I call them in this dissertation, which characterise only the internal state changes of an *agent* (a process). Reactions do not require interaction with the surrounding environment.

The second consists of *labelled transitions*, which characterise the state changes that an agent may undergo in concert with the environment. Each transition is associated with a label, which records the interaction with the environment (for example an output on a channel) that enables the transition. A label is thus an *observation* about the behaviour of an agent. Normally, the reactions of an agent are special labelled transitions for which the interaction with the environment is vacuous (so-called τ-transitions).

The third consists of *operational preorders and equivalences*, which characterise when one computation is respectively behaviourally replaceable by, and behaviourally equal to, another. An equivalence is most useful when it is a *congruence*, i.e. is preserved by the syntactic constructions of the calculus; in this case, proving the equivalence of two large agents can be reduced to proving the equivalence of components. Thus congruence is one of the most desirable properties an equivalence may possess. These remarks apply as well to operational preorders.

As I will describe later, much research has concentrated on finding general definitions of operational preorders and equivalences in terms of labelled transitions. For any specific process calculus, it is then a challenge to prove that the preorder or equivalence in question is a congruence. But what happens when there are no labelled transitions, only reactions? The latter situation has become common since reactions provide a clean way of specifying allowable state changes without commitment to particular observables. Reactions are highly expressive but limited in an important way: they do not lead as naturally as labelled transitions do to congruential operational equivalences and preorders.

The central problem addressed by this dissertation is as follows: From a process calculus with only a reaction relation —called a *reactive system*— can we derive a tractable labelled transition relation? By tractable, I mean two things: the labels come from some small set; the labelled transitions readily yield operational preorders and equivalences that are congruences.

My approach is to take a label to be a context (an agent with a hole in it). For any agent a, a label is any context F which intuitively is just large enough so that the agent Fa ("a in context F") is able to perform a reaction. The key contribution of my work is to make precise the definition of "just large enough" in terms of the categorical notion of *relative pushout* (RPO) and to show that several operational equivalences and preorders (strong bisimulation, weak bisimulation, the traces preorder, and the failures preorder) are congruences when sufficient RPOs exist.

I also present a substantial example of a family of reactive systems based on closed, shallow action calculi (those with no free names and no nesting). I prove that sufficient RPOs exist for a category of such contexts. The proof is carried out indirectly in terms of a category of action graphs and embeddings and gives necessary and sufficient conditions for the existence of RPOs. I conclude by arguing that these conditions are satisfied for a wide class of reaction rules. The thrust of this dissertation is, therefore, towards easing the burden of exploring new models of computation by providing a general method for achieving useful operational congruences.

In the rest of this chapter, I discuss some of the history of operational congruences and offer some background as to why reactions have become prominent. I then sketch the solution strategy presented in this dissertation. I relate my approach to work by other researchers and to material I have jointly published. I conclude with an overview of structure of the chapters.

1.2 Historical background and motivation

It may seem strange at first to investigate the "derivation of operational congruences" since congruence is usually a postulated notion in mathematics: to define a model, one *chooses* which elements are equivalent. But choosing has become difficult for many models of computation, creating a gap between the properties such as reaction that are easy to postulate and those such as equivalence that are useful to have.

To illustrate how this gap opened, let us look at some strands in the history of process calculi. Starting with the seminal work by E.F. Moore [Moo56] which examined “finite automata from the experimental point of view”, theoretical computer scientists have pursued the notion that the *observable behaviour* of a computation is more fundamental than the form in which the computation is written; in this view, two programs are equivalent when they are indistinguishable by an outside observer. In the case of automata, an observation consists of a *labelled transition* of the form $a \xrightarrow{x} a'$: automaton a can input the label x and become a'. (This is a slight simplification of Moore’s original notion, which distinguished input from output labels.)

One automaton refines another if the former has a subset of the labelled transitions of the latter, whether or not the two have different syntactic forms. For example, $a \mathrel{\hat{=}} x.(y+z)$ and $b \mathrel{\hat{=}} x.y + x.z$ both can input the same strings of labels, $\langle\rangle, \langle x\rangle, \langle xy\rangle, \langle xz\rangle$, so are equal (refine each other) despite their syntactic differences. This notion of comparing computations based on their labelled transitions is now known as the *traces preorder*; the interpretation of computations in terms of their traces is the *traces model* (see [Ros98]).

The traces preorder is attractive for its simplicity but is not faithful in comparing exactly when one automata’s labelled transitions are “no worse” than another’s. Some of its inadequacies with respect to particular applications were overcome in the late 1970s by Hoare and Milner in ways that I describe below. Different applications for process calculi have different informal requirements for when the patterns of labelled transitions for two agents are the same (in the case of equivalences) or refine one another (in the case of preorders). No one definition can be faithful to all possible requirements. Given that, we can consider schemas parameterised by a labelled transition relation for defining different equivalences and preorders — and thus take labelled transitions to be the *postulated* part of a process calculus from which equivalences and preorders are *derived*. This difference lends labelled transitions their power: a labelled transition relation captures the important interactions between a computation and its environment without making a commitment to a specific equivalence or preorder. As I will discuss later, many new process calculi for modelling distributed computation do not include axioms for labelled transitions but instead rest on an even simpler base.

The primary weakness of the traces preorder is its inability to handle the subtleties of non-determinism. This is evident if we consider the pair a, b defined earlier. After inputting x, a is in a state that can input either y or z; but b takes a “silent” choice (an internal choice not controllable by external influence) when inputting x and enters one of two states: either a state in which only y can be input or a state in which only z can be input. Even though a and b have identical traces, their non-deterministic behaviour is different.

The problem of handling non-determinism together with causality (the dependence of one transition upon an earlier one) was first addressed by *Petri nets* (see, for example, [Rei85]). In its simplest form, a Petri net consists of a directed hypergraph whose nodes are called *conditions* and whose hyperedges are called *events*. A *token* may reside at a condition. When the preconditions of an event hold (i.e. there is a token on every condition that is a source of the event hyperedge), then the event *fires*: the tokens are removed from the source conditions and places on the target conditions. Non-determinism comes from the possibility that a token may enable several events to fire; causality of events is determined by the graph structure, which constrains the allowable flow of tokens and thus enables certain event firings only after others have occurred. Researchers on Petri nets did not originally consider equivalences or preorders for them. However, *event structures*, which are similar to traces but account for the causality and non-determinism of events, are a setting for modelling Petri nets and other causal systems and yield notions of equivalence [WN95]. Because Petri nets do not have a compositional syntax, it is difficult to understand what it means for an equivalence to be a congruence. For this reason, I do not discuss them further and confine my attention to process calculi with compositional syntax.

Hoare's work on Communicating Sequential Processes (CSP) [Hoa78, Hoa85] and Milner's on Communicating Concurrent Systems (CCS) [Mil80, Mil88] addressed the problem of comparing computations in a way that is sensitive to non-determinism, and hence distinguishes a from b (the examples I gave earlier).

Hoare's approach in CSP was to present every agent as an explicit set-theoretic construction including, amongst other data, *failures*; the latter are traces, each of which is enriched with a set of *refusals* indicating which labels may be refused by the agent after inputting the trace. For example, after inputing x, b can refuse y and can refuse z but a cannot refuse either. Each constructor in CSP (prefixing, parallel composition, choice, etc.) is defined in terms of the manipulation of failures (as a continuous function on the complete partial order of agents). The explicit set-theoretic representation of agents in CSP supports the design of model checkers [Ros94, Ros98] which are, for example, effective in detecting bugs in protocols [Low96]. This explicitness is also an obligation: experimentation with new constructors for combining agents requires the creation of a continuous function that manipulates failures; new preorders other than quotients of the failures preorder are not easily catered for. I concentrate in this dissertation, therefore, on the strand of research originating with CCS, but return to the failures preorder in Section 3.7 when proving that it is a congruence. To be fair, though, the heaviness involved in adding new constructors in CSP is not eliminated in CCS-like languages, but instead replaced by different obligations, as described below.

In Milner's CCS, agents have a free syntax and the labelled transition relation is generated by inference rules in the style of Plotkin's structured operational semantics

[Plo81], e.g.

$$x.a \xrightarrow{x} a \qquad \frac{a \xrightarrow{\alpha} a'}{a + b \xrightarrow{\alpha} a'}$$

Remark: Throughout this dissertation $a, b, c, \ldots$ are used to denote agents even when this contradicts historical conventions. The labels (ranged over by $\alpha \ldots$) comprise input channels $x, y, z, \ldots$ and output channels $\bar{x}, \bar{y}, \bar{z}, \ldots$ and a special distinguished element τ. This terminology is somewhat misleading because no data is actually input or output; they are just complementary flavours. In the π-calculus, which I describe later, the differences are significant.

The key idea in CCS is that of synchronisation. If one agent can output x and the other can input x then their parallel composition can synchronise on x. The result is a τ-transition which records the synchronisation but leaves the name on which it occurs anonymous:

$$\frac{a \xrightarrow{\bar{x}} a' \quad b \xrightarrow{x} b'}{a \mid b \xrightarrow{\tau} a' \mid b'}$$

Milner considered several equivalences for CCS; the main ones were *strong and weak bisimulation*. Both kinds employ a coinductive form of definition which gives a powerful proof technique for comparing agents (an idea that originates with Park [Par81]) and provide a general way of defining an equivalence parameterised by a labelled transition system. Strong bisimulation relies on no assumptions about the underlying structure of the labels; weak bisimulation requires a distinguished τ-transition. I will discuss both in detail later (Section 2.3 and Section 3.5) but summarise the two now.

Both strong and weak bisimulation are sensitive to non-determinism and are able to distinguish the pair a, b described earlier in the discussion of the traces preorder. A *strong bisimulation* $\mathcal{S}$ is a relation on agents satisfying the following coinductive property: for all $(a, b) \in \mathcal{S}$ and all labelled transitions $a \xrightarrow{\alpha} a'$, there exists b' such that $b \xrightarrow{\alpha} b'$ and $(a', b') \in \mathcal{S}$ (and vice versa). Informally, "if a and b are $\mathcal{S}$-related then whatever a can do, b can do too, and they both end up in $\mathcal{S}$-related states; likewise for whatever b can do". The largest strong bisimulation relation (which is the union of all strong bisimulations) is denoted by $\sim$. The largest weak bisimulation, denoted by $\approx$, is coarser because it is flexible in allowing τ-transitions to be collapsed and expanded when comparing two agents. For example, a agent that inputs x and then immediately outputs y would be related by weak bisimulation to one that inputs x, then has several τ-steps (internal computations), and finally outputs y.

Milner proved that both kinds of bisimulation are congruences (though not with respect to sum for weak bisimulation — see Section 3.5 for further discussion). Such proofs require care in CCS and are more difficult in later calculi. This is the price of avoiding CSP-style explicit representations of agents and agent constructors. It is a central theme of this dissertation to provide mathematical tools for easing the burden of proving that operationally defined equivalences (such as bisimulation) are congruences.

The idea of using a τ-like transition to record a primitive computational step plays a central role in later calculi. These transitions are variously called *reaction rules*, *reduction rules*, *firing rules*, etc.; I shall use *reaction rules* throughout. In CCS, the τ transition relation requires the entire collection of labelled transition rules to generate it. A dramatic simplification was proposed in the Chemical Abstract Machine (CHAM) of Berry and Boudol [BB90, BB92] and used in work [Mil90] on the π-calculus of Milner, Parrow, and Walker [MPW89, MPW92]. These calculi were the first to employ a lightweight quotient of the agent terms, called a structural congruence, in order to make their reaction rules easy to define. I shall confine the discussion to the π-calculus. (CHAM treats the quotient more explicitly with "heating and cooling" rules, though the idea is similar.)

Structural congruence is an equivalence relation $\equiv$ on agents that allows the parts of an agent to rearrange themselves, e.g.

$$a \mid b \equiv b \mid a \qquad a \mid (b \mid c) \equiv (a \mid b) \mid c \qquad \cdots$$

The reaction relation $\longrightarrow\!\!\triangleright$, which characterises the primitive computational step (namely communication of a name over a channel), is simple to define; it is the smallest relation satisfying the rule

$$\bar{x}\langle y\rangle.a \mid x(z).b \longrightarrow\!\!\triangleright a \mid \{y/z\}b$$

that respects structural congruence

$$\frac{a \equiv a' \quad a' \longrightarrow\!\!\triangleright b' \quad b' \equiv b}{a \longrightarrow\!\!\triangleright b}$$

and is closed under all non-guarding contexts C (e.g. $c \mid -$; see Section 2.1 for further discussion)

$$\frac{a \longrightarrow\!\!\triangleright b}{C[a] \longrightarrow\!\!\triangleright C[b]}\;.$$

Thus the agent $\bar{x}\langle y\rangle \mid (c \mid x(z).b)$ has a reaction even though the parts needed to enable the reaction — namely $\bar{x}\langle y\rangle$ and $x(z).b$ — are not adjacent:

$$\bar{x}\langle y\rangle \mid (c \mid x(z).b) \quad\equiv\quad (\bar{x}\langle y\rangle \mid x(z).b) \mid c \quad\longrightarrow\!\!\triangleright\quad \{y/z\}b \mid c\;.$$

The ease with which reaction rules are defined in this style facilitated an outpouring of new process calculi for modelling encrypted communication [AG97], secure encapsulation

[SV99], agent migration [CG98, Sew98, FGL+96] and so on. Each isolates a computational phenomenon and presents it via a reaction rule together with a structural congruence over some syntax. Here are two examples of reaction rules:

- In Cardelli and Gordon's ambient calculus, one ambient may move inside another (like a packet through a firewall):

$$y[\mathit{in}\ x.a \mid b] \mid x[c] \longrightarrow\!\!\triangleright\ x[y[a \mid b] \mid c]\ .$$

- In Sewell and Vitek's Box-π calculus, a message may move from inside to outside a wrapper and is decorated with the wrapper's name while doing so:

$$y[\bar{x}^{\uparrow}v \mid b] \longrightarrow\!\!\triangleright\ \bar{x}^{\bar{y}}v \mid y[b]\ .$$

In the π-calculus the communication of names along channels presented subtleties in the design of a labelled transitions system and of equivalences (of which several are now studied [SW01]); the proofs of congruence for these equivalences require care. For many of the newer calculi, such as those listed above, the problem of choosing appropriate labelled transitions and proving that bisimulation is a congruence is difficult, and, in many cases, not attempted. In all these cases, experimentation is costly: a slight modification of the syntax or reaction rules often causes the labelled transition relation to change and breaks congruence proofs, forcing them to be reconstructed.

So the gap first described at the beginning of this section has widened as the data defining a typical process calculus have changed. Labelled transitions are no longer postulated primitives of a calculus but instead are teased out from the four fundamental components:

- syntax (agents and agent contexts);
- structural congruence;
- set of reactive (non-guarding) contexts;
- reaction rules.

I call a process calculus containing these components a *reactive system*.

In this way, a reactive system closely resembles instances of the λ-calculus [Bar84]. The latter consist of a simple syntax, a structural congruence based on α-conversion, a set of reactive contexts (known as "evaluation contexts" [FF86]) chosen to force strategies such as call-by-name or call-by-value, and a reaction rule based on β-reduction.

There is however an important difference which renders the problem of finding useful equivalences for process calculi more difficult: their reaction relations are usually neither confluent nor normalising. As a result, equivalences for λ-calculi, such as those based on normal forms or on termination properties [Bar84, Lév78], do not provide viable routes to follow. Therefore we need to consider equivalences based on bisimulation or other techniques that make no assumptions about confluence or normalisation properties. Yet bisimulation requires labelled transitions, which are not provided in a reactive system, and proofs of congruence, which can be difficult and fragile, as already discussed. The next section outlines the strategy for *deriving* labelled transitions which is the basis for the work in this dissertation.

1.3 Contexts as labels

We wish to answer two questions about an arbitrary reactive system consisting of agents (whose syntax may be quotiented by a structural congruence) and a reaction relation $\longrightarrow\!\!\!\triangleright$ (generated by reaction rules):

1. Can we *derive* a labelled transition relation $\xrightarrow{\lambda}\!\!\!\triangleright$ where λ comes from a small set of labels that intuitively reflect how an agent interacts with its environment?

2. Under what general conditions is strong bisimulation over $\xrightarrow{\lambda}\!\!\!\triangleright$ a congruence?

We can begin to address question 1 by considering CCS. Let a, b range over agents, C, D, F range over contexts (agents with a hole), and x range over names. The usual labelled transitions $\xrightarrow{\lambda}\!\!\!\triangleright$ for $\lambda ::= \bar{x} \;\big|\; x \;\big|\; \tau$ reflect an agent's *capability* to engage in some behaviour, e.g. $\bar{x}.a \mid b$ has the labelled transition $\xrightarrow{\bar{x}}\!\!\!\triangleright$ because $\bar{x}.a \mid b$ can perform an output on x. However, if we shift our emphasis from characterising the capabilities of an agent to the *contexts* that enable the agent to react, then we gain an approximate answer to question 1 by choosing the contexts themselves as labels; namely we define

$$a \xrightarrow{F}\!\!\!\triangleright a' \qquad \text{iff} \qquad Fa \longrightarrow\!\!\!\triangleright a' \tag{1.1}$$

for all contexts F. (We denote context composition and application by juxtaposition throughout. We regard a context with a trivial hole as an agent, so application is a special case of composition.) Instead of observing that $\bar{x}.a \mid b$ "can do an $\bar{x}$" we might instead see that it "interacts with an environment that offers to input on x, i.e. reacts when placed in the context $- \mid x$". Thus, $\bar{x}.a \mid b \xrightarrow{-|x}\!\!\!\triangleright a \mid b$.

The definition of labelled transition in (1.1) is attractive when applied to an arbitrary reactive system because it in no way depends upon the presence or absence of structural congruence. Furthermore, it is generated entirely from the reaction relation $\longrightarrow\!\!\!\triangleright$ (question 1); and, strong bisimulation over the derived labelled transition relation $\xrightarrow{\cdot}\!\!\!\triangleright$ is a congruence, (question 2). The proof of the latter is straightforward: let C be an arbitrary context and suppose $a \sim b$; we show that $Ca \sim Cb$. Suppose $Ca \xrightarrow{F}\!\!\!\triangleright a'$; by definition,

$FCa \rightarrowtriangle a'$, hence $a \xrightarrow{FC}\!\!\!\triangleright a'$. Since $a \sim b$, there exists b' such that $b \xrightarrow{FC}\!\!\!\triangleright b'$ and $a' \sim b'$. Hence $Cb \xrightarrow{F}\!\!\!\triangleright b'$, as desired. The other direction follows by symmetry.

Nonetheless, the definition in (1.1) is unsatisfactory: the label F comes from the set of *all* contexts — not the "small set" asked for in question 1 — thus making strong bisimulation proofs intolerably heavy. Also, the definition fails to capture its intended meaning, namely that $a \xrightarrow{F}\!\!\!\triangleright a'$ holds when a *requires* the context F in order that a reaction is enabled in Fa: there is nothing about the reaction $Fa \rightarrowtriangle a'$ that forces all of F — or indeed any of F — to be used. In particular, if $a \rightarrowtriangle a'$ then for all contexts F that preserve reaction, $Fa \rightarrowtriangle Fa'$, hence $a \xrightarrow{F}\!\!\!\triangleright Fa'$; thus a has many labelled transitions that reflect nothing about the interactions of a itself.

Let us unpack (1.1) to understand in detail where it goes wrong. Consider an arbitrary reactive system equipped with a set Reacts of reaction rules; the reaction relation $\rightarrowtriangle$ contains Reacts and is preserved by all contexts, as formalised by the following axiom and inference rule:

$$l \rightarrowtriangle r \quad \text{if } (l, r) \in \mathsf{Reacts} \qquad\qquad \frac{a \rightarrowtriangle a'}{Ca \rightarrowtriangle Ca'} \quad .$$

Expanding (1.1) according to this definition of $\rightarrowtriangle$ we have:

$$\begin{array}{lll} a \xrightarrow{F}\!\!\!\triangleright a' & \text{iff} & Fa \rightarrowtriangle a' \\ & \text{iff} & \exists (l, r) \in \mathsf{Reacts}, D. \quad Fa = Dl \;\&\; a' = Dr \quad . \end{array} \tag{1.2}$$

The requirement $Fa = Dl$ in (1.2) is rendered by a commuting square (as shown in Figure 1.1) in some category whose arrows are the agents and contexts of the reactive system. This requirement reveals the flaw described earlier: nothing in (1.2) forces F and D to be a "small upper bound" on a and l. The next section explores the challenges involved in making precise what "small upper bound" means.

$$\begin{tikzcd} \cdot \arrow[r, "a"] \arrow[d, "l"'] & \cdot \arrow[d, "F"] \\ \cdot \arrow[r, "D"'] & \cdot \end{tikzcd}$$

1.1

1.4 Relative pushouts

For several years I tried to make precise what "small" means. I was mired in detailed arguments about specific examples of contexts for which I would search for a "dissection lemma", having roughly the following form: given $Fa = Dl$, there exists a "maximum" C, such that for some F' and D' we have $F'a = D'l$, $F = CF'$ and $D = CD'$. In other words, dissection peels off as much as possible from the outside of F and D, decomposing $Fa = Dl$ as $CF'a = CD'l$. If the dissection of $Fa = Dl$ peels off nothing, i.e. produces the decomposition $\mathsf{id}\,Fa = \mathsf{id}\,Dl$, where id is the identity context, then F and D are "a small upper bound". But the problem then remains: when is C the "maximum" possible context that can be peeled off?

In this section I look at some examples of dissection in order to illustrate just how subtle the problem is. These examples motivate a solution that I introduce here and present in detail later in this dissertation.

For some particular classes of contexts, it is easy to understand what to do. Consider this example of two compositions $Fa = Dl$ in a free term algebra:

$$\Big(\alpha\langle\alpha'\langle\beta\langle-\rangle\rangle\rangle\Big) \circ \gamma \quad = \quad \Big(\alpha\langle\alpha'\langle-\rangle\rangle\Big) \circ \Big(\beta\langle\gamma\rangle\Big) \quad = \quad \alpha\langle\alpha'\langle\beta\langle\gamma\rangle\rangle\rangle\ .$$

(For the sake of clarity, I use $\circ$ for composition in this example.) The maximum context C that can be peeled off is $\alpha\langle\alpha'\langle-\rangle\rangle$. Nothing more can be peeled off because $C = D$. The only other possibilities are $-$ and $\alpha\langle-\rangle$, neither of which is as big as C. So C it is! Of course I am arguing informally here, but it is straightforward to make this line of thinking precise.

Free term algebras are the simplest setting in which to consider dissection exactly because they are *free*. When passing to a syntax quotiented by a non-trivial structural congruence, the question becomes difficult, especially when compounded with naming structure such as in the π-calculus. Consider a typical π-calculus agent such as $a \mathrel{\hat{=}} (\nu u)(\bar{x}\langle u\rangle \mid x(z).\bar{y}\langle z\rangle)$. Notice that a contains a parallel constructor $|$, which is associative and commutative and has identity $\mathbf{0}$; also, u and z are bound, so are subject to α-conversion. Furthermore, the name x is used twice (which requires attention when substituting another name for it) and is discarded after the reaction $a \longrightarrow\!\!\!\!\rhd\, a'$, where $a' \mathrel{\hat{=}} (\nu u)(\bar{y}\langle u\rangle)$.

This structure demands careful treatment and makes dissection for the π-calculus difficult: for term algebra, one can incrementally peel off a top-level function symbol, but for the π-calculus, there is no notion of a "top-level" constructor. The structural congruence of the π-calculus equates so many different syntactic representations of the same agent that it is difficult to understand where to begin. Without any naming structure, parallel composition becomes easier to handle, as shown by Sewell [Sew01]. As I explain in Section 1.5, it is the treatment of names that distinguishes the dissection results in this dissertation from his.

A possible approach is to abandon tree-like syntax and to think in terms of graph-like syntax that automatically quotients out many ignorable differences. Even if a dissection result could be proved for some graph-theoretic representation of the π-calculus, it would not necessarily generalise smoothly to other calculi. As a result, I studied dissection for Milner's action calculi [Mil96], which are a family of reactive systems. The syntax of action calculi is sufficiently rich to embrace process calculi such as π-calculus, the λ-calculus, and the ambient calculus. Action calculi are introduced in Section 5.2; here I confine my attention to a few salient features of their graphical form.

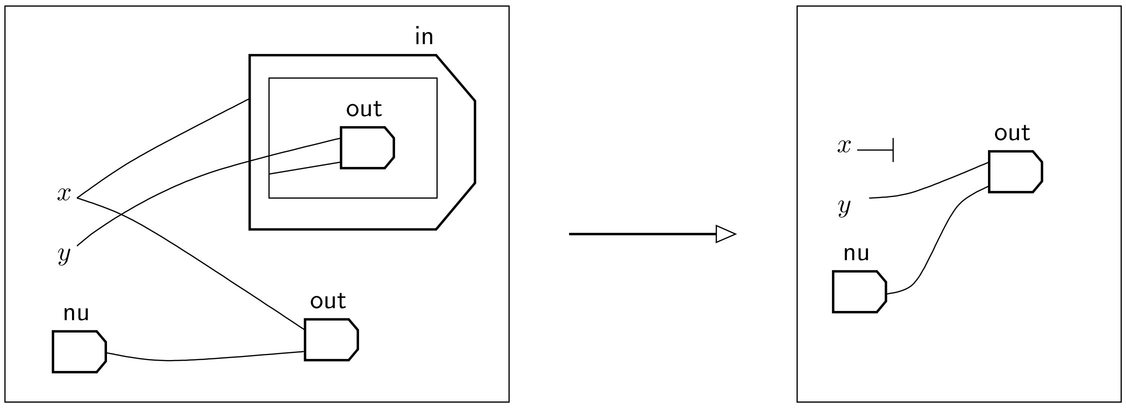


Figure 1.2: A graphical representation of a π-calculus reaction $(\nu u)(\bar{x}\langle u\rangle \mid x(z).\bar{y}\langle z\rangle) \longrightarrow\!\!\!\triangleright$ $(\nu u)(\bar{y}\langle u\rangle)$

Consider the example shown in Figure 1.2; it illustrates a pair of *action graphs* — agents in an action calculus— that represent the π-calculus reaction $a \longrightarrow\!\!\!\triangleright a'$ given earlier. An action graph consists of nodes (rectangles with two blunt corners) and arcs:

- Nodes are labelled with *controls* from a *control signature* of primitives. Each action calculus may have a different control signature, such as $\{\mathsf{nu}, \mathsf{in}, \mathsf{out}\}$ for a simple π-calculus without replication or choice, or $\{\mathsf{ap}, \mathsf{lam}\}$ (application and λ-abstraction) for the λ-calculus.

- Arcs represent names. They connect *source ports* to *target ports*. The source ports are arrayed on the left interface of an action graph (such as is shown for the action graph nested inside in) and on the right side of nodes (e.g. nu); they also include free names (e.g. y). The target ports are arrayed on the right interface of action graphs and the left side of nodes. Arcs may be forked from a source port (such as the arcs from x in the LHS) and may be discarded (such as the arc from x in the RHS).

How can we perform dissection on action graphs? It is the wiring structure (the arcs) of an action graph that makes dissection so difficult. Consider the example of a composition $Fa = Dl$ shown in Figure 1.3. (This example is based on one first suggested by Philippa Gardner.) The contexts F and D are just action graphs with a hole (shown as a shaded rectangle). The composition Fa is formed by placing a inside the hole of F and joining up the corresponding arcs. By reasoning about the controls it is possible to see which ones can be peeled off: F has N, M and D has N, L, so C might have N (the intersection of the controls in F and D). Indeed, the C shown does have just one control. But what arcs should C have? Does the loop shown in C make C "maximal" in some sense? Should C have other forked and discarded arcs? How can we choose?

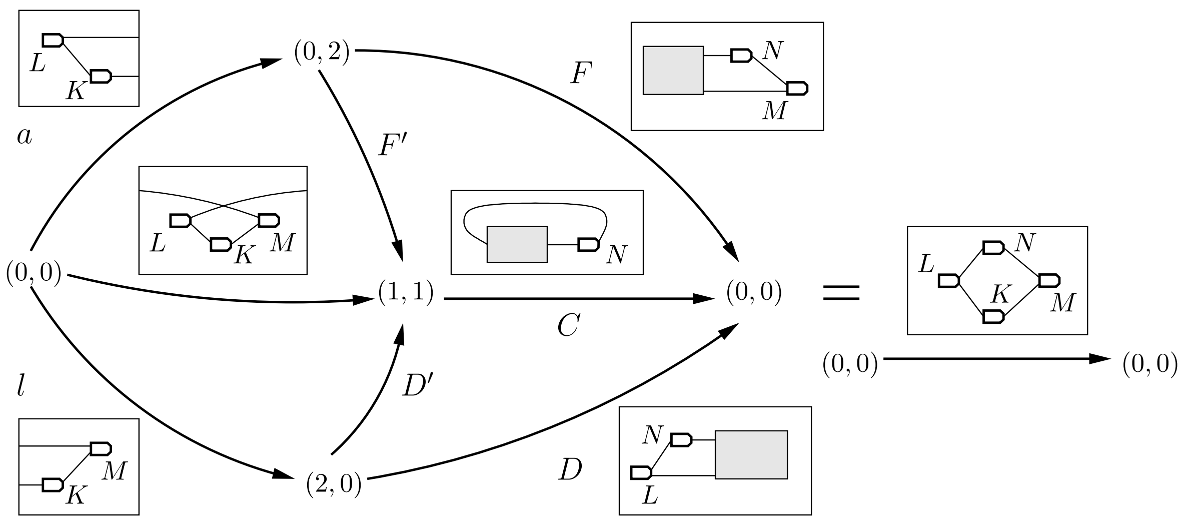


Figure 1.3: A surprising dissection involving reflexion

The example in Figure 1.4 is even more curious: F and D have no nodes at all — they are pure wiring contexts. Since $F = D$ and $a = l$, one might expect that the “maximal” common part that can be peeled off of $Fa = Dl$ is $F = D$ itself. This is not true! The triple F', D', C provides a better dissection in a way that I will make precise below.

The point of these examples is to demonstrate that informal reasoning about dissection is difficult when dealing with contexts containing wiring: There is almost no way to decide just by looking at the arcs themselves whether one dissection is better than another.

Out of this murkiness finally emerged a clearer way. I came upon an *abstract* characterisation, called a *relative pushout*, of which dissections are “best”. By abstract, I mean that the characterisation makes no use of any specific properties of contexts except their composability, thus can be instantly cast as a category theoretic property.

Consider the outside commuting square in Figure 1.5 which shows $Fa = Dl$. A *relative pushout* (RPO) for this square is a triple F', D', C satisfying two properties: first, the triple is a *candidate*, i.e. $F'a = D'l$ and $CF' = F$ and $CD' = D$; second, for any other candidate $\hat{F}', \hat{D}', \hat{C}$, there exists a *unique mediating* J making all the triangles commute (as shown).

RPOs are the key contribution of this dissertation. It is by working with them, rather than trying to come up with *ad hoc* ways of dissecting contexts in specific examples, that we gain two important advantages:

- RPOs are abstract: as I said before they do not rely on any specific properties of contexts except composability. By defining labelled transitions in terms of RPO constructions, it is possible carry out proofs of congruence (see Chapter 2 and Chapter 3) for several operational equivalences and preorders (strong bisimulation,

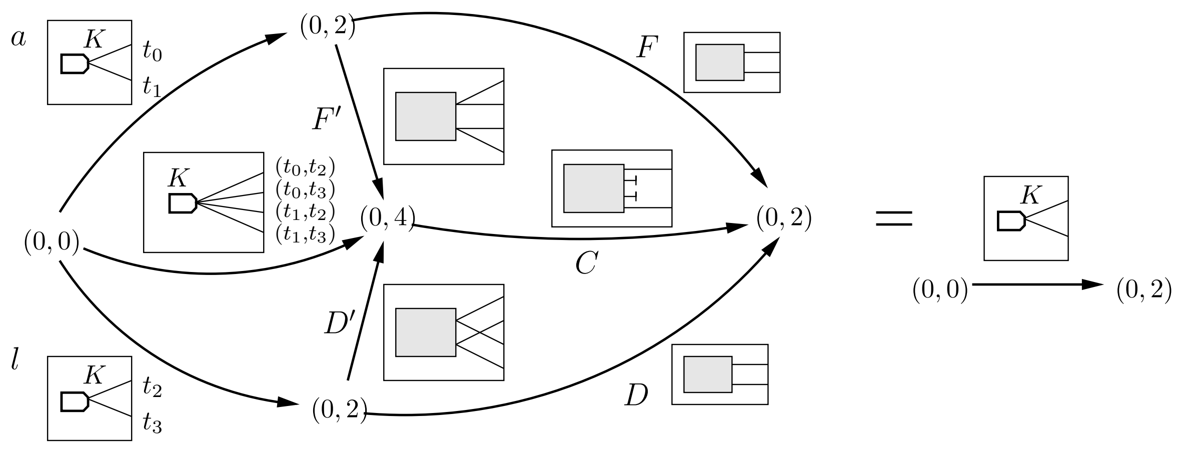


Figure 1.4: A surprising dissection involving forking

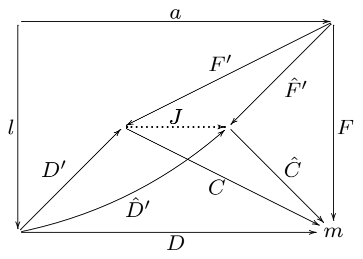


Figure 1.5: A relative pushout

weak bisimulation, the traces preorder, and the failures preorder), for which the proofs *do not depend on any specific properties of contexts except the existence of RPOs*. Thus these results are applicable to any reactive system which possesses sufficient RPOs.

- RPOs provide a unifying discipline for analysing contexts in specific examples. The fuzziness about which dissection is "best" in the examples of action graphs shown earlier disappears: we want the candidate triple F', D', C' from which there is a unique mediator to any other candidate. By this requirement, the candidates shown in Figure 1.3 and Figure 1.4 are "best". (I put quotation marks around best because there may be many best candidates, but all of them are isomorphic to each other by standard categorical reasoning.) The problem of finding RPOs for graphs is non-trivial, as shown in Chapter 6, but one is sustained by the unambiguity of the task: there is no vagueness about the properties required of an RPO.

Thus, the notion of an RPO does not solve the problem of finding a dissection, but it makes the problem well-defined and provides a reward for the effort by virtue of the congruence proofs that rely on the existence of sufficient RPOs.

1.5 Other work

This section brings together some of the important related work. There is a large collection of literature about process calculi, some of which I referred to in previous sections. I will concentrate on those pieces of research that most closely impinge on the specific problems that I address in this dissertation.

The idea of finding conditions under which a labelled transition relation yields an operational congruence has been thoroughly studied in work on *structural operational semantics* (SOS) [GV92, TP97]. The principle is to postulate *rule formats*, conditions on an *inductive* presentation of a labelled transition relation that ensure that operational equivalences (e.g. weak bisimulation [Blo93]) are congruences. There is a fundamental difference between this problem and the one I am looking at. The work on SOS presumes that a labelled transition relation is *already given*: the problem is to show that *if* it satisfies a particular format *then* the congruences follow. My work takes reaction rules as primitive, not labelled transitions: I aim to *derive* a labelled transition relation from the reaction rules for which operational equivalences (and preorders) are congruences. In my case, the derived labelled transition relation is not inductively presented. It is an open question whether it can be inductively presented and, further, whether this presentation satisfies some well-known rule format from SOS theory: this possibility is not explored in the dissertation but is discussed in greater detail in Chapter 8, which covers future work.

The problem of deriving operational congruences for process calculi from a reaction relation and not from a labelled transition relation is studied in the work on *barbed bisimulation* [MS92], *insensitivity observation* [HY95], and *testing equivalence* [DH84]. The first two construct equivalences by augmenting bisimulation over the reaction relation with observations about related states. For example, in the former, the observations are barbs which detect the ability of an agent to perform an output on a channel. The last (testing) compares two agents by their ability to satisfy the same “may” and “must” tests.

Thus all three depend on primitive observations (not just reactions) though these can be simpler than labelled transitions. The main obstacle to their use is the fact that they do not yield congruences but instead need to be “closed up” by all contexts. For example, it is straightforward to prove barbed equivalence in particular cases but difficult to show barbed congruence because of the heavy quantification over all contexts. Work by Fournet [Fou98] and by Fournet and Gonthier [FG98] ease this burden with techniques that allow a proof of barbed congruence to be broken into pieces, each of which may be carried out using other congruence relations.

Jeffrey and Rathke [JR99] used contexts as the basis for the labels of an LTS in the case of the ν-calculus (a variant of the λ-calculus with fresh name creation). They did not derive uniformly these labels from a reaction relation but they were guided by the intuition that the labels are small contexts that enable a reaction. These labels give them the right observational power to obtain useful congruences based on bisimulation.

Sewell's work [Sew01] is the closest to mine of all the material I have cited here. He studied the problem of deriving contextual labels for a family of reactive systems (parameterised by arbitrary reaction rules) whose syntax consists of free terms quotiented by a structural congruence for parallel composition. He defined labelled transitions by explicitly reasoning about how a label overlaps with a redex. From this definition, he proved that bisimulation is a congruence by appealing to his specific dissection results (not motivated by RPOs). There are three important differences in approach:

- He dealt with multi-hole redexes which capture the full uniformity of metavariables, thus leading to lighter labelled transitions than I can derive with my present RPO technology. I discuss possible remedies for this in Chapter 8.

- He invented explicitly his definition of labelled transition and his statement of dissection, both of which are complex, without employing RPOs (which I worked with later). As as result, his proof of congruence is not easily generalisable to other syntactic families and other operational equivalences. It seems likely, however, that his dissection results imply the existence of RPOs for the class of reactive systems he considered, and it would be worth trying to recast them so as to make this precise.

- He confined his attention to free term algebras with parallel composition and did not handle wiring structure such as is shown in Figure 1.4. I do handle wiring in Chapter 6 where the discipline of seeking RPOs guides the dissection involved. Given the complexity of Sewell's definition of labelled transition and of his statement of dissection, it is difficult to see how these could be generalised to embrace wiring without the benefit of the notion of RPOs or other universal constructions.

In this dissertation, I have made no use of the *double pushout* techniques developed in graph rewriting [Ehr79]. These are a way to describe the occurrence of a subgraph —especially a redex— in a graph. To avoid confusion, I should emphasise that *relative* pushouts play quite a different role. In my work, subgraph occurrences are handled by embeddings and contexts; the nature of the graphs (with forked wiring) seems to require a specific approach. But it would be useful to examine in the future how the embeddings relate to the double pushout construction, and how graph-theoretic representations of the syntax of the π-calculus and other calculi formed by quotienting out structural congruence compare to similar work in graph rewriting [CM91, Kön99].

Some of my work has been done in collaboration with Milner and all of it under his supervision. I wish to draw attention to two relevant pieces of published work for which I am a joint author. All other work presented in this dissertation that is not mentioned here is my own.

- Leifer and Milner: "Deriving bisimulation congruences for reactive systems" [LM00a]. This paper introduces RPOs and gives a proof of congruence for strong bisimulation, thus overlapping with some of the material in Chapter 2. Milner collaborated with me on the categorical manipulations, which are critical to the proof of congruence.

- Cattani, Leifer, and Milner: "Contexts and embeddings for closed shallow action graphs" [CLM00]. This technical report demonstrates the existence of RPOs for a category of action graph contexts by relating RPOs to relative coproducts in a category of action graph embeddings, thus overlapping with some of the material in Chapter 5 and Chapter 6. Cattani's contribution was to recast the opfibration (originally investigated by Milner and me) between the embeddings category and the contexts category in terms of the coslice of the latter. Milner and I worked jointly on the definition of context composition and of embedding. I am responsible for the proof of the existence of relative coproducts (given in full detail in Chapter 6).

1.6 Outline

The subsequent chapters of this dissertation are organised in the following way:

Chapter 2: I make precise the notion of a reactive system and give a definition of labelled transition in terms of *idem pushouts* (IPO), a sister notion of RPOs. I then prove a series of results using simple categorical reasoning that shows how to manipulate IPOs and RPOs. The main theorem then follows by direct use of the categorical results from the chapter: if sufficient RPOs exist then strong bisimulation is a congruence. I conclude by reproving the same theorem in a cleaner way by isolating two lemmas which give derived inference rules for labelled transitions.

Chapter 3: I generalise the definition of reactive system by enriching it so as to comprise two categories with a functor $\mathcal{F}$ between them. The idea is that the downstairs category is the one in which one wishes to consider agents and contexts, but for which enough RPOs might not exist. The upstairs category does have RPOs but at the cost of extra intensional information in the arrows and objects. By refining the definition of labelled transition so that it relates arrows downstairs in terms of IPO properties of their preimages upstairs, I obtain congruence results for strong bisimulation, weak bisimulation, the traces preorder, and the failures preorder. Finally, I propose added structure needed in a functorial reactive system to cater explicitly for RPOs that yield multi-hole contexts. I conclude by proving that strong bisimulation is a congruence here as well.

Chapter 4: The proofs of congruence for the preceding chapter rest on some hypotheses about the functor $\mathcal{F}$. The most important one of these is that $\mathcal{F}$ allows an IPO square upstairs to "slide" so as to leave the $\mathcal{F}$-image of the square invariant. This chapter eases the burden of showing that these hypotheses are satisfied by giving an abstract way of generating a functor satisfying them from simpler data. The chapter started by assuming the existence of a *well-supported precategory* $\mathbf{A}$, which is like a category but lacks some arrow compositions and has extra structure for extracting and renaming the *support* of an arrow. The motivation for this comes from the raw contexts in Chapter 5 for which composition of two contexts is not defined unless their node sets (supports) intersect trivially. I derive from $\mathbf{A}$ two categories and a functor between them. The upstairs category is formed from $\mathbf{A}$ by adding extra information to the objects, so as to make composition total. The downstairs category is formed by quotienting away the supports. The functor $\mathcal{F}$ maps arrows to their quotient equivalence classes. By reasoning abstractly, I show that $\mathcal{F}$ allows IPO sliding and has all of the other properties required of functorial reactive systems in Chapter 3. By instantiating these results, as in Chapter 5 for example, one gets "for free" a functorial reactive system provided that one starts with a well-supported precategory, a light burden.

Chapter 5: This chapter gives a significant example of a family of functorial reactive systems. The contexts are derived from closed, shallow action graphs, those with no nesting and no free names. Their graph structure includes forked and discarded wiring connecting interfaces on each node and on the inside and outside of each context. By instantiating the results of the preceding chapter, we construct a functorial reactive system with the following properties: the downstairs category does not distinguishes the occurrences of controls, as desired when modelling agents of a process calculus; the upstairs one does distinguish them, thus providing sufficient RPOs as shown in Chapter 6.

Chapter 6: I take up the task of proving that sufficient RPOs exist in a category of action graph contexts. My strategy is to follow an indirect route: rather than try to construct RPOs directly in a category of contexts, I shift the problem to that of looking for a related construction called a *relative coproduct* in a category of action graphs and embeddings. The motivation for this choice is to avoid some of the heaviness inherent in manipulating contexts: the equality of two decompositions, such as in $C_0a_0 = C_1a_1$, contains much redundant information about the nodes and arcs of the component contexts. The important data is how a_0 and a_1 overlap in the common action graph $C_0a_0 = C_1a_1$. The embeddings category is the right setting for analysing this: each embedding is a tuple of functions mapping the nodes and ports of one action graph into another. Thus, given two embeddings $\eta_i : a_i \rightarrowtail b$ into the same action graph $b \hat{=} C_0a_0 = C_1a_1$, it is clear how the nodes and arcs of a_0 and a_1 overlap. The chapter defines embeddings and gives necessary and sufficient conditions for the existence of relative coproducts. A simple corollary then shows that sufficient RPOs exist in the contexts category provided that the redexes satisfy a constraint on their wiring. As a result, all of the congruence proofs in Chapter 3 are applicable to the family of functorial reactive systems formed from action graphs.

Chapter 7: The preceding chapter showed that sufficient RPOs exist provided the wiring present in redexes is constrained. This chapter argues that this constraint is reasonable and shows an example of undesirable non-determinism in the pattern of reactions when the constraint is not satisfied. It transpires that by confining redexes so as to satisfy two further "reasonable" wiring requirements on redexes, a beautiful categorical property holds, namely that the preimages of redexes are epis. The epi property may be exploited abstractly (Chapter 3) to prove that id-labelled transitions and reactions correspond exactly. The chapter concludes by speculating about other possible concrete and abstract constraints on redexes and the benefits gained when they are satisfied.

Chapter 8: This chapter first reviews some of the accomplishments of this dissertation and then discusses future research. The latter includes two main lines. The first is the extension of the action graph contexts handled in Chapter 5 to include richer features such as nesting, free names, binding, and full reflexion. The second is the extension of graphs and of functorial reactive systems to deal with multi-hole redexes in order to capture the full uniformity of reaction rules with metavariables. The goal of both of these is to create a shared theory that is sufficiently powerful to provide lightweight labelled transitions and useful operational congruences for current and future process calculi

Appendix A: This appendix reviews the category theory required in this dissertation.

Appendix B: The second appendix considers a variation based on retractions of the labelled transitions defined in Chapter 3; the result is that id-labelled transitions match the reaction relation exactly.

Chapter 2

Operational congruences for reactive systems

2.1 Formalisation of reactive systems

In this section I investigate how to give reactive systems, which were introduced informally in Section 1.2, a precise category-theoretic definition. The goal is to include appropriate categorical structure so that it is possible to derive labelled transitions and prove that several operational equivalences and preorders are congruences. To that end, I first study *relative pushouts* (RPOs) and *idem pushouts* (IPOs), which are universal constructions related to pushouts. I then show how to derive labelled transitions from a set of reaction rules, with IPOs playing the central role. Finally I prove by categorical reasoning that if sufficiently many RPOs exist then strong bisimulation is a congruence. The next chapter considers richer notions of equivalences and reactive systems, proving that the former are congruences.

This dissertation employs only basic category theory, such as simple universal constructions, slices (and coslices), monoidal structures, and functors. A summary of all the required theory is given in Appendix A.

Throughout this chapter, I use lowercase roman letters $a, b, \ldots$ for *agents* (processes) and uppercase roman letters $C, D, \ldots$ for *agent contexts* (process contexts). Both are arrows in a category of contexts (as explained below). Juxtaposition is used for categorical composition. Other notation is explained as it comes up.

How do we get at the essence of reactive systems, i.e. how do we find mathematical structure that is simple enough to get general results about congruences and rich enough to encompass significant examples? The key ingredients of a reactive system were shown in Section 1.2 and are recalled here:

- syntax (agents and agent contexts);

- structural congruence;
- set of reactive (non-guarding) contexts;
- reaction rules.

For each, I outline the mathematical design space and explain the decisions I have taken.

syntax: Since contexts are composable, I take them to be the arrows of some category **C**. This presents an immediate question. Are the objects of **C** agents or sorts?

Following the former route leads to a problem: If we think of a context C as an embedding of an agent a into an agent a', i.e. " $C : a \rightarrowtail a'$ ", then there is no easy way to apply C to a different agent. For example, we cannot state the congruence property of an equivalence $\sim$: if $a \sim b$ for agents a and b then C "applied to" a and C "applied to" b are $\sim$-equivalent for all contexts C. In Chapter 6 I return to embeddings (which are critical in proving the existence of universal constructions for categories of graphs) but concentrate now on a category **C** of contexts and sorts. If the objects are the sorts (or "types" or "arities") of the contexts, then agents are a special subclass of contexts. In particular, agents are contexts with a null hole, i.e. contexts whose domain is some distinguished object 0. Now the congruence property of $\sim$ is neatly rendered: if $a \sim b$ for all arrows $a, b : 0 \rightarrowtail m$, then $Ca \sim Cb$ for all arrows C with domain m.

For concreteness, consider as an example a category of contexts for some Algol-like programming language. The objects of the category could comprise the usual types: *bool*, *int*, *cmd*. Then we have the following examples of arrows:

$$\begin{array}{lcll} C_0 & \hat{=} & \textbf{if} - \textbf{then}\ x := 0\ \textbf{else skip} & : \mathit{bool} \rightarrowtail \mathit{cmd} \\ C_1 & \hat{=} & 14 < - & : \mathit{int} \rightarrowtail \mathit{bool} \\ C_0C_1 & = & \textbf{if}\ 14 < - \ \textbf{then}\ x := 0\ \textbf{else skip} & : \mathit{int} \rightarrowtail \mathit{cmd}\ . \end{array}$$

Another example is of a category of linear multi-hole term algebra contexts over some signature Σ. These are considered in more detail in Section 3.8. The objects are natural numbers. The arrows $m \rightarrowtail n$ are n-tuples of terms over $\Sigma \cup \{-_1, \ldots, -_m\}$, where each symbol $-_i$ is used exactly once. For example, if $\Sigma = \{\alpha, \alpha', \beta, \gamma\}$, where α and α' are constants, β is a 1-place function symbol, and γ is a 2-place function symbol, then:

$$\begin{array}{lcll} C_0 & \hat{=} & \langle \gamma\langle -_2, \alpha' \rangle, \alpha, \beta\langle -_1 \rangle \rangle & : 2 \rightarrowtail 3 \\ C_1 & \hat{=} & \langle \alpha, \beta\langle \alpha' \rangle \rangle & : 0 \rightarrowtail 2 \\ C_0C_1 & = & \langle \gamma\langle \beta\langle \alpha' \rangle, \alpha' \rangle, \alpha, \beta\langle \alpha \rangle \rangle & : 0 \rightarrowtail 3\ . \end{array}$$

structural congruence: The main decision here is whether to make structural congruence explicit or implicit. The simplest solution (which is the one taken in this dissertation) is leave it implicit in the definition of arrow equality — thus the arrows are structural equivalence classes of contexts. Consequently, certain categories (such as those of graph contexts, see Chapter 5) do not have enough universal constructions to give the desired congruence results. In these cases, we are forced to look for the universal constructions in less quotiented categories and then exhibit functors with special properties back to the fully quotiented categories (see Section 3.2).

set of reactive contexts: This is modelled by a set $\mathbf{D}$ of arrows. Since reactive contexts are composable and identity contexts are reactive, I take $\mathbf{D}$ to be a subcategory of $\mathbf{C}$. Furthermore, decomposing reactive contexts yields reactive contexts, so $D_1 D_0 \in \mathbf{D}$ implies $D_1, D_0 \in \mathbf{D}$.

For example, in the call-by-value λ-calculus [Plo75], the reactive contexts consist of all compositions of the following contexts:

$$- \qquad \mathsf{ap}(v, -) \qquad \mathsf{ap}(-, a)$$

where v is any value (closed abstraction) and a is any term. In the π-calculus (see [Mil90]), the reactive contexts consist of all compositions of following contexts (closed under structural congruence):

$$- \qquad (\nu x)(-) \qquad - \mid a$$

where x is any name and a is any process. Reactive contexts correspond to *evaluation contexts* of Felleisen and Friedman [FF86].

reaction rules: These are given by a set Reacts of redex-contractum pairs of agents (l, r) with common codomain, i.e. $(l, r) \in \mathsf{Reacts}$ implies that there is an object m of $\mathbf{C}$ such that $l, r : 0 \rightarrow m$. For simplicity, I consider redexes and contractums that are pure agents, not agents with meta-variables (i.e. contexts). Thus, to define the reactions of CCS, we let

$$\mathsf{Reacts} \mathrel{\hat{=}} \big\{ \big(\bar{x}.a \mid x.b \ , \ a \mid b\big) \quad / \quad x \text{ is a name and } a, b \text{ are agents} \big\}$$

rather than:

$$\mathsf{Reacts} \mathrel{\hat{=}} \big\{ \big(\bar{x}.{-_1} \mid x.{-_2} \ , \ {-_1} \mid {-_2}\big) \quad / \quad x \text{ is a name} \big\} \ .$$

I use $\ /\ $ throughout this dissertation for set comprehensions. The latter approach maintains the maximum uniformity present in rules and is considered in detail by Sewell in [Sew01]; however, that approach is complex and would require future work to adapt it to the categorical setting of this dissertation (see Chapter 8).

Distilling the structures described in the past paragraphs yields the following definition of a reactive system and a reaction relation:

Definition 2.1 (reactive system) A *reactive system* consists of a category $\mathbf{C}$ with added structure. We let m, n range over objects. $\mathbf{C}$ has the following extra components:

- a distinguished object 0 (not necessarily initial);
- a set of *reaction rules* called $\mathsf{Reacts} \subseteq \bigcup_{m \in \mathsf{obj}\,\mathbf{C}} \mathbf{C}(0, m)^2$, a relation containing pairs of agents with common codomain;
- a subcategory $\mathbf{D}$ of $\mathbf{C}$, whose arrows are the *reactive contexts*, with the property that $D_1 D_0 \in \mathbf{D}$ implies $D_1, D_0 \in \mathbf{D}$. ■

Definition 2.2 (reaction relation) Given a reactive system $\mathbf{C}$, the *reaction relation* $\longrightarrow\!\!\triangleright \subseteq \bigcup_{m \in \mathsf{obj}\,\mathbf{C}} \mathbf{C}(0, m)^2$ contains pairs of agents with common codomain and is defined by lifting the reaction rules through all reactive contexts:

$$a \longrightarrow\!\!\triangleright a' \qquad \text{iff} \qquad \exists (l, r) \in \mathsf{Reacts}, D \in \mathbf{D}. \quad a = Dl \;\&\; a' = Dr\ . \qquad ■$$

We now have enough definitions to make precise the first approximation for labelled transitions given in (1.1). The only change is that we think of composing arrows rather than "applying contexts" and we are careful about which contexts are reactive (by writing $D \in \mathbf{D}$ below):

Definition 2.3 (labelled transition — first approximation)

$$\begin{aligned} a \xrightarrow{F}\!\!\triangleright a' \qquad &\text{iff} \qquad Fa \longrightarrow\!\!\triangleright a' \\ &\text{iff} \qquad \exists (l, r) \in \mathsf{Reacts}, D \in \mathbf{D}. \quad Fa = Dl \;\&\; a' = Dr\ . \end{aligned} \qquad ■$$

$$\begin{tikzcd} 0 \arrow[r, "a"] \arrow[d, "l"'] & {} \arrow[d, "F"] \\ {} \arrow[r, "D"'] & {} \end{tikzcd} \quad 2.1$$

The commuting square to the right renders the equality $Fa = Dl$. As I argued in Section 1.2, there may be "junk" in F and D, i.e. parts of F and D that do not contribute to the reaction. For example, in a category of CCS contexts, the outside square in Figure 2.2 commutes. So, by the naive definition of labelled transitions given above,

$$\bar{x}.a \xrightarrow{-|x.b|y}\!\!\triangleright a \mid b \mid y \tag{2.1}$$

But the y in the label is superfluous. Is there a general condition on Figure 2.1 that prevents this labelled transition, but still allows the following:

$$\bar{x}.a \xrightarrow{-|x.b}\!\!\triangleright a \mid b \quad ?$$

$$\begin{tikzcd}
0 \arrow[rr, "\bar{x}.a"] \arrow[dd, "\bar{x}.a|x.b"'] & & {} \arrow[dl, "-|x.b"'] \arrow[dd, "-|x.b|y"] \\
& {} \arrow[dr, "-|y"] & \\
{} \arrow[ur, "-"'] \arrow[rr, "-|y"'] & & {}
\end{tikzcd}$$

2.2

Informally, the condition would state that there is no lesser upper bound in Figure 2.1 for a, l than F, D. In Figure 2.2 there clearly *is* a lesser upper bound, as illustrated by the triple of arrows inside the square. In the following sections I render this condition in terms of categorical constructions and incorporate it in a new definition of labelled transitions. I then show that strong bisimulation is a congruence with respect to this labelled transition relation. A variety of preorders and other operational equivalences are discussed in the next chapter.

2.2 Categorical basis for contextual labels

The goal of this section is to find a tractable definition of a labelled transition relation, one which readily leads to congruential equivalences and preorders and moreover facilitates proofs concerning these relations. Intuitively, the labels represent just the information exchanged between an agent and its environment in order to make a reaction.

$$\begin{tikzcd}
{} \arrow[r, "a"] \arrow[d, "l"'] & {} \arrow[d, "F"] \arrow[ddr, bend left, "F'"] & \\
{} \arrow[r, "D"] \arrow[drr, bend right, "D'"'] & {} \arrow[dr, dotted, "G"] & \\
& & {}
\end{tikzcd}$$

2.3

Following the intuitions of the previous sections concerning Figure 2.1, the natural question is as follows. How can F, D be forced to contain no "junk"? A possible solution is to require that F and D are a "least upper bound" for a and l. The typical way to formulate this is to state that the square in Figure 2.3 is a *pushout*, i.e. has the property: $Fa = Dl$, and for every F' and D' satisfying $F'a = D'l$ there exists a unique G such that $GF = F'$ and $GD = D'$, as shown here.

Unfortunately, pushouts rarely exist in the categories that interest us. Consider, for example, a category of term contexts over a signature Σ; its objects consist of 0 and 1; its arrows $0 \rightarrow 1$ are terms over Σ; its arrows $1 \rightarrow 1$ are one-hole contexts over Σ; there are no arrows $1 \rightarrow 0$ and exactly one arrow $\mathsf{id}_0 : 0 \rightarrow 0$. Now, if Σ contains only constant symbols, say $\Sigma = \{\alpha, \alpha'\}$, then there is no pushout completing Figure 2.4(1) because there are no contexts other than the identity. If we introduce a 2-place function symbol β into Σ, we can construct an upper bound for α and α' but still no pushout (Figure 2.4(2)).

A more refined approach is to assert that F and D are a "minimal upper bound" — informally, an upper bound for which there are no lesser upper bounds. Before defining this notion in terms of *idem pushouts* (IPOs), I give a more basic construction, namely that of *relative pushouts* (RPOs). The latter, unlike pushouts, exist in many categories of agent contexts.

The plan for the rest of this section is to develop a sequence of propositions that will serve as a basis for the proofs of congruence by categorical reasoning given in a later section of this chapter and in subsequent chapters of the dissertation.

$$\begin{tikzcd}
0 \arrow[r, "\alpha"] \arrow[d, "\alpha'"'] & 1 \arrow[d, dotted, "\nexists"] \\
1 \arrow[r, dotted, "\nexists"'] & 1
\end{tikzcd}$$

2.4(1)

$$\begin{tikzcd}
0 \arrow[rr, "\alpha"] \arrow[dd, "\alpha'"'] & & 1 \arrow[dl, "\beta\langle\alpha',-\rangle"'] \arrow[dd, "\beta\langle-,\alpha'\rangle"] \\
& 1 \arrow[dr, dotted, bend left, "\nexists"] & \\
1 \arrow[ur, "\beta\langle-,\alpha\rangle"] \arrow[rr, "\beta\langle\alpha,-\rangle"'] & & 1 \arrow[ul, dotted, bend left, "\nexists"]
\end{tikzcd}$$

2.4(2)

Figure 2.4: Non-existence of pushouts

$$\begin{tikzcd}
\cdot \arrow[r, "f_0"] \arrow[d, "f_1"'] & \cdot \arrow[d, "g_0"] \\
\cdot \arrow[r, "g_1"'] & m
\end{tikzcd}$$

2.5(1)

$$\begin{tikzcd}
\cdot \arrow[rr, "f_0"] \arrow[dd, "f_1"'] & & \cdot \arrow[dl, "h_0"'] \arrow[dd, "g_0"] \\
& \cdot \arrow[dr, "h"] & \\
\cdot \arrow[ur, "h_1"] \arrow[rr, "g_1"'] & & m
\end{tikzcd}$$

2.5(2)

$$\begin{tikzcd}
\cdot \arrow[rrr, "f_0"] \arrow[dd, "f_1"'] & & & \cdot \arrow[dll, "h_0"'] \arrow[dl, "h'_0"] \arrow[dd, "g_0"] \\
& \cdot \arrow[r, dotted, "k"] \arrow[drr, "h"] & \cdot \arrow[dr, "h'"] & \\
\cdot \arrow[ur, "h_1"] \arrow[urr, bend right, "h'_1"'] \arrow[rrr, "g_1"'] & & & m
\end{tikzcd}$$

2.5(3)

Figure 2.5: Construction of an RPO

Because RPOs and IPOs are categorical constructions independent of reactive systems, I shall work in this section with an arbitrary category **C** whose arrows and objects I denote by f, g, h, k, x, y, z and m, n; in pictures I omit labels on the objects when possible.

Definition 2.4 (RPO) In any category **C**, consider a commuting square (Figure 2.5(1)) consisting of $g_0 f_0 = g_1 f_1$. An *RPO* is a triple h_0, h_1, h satisfying two properties:

commutation: $h_0 f_0 = h_1 f_1$ and $hh_i = g_i$ for $i = 0, 1$ (Figure 2.5(2));

universality: for any h'_0, h'_1, h' satisfying $h'_0 f_0 = h'_1 f_1$ and $h'h'_i = g_i$ for $i = 0, 1$, there exists a unique mediating arrow k such that $h'k = h$ and $kh_i = h'_i$ (Figure 2.5(3)). ■

A triple, such as h'_0, h'_1, h' given above, that satisfies the commutation property, i.e. $h'_0 f_0 = h'_1 f_1$ and $h'h'_i = g_i$ for $i = 0, 1$, is often called a *candidate*. Thus an RPO triple is a candidate for which there is a unique mediating arrow from it to any other candidate.

An RPO for Figure 2.5(1) is just a pushout in the slice category of **C** over m. Thus an RPO is a standard combination of categorical constructions — though it is not commonly used in category theory and its application to reactive systems is novel.

In later chapters, I illustrate the existence of RPOs for categories of graphs. For concreteness, though, it is worth examining now the example of an RPO and another

$$\begin{tikzcd}
\cdot \arrow[rrr, "\beta\langle\alpha\rangle"] \arrow[dd, "\alpha"'] & & & \cdot \arrow[dll, "-"'] \arrow[dl, "\gamma\langle-\rangle"] \arrow[dd, "\delta\langle\gamma\langle-\rangle\rangle"] \\
& \cdot \arrow[r, dotted, "\gamma\langle-\rangle"] \arrow[drr, "\delta\langle\gamma\langle-\rangle\rangle"] & \cdot \arrow[dr, "\delta\langle-\rangle"] & \\
\cdot \arrow[ur, "\beta\langle-\rangle"] \arrow[urr, "\gamma\langle\beta\langle-\rangle\rangle"] \arrow[rrr, "\delta\langle\gamma\langle\beta\langle-\rangle\rangle\rangle"'] & & & \cdot
\end{tikzcd}$$

Figure 2.6: An example of an RPO and another candidate

candidate shown in Figure 2.6. The arrows are in a category of term algebra contexts over the signature $\{\alpha, \beta, \gamma, \delta\}$, where α is a constant and β, γ, δ are 1-place function symbols. The RPO triple $-, \beta\langle-\rangle, \delta\langle\gamma\langle-\rangle\rangle$ adds just the minimal extra bit of context $\beta\langle-\rangle$ to α in order to get an upper bound for $\beta\langle\alpha\rangle$ and α; the arrow $\delta\langle\gamma\langle-\rangle\rangle$ then provides the extra junk necessary to recover the upper bound provided by the surrounding square. The reader may enjoy checking that the candidate triple $\gamma\langle-\rangle, \gamma\langle\beta\langle-\rangle\rangle, \delta\langle-\rangle$ is the only other non-trivial one possible and that the mediating dotted arrow $\gamma\langle-\rangle$ is unique.

A square is called an IPO if it has an RPO of a special kind:

Definition 2.5 (IPO) The commuting square in Figure 2.5(1) is an IPO if the triple g_0, g_1, id_m is an RPO. ■

The difference between a pushout and an IPO is clearest in a partial order category: a pushout is a least upper bound (i.e. less than any other upper bound) and an IPO is a minimal upper bound (i.e. not greater than any other upper bound). IPOs form the basis of our abstract definition of labelled transition and their existence follows from that of RPOs as shown by the proposition immediately after the following lemma:

Lemma 2.6 If Figure 2.7 is an RPO diagram and $j : n \rightarrow n$ satisfies

$$\begin{tikzcd}
\cdot \arrow[rr, "f_0"] \arrow[dd, "f_1"'] & & \cdot \arrow[dl, "h_0"'] \arrow[dd, "g_0"] \\
& n \arrow[dr, "h"] & \\
\cdot \arrow[ur, "h_1"] \arrow[rr, "g_1"'] & & m
\end{tikzcd} \quad 2.7$$

$$hj = h$$
$$jh_i = h_i \qquad (i = 0, 1)$$

then $j = \mathsf{id}_n$.

Proof If we think of the triple h_0, h_1, h as a candidate itself, then by the RPO property there exists a unique $j' : n \rightarrow n$ such that:

$$\begin{aligned} hj' &= h \\ j'h_i &= h_i \qquad (i = 0, 1) \end{aligned}$$

Therefore $j = j' = \mathsf{id}_n$. ■

Proposition 2.7 (IPOs from RPOs) If Figure 2.8 is an RPO diagram then the square in Figure 2.9 is an IPO.

$$\begin{tikzcd} \cdot \arrow[rr, "f_0"] \arrow[dd, "f_1"'] & & \cdot \arrow[dl, "h_0"'] \arrow[dd, "g_0"] \\ & n \arrow[dr, "h"] & \\ \cdot \arrow[ur, "h_1"] \arrow[rr, "g_1"'] & & m \end{tikzcd}$$

2.8

$$\begin{tikzcd} \cdot \arrow[r, "f_0"] \arrow[d, "f_1"'] & \cdot \arrow[d, "h_0"] \\ \cdot \arrow[r, "h_1"'] & n \end{tikzcd}$$

2.9

Proof Consider any candidate triple h'_0, h'_1, h' inside Figure 2.9; the components satisfy the following equations:

$$\begin{aligned} h'_0 f_0 &= h'_1 f_1 \\ h' h'_i &= h_i \qquad (i = 0, 1) \end{aligned} \tag{2.2}$$

We can therefore shift this candidate over to Figure 2.8 by considering the triple h'_0, h'_1, hh'. Since Figure 2.8 is an RPO, there exists a unique k such that:

$$hh'k = h \tag{2.3}$$

$$kh_i = h'_i \qquad (i = 0, 1) \tag{2.4}$$

Therefore, $h'kh_i =^{(2.4)} h'h'_i =^{(2.2)} h_i$. By Lemma 2.6, $h'k = \mathsf{id}_m$. Uniqueness: suppose k' satisfies: $h'k' = \mathsf{id}_m$ and $k'h_i = h'_i$. Then k' has the same property as k in (2.3) and (2.4), therefore $k' = k$ as desired. ■

The next result provides a partial converse to the previous proposition. It serves as a key part of the proof of IPO pasting which comes afterwards:

Proposition 2.8 (RPOs from IPOs) If Figure 2.10 is an IPO and Figure 2.11 has an RPO then Figure 2.12 is an RPO.

$$\begin{tikzcd} \cdot \arrow[r, "f_0"] \arrow[d, "f_1"'] & \cdot \arrow[d, "h_0"] \\ \cdot \arrow[r, "h_1"'] & m \end{tikzcd}$$

2.10

$$\begin{tikzcd} \cdot \arrow[r, "f_0"] \arrow[d, "f_1"'] & \cdot \arrow[d, "hh_0"] \\ \cdot \arrow[r, "hh_1"'] & \cdot \end{tikzcd}$$

2.11

$$\begin{tikzcd} \cdot \arrow[rr, "f_0"] \arrow[dd, "f_1"'] & & \cdot \arrow[dl, "h_0"'] \arrow[dd, "hh_0"] \\ & m \arrow[dr, "h"] & \\ \cdot \arrow[ur, "h_1"] \arrow[rr, "hh_1"'] & & \cdot \end{tikzcd}$$

2.12

Proof Let g_0, g_1, g be an RPO for Figure 2.11 and suppose $\mathsf{Cod}\, g_0 = \mathsf{Cod}\, g_1 = \mathsf{Dom}\, g = m'$. Then h_0, h_1, h is a candidate, hence there exists a unique $k : m' \rightarrow m$ such that:

$$hk = g \tag{2.5}$$

$$kg_i = h_i \qquad (i = 0, 1) \tag{2.6}$$

$$\begin{tikzcd}
{} \arrow[rrr, "f_0"] \arrow[dd, "f_1"'] & & & {} \arrow[dll, "g_0"'] \arrow[dl, "h_0"] \arrow[dd, "hh_0"] \\
& m' \arrow[r, dotted, bend left, "k"] \arrow[drr, "g"] & m \arrow[l, dotted, bend left, "j"] \arrow[dr, "h"] & \\
{} \arrow[ur, "g_1"] \arrow[urr, "h_1"'] \arrow[rrr, "hh_1"'] & & & {}
\end{tikzcd}$$

Therefore, g_0, g_1, k is a candidate for Figure 2.10, hence there exists a unique $j : m \to m'$ such that:

$$kj = \mathsf{id}_m \tag{2.7}$$

$$jh_i = g_i \qquad (i = 0, 1) \tag{2.8}$$

Now

$$gjk =^{(2.5)} hkjk =^{(2.7)} hk =^{(2.5)} g$$
$$jkg_i =^{(2.6)} jh_i =^{(2.8)} g_i$$

Therefore by Lemma 2.6, $jk = \mathsf{id}_{m'}$, which in conjunction with (2.7), implies that j and k are isos. Thus Figure 2.12 is an RPO, as desired. ■

IPOs can be pasted together as shown by the following proposition, which is analogous to the standard pasting result for pushouts.

Proposition 2.9 (IPO pasting) Suppose that both squares in Figure 2.13 commute and that Figure 2.14 has an RPO. Then the following properties hold of Figure 2.13:

$$\begin{tikzcd}
{} \arrow[r, "f_0"] \arrow[d, "x"'] & {} \arrow[r, "g_0"] \arrow[d, "y"] & {} \arrow[d, "z"] \\
{} \arrow[r, "f_1"'] & n \arrow[r, "g_1"'] & m
\end{tikzcd}$$

2.13

$$\begin{tikzcd}
{} \arrow[rr, "f_0"] \arrow[dd, "x"'] & & {} \arrow[d, "g_0"] \\
& & {} \arrow[d, "z"] \\
{} \arrow[r, "f_1"'] & n \arrow[r, "g_1"'] & m
\end{tikzcd}$$

2.14

1. If the two squares are IPOs then so is the big rectangle.

2. If the big rectangle and the left square are IPOs then so is the right square.

Proof

$$\begin{tikzcd}
{} \arrow[r, "f_0"] \arrow[dd, "x"'] & {} \arrow[r, "g_0"] & {} \arrow[dl, "h_0"'] \arrow[dd, "z"] \\
& n' \arrow[dr, "h"] & \\
{} \arrow[ur, "h_1"] \arrow[r, "f_1"'] & n \arrow[r, "g_1"'] & m
\end{tikzcd}$$

2.15

1. Let h_0, h_1, h be a candidate for the big rectangle of Figure 2.13, i.e. $h_0 g_0 f_0 = h_1 x$, $hh_0 = z$, and $hh_1 = g_1 f_1$, as shown in Figure 2.15.

   By hypothesis, the left square of Figure 2.13 is an IPO and Figure 2.14 has RPOs; therefore by Proposition 2.8, y, f_1, g_1 is an RPO for Figure 2.14. By construction, $h_0 g_0, h_1, h$ is a candidate, hence there exists a unique $k : n \rightarrow n'$ such that:

$$\begin{aligned} hk &= g_1 \\ ky &= h_0 g_0 \\ kf_1 &= h_1 \end{aligned} \tag{2.9}$$

   Therefore h_0, k, h is a candidate for the right square of Figure 2.13. Hence there exists a unique $j : m \rightarrow n'$ such that

$$hj = \mathsf{id}_m \tag{2.10}$$
$$jg_1 = k \tag{2.11}$$
$$jz = h_0 \tag{2.12}$$

$$\begin{tikzcd}
{} \arrow[rrrr, "f_0"] \arrow[ddd, "x"'] & & & & {} \arrow[dlll, "y"'] \arrow[dl, "g_0"'] \arrow[d, "g_0"] \\
& & & {} \arrow[dl, "h_0"'] & {} \arrow[dd, "z"] \\
& n \arrow[r, dotted, "k"] \arrow[drrr, "g_1"'] & n' \arrow[drr, "h"] & & \\
{} \arrow[ur, "f_1"] \arrow[urr, "h_1"'] \arrow[rr, "f_1"'] & & n \arrow[rr, "g_1"'] & & m \arrow[ull, dotted, bend right, "j"']
\end{tikzcd}$$

   Now

$$jg_1 f_1 =^{(2.11)} kf_1 =^{(2.9)} h_1 \tag{2.13}$$

   By (2.10), (2.12), and (2.13) j is a mediating arrow. Uniqueness: suppose $j' : m \rightarrow n'$ satisfies the same specification:

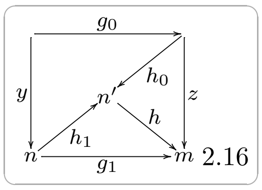
$$hj' = \mathsf{id}_m \tag{2.14}$$
$$j' g_1 f_1 = h_1 \tag{2.15}$$
$$j' z = h_0 \tag{2.16}$$

Then $j'g_1$ satisfies k 's universal property because of (2.15) and

$$hj'g_1 =^{(2.14)} g_1$$
$$j'g_1y =^{\text{Figure 2.13}} j'zg_0 =^{(2.16)} h_0g_0$$

Hence $j'g_1 = k$ by the uniqueness of k. Therefore j' satisfies the universal property of j, and hence $j' = j$ as desired.

2. Let h_0, h_1, h be a candidate for the right square of Figure 2.13, i.e. $h_0g_0 = h_1y$, $hh_0 = z$, and $hh_1 = g_1$, as shown in Figure 2.16. Then h_0, h_1f_1, h is a candidate for the big rectangle of Figure 2.13 therefore there exists a unique $j : m \rightarrowtail n'$ such that



$$\begin{tikzcd} \cdot \arrow[rr, "g_0"] \arrow[dd, "y"'] & & \cdot \arrow[dl, "h_0"] \arrow[dd, "z"] \\ & n' \arrow[dr, "h"] & \\ n \arrow[ur, "h_1"] \arrow[rr, "g_1"'] & & m \end{tikzcd}$$

2.16

$$hj = \mathsf{id}_m \tag{2.17}$$
$$jz = h_0 \tag{2.18}$$
$$jg_1f_1 = h_1f_1 \tag{2.19}$$

By hypothesis, the left square of Figure 2.13 is an IPO and Figure 2.14 has an RPO; therefore by Proposition 2.8, y, f_1, g_1 is an RPO for Figure 2.14. By construction, h_0g_0, h_1f_1, h is a candidate. Therefore there exists a unique $k : n \rightarrowtail n'$ such that:

$$hk = g_1$$
$$ky = h_0g_0$$
$$kf_1 = h_1f_1$$

Then h_1 satisfies the universal property of k, therefore $k = h_1$. Now, jg_1 satisfies the same universal property because of (2.19) and

$$hjg_1 =^{(2.17)} g_1$$
$$jg_1y =^{\text{Figure 2.13}} jzg_0 =^{(2.18)} h_0g_0$$

Therefore $jg_1 = k = h_1$ and hence by (2.17) and (2.18), j is the required mediating arrow. Uniqueness: suppose j' is also a mediating arrow. Then $hj' = \mathsf{id}_m$, $j'z = h_0$, and $j'g_1 = h_1$; the first two are analogues of (2.17) and (2.18); postmultiplying the last equation by f_1 gives the an analogue of (2.19). Therefore $j' = j$ as desired. ■

Finally, I conclude this collection of categorical results with three concerning IPOs; they are not immediately relevant to this chapter, but play an important role later in the dissertation.

The first asserts that if the left-leg of an IPO is an iso then so is the right-leg.

Proposition 2.10 If the outside square in Figure 2.17 is an IPO and f_0 is an iso then f_1 is an iso.

$$\begin{tikzcd}
{} \arrow[rr, "g_0"] \arrow[dd, "f_0"'] & & m \arrow[dl, "\mathsf{id}_m"'] \arrow[dd, "f_1"] \\
& m \arrow[dr, "f_1"] & \\
{} \arrow[ur, "g_0 f_0^{-1}"] \arrow[rr, "g_1"'] & & n
\end{tikzcd}$$

2.17

Proof Let f_0^{-1} be the inverse of f_0. Then all the triangles in Figure 2.17 commute. Therefore, there exists (a unique) $k : n \to m$ such that $f_1 k = \mathsf{id}_n$, $k f_1 = \mathsf{id}_m$, and $k g_1 = g_0 f_0^{-1}$. The first two conditions imply that f_1 is an iso. ■

The second asserts that IPOs can arise from epis.

Proposition 2.11 Suppose f_1 is an epi. Then the outer square in Figure 2.18 is an IPO.

$$\begin{tikzcd}
{} \arrow[rr, "f_0"] \arrow[dd, "f_1"'] & & {} \arrow[dl, "x_0"] \arrow[dd, "\mathsf{id}"] \\
& {} \arrow[dr, "x"] & \\
{} \arrow[ur, "x_1"] \arrow[rr, "g_1"'] & & {} \arrow[ul, dotted, bend left, "x_0"']
\end{tikzcd}$$

2.18

Proof Consider any candidate x_0, x_1, x for Figure 2.18, i.e.

$$x_0 f_0 = x_1 f_1 \tag{2.20}$$

$$x x_0 = \mathsf{id} \tag{2.21}$$

$$x x_1 = g_1 \tag{2.22}$$

Then $x_0 g_1 f_1 =^{\text{Figure 2.18}} x_0 f_0 =^{(2.20)} x_1 f_1$. Since f_1 is an epi, $x_0 g_1 = x_1$. This last equation plus (2.21) imply that x_0 is a mediating arrow. It is unique because any mediating arrow k must satisfy $k\,\mathsf{id} = x_0$, i.e. $k = x_0$. ■

The final result asserts that only a subcategory of $\mathbf{C}$ plays any role in characterising that a particular square is an IPO.

Proposition 2.12 Let m, m' be objects of $\mathbf{C}$ and let $\mathbf{C}'$ be a full subcategory of $\mathbf{C}$ satisfying the following property:

$$\mathsf{obj}\,\mathbf{C}' \supseteq \{n \in \mathsf{obj}\,\mathbf{C} \ \ / \ \ \exists h \in \mathbf{C}(m,n) \ \& \ \exists h' \in \mathbf{C}(n,m')\} \ .$$

$$\begin{tikzcd}
m \arrow[r, "f_0"] \arrow[d, "f_1"'] & {} \arrow[d, "g_0"] \\
{} \arrow[r, "g_1"'] & m'
\end{tikzcd}$$

2.19

Suppose the square in Figure 2.19 commutes, where $f_i, g_i \in \mathbf{C}'$ for $i = 0, 1$. Then the square is an IPO in $\mathbf{C}$ iff it is an IPO in $\mathbf{C}'$.

Proof The only arrows relevant to the square being an IPO in $\mathbf{C}$ are contained in $\mathbf{C}'$. ■

2.3 Labelled transitions and congruence for strong bisimulation

The category theory developed in the previous section provides the machinery needed in this section to accomplish two aims. The first is to improve the unsatisfactory definition of labelled transition given earlier (Definition 2.3). The second is to prove that strong bisimulation over the new labelled transitions is a congruence. I return to the notations of Section 2.1, using $\mathbf{C}$ for a reactive system, with $a, b \in \mathbf{C}$ ranging over arrows with domain 0 (agents) and $C, D, F \in \mathbf{C}$ ranging over arbitrary arrows (contexts).

The new version of labelled transitions is a modification of the approximation given by Definition 2.3, where the condition $Fa = Dl$ is strengthened to require that the square in Figure 2.20 is an IPO:

$$\begin{tikzcd} 0 \arrow[r, "a"] \arrow[d, "l"'] & \cdot \arrow[d, "F"] \\ \cdot \arrow[r, "D"'] & \cdot \end{tikzcd}$$

2.20

Definition 2.13 (labelled transition) $a \xrightarrow{F}\!\!\triangleright a'$ iff there exists $(l, r) \in \mathsf{Reacts}$ and $D \in \mathbf{D}$ such that Figure 2.20 is an IPO and $a' = Dr$. ■

This definition assures that F, D provides a minimal upper bound on a and l, as required in Section 2.1. For suppose there is another upper bound F', D', i.e. $F'a = D'l$, and also $F = RF'$ and $D = RD'$ for some R. Then the IPO property for Figure 2.20 ensures that for some R' (with $RR' = \mathsf{id}$) we have $F' = R'F$ and $D' = R'D$ — so F, D provides a "lesser" upper bound than F', D' after all.

Proposition 2.14 For all contexts F we have that $a \xrightarrow{F}\!\!\triangleright a'$ implies $Fa \longrightarrow\!\!\triangleright a'$. ■

The converse fails in general (which is good, given the remarks made after Definition 2.3 about the first approximation for labelled transitions). I return to the converse property later in Section 3.4 in the special case that F is an iso. Strong bisimulation over $\overset{\cdot}{\longrightarrow}\!\!\triangleright$ follows its usual scheme [Par81]:

Definition 2.15 (strong bisimulation over $\overset{\cdot}{\longrightarrow}\!\!\triangleright$) Let $\mathcal{S} \subseteq \bigcup_{m \in \mathsf{obj}\,\mathbf{C}} \mathbf{C}(0, m)^2$ be a relation that contains pairs of agents with common codomain. $\mathcal{S}$ is a *simulation* over $\overset{\cdot}{\longrightarrow}\!\!\triangleright$ iff $\mathcal{S}$ satisfies the following property for all $(a, b) \in \mathcal{S}$: if $a \xrightarrow{F}\!\!\triangleright a'$ then there exists b' such that $b \xrightarrow{F}\!\!\triangleright b'$ and $(a', b') \in \mathcal{S}$. $\mathcal{S}$ is a *strong bisimulation* iff $\mathcal{S}$ and $\mathcal{S}^{-1}$ are strong simulations. Let $\sim$ be the *largest strong bisimulation* over $\overset{\cdot}{\longrightarrow}\!\!\triangleright$. ■

I now state and prove the congruence result for strong bisimulation, one of the central results of this dissertation: if $\mathbf{C}$ has a sufficiently rich collection of RPOs then $\sim$ is a congruence.

Definition 2.16 (C has all redex-RPOs) Say that $\mathbf{C}$ *has all redex-RPOs* if for all $(l, r) \in \mathsf{Reacts}$ and arrows a, F, D such that $D \in \mathbf{D}$ and $Fa = Dl$, the square in Figure 2.20 has an RPO. ■

Theorem 2.17 (congruence for $\sim$) Let $\mathbf{C}$ be a reactive system which has all redex-RPOs. Then $\sim$ is a congruence, i.e. $a \sim b$ implies $Ca \sim Cb$ for all $C \in \mathbf{C}$ with required domain.

Proof By symmetry, it is sufficient to show that the following relation is a strong simulation:

$$\mathcal{S} \mathrel{\hat{=}} \{(Ca, Cb) \ / \ a \sim b \text{ and } C \in \mathbf{C}\} \,.$$

The proof falls into three parts, each of which is an implication as illustrated in Figure 2.21(1). Dashed lines connect pairs of points contained within the relation annotating the line. Each arrow " $\Downarrow$ " is tagged by the part of the proof below that justifies the implication. Suppose that $a \sim b$ and $C \in \mathbf{C}$, and thus $(Ca, Cb) \in \mathcal{S}$.

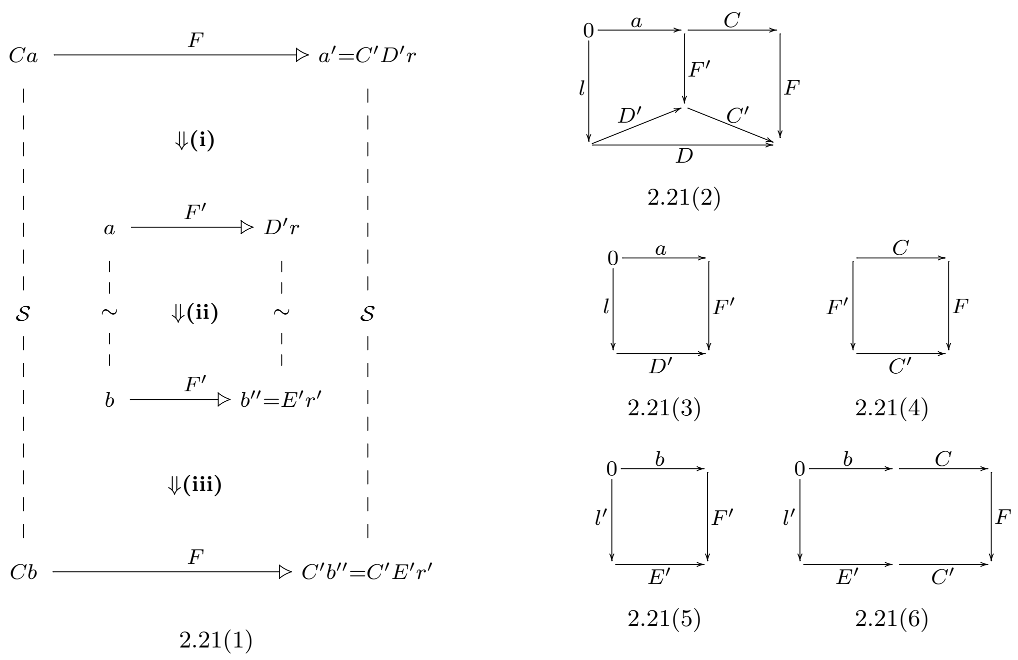


Figure 2.21: Congruence proof for strong bisimulation

(i): If $Ca \xrightarrow{F} a'$ then, by definition, there exists $(l, r) \in \mathsf{Reacts}$ and $D \in \mathbf{D}$ such that the big rectangle in Figure 2.21(2) is an IPO and $a' = Dr$. Because $\mathbf{C}$ has all redex-RPOs, there exists F', D', C' forming an RPO as in Figure 2.21(2); moreover, $D', C' \in \mathbf{D}$ since $C'D' = D \in \mathbf{D}$. By Proposition 2.7, Figure 2.21(3) is an IPO. Because $\mathbf{C}$ has all redex-RPOs, Proposition 2.9 implies that Figure 2.21(4) is an IPO too. By definition, $a \xrightarrow{F'} D'r$ and $a' = C'D'r$.

(ii): Since $a \sim b$, there exists b'' such that $b \xrightarrow{F'} b''$ and $D'r \sim b''$. By definition there exists $(l', r') \in \mathsf{Reacts}$ and $E' \in \mathbf{D}$ such that Figure 2.21(5) is an IPO and $b'' = E'r'$.

(iii): Because $\mathbf{C}$ has all redex-RPOs, Proposition 2.9 implies that we can paste Figure 2.21(5) with Figure 2.21(4) (both IPOs) along F' and conclude that Figure 2.21(6) is an IPO. Hence $Cb \xrightarrow{F} C'E'r'$ and $(C'D'r, C'E'r') \in \mathcal{S}$ because $D'r \sim E'r'$, as desired. ■

The crux of the above proof is that Figure 2.21(4), which mediates between an F'-labelled transition of a and an F-labelled transition of Ca, is "portable", i.e. can be pasted onto a new diagram, serving the same function for b and Cb. This essential idea appears to be robust under variation both of the definition of labelled transition and of the congruence being established. Many examples are shown in Chapter 3.

We can isolate precisely in two lemmas how such portable IPO squares are cut and then pasted. These lemmas are just pieces of the congruence proof above, but their factorisation from the main proof greatly simplifies the latter and lays the ground for tractable presentations of more difficult congruences results in the next chapter.

The first lemma shows that portable IPO squares arise when a composite agent has a labelled transition:

Lemma 2.18 (portable IPO cutting) If **C** has all redex-RPOs then the following inference rule holds:

$$\frac{Ca \xrightarrow{F} a'}{\exists \quad a'' \text{ and an IPO } \begin{tikzcd} {} \arrow[r, "C"] \arrow[d, "F'"'] & {} \arrow[d, "F"] \\ {} \arrow[r, "C'"'] & {} \end{tikzcd} . \qquad a \xrightarrow{F'} a'' \qquad a' = C'a'' \qquad C' \in \mathbf{D}} \ .$$

■

The second shows how to "paste" a portable square to gain a labelled transition of a composite agent:

Lemma 2.19 (portable IPO pasting) If **C** has all redex-RPOs then the following inference rule holds:

$$\frac{\begin{tikzcd} {} \arrow[r, "C"] \arrow[d, "F'"'] & {} \arrow[d, "F"] \\ {} \arrow[r, "C'"'] & {} \end{tikzcd} \text{ is an IPO} \qquad a \xrightarrow{F'} a'' \qquad C' \in \mathbf{D}}{Ca \xrightarrow{F} C'a''} \ .$$

■

We can now replay the proof of Theorem 2.17 in a more concise form by employing these lemmas:

$$\begin{tikzcd} {} \arrow[r, "C"] \arrow[d, "F'"'] & {} \arrow[d, "F"] \\ {} \arrow[r, "C'"'] & {} \end{tikzcd} \quad 2.22$$

(i): If $Ca \xrightarrow{F} a'$ then by Lemma 2.18, there exists a'' and an IPO square shown in Figure 2.22 such that

$$a \xrightarrow{F'} a'' \qquad a' = C'a'' \qquad C' \in \mathbf{D} \ .$$

(ii): Since $a \sim b$, there exists b'' such that $b \xrightarrow{F'} b''$ and $a'' \sim b''$.

(iii): Since Figure 2.22 is an IPO and $C' \in \mathbf{D}$, Lemma 2.19 implies that $Cb \xrightarrow{F} C'b''$. Also, $a'' \sim b''$ implies $(C'a'', C'b'') \in \mathcal{S}$, as desired.

Let us return to Lemma 2.18 in order to expose an odd property. The Lemma contradicts the situation in many process calculi: normally, $Ca \xrightarrow{F}$ *does not necessarily imply* that a has any labelled transitions. In CCS, for example, $\mathbf{0} \mid x \xrightarrow{x}$ (using the traditional CCS non-contextual labels) but $\mathbf{0}$ has no transitions. As a result, in typical proofs of congruence for strong bisimulation, two cases are distinguished when considering the transition $Ca \xrightarrow{F}$ (using the notation of this chapter):

- C and F together conspire to create the transition without reference to a. In this case $Cb \xrightarrow{F}\triangleright$ holds without using the assumption that $a \sim b$. (For example, see "Case 2" on p. 98 of [Mil88].)
- Or, a, C, and F together conspire to create the transition, as in part **(i)** above.

Recasting the CCS example in terms of contextual labels, we have that $\mathbf{0} \mid x \xrightarrow{-|\bar{x}}\triangleright$. But then there *is* a contextual transition for $\mathbf{0}$, namely $\mathbf{0} \xrightarrow{-|x|\bar{x}}\triangleright$, though not a satisfactory one: the label provides the entire redex, without any contribution from $\mathbf{0}$. This is attributable to a defect in the definition of contextual labelled transitions: if $a \xrightarrow{F}\triangleright$, the IPO property requires that F contain parts of the relevant redex and no extra junk, but does not prevent F from containing *all* of the redex.

It is by enriching the categorical structure to express multi-hole contexts (see Section 3.8) that we eliminate this defect of transitions. When we do, exactly the same case analysis (shown above) is carried out when proving congruence.

Chapter 3

Further congruence results

3.1 Introduction

This chapter generalises the definition of labelled transition given in the previous chapter and provides congruence proofs for additional equivalences and preorders. These include weak bisimulation, the traces preorder, and the failures preorder. I will describe them in turn as each congruence proof is presented.

The more important step, though, is the generalisation of reactive systems and labelled transitions. In the previous chapter, the central hypothesis required in the proof of congruence for strong bisimulation is that the reactive system $\mathbf{C}$ has all redex-RPOs. This chapter addresses the problem of what to do if $\mathbf{C}$ does not possess sufficient RPOs. Such a situation arises when considering, for example, a category of graph contexts (see Section 5.3). Roughly, the lack of RPOs is attributable to the absence of enough intensional information about the occurrence of nodes: it is ambiguous which node in a context corresponds to a node in the composition of the context with another. Thus if $C_0B_0 = C_1B_1$, it is ambiguous which nodes are common to both C_0 and C_1 and thus impossible to choose the context to be factored off when constructing an RPO.

What can be done when there are not enough RPOs in a reactive system? In general, it is not a good solution simply to enrich the reactive system to force it to have enough RPOs. The enrichment could yield a category with too much intensional information. For example, the enrichment considered for graph contexts (Section 5.6) forces arrows with the same domain and codomain to have the same number of nodes. Since the definition of strong bisimulation requires that $a \sim b$ implies that $a, b : 0 \rightarrow m$ for some object m, the strong bisimulation relation could only compare arrows of the same number of nodes. Such a restriction is unacceptable because strong bisimulation should include pairs of agents with widely differing static structure.

The solution presented in this chapter is to accommodate two categories $\hat{\mathbf{C}}$ and $\mathbf{C}$

related by a functor:

$$\begin{tikzcd}\hat{\mathbf{C}} \arrow[d, "\mathcal{F}"] \\ \mathbf{C}\end{tikzcd} .$$

The idea is that $\mathbf{C}$ is a reactive system, whose arrows correspond to the agents and agent contexts; $\mathbf{C}$ does not necessarily have enough RPOs. Sitting "above" $\mathbf{C}$ is $\hat{\mathbf{C}}$, a category with sufficient RPOs. The definition of the labelled transition *relates* arrows in $\mathbf{C}$, just as in the previous chapter: i.e. $a \xrightarrow{F} a'$ is defined for a, a' agents in $\mathbf{C}$ and F an agent context of $\mathbf{C}$. But, by contrast with the previous chapter, the definition is given in terms of the existence of an IPO "upstairs" in $\hat{\mathbf{C}}$. (If $\hat{\mathbf{C}} = \mathbf{C}$ and $\mathcal{F}$ is the identity functor, then the new definition of this chapter collapses into the old one given in the previous.)

Thus we get the best of both worlds: the agents whose labelled transition behaviour we consider need not contain any superfluous intensional data; as long as we can construct a more intensional category above containing sufficient RPOs and a suitable functor, then *we can get congruence results downstairs.*

These congruence results require the functor $\mathcal{F} : \hat{\mathbf{C}} \longrightarrow \mathbf{C}$ to satisfy certain properties. Most are trivial but one is interesting, namely that $\mathcal{F}$ *allows IPO sliding.* Recall from the previous chapter that the crux of the congruence proof for strong bisimulation was the portability of the IPO square that related F'-transitions of an agent to F-transitions of C applied to the agent. This square was cut off when passing from $Ca \xrightarrow{F}$ to $a \xrightarrow{F'}$ and then pasted back on when passing from $b \xrightarrow{F'}$ to $Cb \xrightarrow{F}$. In the new definition of labelled transition considered in this chapter, the pasting operation is more complex. The portable square, e.g. Figure 3.1, now lives in $\hat{\mathbf{C}}$ (the upstairs category) and its left leg is $\mathtt{F}'$, some arrow for which $\mathcal{F}(\mathtt{F}') = F'$. (Teletype font is used for arrows in $\hat{\mathbf{C}}$.) However, the transition $b \xrightarrow{F'}$ is justified by an IPO square upstairs whose right-leg is $\mathtt{F}'_0$, an arrow in the preimage of F' not necessarily equal to $\mathtt{F}'$. Thus Figure 3.1 cannot be pasted without first sliding it to a new IPO square, e.g. Figure 3.2, whose left-leg is $\mathtt{F}'_0$ and whose $\mathcal{F}$-image is kept invariant. The present chapter assumes that $\mathcal{F}$ allows IPO sliding; the next chapter proves that this is the case when $\mathcal{F}$ is of a certain general form.

$$\begin{tikzcd} \cdot \arrow[r, "\mathtt{C}"] \arrow[d, "\mathtt{F}'"'] & \cdot \arrow[d, "\mathtt{F}"] \\ \cdot \arrow[r, "\mathtt{C}'"'] & \cdot \end{tikzcd}$$
3.1

$$\begin{tikzcd} \cdot \arrow[r, "\mathtt{C}_0"] \arrow[d, "\mathtt{F}'_0"'] & \cdot \arrow[d, "\mathtt{F}_0"] \\ \cdot \arrow[r, "\mathtt{C}'_0"'] & \cdot \end{tikzcd}$$
3.2

The outline of this chapter is as follows. In the next section, I define the notion of a functorial reactive system, giving precise requirements for $\mathcal{F}$. Then I define the reaction and labelled transition relations. In the following section, I prove some results about portable IPO squares that are direct analogies to those (Lemma 2.18 and Lemma 2.19) at the end of the previous chapter. The main sections are concerned with a series of congruence proofs for strong bisimulation, weak bisimulation, the traces preorder, and the failures preorder. The final section treats a richer notion of functorial reactive system with arrows corresponding to multi-hole contexts and shows that strong bisimulation is indeed a congruence here as well.

3.2 Functorial reactive systems

The first part of the setup is to define precisely the notion of functorial reactive system, which was introduced informally above. Its definition rests on that of a reactive system, given in the previous chapter, which we recall here first for ease of reference:

Definition (reactive system recalled; see Definition 2.1) A *reactive system* consists of a category $\mathbf{C}$ with added structure. We let m, n range over objects. $\mathbf{C}$ has the following extra components:

- a distinguished object 0 (not necessarily initial);
- a set of *reaction rules* called $\mathsf{Reacts} \subseteq \bigcup_{m \in \mathsf{obj}\,\mathbf{C}} \mathbf{C}(0, m)^2$, a relation containing pairs of agents with common codomain;
- a subcategory $\mathbf{D}$ of $\mathbf{C}$, whose arrows are the *reactive contexts*, with the property that $D_1 D_0 \in \mathbf{D}$ implies $D_1, D_0 \in \mathbf{D}$. ■

Definition 3.1 (functorial reactive system) Let $\mathbf{C}$ be a reactive system. A *functorial reactive system over* $\mathbf{C}$ consists of a functor $\mathcal{F} : \hat{\mathbf{C}} \to \mathbf{C}$ which maps a distinguished object $\boldsymbol{\varepsilon} \in \mathsf{obj}\,\hat{\mathbf{C}}$ to 0 (the distinguished object of $\mathbf{C}$) and which satisfies the following properties.

$\mathcal{F}$ **lifts agents:** for any $a : 0 \to m$ there exists $\mathtt{a} : \boldsymbol{\varepsilon} \to \mathtt{m}$ such that $\mathcal{F}(\mathtt{a}) = a$.

$\mathcal{F}$ **creates isos:** if $\mathcal{F}(\mathtt{C})$ is an iso then $\mathtt{C}$ is an iso.

$\mathcal{F}$ **creates compositions:** if $\mathcal{F}(\mathtt{C}) = C_1 C_0$, there exist $\mathtt{C}_0, \mathtt{C}_1 \in \hat{\mathbf{C}}$ such that $\mathtt{C} = \mathtt{C}_1 \mathtt{C}_0$ and $\mathcal{F}(\mathtt{C}_i) = C_i$ for $i = 0, 1$.

$$\begin{tikzcd} {} \arrow[r, "\mathtt{C}"] \arrow[d, "\mathtt{F}'"'] & {} \arrow[d, "\mathtt{F}"] \\ {} \arrow[r, "\mathtt{C}'"'] & {} \end{tikzcd} \quad 3.3 \qquad \begin{tikzcd} {} \arrow[r, "\mathtt{C}_0"] \arrow[d, "\mathtt{F}'_0"'] & {} \arrow[d, "\mathtt{F}_0"] \\ {} \arrow[r, "\mathtt{C}'_0"'] & {} \end{tikzcd} \quad 3.4$$

$\mathcal{F}$ **allows IPO sliding:** for any IPO square as in Figure 3.3 and any arrow $\mathtt{F}'_0$ with $\mathcal{F}(\mathtt{F}'_0) = \mathcal{F}(\mathtt{F}')$ there exist $\mathtt{C}_0, \mathtt{C}'_0, \mathtt{F}_0$ forming an IPO square as in Figure 3.4 with

$$\mathcal{F}(\mathtt{C}_0) = \mathcal{F}(\mathtt{C}) \qquad \mathcal{F}(\mathtt{C}'_0) = \mathcal{F}(\mathtt{C}') \qquad \mathcal{F}(\mathtt{F}_0) = \mathcal{F}(\mathtt{F})\ .$$ ■

Throughout this chapter, I use uppercase teletype characters to denote arrows in $\hat{\mathbf{C}}$ and lowercase teletype characters $(\mathtt{a}, \mathtt{l}, \ldots)$ to denote arrows with domain $\boldsymbol{\varepsilon}$ in $\hat{\mathbf{C}}$.

The $\mathcal{F}$ images of these are agents in $\mathbf{C}$. The special domain requirement of $\mathtt{a}, \mathtt{l}, \ldots$ is left tacit throughout this chapter: thus $(\exists \mathtt{l} \in \hat{\mathbf{C}}.\ \ldots)$ means $(\exists \mathtt{l} \in \hat{\mathbf{C}}.\ \mathsf{Dom}\,\mathtt{l} = \boldsymbol{\varepsilon}\ \&\ \ldots)$.

The definition of the reaction relation is identical to the one given earlier:

Definition (reaction relation ($\longrightarrow\!\!\!\triangleright$) recalled; cf. Definition 2.2) Given a functorial reactive system $\mathcal{F} : \hat{\mathbf{C}} \to \mathbf{C}$, the *reaction relation* $\longrightarrow\!\!\!\triangleright \subseteq \bigcup_{m \in \mathsf{obj}\,\mathbf{C}} \mathbf{C}(0, m)^2$ contains pairs of agents with common codomain and is defined by lifting the reaction rules through all reactive contexts: $a \longrightarrow\!\!\!\triangleright a'$ iff there exists $D \in \mathbf{D}$ and $(l, r) \in \mathsf{Reacts}$ such that $a = Dl$ and $a' = Dr$. ■

This definition has an alternative characterisation given by the following result:

Proposition 3.2 (characterisation of $\rightarrowtriangle$) $a \rightarrowtriangle a'$ iff there exist $\mathtt{a}, \mathtt{l}, \mathtt{D} \in \hat{\mathbf{C}}$ and $r \in \mathbf{C}$ such that $\mathtt{a} = \mathtt{D}\mathtt{l}$ and

$$a' = \mathcal{F}(\mathtt{D})r \qquad \mathcal{F}(\mathtt{D}) \in \mathbf{D} \qquad (\mathcal{F}(\mathtt{l}), r) \in \mathsf{Reacts}$$
$$\mathcal{F}(\mathtt{a}) = a\ .$$

Proof Follows immediately because $\mathcal{F}$ lifts agents and creates compositions. ■

We now turn to the definition of labelled transition. As stated earlier, $\overset{\cdot}{\rightarrowtriangle}$ is a ternary relation whose arguments are all arrows in $\mathbf{C}$. The original requirement that a particular square be an IPO in $\mathbf{C}$ (see Definition 2.13) is replaced here by requiring that there exist a preimage of this square that is an IPO in $\hat{\mathbf{C}}$:

Definition 3.3 (labelled transition ($\overset{\cdot}{\rightarrowtriangle}$); cf. Definition 2.13) $a \overset{F}{\rightarrowtriangle} a'$ iff there exist $\mathtt{a}, \mathtt{l}, \mathtt{F}, \mathtt{D} \in \hat{\mathbf{C}}$ and $r \in \mathbf{C}$ such that Figure 3.5 is an IPO in $\hat{\mathbf{C}}$ and

$$\begin{tikzcd} {} \arrow[r, "\mathtt{a}"] \arrow[d, "\mathtt{l}"'] & {} \arrow[d, "\mathtt{F}"] \\ {} \arrow[r, "\mathtt{D}"'] & {} \end{tikzcd}$$
3.5

$$a' = \mathcal{F}(\mathtt{D})r \qquad \mathcal{F}(\mathtt{D}) \in \mathbf{D} \qquad (\mathcal{F}(\mathtt{l}), r) \in \mathsf{Reacts}$$
$$\mathcal{F}(\mathtt{a}) = a \qquad \mathcal{F}(\mathtt{F}) = F\ .$$
■

Notice that a', the RHS of the transition, is required to be $\mathcal{F}(\mathtt{D})r$, and *not* $\mathcal{F}(\mathtt{Dr})$ for some $\mathtt{r}$ with $\mathcal{F}(\mathtt{r}) = r$. This is important since it allows the reaction rules Reacts to contain pairs (l, r) for which $\mathcal{F}$-preimages of l and r might not have common codomains.

By analogy with Definition 2.16, we can define when a functorial reactive system $\mathcal{F}$ has all redex-RPOs; the primary difference is that here the RPOs exist upstairs in $\hat{\mathbf{C}}$, not downstairs in $\mathbf{C}$:

Definition 3.4 ($\mathcal{F}$ has all redex-RPOs; cf. Definition 2.16) A functorial reactive system $\mathcal{F} : \hat{\mathbf{C}} \longrightarrow \mathbf{C}$ *has all redex-RPOs* if any square, such as in Figure 3.6, has an RPO, provided that $\mathcal{F}(\mathtt{D}) \in \mathbf{D}$ and that there exists $r \in \mathbf{C}$ such that $(\mathcal{F}(\mathtt{l}), r) \in \mathsf{Reacts}$. ■

$$\begin{tikzcd} {} \arrow[r, "\mathtt{a}"] \arrow[d, "\mathtt{l}"'] & {} \arrow[d, "\mathtt{F}"] \\ {} \arrow[r, "\mathtt{D}"'] & {} \end{tikzcd}$$
3.6

Note that we do not demand that *all* RPOs exist, just ones for which the left-leg of the enclosing square is a preimage of a redex and the bottom leg is a preimage of an arrow in $\mathbf{D}$. (This narrowing of the definition is exactly analogous to what happens in the previous chapter.)

3.3 Cutting and pasting portable IPO squares

This section replays the results at the end of the previous chapter which show how to cut and paste portable IPO squares. The main difference lies in the proof of pasting: here we make explicit use of the assumption that $\mathcal{F}$ allows IPO sliding.

The first result shows how the transitions of composite agents yield IPO squares:

Lemma 3.5 (portable IPO cutting; cf. Lemma 2.18) Suppose $\mathcal{F} : \hat{\mathbf{C}} \to \mathbf{C}$ is a functorial reactive system and has all redex-RPOs. The following inference rule holds:

$$\frac{Ca \xrightarrow{F} a'}{\exists \quad a'' \in \mathbf{C} \text{ and an IPO square } \begin{tikzcd} \cdot \arrow[r, "\mathtt{C}"] \arrow[d, "\mathtt{F}'"'] & \cdot \arrow[d, "\mathtt{F}"] \\ \cdot \arrow[r, "\mathtt{C}'"'] & \cdot \end{tikzcd} . \quad \left(\begin{array}{ccc} a \xrightarrow{\mathcal{F}(\mathtt{F}')} a'' & a' = \mathcal{F}(\mathtt{C}')a'' & \mathcal{F}(\mathtt{C}') \in \mathbf{D} \\ & \mathcal{F}(\mathtt{C}) = C & \mathcal{F}(\mathtt{F}) = F \end{array} \right)} \; .$$

Proof By the definition of $\xrightarrow{F}$ and the hypothesis that $\mathcal{F}$ creates compositions, there exists $\mathtt{a}, \mathtt{C}, \mathtt{l}, \mathtt{F}, \mathtt{D} \in \hat{\mathbf{C}}$ and $r \in \mathbf{C}$ such that the big rectangle in Figure 3.7 is an IPO and

$$\begin{tikzcd} \cdot \arrow[r, "\mathtt{a}"] \arrow[dd, "\mathtt{l}"'] & \cdot \arrow[r, "\mathtt{C}"] \arrow[d, "\mathtt{F}'"] & \cdot \arrow[dd, "\mathtt{F}"] \\ & \cdot \arrow[dr, "\mathtt{C}'"] & \\ \cdot \arrow[ur, "\mathtt{D}'"] \arrow[rr, "\mathtt{D}"'] & & \cdot \end{tikzcd}$$

3.7

$$a' = \mathcal{F}(\mathtt{D})r \qquad \mathcal{F}(\mathtt{D}) \in \mathbf{D} \qquad (\mathcal{F}(\mathtt{l}), r) \in \mathsf{Reacts}$$
$$\mathcal{F}(\mathtt{a}) = a \qquad \mathcal{F}(\mathtt{C}) = C \qquad \mathcal{F}(\mathtt{F}) = F \; .$$

Because $\mathcal{F}$ has all redex-RPOs, there exist $\mathtt{F}', \mathtt{D}', \mathtt{C}'$ forming an RPO in $\hat{\mathbf{C}}$, as in Figure 3.7. Then $\mathcal{F}(\mathtt{C}') \in \mathbf{D}$ since $\mathcal{F}(\mathtt{C}')\mathcal{F}(\mathtt{D}') = \mathcal{F}(\mathtt{D}) \in \mathbf{D}$. By Proposition 2.7, the small left-hand square of Figure 3.7 is an IPO. Because $\mathcal{F}$ has all redex-RPOs, Proposition 2.9 implies that the small right-hand square is an IPO too. By definition, $a \xrightarrow{\mathcal{F}(\mathtt{F}')} a''$ and $a' = \mathcal{F}(\mathtt{C}')a''$ where $a'' \mathrel{\hat{=}} \mathcal{F}(\mathtt{D}')r$. ■

The next result shows how the reactions of composite agents can be decomposed. There is no analogue of this result the previous chapter since it is not needed in the congruence proof for strong bisimulation.

Lemma 3.6 (portable IPO cutting for reactions) Suppose $\mathcal{F} : \hat{\mathbf{C}} \to \mathbf{C}$ is a functorial reactive system and has all redex-RPOs. The following inference rule holds:

$$\frac{Ca \longrightarrow a'}{\exists \quad a'' \in \mathbf{C} \text{ and } \mathtt{C}', \mathtt{F}' \in \hat{\mathbf{C}}. \quad \left(\begin{array}{ccc} a \xrightarrow{\mathcal{F}(\mathtt{F}')} a'' & a' = \mathcal{F}(\mathtt{C}')a'' & \mathcal{F}(\mathtt{C}') \in \mathbf{D} \\ & \mathcal{F}(\mathtt{C}'\mathtt{F}') = C & \end{array} \right)} \; .$$

Moreover, if $\mathtt{F}'$ is an iso in the conclusion then it is equal to id.

Proof By Proposition 3.2 and the hypothesis that $\mathcal{F}$ creates compositions, there exist $\mathtt{a}, \mathtt{l}, \mathtt{C}, \mathtt{D} \in \hat{\mathbf{C}}$ and $r \in \mathbf{C}$ such that the big rectangle in Figure 3.8 commutes and

$$\begin{tikzcd} \cdot \arrow[r, "\mathtt{a}"] \arrow[dd, "\mathtt{l}"'] & \cdot \arrow[r, "\mathtt{C}"] \arrow[d, "\mathtt{F}'"] & \cdot \arrow[dd, "\mathsf{id}"] \\ & \cdot \arrow[dr, "\mathtt{C}'"] & \\ \cdot \arrow[ur, "\mathtt{D}'"] \arrow[rr, "\mathtt{D}"'] & & \cdot \end{tikzcd}$$

3.8

$$a' = \mathcal{F}(\mathtt{D})r \qquad \mathcal{F}(\mathtt{D}) \in \mathbf{D} \qquad (\mathcal{F}(\mathtt{l}), r) \in \mathsf{Reacts}$$
$$\mathcal{F}(\mathtt{a}) = a \qquad \mathcal{F}(\mathtt{C}) = C \; .$$

Because $\mathcal{F}$ has all redex-RPOs, there exist $\mathtt{F}', \mathtt{D}', \mathtt{C}'$ forming an RPO in $\hat{\mathbf{C}}$, as in Figure 3.8. Because RPOs are a universal construction, we can assume without loss of generality that if $\mathtt{F}'$ is an iso, it is equal to id. Then $\mathcal{F}(\mathtt{C}') \in \mathbf{D}$ since since $\mathcal{F}(\mathtt{C}')\mathcal{F}(\mathtt{D}') = \mathcal{F}(\mathtt{D}) \in \mathbf{D}$. By Proposition 2.7, the small left-hand square of Figure 3.8 is an IPO. By definition, $a \xrightarrow{\mathcal{F}(\mathtt{F}')} a''$ and $a' = \mathcal{F}(\mathtt{C}')a''$ where $a'' \mathrel{\hat{=}} \mathcal{F}(\mathtt{D}')r$. Finally, since the small right-hand square commutes, $\mathtt{C}'\mathtt{F}' = \mathtt{C}$ as desired. ■

The final result shows how to paste a portable IPO square in order to gain a transition for a composite agent. As stated above, this is where IPO sliding is used:

Lemma 3.7 (portable IPO pasting; cf. Lemma 2.19) Suppose $\mathcal{F} : \hat{\mathbf{C}} \to \mathbf{C}$ is a functorial reactive system and has all redex-RPOs. The following inference rule holds:

$$\frac{\begin{tikzcd} \cdot \arrow[r, "\mathtt{C}"] \arrow[d, "\mathtt{F}'"'] & \cdot \arrow[d, "\mathtt{F}"] \\ \cdot \arrow[r, "\mathtt{C}'"'] & \cdot \end{tikzcd} \text{ is an IPO} \qquad a \xrightarrow{\mathcal{F}(\mathtt{F}')} a' \qquad \mathcal{F}(\mathtt{C}') \in \mathbf{D}}{\mathcal{F}(\mathtt{C})a \xrightarrow{\mathcal{F}(\mathtt{F})} \mathcal{F}(\mathtt{C}')a'} \; .$$

Proof Since $a \xrightarrow{\mathcal{F}(\mathtt{F}')} a'$ there exist $\mathtt{a}, \mathtt{l}, \mathtt{F}'_0, \mathtt{D} \in \hat{\mathbf{C}}$ and $r \in \mathbf{C}$ such that Figure 3.9 is an IPO and

$$\begin{tikzcd} \cdot \arrow[r, "\mathtt{a}"] \arrow[d, "\mathtt{l}"'] & \cdot \arrow[d, "\mathtt{F}'_0"] \\ \cdot \arrow[r, "\mathtt{D}"'] & \cdot \end{tikzcd} \; 3.9 \qquad \begin{tikzcd} \cdot \arrow[r, "\mathtt{C}_0"] \arrow[d, "\mathtt{F}'_0"'] & \cdot \arrow[d, "\mathtt{F}_0"] \\ \cdot \arrow[r, "\mathtt{C}'_0"'] & \cdot \end{tikzcd} \; 3.10$$

$$a' = \mathcal{F}(\mathtt{D})r \qquad \mathcal{F}(\mathtt{D}) \in \mathbf{D} \qquad (\mathcal{F}(\mathtt{l}), r) \in \mathsf{Reacts}$$
$$\mathcal{F}(\mathtt{a}) = a \qquad \mathcal{F}(\mathtt{F}'_0) = \mathcal{F}(\mathtt{F}') \; .$$

Since $\mathcal{F}$ allows IPO sliding, there exist $\mathtt{C}_0, \mathtt{F}_0, \mathtt{C}'_0 \in \hat{\mathbf{C}}$ such that Figure 3.10 is an IPO and

$$\mathcal{F}(\mathtt{C}_0) = \mathcal{F}(\mathtt{C}) \qquad \mathcal{F}(\mathtt{F}_0) = \mathcal{F}(\mathtt{F}) \qquad \mathcal{F}(\mathtt{C}'_0) = \mathcal{F}(\mathtt{C}') \; .$$

$$\begin{tikzcd} \cdot \arrow[r, "\mathtt{a}"] \arrow[d, "\mathtt{l}"'] & \cdot \arrow[r, "\mathtt{C}_0"] & \cdot \arrow[d, "\mathtt{F}_0"] \\ \cdot \arrow[r, "\mathtt{D}"'] & \cdot \arrow[r, "\mathtt{C}'_0"'] & \cdot \end{tikzcd} \; 3.11$$

Because $\mathcal{F}$ has all redex-RPOs, Proposition 2.9 implies that we can paste Figure 3.9 with Figure 3.10 (both IPOs) along $\mathtt{F}'_0$ and conclude that Figure 3.11 is an IPO. Thus $\mathcal{F}(\mathtt{C})a \xrightarrow{\mathcal{F}(\mathtt{F})} \mathcal{F}(\mathtt{C}')\mathcal{F}(\mathtt{D})r = \mathcal{F}(\mathtt{C}')a'$, as desired. ■

That concludes the setup for functorial reactive systems. The rest of the chapter is devoted to a sequence of congruence proofs.

3.4 Strong bisimulation

This section proves that strong bisimulation is a congruence for a functorial reactive system. The definition is straightforward:

Definition 3.8 (strong bisimulation over $\xrightarrow{\cdot}$; cf. Definition 2.15) Let $\sim$ be the largest strong bisimulation over $\xrightarrow{\cdot}$. ■

The proof of congruence is almost identical to the one presented at the end of the previous chapter: the only difference is that the updated IPO cutting and pasting results for functorial reactive systems are substituted for the old ones.

$$\begin{tikzcd}
Ca \arrow[rrr, "F", -latex] \arrow[ddd, dash, dashed, "\mathcal{S}"'] \arrow[drr, phantom, "\Downarrow\textbf{(i)}"] & & & a'{=}\mathcal{F}(\mathtt{C}')a'' \arrow[ddd, dash, dashed, "\mathcal{S}"] \\
& a \arrow[r, "\mathcal{F}(\mathtt{F}')"] \arrow[d, dash, dashed, "\sim"'] \arrow[dr, phantom, "\Downarrow\textbf{(ii)}"] & a'' \arrow[d, dash, dashed, "\sim"] & \\
& b \arrow[r, "\mathcal{F}(\mathtt{F}')"] \arrow[dr, phantom, "\Downarrow\textbf{(iii)}"] & b'' & \\
Cb \arrow[rrr, "F"] & & & \mathcal{F}(\mathtt{C}')b''
\end{tikzcd}$$

Figure 3.12: Schema of the congruence proof for $\sim$

Theorem 3.9 (congruence for $\sim$) Let $\mathcal{F} : \hat{\mathbf{C}} \to \mathbf{C}$ be a functorial reactive system which has all redex-RPOs. Then $\sim$ is a congruence, i.e. $a \sim b$ implies $Ca \sim Cb$ for all $C \in \mathbf{C}$ of the required domain.

Proof By symmetry, it is sufficient to show that the following relation is a strong simulation:

$$\mathcal{S} \mathrel{\hat{=}} \{(Ca, Cb) \;\;/\;\; a \sim b \text{ and } C \in \mathbf{C}\} \;.$$

The proof falls into three parts, each of which is an implication as illustrated in Figure 3.12. Dashed lines connect pairs of points contained within the relation annotating the line. Each arrow "$\Downarrow$" is tagged by the part of the proof below that justifies the implication. Suppose that $a \sim b$ and $C \in \mathbf{C}$, and thus $(Ca, Cb) \in \mathcal{S}$.

$$\begin{tikzcd}
\cdot \arrow[r, "\mathtt{C}"] \arrow[d, "\mathtt{F}'"'] & \cdot \arrow[d, "\mathtt{F}"] \\
\cdot \arrow[r, "\mathtt{C}'"'] & \cdot
\end{tikzcd} \quad 3.13$$

(i): If $Ca \xrightarrow{F} a'$ then by Lemma 3.5, there exist $a'' \in \mathbf{C}$ and an IPO square shown in Figure 3.13 such that $a \xrightarrow{\mathcal{F}(\mathtt{F}')} a''$ and

$$a' = \mathcal{F}(\mathtt{C}')a'' \qquad \mathcal{F}(\mathtt{C}') \in \mathbf{D}$$
$$\mathcal{F}(\mathtt{C}) = C \qquad \mathcal{F}(\mathtt{F}) = F \;.$$

(ii): Since $a \sim b$, there exists b'' such that $b \xrightarrow{\mathcal{F}(\mathtt{F}')} b''$ and $a'' \sim b''$.

(iii): Since Figure 3.13 is an IPO and $\mathcal{F}(\mathtt{C}') \in \mathbf{D}$, Lemma 3.7 implies that $Cb \xrightarrow{F} \mathcal{F}(\mathtt{C}')b''$. Also, $a'' \sim b''$ implies $(\mathcal{F}(\mathtt{C}')a'', \mathcal{F}(\mathtt{C}')b'') \in \mathcal{S}$, as desired. ■

In most process calculi the reaction relation and the τ-labelled transition relation coincide. See, for example Proposition 5.2 in [Mil92] and Theorem 2 in [Sew00]. A τ

transition is a "silent move": a transition that takes place without interacting with the external environment. Intuitively, a τ-labelled transition corresponds to an id-labelled transition when using contexts as labels, i.e. $a \xrightarrow{\mathsf{id}} a'$ iff the environment need only supply a vacuous identity context in order to enable a to react. However, if we look carefully at the definition of labelled transition given in Definition 3.3 and the characterisation of reaction in Proposition 3.2, we see that $\xrightarrow{\mathsf{id}}$ and $\longrightarrow$ are not necessarily identical. There is an implication in one direction, namely $\xrightarrow{\mathsf{id}} \subseteq \longrightarrow$, since every IPO square is also a commuting square, but not necessarily the converse. Indeed, Example 5.2 (p. 77) contains a non-IPO commuting square whose bottom- and right-legs are both identity arrows.

In the special situation when all preimages of redexes are epis, $\xrightarrow{\mathsf{id}}$ and $\longrightarrow$ *do* coincide, thanks to Proposition 2.11. This situation is explored at the end of this section in Proposition 3.15.

Before taking the epi hypothesis on board let us a consider an alternate definition of labelled transition for which we do recover the reaction relation:

Definition 3.10 (labelled transition by cases ($\xrightarrow[\mathsf{c}]{\cdot}$))

$$a \xrightarrow[\mathsf{c}]{F} a' \quad \text{iff} \quad \begin{cases} Fa \longrightarrow a' & \text{if } F \text{ is an iso} \\ a \xrightarrow{F} a' & \text{if } F \text{ is not an iso .} \end{cases}$$

■

It follows immediately from the definition that $a \xrightarrow[\mathsf{c}]{\mathsf{id}} a'$ iff $a \longrightarrow a'$. Furthermore, the induced strong bisimulation (defined next) is a congruence, as shown below. It is worth considering whether there are other definitions that recover the reaction relation but do *not* involve case analysis. This point is taken up in Appendix B where I present a definition of labelled transition that satisfies this requirement; I show furthermore that this definition induces a congruence which includes $\sim_{\mathsf{c}}$. Let us return to the main flow of the argument now and consider $\sim_{\mathsf{c}}$ in detail.

Definition 3.11 (strong bisimulation over $\xrightarrow[\mathsf{c}]{\cdot}$) Let $\sim_{\mathsf{c}}$ be the largest strong bisimulation over $\xrightarrow[\mathsf{c}]{\cdot}$. ■

Theorem 3.12 (congruence for $\sim_{\mathsf{c}}$) Let $\mathcal{F} : \hat{\mathbf{C}} \to \mathbf{C}$ be a functorial reactive system which has all redex-RPOs. Then $\sim_{\mathsf{c}}$ is a congruence, i.e. $a \sim_{\mathsf{c}} b$ implies $Ca \sim_{\mathsf{c}} Cb$ for all $C \in \mathbf{C}$ of the required domain.

Proof By symmetry, it is sufficient to show that the following relation is a strong simulation:

$$\mathcal{S} \mathrel{\hat{=}} \{(Ca, Cb) \ / \ a \sim_{\mathsf{c}} b \text{ and } C \in \mathbf{C}\} \ .$$

Suppose that $a \sim_{\mathsf{c}} b$ and $C \in \mathbf{C}$, and thus $(Ca, Cb) \in \mathcal{S}$. Suppose $Ca \xrightarrow[\mathsf{c}]{F} a'$.

Case F is an iso: Then $Ca \longrightarrow F^{-1}a'$. By Lemma 3.6, there exist $\mathtt{C}', \mathtt{F}' \in \hat{\mathbf{C}}$ and $a'' \in \mathbf{C}$ such that $a \xrightarrow{\mathcal{F}(\mathtt{F}')} a''$, thus $a \xrightarrow[\mathsf{c}]{\mathcal{F}(\mathtt{F}')} a''$, and

$$a' = F\mathcal{F}(\mathtt{C}')a'' \qquad \mathcal{F}(\mathtt{C}') \in \mathbf{D} \qquad \mathcal{F}(\mathtt{C}'\mathtt{F}') = C \ .$$

Since $a \sim_{\mathsf{c}} b$, there exists b'' such that $b \xrightarrow[\mathsf{c}]{\mathcal{F}(\mathtt{F}')}\!\triangleright\, b''$ and $a'' \sim_{\mathsf{c}} b''$. Since $\mathcal{F}(\mathtt{C}') \in \mathbf{D}$, we have that $FCb = F\mathcal{F}(\mathtt{C}')\mathcal{F}(\mathtt{F}')b \longrightarrow\!\triangleright\, F\mathcal{F}(\mathtt{C}')b''$. Thus $Cb \xrightarrow[\mathsf{c}]{F}\!\triangleright\, F\mathcal{F}(\mathtt{C}')b''$. Since $a'' \sim_{\mathsf{c}} b''$, we have $(F\mathcal{F}(\mathtt{C}')a'', F\mathcal{F}(\mathtt{C}')b'') \in \mathcal{S}$, as desired.

Case F is not an iso: By definition $Ca \xrightarrow{F}\!\triangleright\, a'$. By Lemma 3.5, there exist $a'' \in \mathbf{C}$ and an IPO square shown in Figure 3.14 such that $a \xrightarrow{\mathcal{F}(\mathtt{F}')}\!\triangleright\, a''$ and

$$\begin{tikzcd} \cdot \arrow[r, "\mathtt{C}"] \arrow[d, "\mathtt{F}'"'] & \cdot \arrow[d, "\mathtt{F}"] \\ \cdot \arrow[r, "\mathtt{C}'"'] & \cdot \end{tikzcd}$$
3.14

$$a' = \mathcal{F}(\mathtt{C}')a'' \qquad \mathcal{F}(\mathtt{C}') \in \mathbf{D} \qquad \mathcal{F}(\mathtt{C}) = C \qquad \mathcal{F}(\mathtt{F}) = F\ .$$

Thus $a \xrightarrow[\mathsf{c}]{\mathcal{F}(\mathtt{F}')}\!\triangleright\, a''$. Since $a \sim_{\mathsf{c}} b$, there exists b'' such that $b \xrightarrow[\mathsf{c}]{\mathcal{F}(\mathtt{F}')}\!\triangleright\, b''$ and $a'' \sim_{\mathsf{c}} b''$. Since F is not an iso, $\mathtt{F}$ is not an iso; Proposition 2.10 implies that $\mathtt{F}'$ is also not an iso; since $\mathcal{F}$ creates isos, $\mathcal{F}(\mathtt{F}')$ is not an iso, thus $b \xrightarrow{\mathcal{F}(\mathtt{F}')}\!\triangleright\, b''$. Since Figure 3.14 is an IPO and $\mathcal{F}(\mathtt{C}') \in \mathbf{D}$, Lemma 3.7 implies that $Cb \xrightarrow{F}\!\triangleright\, \mathcal{F}(\mathtt{C}')b''$, so $Cb \xrightarrow[\mathsf{c}]{F}\!\triangleright\, \mathcal{F}(\mathtt{C}')b''$. Since $a'' \sim_{\mathsf{c}} b''$, we have $(\mathcal{F}(\mathtt{C}')a'', \mathcal{F}(\mathtt{C}')b'') \in \mathcal{S}$, as desired. ■

Because $\xrightarrow{\cdot}\!\triangleright$ and $\xrightarrow[\mathsf{c}]{\cdot}\!\triangleright$ are so closely related, it is not surprising that the induced congruences are also related. Indeed the following result shows that $\sim_{\mathsf{c}}$ is a coarser equivalence than $\sim$. It is an open question whether there exists a functorial reactive system for which the inequality is strict.

Proposition 3.13 ($\sim \subseteq \sim_{\mathsf{c}}$) Let $\mathcal{F} : \hat{\mathbf{C}} \to \mathbf{C}$ be a functorial reactive system which has all redex-RPOs. Then $\sim \subseteq \sim_{\mathsf{c}}$.

Proof We show that $\sim$ is a strong bisimulation with respect to the labelled transition relation $\xrightarrow[\mathsf{c}]{\cdot}\!\triangleright$. By symmetry, it is enough to shown that $\sim$ is a strong simulation over $\xrightarrow[\mathsf{c}]{\cdot}\!\triangleright$. Consider any a, b for which $a \sim b$. Suppose $a \xrightarrow[\mathsf{c}]{F}\!\triangleright\, a'$.

Case F is an iso: Then $Fa \longrightarrow\!\triangleright\, a'$. By Lemma 3.6, there exist $\mathtt{C}', \mathtt{F}' \in \hat{\mathbf{C}}$ and $a'' \in \mathbf{C}$ such that $a \xrightarrow{\mathcal{F}(\mathtt{F}')}\!\triangleright\, a''$ and

$$a' = \mathcal{F}(\mathtt{C}')a'' \qquad \mathcal{F}(\mathtt{C}') \in \mathbf{D} \qquad \mathcal{F}(\mathtt{C}'\mathtt{F}') = F\ .$$

Since $a \sim b$, there exists b'' such that $b \xrightarrow{\mathcal{F}(\mathtt{F}')}\!\triangleright\, b''$ and $a'' \sim b''$. Since $\mathcal{F}(\mathtt{C}') \in \mathbf{D}$, we have that $Fb = \mathcal{F}(\mathtt{C}'\mathtt{F}')b \longrightarrow\!\triangleright\, \mathcal{F}(\mathtt{C}')b''$, so, $b \xrightarrow[\mathsf{c}]{F}\!\triangleright\, \mathcal{F}(\mathtt{C}')b''$; also $a' = \mathcal{F}(\mathtt{C}')a'' \sim \mathcal{F}(\mathtt{C}')b''$ since $\sim$ is a congruence.

Case F is not an iso: Then $a \xrightarrow{F}\!\triangleright\, a'$. Since $a \sim b$, there exists b' such that $b \xrightarrow{F}\!\triangleright\, b'$ and $a' \sim b'$. Also $b \xrightarrow[\mathsf{c}]{F}\!\triangleright\, b'$, as desired. ■

I now return to the epi hypothesis discussed earlier and show that if it is satisfied then $\xrightarrow{\cdot}\!\triangleright = \longrightarrow\!\triangleright$. A corollary is that $\xrightarrow{\cdot}\!\triangleright = \xrightarrow[\mathsf{c}]{\cdot}\!\triangleright$ and thus $\sim = \sim_{\mathsf{c}}$. In Chapter 7, I show exact conditions on redexes of a functorial reactive system of graph contexts which hold iff the epi hypothesis is satisfied.

Definition 3.14 (redexes have epi preimages) Let $\mathcal{F} : \hat{\mathbf{C}} \to \mathbf{C}$ be a functorial reactive system. Say that *redexes have epi preimages* iff for all $\mathtt{l} \in \hat{\mathbf{C}}$, if there exists $r \in \mathbf{C}$ such that $(\mathcal{F}(\mathtt{l}), r) \in \mathsf{Reacts}$, then $\mathtt{l}$ is an epi. (Recall that lowercase teletype letters stand for arrows in $\hat{\mathbf{C}}$ with domain ε.) ■

$$\begin{tikzcd} \cdot \arrow[r, "\mathtt{a}"] \arrow[d, "\mathtt{l}"'] & \cdot \arrow[d, "\mathsf{id}"] \\ \cdot \arrow[r, "\mathtt{D}"'] & \cdot \end{tikzcd} \quad 3.15$$

Proposition 3.15 Let $\mathcal{F} : \hat{\mathbf{C}} \longrightarrow \mathbf{C}$ be a functorial reactive system. Suppose that redexes have epi preimages. Then $\overset{\mathsf{id}}{\longrightarrow\!\!\triangleright} = \longrightarrow\!\!\triangleright$.

Proof As discussed earlier, we know that $\overset{\mathsf{id}}{\longrightarrow\!\!\triangleright} \subseteq \longrightarrow\!\!\triangleright$. So we consider the converse. Let $a \longrightarrow\!\!\triangleright a'$. By Proposition 3.2 there exist $\mathtt{a}, \mathtt{l}, \mathtt{D} \in \hat{\mathbf{C}}$ and $r \in \mathbf{C}$ such that $\mathtt{a} = \mathtt{D}\mathtt{l}$, i.e. Figure 3.15 commutes, and

$$a' = \mathcal{F}(\mathtt{D})r \qquad \mathcal{F}(\mathtt{D}) \in \mathbf{D} \qquad (\mathcal{F}(\mathtt{l}), r) \in \mathsf{Reacts}$$
$$\mathcal{F}(\mathtt{a}) = a\ .$$

By the epi hypothesis, $\mathtt{l}$ is an epi. By Proposition 2.11, Figure 3.15 is an IPO. By definition, $a \overset{\mathsf{id}}{\longrightarrow\!\!\triangleright} a'$, as desired. ■

Corollary 3.16 Let $\mathcal{F} : \hat{\mathbf{C}} \longrightarrow \mathbf{C}$ be a functorial reactive system which has all redex-RPOs. Suppose the epi hypothesis of Proposition 3.15 is satisfied, namely, the preimage of every redex is an epi. Then $\overset{\cdot}{\longrightarrow\!\!\triangleright} = \underset{\mathsf{c}}{\overset{\cdot}{\longrightarrow\!\!\triangleright}}$ and thus $\sim = \sim_{\mathsf{c}}$.

$$\begin{tikzcd} \cdot \arrow[r, "\mathtt{Fa}"] \arrow[d, "\mathtt{l}"'] & \cdot \arrow[d, "\mathtt{H}"] \\ \cdot \arrow[r, "\mathtt{D}"'] & \cdot \end{tikzcd} \quad 3.16 \qquad \begin{tikzcd} \cdot \arrow[r, "\mathtt{a}"] \arrow[d, "\mathtt{l}"'] & \cdot \arrow[d, "\mathtt{HF}"] \\ \cdot \arrow[r, "\mathtt{D}"'] & \cdot \end{tikzcd} \quad 3.17$$

Proof The only interesting part is to show that $\underset{\mathsf{c}}{\overset{F}{\longrightarrow\!\!\triangleright}} \subseteq \overset{F}{\longrightarrow\!\!\triangleright}$ for F an iso. Suppose $a \underset{\mathsf{c}}{\overset{F}{\longrightarrow\!\!\triangleright}} a'$. By definition, $Fa \longrightarrow\!\!\triangleright a'$. By Proposition 3.15, $Fa \overset{\mathsf{id}}{\longrightarrow\!\!\triangleright} a'$. By definition, and since $\mathcal{F}$ creates compositions, there exist $\mathtt{a}, \mathtt{l}, \mathtt{F}, \mathtt{D}, \mathtt{H} \in \hat{\mathbf{C}}$ and $r \in \mathbf{C}$ such that Figure 3.16 is an IPO in $\hat{\mathbf{C}}$ and

$$a' = \mathcal{F}(\mathtt{D})r \qquad \mathcal{F}(\mathtt{D}) \in \mathbf{D} \qquad (\mathcal{F}(\mathtt{l}), r) \in \mathsf{Reacts}$$
$$\mathcal{F}(\mathtt{a}) = a \qquad \mathcal{F}(\mathtt{F}) = F \qquad \mathcal{F}(\mathtt{H}) = \mathsf{id}\ .$$

Since $\mathcal{F}$ creates isos and F is an iso, then $\mathtt{F}$ is an iso. Thus Figure 3.17 is also an IPO, whence $a \overset{F}{\longrightarrow\!\!\triangleright} a'$ as desired.

Finally, since $\overset{\cdot}{\longrightarrow\!\!\triangleright} = \underset{\mathsf{c}}{\overset{\cdot}{\longrightarrow\!\!\triangleright}}$ the strong bisimulation relations induced by both are equal, thus $\sim = \sim_{\mathsf{c}}$. ■

3.5 Weak bisimulation

Weak bisimulation [Mil88] is a coarser equivalence than strong bisimulation and is less sensitive to the number of silent steps made by the agents it compares. A single labelled transition by one agent may be matched by a *weak labelled transition* of another, namely a sequence of reactions, followed by a like transition, followed by further reactions.

Definition 3.17 (weak labelled transition ($\overset{\cdot}{\Longrightarrow\!\!\triangleright}$)) The *weak labelled transition relation* is defined as follows:

$$a \overset{F}{\Longrightarrow\!\!\triangleright} a' \quad \text{iff} \quad a \longrightarrow\!\!\triangleright^{*} \overset{F}{\longrightarrow\!\!\triangleright} \longrightarrow\!\!\triangleright^{*} a'\ .$$ ■

The definition of weak bisimulation follows exactly the form set out by Milner:

Definition 3.18 (weak bisimulation over $\overset{\cdot}{\rightarrowtriangle}$; cf. Definition 5 on p. 108 in [Mil88]) Let $\mathcal{S} \subseteq \bigcup_{m \in \mathsf{obj}\,\mathbf{C}} \mathbf{C}(0,m)^2$ be a relation that contains pairs of agents with common codomain. $\mathcal{S}$ is a *weak simulation* over $\overset{\cdot}{\rightarrowtriangle}$ iff it satisfies the following properties for all $(a,b) \in \mathcal{S}$:

1. If $a \rightarrowtriangle a'$, then there exists b' such that $b \rightarrowtriangle^{*} b'$ and $(a',b') \in \mathcal{S}$.
2. If $a \xrightarrow{F} a'$ where F is not an iso, then there exists b' such that $b \xRightarrow{F} b'$ and $(a',b') \in \mathcal{S}$.

$\mathcal{S}$ is a *weak bisimulation* iff $\mathcal{S}$ and $\mathcal{S}^{-1}$ are weak simulations. Let $\approx$ be the *largest weak bisimulation* over $\overset{\cdot}{\rightarrowtriangle}$. ■

In the second clause of the definition of weak simulation, the label F is required to be not an iso. Without this requirement, the definition would force $a \xrightarrow{F}$ to be matched by $b \xRightarrow{F}$, for F an iso. By the definition of IPOs, iso-labelled transitions are essentially like id labelled transitions. As argued in the previous section $\xrightarrow{\mathsf{id}} \subseteq \rightarrowtriangle$, and moreover the converse holds in certain cases. So allowing F to be an iso in clause 2 would override clause 1. But clause 1 embodies a basic principle, namely that a silent transitions is matched by *zero or more* silent transitions and not by *one or more.*

The congruence property of weak bisimulation is more limited than that of strong bisimulation: $\approx$ is a congruence with respect to arrows only in $\mathbf{D}$, a subcategory of $\mathbf{C}$. Recall that $\mathbf{D}$ consists of the “reactive contexts”, i.e. the contexts that allow reaction under them: $a \rightarrowtriangle a'$ implies $Da \rightarrowtriangle Da'$ for $D \in \mathbf{D}$. (See Definition 2.) This limitation is not surprising. In CCS, for example, weak bisimulation is not a congruence with respect to summation contexts, which are not reactive, i.e. we do not have that $a \rightarrowtriangle a'$ implies $a + b \rightarrowtriangle a' + b$. (I am using a, b for agents *and not for names* in order to maintain consistency with the notation of the rest of this dissertation.) Use of the hypothesis $C \in \mathbf{D}$ occurs twice in the proof: see (3.1) and (3.2) below.

In CCS, weak bisimulation *is* a congruence with respect to some non-reactive contexts, namely the prefixing contexts $x.-$ and $\bar{x}.-$. We would require richer structure than is contained in Chapter 5 in order to have a category of CCS contexts, namely the nesting of graphs to represent prefixing, some added data (not yet well understood) to represent summation, and the inclusion of free names to represent the naming structure of CCS. If we could construct such a category then it is likely that proving explicitly that weak bisimulation is preserved by prefixing would be easy since the only initial labelled transition of a prefixed agent is based on the prefix itself. Nonetheless, it is worth considering whether we could get a better general congruence result for weak bisimulation by dividing the set of reactive contexts in two, with one set containing prefixing-like contexts and another containing sum-like contexts. I am not sure how this would work, but something similar is done in work on rule formats for structural operational semantics by Bloom in [Blo93] who distinguishes "tame" from "wild" contexts.

Theorem 3.19 (congruence for $\approx$ w.r.t. $\mathbf{D}$) Let $\mathcal{F} : \hat{\mathbf{C}} \to \mathbf{C}$ be a functorial reactive system which has all redex-RPOs. Then $\approx$ is a congruence with respect to all contexts in $\mathbf{D}$, i.e. $a \approx b$ implies $Ca \approx Cb$ for all $C \in \mathbf{D}$ of the required domain.

Proof By symmetry, it is sufficient to show that the following relation is a weak simulation:

$$\mathcal{S} \mathrel{\hat{=}} \{(Ca, Cb) \ / \ a \approx b \text{ and } C \in \mathbf{D}\} .$$

The proof falls into two cases corresponding to those of Definition 3.18. Suppose that $a \approx b$ and $C \in \mathbf{D}$, and thus $(Ca, Cb) \in \mathcal{S}$,

Case $Ca \longrightarrow\!\!\triangleright a'$: By Lemma 3.6, there exist $a'' \in \mathbf{C}$ and $\mathtt{C}', \mathtt{F}' \in \hat{\mathbf{C}}$ such that $a \xrightarrow{\mathcal{F}(\mathtt{F}')}\!\!\triangleright a''$ and

$$a' = \mathcal{F}(\mathtt{C}')a'' \qquad \mathcal{F}(\mathtt{C}') \in \mathbf{D} \qquad \mathcal{F}(\mathtt{C}'\mathtt{F}') = C .$$

Moreover, if $\mathtt{F}'$ is an iso then it is equal to id. We distinguish between two cases.

Case $\mathtt{F}'$ is an iso: By definition $a \longrightarrow\!\!\triangleright a''$. Since $a \approx b$, there exists b'' such that $b \longrightarrow\!\!\triangleright^* b''$ and $a'' \approx b''$. Since $\mathcal{F}(\mathtt{C}') \in \mathbf{D}$, we have that $Cb = \mathcal{F}(\mathtt{C}')b \longrightarrow\!\!\triangleright^* \mathcal{F}(\mathtt{C}')b''$.

Case $\mathtt{F}'$ is not an iso: Since $\mathcal{F}$ creates isos, $\mathcal{F}(\mathtt{F}')$ is not an iso. Since $a \approx b$, there exist b_0, b_1, b'' such that $b \longrightarrow\!\!\triangleright^* b_0 \xrightarrow{\mathcal{F}(\mathtt{F}')}\!\!\triangleright b_1 \longrightarrow\!\!\triangleright^* b''$ with $a'' \approx b''$. By hypothesis, $C \in \mathbf{D}$, therefore:

$$Cb \longrightarrow\!\!\triangleright^* Cb_0 . \tag{3.1}$$

Since $\mathcal{F}(\mathtt{C}') \in \mathbf{D}$ we have that $Cb_0 = \mathcal{F}(\mathtt{C}')\mathcal{F}(\mathtt{F}')b_0 \longrightarrow\!\!\triangleright \mathcal{F}(\mathtt{C}')b_1 \longrightarrow\!\!\triangleright^* \mathcal{F}(\mathtt{C}')b''$.

In both cases, $a'' \approx b''$, so $(\mathcal{F}(\mathtt{C}')a'', \mathcal{F}(\mathtt{C}')b'') \in \mathcal{S}$ as desired.

Case $Ca \xrightarrow{F}\!\!\triangleright a'$ for F not an iso: By Lemma 3.5, there exist $a'' \in \mathbf{C}$ and an IPO square shown in Figure 3.18 such that $a \xrightarrow{\mathcal{F}(\mathtt{F}')}\!\!\triangleright a''$ and

$$\begin{tikzcd} \cdot \arrow[r, "\mathtt{C}"] \arrow[d, "\mathtt{F}'"'] & \cdot \arrow[d, "\mathtt{F}"] \\ \cdot \arrow[r, "\mathtt{C}'"'] & \cdot \end{tikzcd}$$

3.18

$$a' = \mathcal{F}(\mathtt{C}')a'' \qquad \mathcal{F}(\mathtt{C}') \in \mathbf{D} \qquad \mathcal{F}(\mathtt{C}) = C \qquad \mathcal{F}(\mathtt{F}) = F .$$

Since F is not an iso, $\mathtt{F}$ is not an iso; Proposition 2.10 implies that $\mathtt{F}'$ is also not an iso; since $\mathcal{F}$ creates isos, $\mathcal{F}(\mathtt{F}')$ is not an iso, thus $a \approx b$ implies that there exist b_0, b_1, b'' such that $b \longrightarrow\!\!\triangleright^* b_0 \xrightarrow{\mathcal{F}(\mathtt{F}')}\!\!\triangleright b_1 \longrightarrow\!\!\triangleright^* b''$ with $a'' \approx b''$. By hypothesis, $C \in \mathbf{D}$, therefore:

$$Cb \longrightarrow\!\!\triangleright^* Cb_0 . \tag{3.2}$$

Since Figure 3.18 is an IPO and $\mathcal{F}(\mathtt{C}') \in \mathbf{D}$, Lemma 3.7 implies that $Cb_0 \xrightarrow{F}\!\!\triangleright \mathcal{F}(\mathtt{C}')b_1$. Since $\mathcal{F}(\mathtt{C}') \in \mathbf{D}$, $\mathcal{F}(\mathtt{C}')b_1 \longrightarrow\!\!\triangleright^* \mathcal{F}(\mathtt{C}')b''$. Moreover, $(\mathcal{F}(\mathtt{C}')a'', \mathcal{F}(\mathtt{C}')b'') \in \mathcal{S}$, as desired.

■

How can we get a congruence with respect to $\mathbf{C}$, not just $\mathbf{D}$? One possibility is to replace the $\overset{F}{\Longrightarrow}$ of clause 2 in Definition 3.18 with $\xrightarrow{F}\!\!\triangleright\longrightarrow\!\!\triangleright^{*}$. The largest symmetric relation satisfying this new definition is a congruence with respect to all of $\mathbf{C}$. A second possibility is to make this change for the *first step only* of the weak bisimulation relation and then revert to the normal definition in Definition 3.18:

Definition 3.20 (greedy weak bisimulation ($\approx_{\mathsf{gr}}$)) Let $\approx_{\mathsf{gr}} \subseteq \bigcup_{m\in \mathsf{obj}\,\mathbf{C}} \mathbf{C}(0,m)^2$ be the largest symmetric relation that contains pairs of agents with common codomain and which satisfies the following properties:

1. If $a \longrightarrow\!\!\triangleright\, a'$, then there exists b' such that $b \longrightarrow\!\!\triangleright^{*} b'$ and $a' \approx b'$.
2. If $a \xrightarrow{F}\!\!\triangleright\, a'$ where F is not an iso, then there exists b' such that $b \xrightarrow{F}\!\!\triangleright\longrightarrow\!\!\triangleright^{*} b'$ and $a' \approx b'$.

We call $\approx_{\mathsf{gr}}$ *greedy weak bisimulation.* ■

Notice that this definition requires that $a' \approx b'$, not the stronger condition that $a' \approx_{\mathsf{gr}} b'$. The relation $\approx_{\mathsf{gr}}$ is also a congruence with respect to $\mathbf{C}$. The proof is almost identical to that of Theorem 3.19.

The idea of having a special requirement for the first step of a weak bisimulation followed by the use of standard weak bisimulation to compare the continuations is well-established. See, for example, the definition of *observational congruence* (Definition 2 on p. 153 in [Mil88]), also known as *rooted weak bisimulation* in the literature. Experience with CCS suggests that a congruence in ensured by changing clause 1 (to require $b \longrightarrow\!\!\triangleright\longrightarrow\!\!\triangleright^{*} b'$) and not clause 2 in Definition 3.18 — exactly the opposite of what is done in Definition 3.20. This anomaly requires further research, both to find categorical theory to model different kinds of non-reactive contexts and to show that RPOs exist for categories of graph-like contexts that *contain summation.*

3.6 Traces preorder

This section addresses the *traces preorder*, a simple preorder that compares agents based on their finite traces. A *trace* (see p. 41 in [Hoa85]) is a sequence of labelled transitions.

The traces preorder is insensitive to non-determinism and deadlock so is of limited use. Nonetheless, traces are good for specifying security properties since it is easy to formulate that an agent does not possess certain "bad" traces (see, for example, [Pau98, SV00]).

The main motivation for this section is to provide a warmup for the next one, which looks at the *failures preorder*. As a result, the traces considered here are all strong (not interspersed with reaction steps) since the way to handle the weak case is subsumed by the results presented in the next section.

$$\begin{tikzcd}
C_0a_0 \arrow[dddddd, dash, dashed, "\gtrsim_{\mathsf{tr}}"] \arrow[rr, "F_1"] & & \cdots \arrow[r, "F_n"] & C_na_n \\
& & \Downarrow\textbf{(i)} & \\
& a_0 \arrow[r, "\mathcal{F}(\mathtt{F}'_1)"] \arrow[dd, dash, "\gtrsim_{\mathsf{tr}}"] & \cdots \arrow[r, "\mathcal{F}(\mathtt{F}'_n)"] & a_n \\
& & \Downarrow\textbf{(ii)} & \\
& b_0 \arrow[r, "\mathcal{F}(\mathtt{F}'_1)"] & \cdots \arrow[r, "\mathcal{F}(\mathtt{F}'_n)"] & b_n \\
& & \Downarrow\textbf{(iii)} & \\
C_0b_0 \arrow[rr, "F_1"] & & \cdots \arrow[r, "F_n"] & C_nb_n
\end{tikzcd}$$

Figure 3.19: Schema of the congruence proof for $\gtrsim_{\mathsf{tr}}$

Definition 3.21 (traces preorder ($\gtrsim_{\mathsf{tr}}$); cf. p. 45 in [Ros98]) A pair of agents b and a with common codomain are related by the *traces preorder*, written $a \gtrsim_{\mathsf{tr}} b$, iff all the traces of a are traces of b: for every trace $\langle F_1, \ldots, F_n \rangle$,

$$a \xrightarrow{F_1} \cdots \xrightarrow{F_n} \quad \text{implies} \quad b \xrightarrow{F_1} \cdots \xrightarrow{F_n} . \qquad ■$$

(In this section and the next, n, m are natural numbers and not objects of a category.)

The proof of congruence is more complicated that that of strong bisimulation (Theorem 3.9) because we need to consider traces, not just individual labelled transitions. The heart of the argument is an inductive construction of a trace of a from a trace of Ca. Each inductive step cuts off a portable IPO square (see Lemma 3.5) which is subsequently pasted back on (Lemma 3.7) when constructing a trace of Cb from a trace of b.

Theorem 3.22 (congruence for $\gtrsim_{\mathsf{tr}}$) Let $\mathcal{F} : \hat{\mathbf{C}} \to \mathbf{C}$ be a functorial reactive system which has all redex-RPOs. Then $\gtrsim_{\mathsf{tr}}$ is a congruence, i.e. $a \gtrsim_{\mathsf{tr}} b$ implies $Ca \gtrsim_{\mathsf{tr}} Cb$ for all $C \in \mathbf{C}$ of the required domain.

Proof Suppose that $a \gtrsim_{\mathsf{tr}} b$. Let C be any context of appropriate domain. We wish to prove that $Ca \gtrsim_{\mathsf{tr}} Cb$. The proof is divided into three parts, which are shown schematically in Figure 3.19.

$$\begin{tikzcd} \cdot \arrow[r, "\mathtt{C}_i"] \arrow[d, "\mathtt{F}'_i"'] & \cdot \arrow[d, "\mathtt{F}_i"] \\ \cdot \arrow[r, "\mathtt{C}'_i"'] & \cdot \end{tikzcd}$$
3.20

(i): Let $\bar{a}_0 \mathrel{\hat{=}} Ca$ and consider any trace $\bar{a}_0 \xrightarrow{F_1} \cdots \xrightarrow{F_n} \bar{a}_n$, where $n \geq 0$. We construct $(a_0, C_0) \dots (a_n, C_n)$ and the square shown in Figure 3.20 for $1 < i \leq n$ such that the following conditions hold for $0 \leq i \leq n$:

$$\begin{array}{rll} \bar{a}_i = C_i a_i & & \text{TR-A} \\ a_{i-1} \xrightarrow{\mathcal{F}(\mathtt{F}'_i)} a_i & \text{for } i \neq 0 & \text{TR-LAB} \\ C_i \in \mathbf{D} & \text{for } i \neq 0 & \text{TR-D} \\ \mathcal{F}(\mathtt{C}_i) = C_{i-1} & \text{for } i \neq 0 & \text{TR-C} \\ \mathcal{F}(\mathtt{C}'_i) = C_i & \text{for } i \neq 0 & \text{TR-CPRIME} \\ \mathcal{F}(\mathtt{F}_i) = F_i & \text{for } i \neq 0 & \text{TR-F} \\ \text{Figure 3.20 is an IPO} & \text{for } i \neq 0\,. & \text{TR-IPO} \end{array}$$

base: Let $a_0 \mathrel{\hat{=}} a$ and $C_0 \mathrel{\hat{=}} C$. Then TR-A holds and the other conditions are vacuous.

$$\begin{tikzcd} \cdot \arrow[r, "\mathtt{C}_{i+1}"] \arrow[d, "\mathtt{F}'_{i+1}"'] & \cdot \arrow[d, "\mathtt{F}_{i+1}"] \\ \cdot \arrow[r, "\mathtt{C}'_{i+1}"'] & \cdot \end{tikzcd}$$
3.21

step: We construct a_{i+1}, C_{i+1} and the square in Figure 3.21 from a_i, C_i as follows, assuming $0 \leq i < n$. By the inductive hypothesis TR-A holds for i, thus $\bar{a}_i = C_i a_i$. Since $\bar{a}_i \xrightarrow{F_i} \bar{a}_{i+1}$, Lemma 3.5 implies that there exist $a_{i+1} \in \mathbf{C}$ and an IPO square shown in Figure 3.21 such that $a_i \xrightarrow{\mathcal{F}(\mathtt{F}'_{i+1})} a_{i+1}$ and

$$\bar{a}_{i+1} = C_{i+1} a_{i+1} \qquad C_{i+1} \in \mathbf{D} \qquad \mathcal{F}(\mathtt{C}_{i+1}) = C_i \qquad \mathcal{F}(\mathtt{F}_{i+1}) = F_{i+1}\,.$$

where we let $C_{i+1} \mathrel{\hat{=}} \mathcal{F}(\mathtt{C}'_{i+1})$. Then all of the inductive properties are satisfied for $i+1$.

Thus by TR-LAB $a = a_0 \xrightarrow{\mathcal{F}(\mathtt{F}'_1)} \cdots \xrightarrow{\mathcal{F}(\mathtt{F}'_n)} a_n$.

(ii): Since $a \gtrsim_{\mathrm{tr}} b$ there exist $b_0 \dots b_n$ such that $b = b_0 \xrightarrow{\mathcal{F}(\mathtt{F}'_1)} \cdots \xrightarrow{\mathcal{F}(\mathtt{F}'_n)} b_n$.

(iii): We now claim that $C_i b_i \xrightarrow{F_{i+1}} C_{i+1} b_{i+1}$ for $0 \leq i < n$.

Since $b_i \xrightarrow{\mathcal{F}(\mathtt{F}'_{i+1})} b_{i+1}$ and Figure 3.21 is an IPO (by TR-IPO), with $\mathcal{F}(\mathtt{C}'_{i+1}) =^{\text{TR-CPRIME}} C_{i+1} \in \mathbf{D}$ by TR-D, then Lemma 3.7 implies

$$C_i b_i =^{\text{TR-C}} \mathcal{F}(\mathtt{C}_{i+1}) b_i \xrightarrow{\mathcal{F}(\mathtt{F}_{i+1})} \mathcal{F}(\mathtt{C}'_{i+1}) b_{i+1} =^{\text{TR-CPRIME}} C_{i+1} b_{i+1}\,,$$

and thus by TR-F, $C_i b_i \xrightarrow{F_{i+1}} C_{i+1} b_{i+1}$. So

$$Cb = C_0 b_0 \xrightarrow{F_1} \cdots \xrightarrow{F_n}$$

as desired. ■

3.7 Failures preorder

This section looks at the failures preorder, which is a fundamental part of the failures and divergences model of CSP [Hoa85]. I do not consider divergences here, so the definition I use only employs failures. The failures preorder is sensitive to non-determinism and deadlock (see Section 3.3 in [Ros98]). The failures of an agent provide a domain-theoretic interpretation, assigning a meaning to each agent independently of the others (unlike for bisimulation). This makes failures properties well-suited to model checking [Ros94, Low96].

In order to define a failure of an agent, I first extend the notion of a weak labelled transition to allow for sequences of labels (not just single labels):

Definition 3.23 (weak labelled transition extended; cf. Definition 3.17)

$$\begin{array}{rcl} a \xRightarrow{\langle\,\rangle} a' & \text{iff} & a \longrightarrow^{*} a' \\ a \xRightarrow{\langle F\rangle\,\hat{}\,t} a' & \text{iff} & a \xRightarrow{F}\xRightarrow{t} a' \ , \end{array}$$

where t is a sequence of arrows of appropriate domain and $\hat{}$ is the concatenation operator. ■

A failure of a consists of a sequence of weak labelled transitions t and a set of labels X such that a evolves by $\xRightarrow{t}$ to a *stable state* (one for which no reactions are possible) which *refuses* X, i.e. cannot engage in a transition labelled by any of the arrows in X. To prevent reactions from masquerading as labelled transitions, every arrow in t and X is not an iso (cf. the discussion immediately following Definition 3.18).

Definition 3.24 (failure; cf. p. 171 in [Ros98]) A *failure* of a is a pair (t, X) where t is a finite sequence of arrows (each not an iso) and X is a set of arrows (each not an iso) for which there exists a' such that the following conditions hold:

$$\begin{array}{ll} a \xRightarrow{t} a' & a \text{ has a weak trace } t; \\ a' \not\longrightarrow & a' \text{ is stable;} \\ \forall F \in X.\ a' \overset{F}{\not\longrightarrow} & a' \text{ refuses } X. \end{array}$$ ■

Definition 3.25 (failures preorder ($\gtrsim_{\mathsf{f}}$); cf. p. 193 in [Ros98]) A pair of agents b and a with common codomain are related by the *failures preorder*, written $a \gtrsim_{\mathsf{f}} b$, iff all the failures of a are possessed by b. ■

The relation $\gtrsim_{\mathsf{f}}$ is only a congruence with respect $\mathbf{D}$, the subcategory of $\mathbf{C}$ consisting of reactive contexts (cf. Theorem 3.19). The only use of the hypothesis $C \in \mathbf{D}$ occurs in the base case of the induction.

The proof is similar to that of the traces preorder however there are two aspects that require care: the cutting and pasting of portable IPO squares for weak labelled transitions and the propagation of refusal sets.

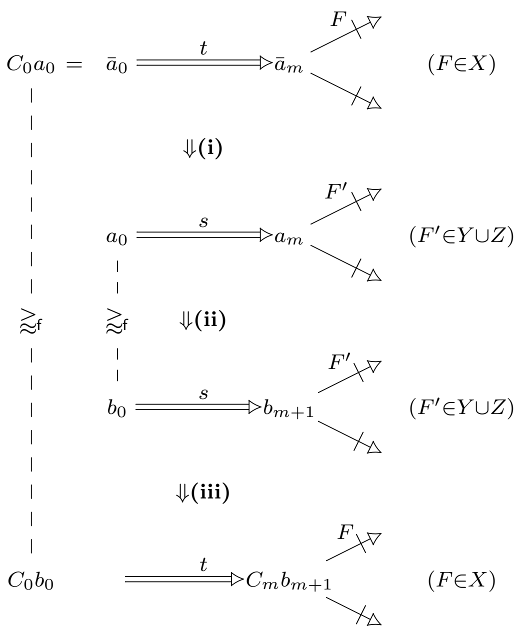

$C_0a_0 = \bar{a}_0 \overset{t}{\Longrightarrow} \bar{a}_m$
F
$(F \in X)$
$\Downarrow$(i)
$a_0 \overset{s}{\Longrightarrow} a_m$
F'
$(F' \in Y \cup Z)$
$\gtrapprox_{\mathsf{f}}$
$\gtrapprox_{\mathsf{f}}$
$\Downarrow$(ii)
$b_0 \overset{s}{\Longrightarrow} b_{m+1}$
F'
$(F' \in Y \cup Z)$
$\Downarrow$(iii)
$C_0b_0 \overset{t}{\Longrightarrow} C_m b_{m+1}$
F
$(F \in X)$

Figure 3.22: Schema of the congruence proof for $\gtrapprox_{\mathsf{f}}$

Theorem 3.26 (congruence for $\gtrapprox_{\mathsf{f}}$ w.r.t. D) Let $\mathcal{F} : \hat{\mathbf{C}} \to \mathbf{C}$ be a functorial reactive system which has all redex-RPOs. Then $\gtrapprox_{\mathsf{f}}$ is a congruence with respect to all contexts in $\mathbf{D}$, i.e. $a \gtrapprox_{\mathsf{f}} b$ implies $Ca \gtrapprox_{\mathsf{f}} Cb$ for all $C \in \mathbf{D}$ of the required domain.

Proof Suppose that $a \gtrapprox_{\mathsf{f}} b$. Let $C \in \mathbf{D}$ be any (reactive) context of appropriate domain. We wish to prove that $Ca \gtrapprox_{\mathsf{f}} Cb$. The proof is divided into three parts, which are shown schematically in Figure 3.22.

(i): Let $\bar{a}_0 \mathrel{\hat{=}} Ca$ and consider any failure (t, X) of $\bar{a}_0$, where $t = \langle F_1, \dots, F_n \rangle$, for $n \geq 0$. By Definition 3.24 and Definition 3.23, there exist natural numbers $0 \leq m$ and $0 < p_1 < \cdots < p_n \leq m$ and a sequence $\bar{a}_0, \dots, \bar{a}_m$ such that for $0 < i \leq m$:

$$\begin{aligned} &\bar{a}_{i-1} \longrightarrow \bar{a}_i && \text{if } i \notin \{p_1, \dots, p_n\}; \\ &\bar{a}_{i-1} \xrightarrow{F_j} \bar{a}_i && \text{if } i = p_j \text{ for some } j, \text{ where } 1 \leq j \leq n. \end{aligned}$$

Moreover, $\bar{a}_m$ is stable and refuses X.

We construct inductively (a_i, C_i) for $0 \leq i \leq m$, and $(\mathtt{F}'_i, \mathtt{C}'_i, \mathtt{C}_i)$ for $1 \leq i \leq m$, and $\mathtt{F}_j$, for $1 \leq j \leq n$ such that the following conditions

$$\begin{tikzcd} \cdot \arrow[r, "\mathtt{C}_i"] \arrow[d, "\mathtt{F}'_i"'] & \cdot \arrow[d, "\mathtt{F}_j"] \\ \cdot \arrow[r, "\mathtt{C}'_i"'] & \cdot \end{tikzcd} \quad 3.23$$

hold for $0 \le i \le m$:

$\bar{a}_i = C_i a_i$		FAIL-A
$C_i \in \mathbf{D}$		FAIL-D
$a_{i-1} \xrightarrow{\mathcal{F}(\mathtt{F}'_i)} a_i$	for $i \neq 0$ and $\mathtt{F}'_i \neq \mathsf{id}$	FAIL-LAB
$\mathtt{F}'_i$ is not an iso	for $i \neq 0$ and $\mathtt{F}'_i \neq \mathsf{id}$	FAIL-NON-ISO
$a_{i-1} \longrightarrow a_i$	for $i \neq 0$ and $\mathtt{F}'_i = \mathsf{id}$	FAIL-REACT
$\mathcal{F}(\mathtt{C}_i) = C_{i-1}$	for $i \neq 0$	FAIL-C
$\mathcal{F}(\mathtt{C}'_i) = C_i$	for $i \neq 0$	FAIL-CPRIME
$\mathcal{F}(\mathtt{F}_j) = F_j$	for $i = p_j$, where $1 \le j \le n$	FAIL-F
Figure 3.23 is an IPO	for $i = p_j$, where $1 \le j \le n$	FAIL-IPO
$\mathtt{C}_i = \mathtt{C}'_i \mathtt{F}'_i$	for $i \neq 0$ and $i \notin \{p_1, \dots, p_n\}$	FAIL-COMMUTE

base: Let $a_0 \mathrel{\hat{=}} a$ and $C_0 \mathrel{\hat{=}} C$. Then FAIL-A and FAIL-D hold and the other conditions are vacuously true.

step: Given a_i, C_i for $0 \le i < m$, we construct all the required data for $i+1$.

case $i+1 = p_j$ for some j, where $1 \le j \le n$: In this case, FAIL-COMMUTE for $i+1$ is vacuously true. By the inductive hypothesis, FAIL-A holds for i, thus $\bar{a}_i = C_i a_i$. Since $\bar{a}_i \xrightarrow{F_j} \bar{a}_{i+1}$, Lemma 3.5 implies that there exist $a_{i+1} \in \mathbf{C}$ and an IPO square shown in Figure 3.24 such that $a_i \xrightarrow{\mathcal{F}(\mathtt{F}'_{i+1})} a_{i+1}$ and

$$\begin{tikzcd}
\cdot \arrow[r, "\mathtt{C}_{i+1}"] \arrow[d, "\mathtt{F}'_{i+1}"'] & \cdot \arrow[d, "\mathtt{F}_j"] \\
\cdot \arrow[r, "\mathtt{C}'_{i+1}"'] & \cdot
\end{tikzcd}$$

3.24

$$\bar{a}_{i+1} = C_{i+1} a_{i+1} \qquad C_{i+1} \in \mathbf{D} \qquad \mathcal{F}(\mathtt{C}_{i+1}) = C_i \qquad \mathcal{F}(\mathtt{F}_j) = F_j \ .$$

where we let $C_{i+1} \mathrel{\hat{=}} \mathcal{F}(\mathtt{C}'_{i+1})$. Since F_j is not an iso, $\mathtt{F}_j$ is not an iso; Proposition 2.10 implies that $\mathtt{F}'_{i+1}$ is also not an iso. Then all of the inductive properties are satisfied for $i+1$.

case $i+1 \notin \{p_1, \dots, p_n\}$: In this case, FAIL-IPO and FAIL-F for $i+1$ are vacuously true. By the inductive hypothesis, FAIL-A holds for i, thus $\bar{a}_i = C_i a_i$. Since $\bar{a}_i \longrightarrow \bar{a}_{i+1}$, Lemma 3.6 implies that there exist $a_{i+1} \in \mathbf{C}$ and $\mathtt{C}_{i+1}, \mathtt{C}'_{i+1}, \mathtt{F}'_{i+1}$ such that

$$a_i \xrightarrow{\mathcal{F}(\mathtt{F}'_{i+1})} a_{i+1} \qquad \mathtt{C}'_{i+1}\mathtt{F}'_{i+1} = \mathtt{C}_{i+1}$$
$$\bar{a}_{i+1} = C_{i+1} a_{i+1} \qquad C_{i+1} \in \mathbf{D} \qquad \mathcal{F}(\mathtt{C}_{i+1}) = C_i \ ,$$

where we let $C_{i+1} \mathrel{\hat{=}} \mathcal{F}(\mathtt{C}'_{i+1})$; furthermore, $\mathtt{F}'_{i+1} = \mathsf{id}$ if $\mathtt{F}'_{i+1}$ is an iso. All the inductive properties are satisfied for $i+1$.

If we let s be the sequence $\langle \mathcal{F}(\mathtt{F}'_1), \dots, \mathcal{F}(\mathtt{F}'_m) \rangle$ with all the id arrows removed, then FAIL-REACT, FAIL-LAB, and FAIL-NON-ISO imply that $a = a_0 \overset{s}{\Longrightarrow} a_m$ and that each label in s is not an iso.

Claim: a_m is stable. Suppose for contradiction $a_m \longrightarrow\!\!\!\triangleright$. Then $\bar{a}_m = C_m a_m \longrightarrow\!\!\!\triangleright$ because by FAIL-D, $C_m \in \mathbf{D}$. But, by hypothesis, $\bar{a}_m$ is stable, a contradiction.

Now let

$$Y \mathrel{\hat{=}} \left\{ \mathcal{F}(\mathtt{F}') \;\middle/\; \begin{array}{l} \mathtt{F}' \text{ is not an iso} \\ \& \left(\exists \mathtt{C}, \mathtt{D}, \mathtt{F}. \begin{array}{l} \mathcal{F}(\mathtt{C}) = C_m \;\&\; \mathcal{F}(\mathtt{D}) \in \mathbf{D} \;\&\; \mathcal{F}(\mathtt{F}) \in X \\ \& \text{ Figure 3.25 is an IPO} \end{array} \right) \end{array} \right\}$$

$$Z \mathrel{\hat{=}} \left\{ \mathcal{F}(\mathtt{F}') \;\middle/\; \mathtt{F}' \text{ is not an iso } \& \left(\exists \mathtt{D}.\, \mathcal{F}(\mathtt{D}) \in \mathbf{D} \;\&\; C_m = \mathcal{F}(\mathtt{D}\mathtt{F}')\right) \right\}$$

$$\begin{tikzcd} \cdot \arrow[r, "\mathtt{C}"] \arrow[d, "\mathtt{F}'"'] & \cdot \arrow[d, "\mathtt{F}"] \\ \cdot \arrow[r, "\mathtt{D}"'] & \cdot \end{tikzcd} \quad 3.25$$

Claim: a_m refuses $Y \cup Z$. Suppose $\mathtt{F}'$ is not an iso. Since $\mathcal{F}$ creates isos, $\mathcal{F}(\mathtt{F}')$ is not an iso. Consider $\mathcal{F}(\mathtt{F}') \in Y$, as witnessed by $\mathcal{F}(\mathtt{C}) = C_m$, $\mathcal{F}(\mathtt{D}) \in \mathbf{D}$, and $\mathcal{F}(\mathtt{F}) \in X$ such that Figure 3.25 is an IPO. Suppose for contradiction that $a_m \xrightarrow{\mathcal{F}(\mathtt{F}')}\!\!\!\triangleright$. By Lemma 3.7, $\bar{a}_m = C_m a_m \xrightarrow{\mathcal{F}(\mathtt{F})}\!\!\!\triangleright$, a contradiction since $\bar{a}_m$ refuses $X \ni \mathcal{F}(\mathtt{F})$.

Consider $\mathcal{F}(\mathtt{F}') \in Z$, as witnessed by $\mathcal{F}(\mathtt{D}) \in \mathbf{D}$ such that $C_m = \mathcal{F}(\mathtt{D}\mathtt{F}')$. Suppose for contradiction that $a_m \xrightarrow{\mathcal{F}(\mathtt{F}')}\!\!\!\triangleright$. Then $\bar{a}_m = C_m a_m = \mathcal{F}(\mathtt{D})\mathcal{F}(\mathtt{F}')a_m \longrightarrow\!\!\!\triangleright$ since $\mathcal{F}(\mathtt{D}) \in \mathbf{D}$. This contradicts the assumption that $\bar{a}_m$ is stable.

Thus a has failure $(s, Y \cup Z)$.

(ii): Since $a \gtrsim_{\mathrm{f}} b$, it follows that b has failure $(s, Y \cup Z)$. By FAIL-LAB, FAIL-NON-ISO, and FAIL-REACT, there exist $b = b_0, \ldots, b_{m+1}$ such that:

$$\begin{array}{ll} b_{i-1} \xRightarrow{\mathcal{F}(\mathtt{F}'_i)}\!\!\!\triangleright\, b_i & \text{if } \mathtt{F}'_i \neq \mathsf{id} \text{ and } 0 < i \leq m; \\ b_{i-1} \longrightarrow\!\!\!\triangleright^{*} b_i & \text{if } \mathtt{F}'_i = \mathsf{id} \text{ and } 0 < i \leq m; \\ b_m \longrightarrow\!\!\!\triangleright^{*} b_{m+1}. & \end{array}$$

Furthermore b_{m+1} is stable and refuses $Y \cup Z$.

(iii): In the final part, we argue that $Cb = C_0 b_0$ has failure (t, X). First we claim that for $0 < i \leq m$

$$\begin{array}{rl} C_{i-1}b_{i-1} \xRightarrow{F_j}\!\!\!\triangleright\, C_i b_i & \text{if } i = p_j \text{ for some } j, \text{ where } 1 \leq j \leq n. \\ C_{i-1}b_{i-1} \longrightarrow\!\!\!\triangleright^{*} C_i b_i & \text{if } i \notin \{p_1, \ldots, p_n\}; \end{array}$$

We consider two cases:

$$\begin{tikzcd} \cdot \arrow[r, "\mathtt{C}_i"] \arrow[d, "\mathtt{F}'_i"'] & \cdot \arrow[d, "\mathtt{F}_j"] \\ \cdot \arrow[r, "\mathtt{C}'_i"'] & \cdot \end{tikzcd} \quad 3.26$$

case $i = p_j$ **for some** j**, where** $1 \leq j \leq n$**:** By FAIL-IPO, Figure 3.26 is an IPO. By FAIL-F, $\mathtt{F}_j$ is not an iso, so Proposition 2.10 implies that $\mathtt{F}'_i$ is also not an iso. By construction, $b_{i-1} \xRightarrow{\mathcal{F}(\mathtt{F}'_i)}\!\!\!\triangleright\, b_i$, so there exist b', b'' such that:

$$b_{i-1} \longrightarrow\!\!\!\triangleright^{*} b' \xrightarrow{\mathcal{F}(\mathtt{F}'_i)}\!\!\!\triangleright\, b'' \longrightarrow\!\!\!\triangleright^{*} b_i \; .$$

By FAIL-D, $C_{i-1}b_{i-1} \longrightarrow\!\!\!\triangleright^* C_{i-1}b'$. Since Figure 3.26 is an IPO, Lemma 3.7 and FAIL-F imply that $C_{i-1}b' =^{\text{FAIL-C}} \mathcal{F}(\mathtt{C}_i)b' \xrightarrow{F_j} \mathcal{F}(\mathtt{C}'_i)b'' =^{\text{FAIL-CPRIME}} C_i b''$. By FAIL-D, $C_i b'' \longrightarrow\!\!\!\triangleright^* C_i b_i$. Thus $C_{i-1}b_{i-1} \xRightarrow{F_j} C_i b_i$, as desired.

case $i \notin \{p_1, \ldots, p_m\}$**:** This case falls into two subcases:

case $\mathtt{F}'_i \neq \mathsf{id}$**:** By construction $b_{i-1} \xRightarrow{\mathcal{F}(\mathtt{F}'_i)} b_i$, so there exist b', b'' such that:

$$b_{i-1} \longrightarrow\!\!\!\triangleright^* b' \xrightarrow{\mathcal{F}(\mathtt{F}'_i)} b'' \longrightarrow\!\!\!\triangleright^* b_i \ .$$

By FAIL-D, $C_{i-1}b_{i-1} \longrightarrow\!\!\!\triangleright^* C_{i-1}b'$. By definition, $\mathcal{F}(\mathtt{F}'_i)b' \longrightarrow\!\!\!\triangleright b''$. Hence by FAIL-D, $C_{i-1}b' =^{\text{FAIL-C}} \mathcal{F}(\mathtt{C}_i)b' =^{\text{FAIL-COMMUTE}} \mathcal{F}(\mathtt{C}'_i)\mathcal{F}(\mathtt{F}'_i)b' =^{\text{FAIL-CPRIME}} C_i\mathcal{F}(\mathtt{F}'_i)b' \longrightarrow\!\!\!\triangleright C_i b'' \longrightarrow\!\!\!\triangleright^* C_i b_i$. Thus, $C_{i-1}b_{i-1} \longrightarrow\!\!\!\triangleright^* C_i b_i$, as desired.

case $\mathtt{F}'_i = \mathsf{id}$**:** By construction $b_{i-1} \longrightarrow\!\!\!\triangleright^* b_i$; by FAIL-D, $C_i \in \mathbf{D}$, so $C_{i-1}b_{i-1} \longrightarrow\!\!\!\triangleright^* C_{i-1}b_i =^{\text{FAIL-C}} \mathcal{F}(\mathtt{C}_i)b_i =^{\text{FAIL-COMMUTE}} \mathcal{F}(\mathtt{C}'_i)b_i =^{\text{FAIL-CPRIME}} C_i b_i$.

Thus $Cb = C_0 b_0 \xRightarrow{t} C_m b_m$. By FAIL-D, $C_m \in \mathbf{D}$, so $Cb \xRightarrow{t} C_m b_{m+1}$.

Claim: $C_m b_{m+1}$ refuses X. Suppose for contradiction $C_m b_{m+1} \xrightarrow{F}$ for some $F \in X$. By Lemma 3.5, there exists an IPO square, as in Figure 3.25, such that $b_{m+1} \xrightarrow{\mathcal{F}(\mathtt{F}')}$ and

$$\mathcal{F}(\mathtt{D}) \in \mathbf{D} \qquad \mathcal{F}(\mathtt{C}) = C_m \qquad \mathcal{F}(\mathtt{F}) = F \ .$$

By hypothesis, F is not an iso. Hence $\mathtt{F}$ is not an iso; Proposition 2.10 implies that $\mathtt{F}'$ is also not an iso. Furthermore $\mathcal{F}(\mathtt{F}') \in Y$, which contradicts the hypothesis that b_{m+1} refuses Y.

Claim: $C_m b_{m+1}$ is stable. Suppose for contradiction that $C_m b_{m+1} \longrightarrow\!\!\!\triangleright$. By Lemma 3.6, there exists $\mathtt{D}, \mathtt{F}'$ such that $b_{m+1} \xrightarrow{\mathcal{F}(\mathtt{F}')}$ and

$$\mathcal{F}(\mathtt{D}) \in \mathbf{D} \qquad \mathcal{F}(\mathtt{D}\mathtt{F}') = C_m \ .$$

Moreover, if $\mathtt{F}'$ is an iso then it is equal to id. We consider two cases:

case $\mathtt{F}' \neq \mathsf{id}$**:** Then $\mathcal{F}(\mathtt{F}') \in Z$, which contradicts the hypothesis that b_{m+1} refuses Z.

case $\mathtt{F}' = \mathsf{id}$**:** Then $b_{m+1} \longrightarrow\!\!\!\triangleright$, which contradicts the hypothesis that b_{m+1} is stable.

Thus Cb has failure (t, X), as desired. ■

3.8 Multi-hole contexts

As anticipated at the end of the previous chapter, this section enriches the definition of functorial reactive systems to model explicitly multi-hole contexts. At first glance, there is no obstacle in accommodating multi-hole contexts directly within the existing theory of reactive systems. (For simplicity, let us ignore functorial reactive systems until later.) However this does not work well as I illustrate in the next few paragraphs.

If we consider multi-hole term algebra contexts (as used in term rewriting), say, then we can choose the objects of **C** to be natural numbers and the arrows to be tuples of contexts (that use each hole exactly once) constructed from function symbols taken from some signature Σ. Concretely, if $\Sigma = \{\alpha, \alpha', \beta, \gamma\}$, where α and α' are constants, β is a 1-place function symbol, and γ is a 2-place function symbol, then, we have the following examples of arrows:

$$\begin{aligned} C_0 &\mathrel{\hat{=}} \langle -_1, \beta(\alpha')\rangle &&: 1 \to 2 \\ C_1 &\mathrel{\hat{=}} \langle \gamma(-_2, \alpha'), \alpha, \beta(-_1)\rangle &&: 2 \to 3 \\ C_1 C_0 &= \langle \gamma(\beta(\alpha'), \alpha'), \alpha, \beta(-_1)\rangle &&: 1 \to 3 \end{aligned}$$

But what is an *agent*? A natural choice is to take agents to be pure terms, i.e. arrows $0 \to 1$ (a 1-tuple containing a term context with 0 holes). But this is not supported by the definition of reactive system, where agents consist of all arrows $0 \to m$ for m an arbitrary object of **C**. This discrepancy is non trivial: if we try to confine the definition of labelled transition and bisimulation to use only a limited set of agents, say those arrows $0 \to m$ for $m \in A$, where A is some subset of the objects of **C**, then the proof of congruence goes wrong. The problem is that even if $m, m', n \in A$ in Figure 3.27, it is not necessarily the case that an RPO F', D', C yields an object $u \in A$. Thus even though F', D' forms an IPO with respect to a, l, it is not the case that $a \xrightarrow{F'}$ since the codomain of F' is not in A.

$$\begin{tikzcd} 0 \arrow[rr, "a"] \arrow[dd, "l"'] & & m \arrow[dl, "F'"'] \arrow[dd, "F"] \\ & u \arrow[dr, "C"] & \\ m' \arrow[ur, "D'"] \arrow[rr, "D"'] & & n \end{tikzcd}$$

3.27

However, it is exactly the fact that RPOs sometimes do *not* produce an object in A that gives multi-hole contexts their power and that makes them worth considering. To see why, suppose that we take **C** to be a category of exclusively 0- and 1-hole contexts. Then RPOs exist, as a corollary of Sewell's dissection result for terms (Lemma 1 in [Sew01]). Consequently, bisimulation is a congruence for term rewriting systems. The resulting labels are unnecessarily heavy, though. For consider the reaction rule $(\beta(\alpha), \alpha')$; we have $\alpha \xrightarrow{\beta(-)} \alpha'$ which corresponds to our intuition that α needs $\beta(-)$ to react. (When there is no confusion I use $-$ for $-_1$.) Unfortunately, we also have a labelled transition where the label contains a *complete copy of the redex*:

$$\alpha' \xrightarrow{\gamma(-,\beta(\alpha))} \gamma(\alpha', \alpha') \; .$$

This was discussed at the end of the previous chapter. This heavy labelled transition is absent when we look at multi-hole contexts, as illustrated with the help of the diagram below. (Tuple brackets $\langle\ \rangle$ are omitted from singletons.)

$$\begin{tikzcd}
0 \arrow[rr, "\alpha'"] \arrow[dd, "\beta(\alpha)"'] & & 1 \arrow[dl, "\langle -,\beta(\alpha)\rangle"'] \arrow[dd, "\gamma(-,\beta(\alpha))"] \\
& 2 \arrow[dr, "\gamma(-_1,-_2)"] & \\
1 \arrow[ur, "\langle \alpha',-\rangle"] \arrow[rr, "\gamma(\alpha',-)"'] & & 1
\end{tikzcd}$$

If we work in the category of 0- and 1-hole contexts then the outer square is an IPO, which gives rise to the transition $\xrightarrow{\gamma(-,\beta(a))}$ mentioned earlier. By admitting multi-hole contexts we have given the outer-square a simpler RPO.

It is now possible to make precise the motivation for the results developed in this section. The goal is to reconcile two things: (i) we want to restrict of the collection of agents of $\mathbf{C}$ to arrows $0 \rightarrow m$ for $m \in A$, where A can be a proper subset of the objects of $\mathbf{C}$ (for example $A = \{1\}$ in the case of multi-hole term contexts); (ii) we want to admit RPOs that yield objects which are not contained in A.

The key idea is to consider the notion of a *strict monoidal category* $(\mathbf{C}, \otimes, 0)$, a category $\mathbf{C}$ equipped with a functor $\otimes : \mathbf{C}\times\mathbf{C} \rightarrow \mathbf{C}$ and an object 0 such that $\otimes$ is strictly associative and has unit 0. The role of the tensor $\otimes$ is to "tuple" arrows and objects, e.g. recalling the term contexts C_0, C_1 from above, we have that:

$$\begin{aligned}
&C_0 && \hat{=}\ \langle -_1, \beta(\alpha')\rangle && : 1 \rightarrow 2\\
&C_1 && \hat{=}\ \langle \gamma(-_2,\alpha'), \alpha, \beta(-_1)\rangle && : 2 \rightarrow 3\\
&C_0 \otimes C_1 && = \langle -_1, \beta(\alpha'), \gamma(-_3,\alpha'), \alpha, \beta(-_2)\rangle && : 3 \rightarrow 5\\
&C_1 \otimes C_0 && = \langle \gamma(-_2,\alpha'), \alpha, \beta(-_1), -_3, \beta(\alpha')\rangle && : 3 \rightarrow 5
\end{aligned}$$

The following definition extends that of a functorial reactive system by postulating that both the upstairs and downstairs categories are monoidal and requiring that the functor between them respects the monoidal structure. The same enrichment could be performed on a reactive system. There is, however, no reason not to take more general approach shown here.

Definition 3.27 (functorial monoidal reactive system; cf. Definition 3.1) A *functorial monoidal reactive system* consists of a functorial reactive system $\mathcal{F} : \hat{\mathbf{C}} \rightarrow \mathbf{C}$ with the following added structure:

- Both $\hat{\mathbf{C}}$ and $\mathbf{C}$ are strict monoidal categories with unit objects ε and 0, respectively, and tensor $\otimes$ (the same symbol being used for both categories).
- $\mathcal{F}$ preserves tensors, i.e. $\mathcal{F}(\mathtt{C}_1 \otimes \mathtt{C}_0) = \mathcal{F}(\mathtt{C}_1) \otimes \mathcal{F}(\mathtt{C}_0)$.

- $\mathcal{F}$ preserves and creates units, i.e. for all $\mathtt{u} \in \mathsf{obj}\,\hat{\mathbf{C}}$, $\mathcal{F}(\mathtt{u}) = 0$ iff $\mathtt{u} = \varepsilon$.
- There is a subset A of $\mathbf{C}$-objects and a subset $\mathtt{A}$ of $\hat{\mathbf{C}}$-objects, where $\mathtt{A}$ is the preimage under $\mathcal{F}$ of A. We use $m, m', \ldots$ to range over A and $\mathtt{m}, \mathtt{m}', \ldots$ to range over $\mathtt{A}$.
- $\mathsf{Reacts} \subseteq \bigcup_{m \in A} \mathbf{C}(0, m)^2$.
- Pairing with an agent yields a reactive context: $a \otimes \mathsf{id}_{m'} \in \mathbf{D}$ for $a : 0 \rightarrow m$. ■

The *agents* are arrows $0 \rightarrow m$ where $m \in A$ and the *agent contexts* are arrows $m \rightarrow m'$, for $m, m' \in A$. Thus the reaction rules of Reacts are only between agents. We overload the terminology for $\hat{\mathbf{C}}$ in a straightforward way: arrows $\varepsilon \rightarrow \mathtt{m}$ of $\hat{\mathbf{C}}$ are also agents for $\mathtt{m} \in \mathtt{A}$; arrows $\mathtt{m} \rightarrow \mathtt{m}'$ of $\hat{\mathbf{C}}$ are also agent contexts for $\mathtt{m}, \mathtt{m}' \in \mathtt{A}$.

The definition of labelled transition confines the arguments to be agents:

Definition 3.28 (labelled transition for functorial monoidal reactive systems ($\xrightarrow[\mathtt{m}]{\cdot}\triangleright$); cf. Definition 3.3) $a \xrightarrow[\mathtt{m}]{F}\triangleright a'$ iff $a \xrightarrow{F}\triangleright a'$ and a, a' are agents (thus forcing F to be an agent context). ■

Two properties now replace the hypothesis that $\mathcal{F}$ has all redex-RPOs. The first asserts that the RPO consists either of agent contexts or of pairing operations that place disjoint instances of $\mathtt{a}$ and $\mathtt{l}$ into the composite $\mathtt{Fa} = \mathtt{Dl}$:

Definition 3.29 ($\mathcal{F}$ has all monoidal-redex-RPOs; cf. Definition 3.4) Suppose $\mathcal{F} : \hat{\mathbf{C}} \rightarrow \mathbf{C}$ is a functorial monoidal reactive system and that $\mathtt{a}, \mathtt{l}$ are agents, $\mathtt{F}, \mathtt{D}$ are agent contexts, $\mathcal{F}(\mathtt{D}) \in \mathbf{D}$, and there exists $r \in \mathbf{C}$ such that $(\mathcal{F}(\mathtt{l}), r) \in \mathsf{Reacts}$. Then $\mathcal{F}$ *has all monoidal-redex-RPOs* if any square, such as in Figure 3.28, has an RPO (as shown) such that if $\mathtt{u} \notin \mathtt{A}$ then $\mathtt{u} = \mathtt{m} \otimes \mathtt{m}'$, $\mathtt{F}' = \mathsf{id}_{\mathtt{m}} \otimes \mathtt{l}$, and $\mathtt{D}' = \mathtt{a} \otimes \mathsf{id}_{\mathtt{m}'}$. ■

$$\begin{tikzcd}
\varepsilon \arrow[rr, "\mathtt{a}"] \arrow[dd, "\mathtt{l}"'] & & \mathtt{m} \arrow[dl, "\mathtt{F}'"'] \arrow[dd, "\mathtt{F}"] \\
& \mathtt{u} \arrow[dr, "\mathtt{C}"] & \\
\mathtt{m}' \arrow[ur, "\mathtt{D}'"] \arrow[rr, "\mathtt{D}"'] & & \mathtt{n}
\end{tikzcd}$$

3.28

The second property asserts that pairing agent contexts yields an IPO:

Definition 3.30 ($\mathcal{F}$ has all monoidal-redex-IPOs) A functorial monoidal reactive system $\mathcal{F} : \hat{\mathbf{C}} \rightarrow \mathbf{C}$ *has all monoidal-redex-IPOs* if any square, such as in Figure 3.29, is an IPO, provided $\mathtt{a}, \mathtt{l}$ are agents and there exists $r \in \mathbf{C}$ such that $(\mathcal{F}(\mathtt{l}), r) \in \mathsf{Reacts}$. ■

$$\begin{tikzcd}
\varepsilon \arrow[r, "\mathtt{a}"] \arrow[d, "\mathtt{l}"'] & \mathtt{m} \arrow[d, "\mathsf{id}_{\mathtt{m}} \otimes \mathtt{l}"] \\
\mathtt{m}' \arrow[r, "\mathtt{a} \otimes \mathsf{id}_{\mathtt{m}'}"'] & \mathtt{m} \otimes \mathtt{m}'
\end{tikzcd}$$

3.29

Just before giving a proof of congruence for strong bisimulation we need to consider a corollary of Lemma 3.7 for functorial monoidal reactive systems:

Corollary 3.31 (portable IPO pasting for functorial monoidal reactive systems; cf. Lemma 3.7) Suppose $\mathcal{F} : \hat{\mathbf{C}} \rightarrow \mathbf{C}$ is a functorial monoidal reactive system and has all monoidal-redex-RPOs. The following inference rule holds:

$$\frac{\begin{tikzcd} \cdot \arrow[r, "\mathtt{C}"] \arrow[d, "\mathtt{F'}"'] & \cdot \arrow[d, "\mathtt{F}"] \\ \cdot \arrow[r, "\mathtt{C'}"'] & \cdot \end{tikzcd} \text{ is an IPO consisting of agent contexts} \qquad a \xrightarrow[\mathsf{m}]{\mathcal{F}(\mathtt{F'})} a' \qquad \mathcal{F}(\mathtt{C'}) \in \mathbf{D}}{\mathcal{F}(\mathtt{C})a \xrightarrow[\mathsf{m}]{\mathcal{F}(\mathtt{F})} \mathcal{F}(\mathtt{C'})a'} \; .$$

■

Strong bisimulation compares pairs of agents:

Definition 3.32 (strong bisimulation over $\xrightarrow[\mathsf{m}]{\cdot}$; cf. Definition 3.8) Let $\sim_{\mathsf{m}} \subseteq \bigcup_{m \in A} \mathbf{C}(0, m)^2$ be the largest strong bisimulation over $\xrightarrow[\mathsf{m}]{\cdot}$ such that $\sim_{\mathsf{m}}$ contains only pairs of agents with common codomain. ■

Finally we prove congruence. As promised at the end of the previous chapter, the argument mirrors closely congruence proofs in typical process calculi. In particular, two cases are distinguished when analysing the transition $Ca \xrightarrow[\mathsf{m}]{F}$: (i) a, C, and F all contribute, in which case a itself has a labelled transition; (ii) only C and F contribute, in which case $Cb \xrightarrow[\mathsf{m}]{F}$ without needing to consider the behaviour of a and b.

Theorem 3.33 (congruence for $\sim_{\mathsf{m}}$ w.r.t. agent contexts) Let $\mathcal{F} : \hat{\mathbf{C}} \to \mathbf{C}$ be a functorial monoidal reactive system which has all monoidal-redex-RPOs and has all monoidal-redex-IPOs. Then $\sim_{\mathsf{m}}$ is a congruence with respect to agent contexts, i.e. $a \sim_{\mathsf{m}} b$ implies $Ca \sim_{\mathsf{m}} Cb$ for any agent context $C \in \mathbf{C}$ of the required domain.

Proof By symmetry, it is sufficient to show that the following relation is a strong simulation:

$$\mathcal{S} \mathrel{\hat{=}} \{(Ca, Cb) \;\; / \;\; a \sim_{\mathsf{m}} b \text{ and } C \in \mathbf{C} \text{ is an agent context}\} \; .$$

$$\begin{tikzcd} \varepsilon \arrow[r, "\mathtt{a}"] \arrow[dd, "\mathtt{l}"'] & \mathtt{m} \arrow[r, "\mathtt{C}"] \arrow[d, "\mathtt{F'}"] & \mathtt{n} \arrow[dd, "\mathtt{F}"] \\ & \mathtt{u} \arrow[dr, "\mathtt{C'}"] & \\ \mathtt{m'} \arrow[ur, "\mathtt{D'}"] \arrow[rr, "\mathtt{D}"'] & & {} \end{tikzcd}$$

3.30

Suppose that $a \sim_{\mathsf{m}} b$ and $C \in \mathbf{C}$, and thus $(Ca, Cb) \in \mathcal{S}$, where $a, b : 0 \to m$ and $C : m \to n$. Suppose $Ca \xrightarrow[\mathsf{m}]{F} a'$. By the definition of $\xrightarrow[\mathsf{m}]{F}$ and the hypothesis that $\mathcal{F}$ creates compositions, there exist $\hat{\mathbf{C}}$-arrows $\mathtt{a} : \varepsilon \to \mathtt{m}, \mathtt{C} : \mathtt{m} \to \mathtt{n}, \mathtt{l} : \varepsilon \to \mathtt{m'}, \mathtt{F}, \mathtt{D}$ and a $\mathbf{C}$-arrow $r : 0 \to \mathcal{F}(\mathtt{m'})$ such that the big rectangle in Figure 3.30 is an IPO and

$$a' = \mathcal{F}(\mathtt{D})r \qquad \mathcal{F}(\mathtt{D}) \in \mathbf{D} \qquad (\mathcal{F}(\mathtt{l}), r) \in \mathsf{Reacts}$$
$$\mathcal{F}(\mathtt{a}) = a \qquad \mathcal{F}(\mathtt{C}) = C \qquad \mathcal{F}(\mathtt{F}) = F \; .$$

Because $\mathcal{F}$ has all monoidal-redex-RPOs, there exist $\mathtt{F'}, \mathtt{D'}, \mathtt{C'}$ forming an RPO in $\hat{\mathbf{C}}$, as in Figure 3.30. Note that $\mathcal{F}(\mathtt{C'}) \in \mathbf{D}$ since $\mathcal{F}(\mathtt{C'})\mathcal{F}(\mathtt{D'}) = \mathcal{F}(\mathtt{D}) \in \mathbf{D}$. By Proposition 2.7, the small left-hand square of Figure 3.30 is an IPO; Proposition 2.9 implies that the small right-hand square is an IPO too. Since $\mathcal{F}$ has all monoidal-redex-RPOs, we have additional information depending on whether $\mathtt{u} \in \mathtt{A}$. We consider both cases.

Case $\mathtt{u} \in \mathtt{A}$: By definition, $a \xrightarrow[\mathtt{m}]{\mathcal{F}(\mathtt{F}')}\!\!\triangleright\, a''$ and $a' = \mathcal{F}(\mathtt{C}')a''$, where $a'' \mathrel{\hat{=}} \mathcal{F}(\mathtt{D}')r$. Since $a \sim_{\mathsf{m}} b$, there exists b'' such that $b \xrightarrow[\mathtt{m}]{\mathcal{F}(\mathtt{F}')}\!\!\triangleright\, b''$ and $a'' \sim_{\mathsf{m}} b''$. Since the small right-hand square in Figure 3.30 is an IPO and $\mathcal{F}(\mathtt{C}') \in \mathbf{D}$, Corollary 3.31 implies that $Cb \xrightarrow[\mathtt{m}]{F}\!\!\triangleright\, \mathcal{F}(\mathtt{C}')b''$. Also, $a'' \sim_{\mathsf{m}} b''$ implies $(\mathcal{F}(\mathtt{C}')a'', \mathcal{F}(\mathtt{C}')b'') \in \mathcal{S}$, as desired.

Case $\mathtt{u} \notin \mathtt{A}$: We have that $\mathtt{F}' = \mathsf{id}_{\mathtt{m}} \otimes \mathtt{1}$ and $\mathtt{D}' = \mathtt{a} \otimes \mathsf{id}_{\mathtt{m}'}$. Since $\mathcal{F}$ lifts agents there exists $\mathtt{b} : \varepsilon \rightarrow \mathtt{m}_0$ such that $\mathcal{F}(\mathtt{b}) = b$, and thus $\mathcal{F}(\mathtt{m}_0) = m = \mathcal{F}(\mathtt{m})$. Since $\mathcal{F}$ preserves tensors, $\mathcal{F}(\mathtt{F}') = \mathsf{id}_{\mathcal{F}(\mathtt{m})} \otimes \mathcal{F}(\mathtt{1}) = \mathsf{id}_{\mathcal{F}(\mathtt{m}_0)} \otimes \mathcal{F}(\mathtt{1}) = \mathcal{F}(\mathsf{id}_{\mathtt{m}_0} \otimes \mathtt{1})$. Since $\mathcal{F}$ allows IPO sliding, the small right-hand IPO square of Figure 3.30 can be slid to form the small right-hand IPO square in Figure 3.31, where

$$\begin{tikzcd}
\varepsilon \arrow[r, "\mathtt{b}"] \arrow[d, "\mathtt{1}"'] & \mathtt{m}_0 \arrow[r, "\mathtt{C}_0"] \arrow[d, "\mathsf{id}_{\mathtt{m}_0}\otimes\mathtt{1}"] & {} \arrow[d, "\mathtt{F}_0"] \\
\mathtt{m}' \arrow[r, "\mathtt{b}\otimes\mathsf{id}_{\mathtt{m}'}"'] & \mathtt{m}_0\otimes\mathtt{m}' \arrow[r, "\mathtt{C}_0'"'] & {}
\end{tikzcd}$$

3.31

$$\mathcal{F}(\mathtt{C}_0) = \mathcal{F}(\mathtt{C}) \qquad \mathcal{F}(\mathtt{F}_0) = \mathcal{F}(\mathtt{F}) \qquad \mathcal{F}(\mathtt{C}_0') = \mathcal{F}(\mathtt{C}')\ .$$

Since $\mathcal{F}$ has all monoidal-redex-IPOs, the small left-hand square of Figure 3.31 is an IPO. Since $\mathcal{F}$ has all monoidal-redex-RPOs, Proposition 2.9 implies that the big rectangle in Figure 3.31 is an IPO. Since $\mathcal{F}$ preserves tensors and since pairing with an agent yields a reactive context, $\mathcal{F}(\mathtt{C}_0'(\mathtt{b} \otimes \mathsf{id}_{\mathtt{m}'})) = \mathcal{F}(\mathtt{C}')(b \otimes \mathsf{id}_{\mathcal{F}(\mathtt{m}')}) \in \mathbf{D}$ so:

$$Cb \xrightarrow[\mathtt{m}]{F}\!\!\triangleright\, \mathcal{F}(\mathtt{C}_0'(\mathtt{b} \otimes \mathsf{id}_{\mathtt{m}'}))r = \mathcal{F}(\mathtt{C}')(b \otimes r) = \mathcal{F}(\mathtt{C}')(\mathsf{id}_m \otimes r)b\ .$$

The last equality is a standard property of strict monoidal categories. Furthermore,

$$a' = \mathcal{F}(\mathtt{D})r = \mathcal{F}(\mathtt{C}')\mathcal{F}(\mathtt{D}')r = \mathcal{F}(\mathtt{C}')\mathcal{F}(\mathtt{a} \otimes \mathsf{id}_{\mathtt{m}'})r = \mathcal{F}(\mathtt{C}')(a \otimes r) = \mathcal{F}(\mathtt{C}')(\mathsf{id}_m \otimes r)a\ .$$

Thus $a \sim_{\mathsf{m}} b$ implies $(\mathcal{F}(\mathtt{C}')(\mathsf{id}_m \otimes r)a, \mathcal{F}(\mathtt{C}')(\mathsf{id}_m \otimes r)b) \in \mathcal{S}$ as desired. ■

To round out this section, let us look again at the example of multi-hole term contexts over a signature Σ.

Definition 3.34 (multi-hole term contexts ($\mathbf{T}^*(\Sigma)$)) Given a *signature* Σ of function symbols then the category of *multi-hole term contexts* $\mathbf{T}^*(\Sigma)$ over Σ is constructed as follows: the objects are the natural numbers; an arrow $j \rightarrow k$ is a k-tuple of terms over the signature $\Sigma \cup \{-_1, \ldots, -_j\}$ containing exactly one use of each hole $-_i$ $(1 \leq i \leq j)$. (When $j = 1$, I often write $-_1$ as $-$.) The *identities* are: $\mathsf{id}_j \mathrel{\hat{=}} \langle -_1, \ldots, -_j \rangle : j \rightarrow j$. For $C = \langle t_1, \ldots, t_k \rangle : j \rightarrow k$ and $D : k \rightarrow m$, their *composition* is the substitution

$$DC \mathrel{\hat{=}} \{t_1/-_1, \cdots, t_k/-_k\}D\ .$$ ■

In order to apply Theorem 3.33 to $\mathbf{T}^*(\Sigma)$, we let $\mathbf{C} = \hat{\mathbf{C}} = \mathbf{T}^*(\Sigma)$ and $\mathcal{F}$ be the identity functor. If we let $A = \{1\}$ then

- an agent of $\mathbf{T}^*(\Sigma)$ is a term $a : 0 \rightarrow 1$;

- agent context of $\mathbf{T}^*(\Sigma)$ is a term context $C : 1 \rightarrow 1$, i.e. a term containing a single hole.

We may choose any subcategory of $\mathbf{T}^*(\Sigma)$ to be the reactive contexts, subject to the conditions in Definition 3.27. The labelled transitions of $\mathbf{T}^*(\Sigma)$ depend, of course, on the reaction rules. Once these are specified, the labelled transition relation $\xrightarrow[\mathsf{m}]{\cdot}$ is determined by Definition 3.28 and the induced strong bisimulation $\sim_{\mathsf{m}}$ by Definition 3.32.

As a corollary of Sewell's dissection result for terms (Lemma 1 in [Sew01]), $\mathcal{F}$ has all monoidal-redex-RPOs and has all monoidal-redex-IPOs. Hence from Theorem 3.33 the induced strong bisimulation $\sim_{\mathsf{m}}$ is preserved by all term contexts. Let us now revisit the reactive system whose only reaction rule is $(\beta(\alpha), \alpha')$. It contains exactly the following labelled transitions:

$$D(\beta(\alpha)) \xrightarrow[\mathsf{m}]{-} D(\alpha') \qquad \text{for all reactive term contexts } D \in \mathbf{D}$$
$$\alpha \xrightarrow[\mathsf{m}]{\beta(-)} \alpha'$$

These agree with the transitions found by Sewell in the case of ground-term rewriting. I believe for any reaction rules specified by Reacts the derived labelled transitions for $\mathbf{T}^*(\Sigma)$ coincide exactly with Sewell's.

Chapter 4

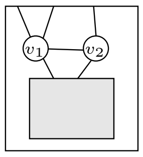
Sliding IPO squares

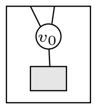
4.1 Introduction and motivation

The previous chapter shows a series of congruence proofs, each one for a different operational equivalence. The theme throughout is the separation of "reactive system" into two categories with a functor $\mathcal{F}$ between them: the domain of $\mathcal{F}$, i.e. $\hat{\mathbf{C}}$, is a category in which RPOs and IPOs exist; the codomain of $\mathcal{F}$, i.e. $\mathbf{C}$, is a category containing (i) the agents that perform reactions and labelled transitions and (ii) the agent contexts that serve as the labels and specify the closure condition for congruence. This separation is useful because the category for which we prove a congruence result is typically not one in which RPOs exist, as I show in the next chapter when considering categories of graph contexts. Thus functorial reactive systems are a powerful generalisation of reactive systems.

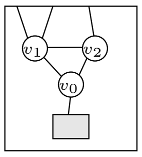
This separation imposes a burden, though, in the form of the list of requirements given in Definition 3.1, i.e.: $\mathcal{F}$ lifts agents, creates isos, creates compositions, and allows IPO sliding. The motivation for the current chapter is to alleviate this burden by showing that all of these properties follow directly if we construct $\hat{\mathbf{C}}$, $\mathbf{C}$, and $\mathcal{F}$ from a precategory $\mathbf{A}$ in a particular way. The assumption is that $\mathbf{A}$ is easier to construct than the others. (The next chapter gives a concrete instance of $\mathbf{A}$.)

Precategories are defined formally below. By way of motivation, though, let us look informally at an example we have in mind when deriving $\mathcal{F} : \hat{\mathbf{C}} \rightarrow \mathbf{C}$ from $\mathbf{A}$. A precategory is just like a category but has a composition operation that is partial, i.e. not always defined. For example, consider a precategory $\mathbf{A}$ of "raw contexts". For the purpose of this example, take the objects to be natural numbers (representing the number of ports in an interface). Take the arrows $m \rightarrow n$ to be just like normal graphs but with some added structure, namely an inner and outer interface. An arrow is thus a doughnut-shaped graph consisting of a set of nodes (which we call the *support*) and an arc relation; the arcs link nodes to one another and to the interface ports; m (ordered) interface ports sit on the inside "hole" and n (ordered ones) on the outside. (These are simpler than the graph contexts that appear in the next chapter.) Composition consists of placing one raw context inside the hole of another and joining together common ports. To see how this works, consider arrows $A_0 : 1 \rightarrowtail 2$ and $A_1 : 2 \rightarrowtail 3$:

$A_1 =$ 

$A_0 =$ 

Then their composition, which we denote $A_1 \oplus A_0 : 1 \rightarrowtail 3$, is as follows:

$A_1 \oplus A_0 =$ 

This example reveals why composition is partial in $\mathbf{A}$. If we form A'_0 from A_0 by renaming the node v_0 by v_2 then the supports of A'_0 and A_1 are *not disjoint*. Thus the composition $A_1 \oplus A'_0$ is undefined since there is no clear choice for the support of the composite.

There are several possible ways of building a true category (i.e. with a total composition relation) from the data provided by $\mathbf{A}$. Two possible ways are as follows:

- We can construct a category $\hat{\mathbf{C}}$ whose objects are pairs (m, U) where m is an object of $\mathbf{A}$ (in this case a natural number) and U is a set. An arrow $(m_0, U_0) \rightarrowtail (m_1, U_1)$ consists of an $\mathbf{A}$-arrow $A : m_0 \rightarrowtail m_1$ for which $U_0 \subseteq U_1$ and $U_1 \setminus U_0$ is equal to the support of A. Thus we can incorporate A_1 (given above) into many possible arrows in $\hat{\mathbf{C}}$, e.g. $(2, \varnothing) \xrightarrow{A_1} (3, \{v_1, v_2\})$ and $(2, \{w\}) \xrightarrow{A_1} (3, \{w, v_1, v_2\})$. As a result composition is always well-defined: if $(m_i, U_i) \xrightarrow{A_i} (m_{i+1}, U_{i+1})$ are $\hat{\mathbf{C}}$-arrows for $i = 0, 1$ then the supports of A_0 and A_1 are disjoint.

- We can construct a category $\mathbf{C}$, whose objects are the objects of $\mathbf{A}$ and whose arrows are $\simeq$-equivalence classes of $\mathbf{A}$-arrows. Two $\mathbf{A}$-arrows are $\simeq$-equivalent iff they are graph-isomorphic, i.e. have (potentially) different supports but look like the same graphs. Composition for this category is also well-defined since it is always possible when composing arrows to find representatives of each equivalence class with disjoint supports.

One might consider a third way, i.e. to use the arrows of $\mathbf{A}$ but rename the supports so as to make them disjoint when composing arrows; but this yields a bicategory, since composition is not associative. This approach is being investigated by Peter Sewell; I do not consider this line of research in this dissertation.

What is the relationship between $\hat{\mathbf{C}}$ and $\mathbf{C}$? There is a simple functor $\mathcal{F}$ which discards the set component of each $\hat{\mathbf{C}}$-object (i.e. $\mathcal{F} : (m, U) \mapsto m$) and maps each arrow to its

$\hat{=}$-equivalence class. As we see later in this chapter, this functor has all of the desired properties listed earlier, e.g. $\mathcal{F}$ allows IPO sliding.

The rest of this chapter repeats the above constructions in full formality.

4.2 Properties of A

First we formally define a precategory (see [MSS00]):

Definition 4.1 (precategory) A *precategory* $\mathbf{A}$ consists of similar data to that of a category: a collection of objects $m, n, \ldots$; a collection of arrows $\mathbf{A}(m, m')$ between objects m and m'; an identity arrow $\mathsf{id}_m \in \mathbf{A}(m, m)$ for all m; and a *partial* composition operation, which we write here as $\oplus : \mathbf{A}(m_1, m_2) \times \mathbf{A}(m_0, m_1) \rightharpoonup \mathbf{A}(m_0, m_2)$ on arrows. Identity: composition with an identity arrow is always well-defined, i.e. for all $A : m_0 \rightarrow m_1$, we have that $\mathsf{id}_{m_1} \oplus A = A = A \oplus \mathsf{id}_{m_0}$ and both compositions are defined. Associativity: if $A_2 \oplus A_1$ and $A_1 \oplus A_0$ are defined then either both $A_2 \oplus (A_1 \oplus A_0)$ and $(A_2 \oplus A_1) \oplus A_0$ are undefined or both are defined and equal. ■

Next we define some special properties of a precategory. These properties form a specification (used in Chapter 5) which any precategory is required to satisfy in order to make use of the constructions and propositions of this chapter.

Definition 4.2 (well-supported precategory) $\mathbf{A}$ is a *well-supported precategory* iff it is a precategory and it satisfies the following properties:

- $\mathbf{A}$ has a *support function* $|\cdot|$ that maps an arrow to a set such that $A_1 \oplus A_0$ is defined iff $|A_1| \cap |A_0| = \varnothing$ and $\mathsf{Dom}\, A_1 = \mathsf{Cod}\, A_0$. The support function satisfies additionally two axioms:

$$|A_1 \oplus A_0| = |A_1| \uplus |A_0| \qquad \text{(provided } A_1 \oplus A_0 \text{ is defined)} \qquad \textsc{Supp-comp}$$
$$|\mathsf{id}_m| = \varnothing\ . \qquad \textsc{Supp-id}$$

- For any arrow $A \in \mathbf{A}(m_0, m_1)$ and any injective map ρ for which $\mathsf{Dom}\, \rho \supseteq |A|$, there exists an arrow $\rho \bullet A \in \mathbf{A}(m_0, m_1)$, which is called the *support translation by* ρ *of* A, where:

$$\rho \bullet \mathsf{id}_m = \mathsf{id}_m \qquad \textsc{Trans-id-r}$$
$$\rho \bullet (A_1 \oplus A_0) = \rho \bullet A_1 \oplus \rho \bullet A_0 \qquad \textsc{Trans-comp-r}$$
$$\mathsf{Id}_{|A|} \bullet A = A \qquad \textsc{Trans-id-l}$$
$$(\rho_1 \circ \rho_0) \bullet A = \rho_1 \bullet (\rho_0 \bullet A) \qquad \textsc{Trans-comp-l}$$
$$\rho_0 \restriction |A| = \rho_1 \restriction |A| \text{ implies } \rho_0 \bullet A = \rho_1 \bullet A \qquad \textsc{Trans-res}$$
$$|\rho \bullet A| = \rho |A|\ . \qquad \textsc{Trans-supp}$$

A note about TRANS-COMP-R: Since ρ is injective, TRANS-SUPP implies that the LHS is defined iff the RHS is defined. (These axioms are similar to those of Honda's Rooted P-Sets [Hon00], though his application concerns free names and renaming.) ■

4.3 Construction of $\hat{\mathbf{C}}$

We now turn to problem of building a genuine category from a well-supported precategory $\mathbf{A}$. The idea is to enrich the object structure with enough data so that composition is always defined. This construction is captured in the following definition:

Definition 4.3 (track) Given a well-supported precategory $\mathbf{A}$, the *track* of $\mathbf{A}$ is a category $\hat{\mathbf{C}}$. An object of $\hat{\mathbf{C}}$ is a *profile*: a pair (m, U) where m is an object of $\mathbf{A}$ and U is a set. We let p range over profiles and adopt the firm convention that the components of a profile p are always written (m, U) with suitable decoration, e.g. $p' = (m', U')$ and $p_i = (m_i, U_i)$. An arrow $p_0 \xrightarrow{A} p_1$ consists of an arrow $A \in \mathbf{A}(m_0, m_1)$ such that $U_0 \subseteq U_1$ and $|A| = U_1 \setminus U_0$. We always include the profiles when referring to an arrow of $\hat{\mathbf{C}}$ since different $\hat{\mathbf{C}}$-arrows can be constructed from the same $\mathbf{A}$-arrow, each with different profiles. The identities of $\hat{\mathbf{C}}$ are defined by $\mathsf{id}_p \;\hat{=}\; p \xrightarrow{\mathsf{id}_m} p$. Composition is defined in terms of the underlying composition in $\mathbf{A}$:

$$p_0 \xrightarrow{A_0} p_1 \xrightarrow{A_1} p_2 \;\hat{=}\; p_0 \xrightarrow{A_1 \oplus A_0} p_2 \;.$$ ■

Proposition 4.4 If $\hat{\mathbf{C}}$ is the track of $\mathbf{A}$ then $\hat{\mathbf{C}}$ is a category.

Proof

Composition is well-defined: if $p_i \xrightarrow{A_i} p_{i+1}$ are arrows for $i = 0, 1$, then $|A_1| \cap |A_0| = (U_2 \setminus U_1) \cap (U_1 \setminus U_0) = \varnothing$, so $A_1 \oplus A_0$ is defined. Furthermore, $U_0 \subseteq U_1 \subseteq U_2$ and

$$U_2 \setminus U_0 \;=\; (U_2 \setminus U_1) \uplus (U_1 \setminus U_0) \;=\; |A_1| \uplus |A_0| \;=^{\textsc{Supp-comp}}\; |A_1 \oplus A_0| \;,$$

so $p_0 \xrightarrow{A_1 \oplus A_0} p_2$ is an arrow of $\hat{\mathbf{C}}$ as desired.

The identity arrows are well-defined: By SUPP-ID, $|\mathsf{id}_m| = \varnothing$, so for any $p = (m, U)$, $p \xrightarrow{\mathsf{id}_m} p$ is an arrow of $\hat{\mathbf{C}}$.

Composition is associative and respects identities: Immediate because these same properties hold in $\mathbf{A}$. ■

4.4 Operations on $\hat{\mathbf{C}}$

In order to motivate the following results, let us recall the intuitions about the functor $\mathcal{F}$ (defined formally later): $\mathcal{F}$ maps $p_0 \xrightarrow{A} p_1$ to an isomorphism equivalence class of A and throws away the set data contained in the profiles p_0, p_1. To prove that $\mathcal{F}$ allows IPO sliding, we require two things. (i) We have to understand how two $\hat{\mathbf{C}}$-arrows are related when they have the same $\mathcal{F}$ image, i.e. $\mathcal{F}(p_0 \xrightarrow{A} p_1) = \mathcal{F}(p'_0 \xrightarrow{A'} p'_1)$. (ii) From the first piece of information, we have to slide similarly an IPO square whose left leg is $p_0 \xrightarrow{A} p_1$ to one whose left leg is $p'_0 \xrightarrow{A'} p'_1$.

For (i), it is clear that we can perform some *profile translation* (defined precisely later) on $p_0 \xrightarrow{A} p_1$ to replace the set components of p_0 and p_1 and then perform a support translation on the resulting arrow to arrive at $p'_0 \xrightarrow{A'} p'_1$. If these two operations (profile translation and support translation) are iso functors (and thus preserve categorical constructions) then we can accomplish (ii).

These two operations are not in fact iso functors on the whole of $\hat{\mathbf{C}}$ but they are on certain *convex subcategories*, as defined next. Fortunately, the subcategories in question are rich enough to be proxies for $\hat{\mathbf{C}}$ with respect to IPO squares, as shown in the result immediately following the definition:

Definition 4.5 (convex subcategories of $\hat{\mathbf{C}}$) For any sets $U_0 \subseteq U_1$, we write $\hat{\mathbf{C}}_{U_0,U_1}$ for the *convex subcategory of* $\hat{\mathbf{C}}$ *w.r.t.* U_0, U_1, namely the full subcategory of $\hat{\mathbf{C}}$ formed by taking only those profiles (m, U) for which $U_0 \subseteq U \subseteq U_1$. ■

Proposition 4.6 Suppose that Figure 4.1 commutes in $\hat{\mathbf{C}}$. Then it is an IPO in $\hat{\mathbf{C}}$ iff it is an IPO in $\hat{\mathbf{C}}_{U,U_2}$.

$$\begin{tikzcd} p \arrow[r, "A_0"] \arrow[d, "A_1"'] & p_0 \arrow[d, "B_0"] \\ p_1 \arrow[r, "B_1"'] & p_2 \end{tikzcd} \quad 4.1$$

Proof For any pair of arrows $p \xrightarrow{C} p' \xrightarrow{C'} p_2$, we have that $U \subseteq U' \subseteq U_2$, so $p' \in \mathsf{obj}\, \hat{\mathbf{C}}_{U,U_2}$ (Definition 4.5) hence $\hat{\mathbf{C}}_{U,U_2}$ satisfies the hypothesis of Proposition 2.12, whence the result follows. ■

Now we now define precisely profile translation and establish that it is an iso functor:

Proposition 4.7 (profile translation is an iso functor) If $W_1 = W_0 \uplus W$ and $W'_1 = W'_0 \uplus W$ then the following operation, called *profile translation*, induces an isomorphism of categories $\mathcal{H} : \hat{\mathbf{C}}_{W_0,W_1} \to \hat{\mathbf{C}}_{W'_0,W'_1}$,

$$\begin{array}{llll} \mathcal{H}: & (m, W_0 \uplus V) & \mapsto & (m, W'_0 \uplus V) \qquad \text{for } V \subseteq W \\ \mathcal{H}: & (p_0 \xrightarrow{A} p_1) & \mapsto & \mathcal{H}(p_0) \xrightarrow{A} \mathcal{H}(p_1) . \end{array}$$

Proof

$\mathcal{H}$ is well-defined on arrows: Suppose that $(p_0 \xrightarrow{A} p_1) \in \hat{\mathbf{C}}_{W_0,W_1}$ where $U_i = W_0 \uplus V_i$, $i = 0, 1$ for some $V_0 \subseteq V_1 \subseteq W$. Now, $W'_0 \uplus V_0 \subseteq W'_0 \uplus V_1$ and

$$(W'_0 \uplus V_1) \setminus (W'_0 \uplus V_0) \;=\; V_1 \setminus V_0 \;=\; |A|$$

so $(\mathcal{H}(p_0) \xrightarrow{A} \mathcal{H}(p_1)) \in \hat{\mathbf{C}}_{W'_0,W'_1}$ as desired.

$\mathcal{H}$ is a functor: Consider the action of $\mathcal{H}$ on identities:

$$\mathcal{H}(\mathsf{id}_p) \;=\; \mathcal{H}(p \xrightarrow{\mathsf{id}_m} p) \;=\; \mathcal{H}(p) \xrightarrow{\mathsf{id}_m} \mathcal{H}(p) \;=\; \mathsf{id}_{\mathcal{H}(p)}$$

and on compositions:

$$\begin{aligned}\mathcal{H}(p_0 \xrightarrow{A_0} p_1 \xrightarrow{A_1} p_2) &= \mathcal{H}(p_0 \xrightarrow{A_1 \oplus A_0} p_2)\\ &= \mathcal{H}(p_0) \xrightarrow{A_1 \oplus A_0} \mathcal{H}(p_2)\\ &= \mathcal{H}(p_0) \xrightarrow{A_1} \mathcal{H}(p_1) \xrightarrow{A_0} \mathcal{H}(p_2)\end{aligned}$$

$\mathcal{H}$ is an isomorphism of categories: By symmetry, the map $\mathcal{H}' : \hat{\mathbf{C}}_{W'_0,W'_1} \to \hat{\mathbf{C}}_{W_0,W_1}$ defined by

$$\begin{aligned}\mathcal{H}' :\ & (m, W'_0 \uplus V) & \mapsto\ & (m, W_0 \uplus V) \qquad \text{for } V \subseteq W\\ \mathcal{H}' :\ & (p_0 \xrightarrow{A} p_1) & \mapsto\ & \mathcal{H}'(p_0) \xrightarrow{A} \mathcal{H}'(p_1)\ .\end{aligned}$$

is also a functor and clearly inverts $\mathcal{H}$, as desired. ■

Finally we lift the support translation operation from $\mathbf{A}$ to $\hat{\mathbf{C}}$ in a straightforward way. This definition induces a functor on convex subcategories of $\hat{\mathbf{C}}$

Proposition 4.8 (support translation is an iso functor) Given $W_0 \subseteq W_1$ and an injection ρ with $\mathsf{Dom}\,\rho \supseteq W_1$, the following operation, called *support translation*, induces an isomorphism of categories $\rho\bullet(\cdot) : \hat{\mathbf{C}}_{W_0,W_1} \to \hat{\mathbf{C}}_{\rho W_0,\rho W_1}$,

$$\begin{aligned}\rho\bullet(m,U) &\ \hat{=}\ (m, \rho U)\\ \rho\bullet(p_0 \xrightarrow{A} p_1) &\ \hat{=}\ \rho\bullet p_0 \xrightarrow{\rho\bullet A} \rho\bullet p_1\ ,\end{aligned}$$

where $\rho\bullet A$ is the support translation by ρ of A in $\mathbf{A}$.

Proof

$\rho\bullet(\cdot)$ is well-defined on arrows: Suppose that $(p_0 \xrightarrow{A} p_1) \in \hat{\mathbf{C}}_{W_0,W_1}$. Then $W_0 \subseteq U_0 \subseteq U_1 \subseteq W_1 \subseteq \mathsf{Dom}\,\rho$. Thus $\rho U_0 \subseteq \rho U_1$ and

$$\rho U_1 \setminus \rho U_0 \ =^{\rho \text{ injective}}\ \rho(U_1 \setminus U_0) = \rho|A| \ =^{\textsc{Trans-supp}}\ |\rho\bullet A|\ ,$$

so $\rho\bullet(\cdot)$ maps an arrow to an arrow.

$\rho\bullet(\cdot)$ is a functor: By Trans-id-r and Trans-comp-r, $\rho\bullet(\cdot)$ preserves identities and compositions, so is a functor.

$\rho\bullet(\cdot)$ is an iso functor: Note that ρ has an inverse $\rho^{-1} : \mathsf{Im}\,\rho \rightarrowtail \mathsf{Dom}\,\rho$. Furthermore,

$$\rho^{-1}\bullet(\cdot) : \hat{\mathbf{C}}_{\rho W_0,\rho W_1} \to \hat{\mathbf{C}}_{W_0,W_1}$$

is a functor for the same reasons (given above) that $\rho\bullet(\cdot)$ is. By Trans-comp-l and Trans-id-l, the functors $\rho\bullet(\cdot)$ and $\rho^{-1}\bullet(\cdot)$ invert each other, so $\rho\bullet(\cdot)$ is an isomorphism of categories, as required. ■

4.5 Construction of C

We now turn to the construction of $\mathbf{C}$, which was described informally in Section 4.1. Recall that the arrows of $\mathbf{C}$ are equivalence classes of arrows of $\mathbf{A}$. In this section we make precise the underlying equivalence relation and the construction of $\mathbf{C}$ and verify that $\mathbf{C}$ is a well-defined category.

Definition 4.9 ($\simeq$-equivalence for A) Given two arrows $A, A' : m_0 \rightarrowtriangle m_1$ in $\mathbf{A}$, we say that they are *$\simeq$-equivalent*, written $A \simeq A'$, iff there exists a bijection $\rho : |A| \rightarrowtail\!\!\!\!\twoheadrightarrow |A'|$ such that $\rho \bullet A = A'$. By Trans-id-l and Trans-comp-l, $\simeq$ is an equivalence relation. ■

Now the construction of $\mathbf{C}$ is straightforward:

Definition 4.10 (support quotient) Given a well-supported precategory $\mathbf{A}$, the *support quotient* of $\mathbf{A}$ is a category $\mathbf{C}$. The objects of $\mathbf{C}$ are the objects of $\mathbf{A}$. The arrows $m_0 \rightarrowtriangle m_1$ of $\mathbf{C}$ are $\simeq$-equivalence classes of arrows in $\mathbf{A}$:

$$\mathbf{C}(m_0, m_1) \ \hat{=}\ \{[A]_{\simeq} \ / \ A \in \mathbf{A}(m_0, m_1)\} \ .$$

Identities: $\mathsf{id}_m \in \mathbf{C}(m,m) \ \hat{=}\ [\mathsf{id}_m]_{\simeq}$. Composition: if $A_1 \oplus A_0$ is defined in $\mathbf{A}$ then $[A_1]_{\simeq}[A_0]_{\simeq} \mathrel{\hat{=}} [A_1 \oplus A_0]_{\simeq}$. ■

This definition yields a well-defined category:

Proposition 4.11 If $\mathbf{C}$ is the support quotient of $\mathbf{A}$ then $\mathbf{C}$ is a category.
Proof

Composition is total: Consider any two arrows in $\mathbf{C}$ such as $[A_i]_{\simeq} : m_i \rightarrowtriangle m_{i+1}$ for $i = 0, 1$. Let W be a fresh set in bijection with $|A_1|$, as witnessed by $\rho : |A_1| \rightarrowtail\!\!\!\!\twoheadrightarrow W$. Then $\rho \bullet A_1 \oplus A_0$ is defined since $|\rho \bullet A_1| \cap |A_0| =^{\text{Trans-supp}} W \cap |A_0| = \varnothing$; thus

$$[A_1]_{\simeq}[A_0]_{\simeq} \ = \ [\rho \bullet A_1]_{\simeq}[A_0]_{\simeq} \ = \ [\rho \bullet A_1 \oplus A_0]_{\simeq} \ ,$$

as desired.

Composition is well-defined: Let $[A_i]_{\simeq} = [A'_i]_{\simeq}$ for $i = 0, 1$ with both $A_1 \oplus A_0$ and $A'_1 \oplus A'_0$ defined. Claim: $[A_1]_{\simeq}[A_0]_{\simeq} = [A'_1]_{\simeq}[A'_0]_{\simeq}$. By hypothesis, there exist bijections $\rho_i : |A_i| \rightarrowtail\!\!\!\!\twoheadrightarrow |A'_i|$ such that $A'_i = \rho_i \bullet A_i$ for $i = 0, 1$. Since $|A_1| \cap |A_0| = \varnothing$ and $|A'_1| \cap |A'_0| = \varnothing$, we can define $\rho \mathrel{\hat{=}} \rho_0 \uplus \rho_1$, a union of bijections with disjoint domains and disjoint codomains. Now,

$$\rho \bullet (A_1 \oplus A_0) \ =^{\text{Trans-comp-r}} \ \rho \bullet A_1 \oplus \rho \bullet A_0 \ =^{\text{Trans-res}} \ \rho_1 \bullet A_1 \oplus \rho_0 \bullet A_0 \ = \ A'_1 \oplus A'_0$$

so

$$[A_1]_{\simeq}[A_0]_{\simeq} \ = \ [A_1 \oplus A_0]_{\simeq} \ = \ [A'_1 \oplus A'_0]_{\simeq} \ = \ [A'_1]_{\simeq}[A'_0]_{\simeq} \ ,$$

as desired.

Composition is associative: Follows from the associativity of the underlying composition in $\mathbf{A}$.

Composition respects identities: Follows from the fact that composition respects identities in $\mathbf{A}$. ■

4.6 Construction of $\mathcal{F}$

In this final section we define a functor $\mathcal{F}$ from $\hat{\mathbf{C}}$ (the track of $\mathbf{A}$) to $\mathbf{C}$ (the support quotient of $\mathbf{A}$). We then verify that $\mathcal{F}$ has all the required properties, i.e.: $\mathcal{F}$ lifts agents, creates isos, creates compositions, creates left inverses, and allows IPO sliding. Two comments are in order. (i) For the first we verify a stronger property defined below, namely $\mathcal{F}$ lifts arrows by their domain. The reason for this choice is that there is no postulated distinguished object in $\mathbf{A}$ or $\mathbf{C}$ corresponding to the 0 of a functorial reactive system (see Definition 3.1), which is required when defining the property "$\mathcal{F}$ lifts agents". However, the stronger property "$\mathcal{F}$ lifts arrows by their domain" is well-defined. (ii) The penultimate property, "$\mathcal{F}$ creates left inverses" does not appear in the definition of a functorial reactive system but is used in Appendix B.

Definition 4.12 (support-quotienting functor) Let $\mathbf{A}$ be a well-supported precategory and $\hat{\mathbf{C}}$ and $\mathbf{C}$ be as in Definition 4.3 and Definition 4.10. Then we define a map $\mathcal{F} : \hat{\mathbf{C}} \to \mathbf{C}$ called the *support-quotienting functor*:

$$\begin{aligned} \mathcal{F} :&\ (m, U) &\mapsto&\ m \\ \mathcal{F} :&\ (p_0 \xrightarrow{A} p_1) &\mapsto&\ \mathcal{F}(p_0) \xrightarrow{[A]_{\simeq}} \mathcal{F}(p_1)\ . \end{aligned}$$ ■

Lemma 4.13 $\mathcal{F}$ is a functor.
Proof Observe the action on identities:

$$\mathcal{F}(\mathsf{id}_p) = \mathcal{F}(p \xrightarrow{\mathsf{id}_m} p) = [\mathsf{id}_m]_{\simeq} = \mathsf{id}_m$$

and on compositions:

$$\begin{aligned} \mathcal{F}(p_0 \xrightarrow{A_0} p_1 \xrightarrow{A_1} p_2) &= \mathcal{F}(p_0 \xrightarrow{A_1 \oplus A_0} p_2) \\ &= [A_1 \oplus A_0]_{\simeq} \\ &= [A_1]_{\simeq} [A_0]_{\simeq} \\ &= \mathcal{F}(p_1 \xrightarrow{A_1} p_2)\, \mathcal{F}(p_0 \xrightarrow{A_0} p_1) \end{aligned}$$ ■

Now we prove all the easy properties of $\mathcal{F}$:

Theorem 4.14 Let $\mathcal{F} : \hat{\mathbf{C}} \to \mathbf{C}$ be the support-quotienting functor constructed from $\mathbf{A}$. Then:

- $\mathcal{F}$ lifts arrows by their domain: if $\mathcal{F}(p_0) = \mathsf{Dom}\,[A]_{\simeq}$ then there exists $p_0 \xrightarrow{B} p_1$ such that $\mathcal{F}(p_0 \xrightarrow{B} p_1) = [A]_{\simeq}$.

- $\mathcal{F}$ creates isos: if $\mathcal{F}(p_0 \xrightarrow{A} p_1)$ is an iso then $p_0 \xrightarrow{A} p_1$ is an iso.

- $\mathcal{F}$ creates compositions: if

$$\mathcal{F}(p'_0 \xrightarrow{B} p'_2) = [A_1]_{\simeq} [A_0]_{\simeq}$$

then there exist $\hat{\mathbf{C}}$-arrows $p'_i \xrightarrow{B_i} p'_{i+1}$ with

$$\mathcal{F}(p'_i \xrightarrow{B_i} p'_{i+1}) = [A_i]_{\simeq} \qquad \text{for } i = 0, 1 \tag{4.1}$$

$$p'_0 \xrightarrow{B} p'_2 = p'_0 \xrightarrow{B_0} p'_1 \xrightarrow{B_1} p'_2 \tag{4.2}$$

- $\mathcal{F}$ creates left inverses: if $\mathsf{id}_{m_0} = [A_1]_{\doteq}\mathcal{F}(p_0 \xrightarrow{A_0} p_1)$ then there exists $p_1 \xrightarrow{A_1'} p_0$ such that $\mathsf{id}_{p_0} = p_0 \xrightarrow{A_0} p_1 \xrightarrow{A_1'} p_0$ and $\mathcal{F}(p_1 \xrightarrow{A_1'} p_0) = [A_1]_{\doteq}$.

Proof

$\mathcal{F}$ lifts arrows by their domain: Suppose $[A]_{\doteq} : (m_0, n_0) \longrightarrow (m_1, n_1)$, thus $A : (m_0, n_0) \longrightarrow (m_1, n_1)$ in $\mathbf{A}$. Let W be a fresh set in bijection with $|A|$, as witnessed by $\rho : |A| \rightarrowtail\!\!\!\!\rightarrow W$. Then $p_0 \xrightarrow{\rho \bullet A} p_1$ is an arrow in $\hat{\mathbf{C}}$, where $U_1 \mathrel{\hat{=}} U_0 \uplus W$. Furthermore $\mathcal{F}(p_0 \xrightarrow{\rho\bullet A} p_1) = [A]_{\doteq}$ as desired.

$\mathcal{F}$ creates isos: Suppose $\mathcal{F}(p_0 \xrightarrow{A} p_1)$ is an iso, i.e. there exists an $\mathbf{A}$-arrow $A' : m_1 \longrightarrow m_0$ such that:

$$\begin{aligned} [A']_{\doteq}[A]_{\doteq} &= \mathsf{id}_{m_0} = [\mathsf{id}_{m_0}]_{\doteq} \\ [A]_{\doteq}[A']_{\doteq} &= \mathsf{id}_{m_1} = [\mathsf{id}_{m_1}]_{\doteq} \end{aligned}$$

Without loss of generality, assume that $|A| \cap |A'| = \varnothing$. Then $A' \oplus A \doteq \mathsf{id}_{m_0}$ and $A \oplus A' \doteq \mathsf{id}_{m_1}$. By Supp-id and Trans-id-r, $A' \oplus A = \mathsf{id}_{m_0}$ and $A \oplus A' = \mathsf{id}_{m_1}$. By Supp-comp, $|A| = |A'| = \varnothing$. Thus $U_1 = U_0$ and $p_1 \xrightarrow{A'} p_0$ is a $\hat{\mathbf{C}}$-arrow. Moreover,

$$p_0 \xrightarrow{A} p_1 \xrightarrow{A'} p_0 = p_0 \xrightarrow{A' \oplus A} p_0 = p_0 \xrightarrow{\mathsf{id}_{m_0}} p_0 = \mathsf{id}_{p_0}$$

and symmetrically. Thus $p_0 \xrightarrow{A} p_1$ is an iso in $\hat{\mathbf{C}}$ as desired.

$\mathcal{F}$ creates compositions: Without loss of generality, assume that $|A_1| \cap |A_0| = \varnothing$. Then there exist p_0, p_1, p_2 such that $p_i \xrightarrow{A_i} p_{i+1}$ are arrows in $\hat{\mathbf{C}}$ for $i = 0, 1$. By the definition of $\mathcal{F}$, there exists a bijection $\rho : |A_1| \uplus |A_0| \rightarrowtail\!\!\!\!\rightarrow |B|$ such that $\rho\bullet(A_1 \oplus A_0) = B$; moreover $m_0 = m_0'$ and $m_2 = m_2'$. Let $B_i \mathrel{\hat{=}} \rho\bullet A_i$ for $i = 0, 1$. Let $(m_1', U_1') \mathrel{\hat{=}} (m_1, U_0' \uplus |B_0|)$, thus defining p_1'. We claim that $p_i' \xrightarrow{B_i} p_{i+1}'$ are arrows in $\hat{\mathbf{C}}$ for $i = 0, 1$; there are three things that we need to check:

$$\begin{gathered} U_0' \subseteq U_0' \uplus |B_0| \subseteq U_0' \uplus |B| = U_2' \\ U_1' \setminus U_0' = |B_0| \\ U_2' \setminus U_1' = (U_0' \uplus |B|) \setminus (U_0' \uplus |B_0|) = |B| \setminus |B_0| = |B_1| \,. \end{gathered}$$

Now, $B_1 \oplus B_0 = B$ by Trans-comp-r, from which (4.2) follows. Also, by Trans-res, $B_i \doteq A_i$ for $i = 0, 1$, from which (4.1) follows.

$\mathcal{F}$ creates left inverses: By hypothesis, $\mathsf{id}_{m_0} = [A_1]_{\doteq}[A_0]_{\doteq} = [A_1' \oplus A_0]_{\doteq}$ for some $A_1' : m_1 \longrightarrow m_0$ with $A_1 \doteq A_1'$, since composition in $\mathbf{C}$ is total. Thus $\mathsf{id}_{m_0} \doteq A_1' \oplus A_0$. By Supp-id, Supp-comp, and Trans-id-l, $\mathsf{id}_{m_0} = A_1' \oplus A_0$ and $|A_1'| = |A_0| = \varnothing$. Thus $U_0 = U_1$. Thus $p_1 \xrightarrow{A_1'} p_0$ is an arrow in $\hat{\mathbf{C}}$ and moreover $\mathcal{F}(p_1 \xrightarrow{A_1'} p_0) = [A_1']_{\doteq} = [A_1]_{\doteq}$ and $p_0 \xrightarrow{A_0} p_1 \xrightarrow{A_1'} p_0 = \mathsf{id}_{p_0}$, as desired. ■

An aside on the condition "$\mathcal{F}$ creates compositions": This looks tantalisingly close to the Conduché fibration property [Con72, Joh99, BF00], especially if one says that two decompositions in $\hat{\mathbf{C}}$ are equivalent if one is the result of a support translation of the other. Perhaps there is some 2-category version of the Conduché property which works exactly if one thinks of $\hat{\mathbf{C}}$ as a 2-category with support translations as 2-cells. Let us now return to the main flow of the argument.

Finally, we prove the key property, namely that $\mathcal{F}$ allows IPO sliding. The proof follows the outline given in Section 4.4. We start with two $\hat{\mathbf{C}}$-arrows with the same image under $\mathcal{F}$, namely $\mathcal{F}(p \xrightarrow{A_1} p_1) = \mathcal{F}(p' \xrightarrow{A_1'} p_1')$. The first arrow is the left leg of an IPO square in $\hat{\mathbf{C}}$. This square is also an IPO in $\hat{\mathbf{C}}_{U,U_2}$, the convex subcategory w.r.t. U, U_2 (Definition 4.5), where $U \subseteq U_2$ are the sets in, respectively, the upper-left and lower-right profiles of the square. We isomorphically transform this subcategory by profile translation and then support translation, gaining a new square that has three properties: it has the same image under $\mathcal{F}$ as the original; its left leg is $p' \xrightarrow{A_1'} p_1'$; it is an IPO in a convex subcategory, so is an IPO in $\hat{\mathbf{C}}$.

Before looking at the technical details, it is useful to consider a concrete case of sliding. Because we have not formally defined the graph contexts referred to at the beginning of this chapter, it is impossible to be precise about which commuting squares are IPOs and which are not. Nonetheless, the activity of "sliding" is relevant for all commuting squares, whether or not they are IPOs.

Let us consider a category of graph contexts formed as the track (Definition 4.3) of the precategory or raw contexts informally defined in Section 4.1. The arrows of this category are just like the raw contexts (doughnut-shaped graphs with an inner and outer interface) but with profiles (Definition 4.3) rather than just natural numbers as objects.

Consider the square in Figure 4.2(1). Its left leg has the same $\mathcal{F}$ image as the left leg of the square in Figure 4.2(3): the two graph contexts look essentially the same, the only difference being the supports. With a profile translation, we can replace the singleton set $\{u\}$ in the top-left corner of Figure 4.2(1) with a fresh 2-element set $\{u_0'', u_1''\}$, as shown in Figure 4.2(2). The freshness is essential to prevent clashes with the other nodes present in the square, namely v_0, v_1. Now, if ρ is defined as follows:

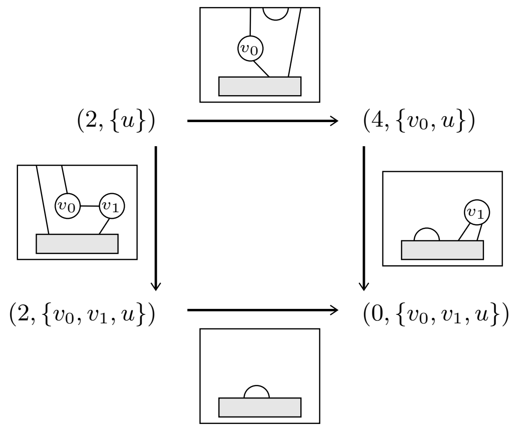
$$\begin{aligned} \rho : v_i &\mapsto v_i' \qquad i = 1, 2 \\ \rho : u_i'' &\mapsto u_i' \qquad i = 1, 2 \end{aligned}$$

then the support translation by ρ of Figure 4.2(2) yields Figure 4.2(3), as desired. Since the passage from Figure 4.2(1) to Figure 4.2(2) and then to Figure 4.2(3) was effected by iso functors, all the universal properties of Figure 4.2(1) (e.g. being an IPO) hold of Figure 4.2(3) too.

Theorem 4.15 ($\mathcal{F}$ allows IPO sliding) Let $\mathcal{F} : \hat{\mathbf{C}} \to \mathbf{C}$ be the support-quotienting functor constructed from $\mathbf{A}$. Then $\mathcal{F}$ allows IPO sliding (Definition 3.1).

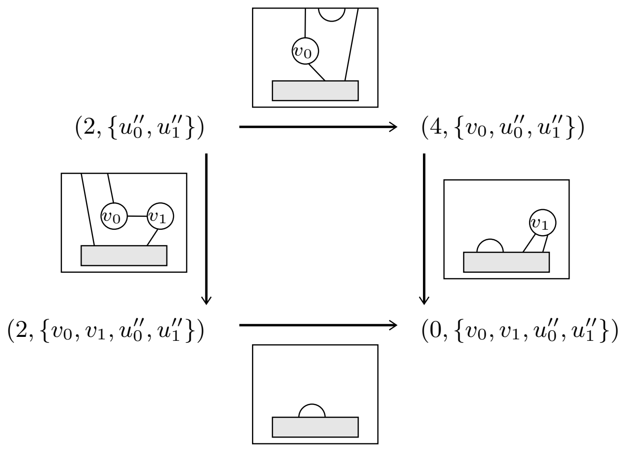
$$\begin{tikzcd} p \arrow[r, "A_0"] \arrow[d, "A_1"'] & p_0 \arrow[d, "B_0"] \\ p_1 \arrow[r, "B_1"'] & p_2 \end{tikzcd}$$
4.3

Proof Consider any IPO square in $\hat{\mathbf{C}}$, as in Figure 4.3, and any arrow $p' \xrightarrow{A_1'} p_1'$ with $\mathcal{F}(p' \xrightarrow{A_1'} p_1') = \mathcal{F}(p \xrightarrow{A_1} p_1)$; thus $A_1' = \alpha \bullet A_1$ for some bijection $\alpha : |A_1| \to |A_1'|$ and $U_1' = U' \uplus \alpha|A_1|$.



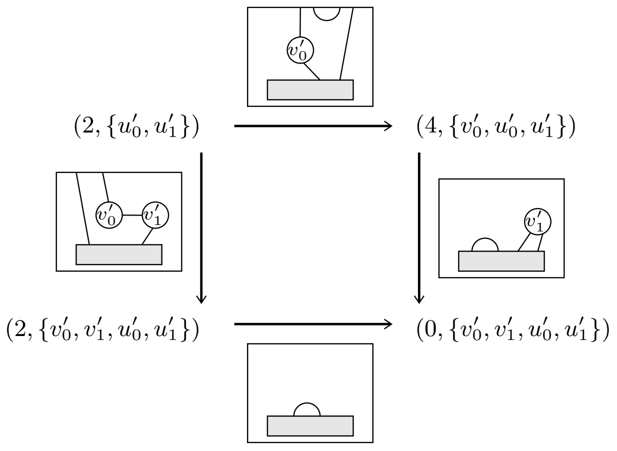
$$\begin{tikzcd}
(2,\{u\}) \arrow[r] \arrow[d] & (4,\{v_0,u\}) \arrow[d] \\
(2,\{v_0,v_1,u\}) \arrow[r] & (0,\{v_0,v_1,u\})
\end{tikzcd}$$

4.2(1) A commuting square before sliding



$$\begin{tikzcd}
(2,\{u_0'',u_1''\}) \arrow[r] \arrow[d] & (4,\{v_0,u_0'',u_1''\}) \arrow[d] \\
(2,\{v_0,v_1,u_0'',u_1''\}) \arrow[r] & (0,\{v_0,v_1,u_0'',u_1''\})
\end{tikzcd}$$

4.2(2) First we apply profile translation ...



$$\begin{tikzcd}
(2,\{u_0',u_1'\}) \arrow[r] \arrow[d] & (4,\{v_0',u_0',u_1'\}) \arrow[d] \\
(2,\{v_0',v_1',u_0',u_1'\}) \arrow[r] & (0,\{v_0',v_1',u_0',u_1'\})
\end{tikzcd}$$

4.2(3) ... and then support translation by ρ

Figure 4.2: Sliding

By Proposition 4.6, Figure 4.3 is an IPO in $\hat{\mathbf{C}}_{U,U_2}$. Let U'' be a fresh set in bijection with U', as witnessed by $\mu : U'' \rightarrowtail\!\!\!\!\twoheadrightarrow U'$. Let $U_2'' \mathrel{\hat{=}} U'' \uplus (U_2 \setminus U)$. Then $U_2 \setminus U = U_2'' \setminus U''$ so by Proposition 4.7, there is a profile translation $\mathcal{H} : \hat{\mathbf{C}}_{U,U_2} \longrightarrow \hat{\mathbf{C}}_{U'',U_2''}$ which is an iso functor and whose action on profiles is:

$$\begin{tikzcd}
\mathcal{H}(p) \arrow[r, "A_0"] \arrow[d, "A_1"'] & \mathcal{H}(p_0) \arrow[d, "B_0"] \\
\mathcal{H}(p_1) \arrow[r, "B_1"'] & \mathcal{H}(p_2)
\end{tikzcd} \quad 4.4$$

$$\mathcal{H} : (m, U \uplus V) \mapsto (m, U'' \uplus V) \qquad \text{for } V \subseteq U_2 \setminus U$$

and whose action on arrows leaves the underlying $\mathbf{A}$-arrow component unchanged. Since isomorphisms of categories preserve universal constructions, Figure 4.4 is an IPO in $\hat{\mathbf{C}}_{U'',U_2''}$ and has the same image under $\mathcal{F}$ as Figure 4.3 does.

Let W be a fresh set in bijection with $|B_1|$, as witnessed by $\beta : |B_1| \rightarrowtail\!\!\!\!\twoheadrightarrow W$. Let $\rho \mathrel{\hat{=}} \mu \uplus \alpha \uplus \beta$, a union of bijections with mutually disjoint domains and codomains. Also $\mathsf{Dom}\,\rho = U'' \uplus |A_1| \uplus |B_1| = U_2''$. Because ρ is bijective, Proposition 4.8 implies that there is a support translation $\rho\bullet(\cdot) : \hat{\mathbf{C}}_{U'',U_2''} \longrightarrow \hat{\mathbf{C}}_{\rho U'',\rho U_2''}$ which is an iso functor. Iso functors preserve universal constructions, so Figure 4.5 is a IPO in $\hat{\mathbf{C}}_{\rho U'',\rho U_2''}$ and has the same image under $\mathcal{F}$. By Proposition 4.6, this square is an IPO in $\hat{\mathbf{C}}$. Moreover,

$$\begin{tikzcd}
\rho\bullet\mathcal{H}(p) \arrow[r, "\rho\bullet A_0"] \arrow[d, "\rho\bullet A_1"'] & \rho\bullet\mathcal{H}(p_0) \arrow[d, "\rho\bullet B_0"] \\
\rho\bullet\mathcal{H}(p_1) \arrow[r, "\rho\bullet B_1"'] & \rho\bullet\mathcal{H}(p_2)
\end{tikzcd} \quad 4.5$$

$$\rho\bullet(\mathcal{H}(p) \xrightarrow{A_1} \mathcal{H}(p_1)) \;=\; \mu\bullet\mathcal{H}(p) \xrightarrow{\alpha\bullet A_1} (\mu \uplus \alpha)\bullet\mathcal{H}(p_1) \;=\; p' \xrightarrow{A_1'} p_1'$$

as desired. ■

Chapter 5

Action graph contexts

5.1 Introduction

As promised in the previous chapter, the present one gives a substantial example of a precategory **A-Ixt** of raw contexts. (The "Ixt" in **C-Ixt** is for "insertion context", a terminology explained in Section 5.6.) The need to handle graphs was the original motivation for functorial reactive systems: as we will see in Section 5.3, RPOs do not always exist for **C-Ixt**, the support quotient (Definition 4.10) of **A-Ixt**, so it is necessary to consider an upstairs category which does possess sufficient RPOs and a functor down to **C-Ixt**.

The precategory **A-Ixt** has all the extra structure required of **A**, namely a support function and a support translation operation, so is a well-supported precategory (Definition 4.2). By direct instantiation of the results of the previous chapter, we can construct three things: the track of **A-Ixt**, which we call $\hat{\mathbf{C}}$**-Ixt**; the support quotient of **A-Ixt**, which we call **C-Ixt**; and a functor $\mathcal{F} : \hat{\mathbf{C}}\textbf{-Ixt} \longrightarrow \mathbf{C}\textbf{-Ixt}$. These are related in the following table to their counterparts from the Chapter 4:

well-supported precategory:	$\mathbf{A}$	**A-Ixt**
track:	$\hat{\mathbf{C}}$	$\hat{\mathbf{C}}$**-Ixt**
support-quotienting functor:	$\downarrow \mathcal{F}$	$\downarrow \mathcal{F}$
support quotient:	$\mathbf{C}$	**C-Ixt**

Thus by Theorem 4.14 and Theorem 4.15, $\mathcal{F}$ lifts arrows by their domain, creates isos, creates compositions, and allows IPO sliding. Furthermore **A-Ixt** has a distinguished object 0, hence there are distinguished objects 0 and ε of **C-Ixt** and $\hat{\mathbf{C}}$**-Ixt**, with $\mathcal{F}(\varepsilon) = 0$, whence $\mathcal{F}$ lifts agents. Thus, any choice of reaction rules Reacts for **C-Ixt** and reactive context subcategory **D** of **C-Ixt** (Definition 2.1) yields a functorial reactive system (Definition 3.1).

The main hurdle then in using the congruence results of Chapter 3 is proving that $\mathcal{F}$ has all redex-RPOs (Definition 3.4). It turns out that this property fails for some choices of reaction rules, but that it *does* hold for a rich class of them.

As promised in Section 3.1, the category **C-Ixt** of graph contexts (the *codomain* of $\mathcal{F}$) does *not* admit enough RPOs for subtle reasons. Examples are shown in Section 5.3. The cause is the lack of sufficient intensional information as to *which* node in contexts C_0 or C_1, say, corresponds to a node in the composite C_0C_1. It is exactly **Ĉ-Ixt**, the *domain* of $\mathcal{F}$, that contains just enough structure to track how the nodes of two contexts are related to the nodes of their composition. By the definition of "$\mathcal{F}$ has all redex-RPOs", the proof obligation is to show that sufficient RPOs exist in **Ĉ-Ixt**. It transpires that a frontal attack on this is difficult (and not attempted in this dissertation): the combinatorial complexity of context composition, for example, is daunting, and the manipulation of the compositions required for verifying the existence of RPOs is too much to handle. However, an indirect approach based on a category **G-Inc** of graphs and inclusion embeddings (a kind of graph embedding) works smoothly. This is the subject of the next chapter, which shows that RPOs in **Ĉ-Ixt** correspond to *relative coproducts* in **G-Inc**; gives a direct construction of the latter subject to some conditions; and, finally, shows that these conditions are necessary for the existence of relative coproducts. Thus $\mathcal{F}$ has all redex-RPOs for a wide variety of reaction rules.

The development of the theory presented in this chapter and the next is adapted from [CLM00] but differs from it in the following ways: I omit extraneous material (not relevant to labelled transitions) and include full proofs of the existence of relative coproducts (Theorem 6.27 and Theorem 6.26). I also have renamed some of the structures: in this dissertation I write **Ĉ-Ixt** and **C-Ixt**; in [CLM00], these are denoted $\mathbf{PIns}_0$ and $\mathbf{ACxt}_0$ respectively.

5.2 Action calculi reviewed

I review a restricted class of the action calculi which were presented in [Mil96]. The rest of this chapter and the next make no specific use of the algebra of action calculi shown here since all of the work is done directly on graphs. Nonetheless, the design decisions taken when defining graphs are guided by the algebraic presentation of action calculi, so the latter provide valuable motivation.

A *closed, shallow action calculus* is a strict monoidal category whose objects are natural numbers $k, m, n, o \ldots$, and whose arrows are called *action graphs* and written $a : (k, o)$, $b : (m, n)$. (I avoid the usual arrow notation $a : k \to o$, reserving it for the context arrows of reactive systems.) The *tensor product* of these two action graphs is $a \otimes b : (k+m, o+n)$; the *composition* of $a : (k, o)$ and $b : (o, m)$ is $a \cdot b : (k, m)$; the *identity* action graph of arity (m, m) is i_m. The order of composition is not conventional in category theory: it connotes the orientation of the action graphs we work with. (Note that this composition is the horizontal juxtaposition of action graphs and has nothing to do with contextual composition which we consider later.) A pair of natural numbers (k, o) is an *arity*; let $\alpha, \beta, \ldots$ range over arities. I deal only with closed, shallow action calculi and so usually omit these adjectives from now on.

An action calculus has a *control signature* $\mathcal{K} = \{K, L, \ldots\}$ consisting of *controls*, each of which has an arity. There are constants $\mathsf{p} : (2, 2)$, $\mathsf{c} : (1, 2)$ and $\omega : (1, 0)$ for permutation, copy and discard. These constants represent only the swapping, sharing and elimination of *arcs*, not of *nodes*. They satisfy simple equations, e.g. $\mathsf{c}{\cdot}\mathsf{p} = \mathsf{c}$ representing the commutativity of copying. There is also an operator called *reflexion* [Mil94] (similar to the "trace" of Joyal et al. [JSV96]) which we need not detail here.

Finally, each action calculus has a binary *reaction relation* $\longrightarrow\!\!\triangleright$, relating action graphs of equal arity. This relation is preserved by all constructions, i.e. by composition, tensor product and reflexion.

For examples of action graphs, let $K : (0, 1)$, $M : (1, 1)$ and $L : (1, 0)$ be controls. Then the following are action graphs, with their arities:

$$\begin{aligned} K \otimes M &: (1, 2) \\ K{\cdot}\mathsf{c}{\cdot}(M \otimes L) &: (0, 1) \\ (K{\cdot}M) \otimes (M{\cdot}L) &: (1, 1) \ . \end{aligned}$$

Composition $\cdot$ binds tighter than tensor $\otimes$, so the last can be written $K{\cdot}M \otimes M{\cdot}L$.

A *context* C is an action graph containing a single hole with arity α, written $-_\alpha$. I omit the arity, writing $-$, if it is determined by the rest of the context or the surrounding discussion. Thus a context $C : \alpha \rightarrow \beta$ is an action graph of arity β with a hole of arity α. Here are two contexts along with their domain and codomain arities (the arity of the hole being fully determined in the second case):

$$\begin{aligned} -_{1,1}{\cdot}\mathsf{c}{\cdot}(M \otimes M) &: (1, 1) \rightarrow (1, 2) \\ K{\cdot}-{\cdot}L &: (1, 1) \rightarrow (0, 0) \ . \end{aligned}$$

Figure 5.1 shows an action graph and a context using a graphical notation. It uses nodes (rectangles with two blunt corners) to represent occurrences of controls, and arcs to represent composition. An action graph with arity (m, n) has m *source* ports on its left side and n *target* ports on its right side. A control node or hole of arity (m, n) has m target ports at its left side n source ports at its right side. At a source node, branching of arcs represents c and absence of arcs represents ω.

Two contexts are equal if the algebraic theory equates them, treating the hole as a control distinct from all others. The composition of two contexts $C : \alpha \rightarrow \beta$ and

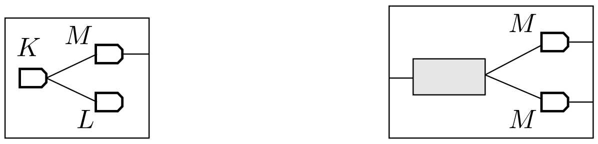


Figure 5.1: The action graph $K{\cdot}\mathsf{c}{\cdot}(M \otimes L)$ and the context $-_{1,1}{\cdot}\mathsf{c}{\cdot}(M \otimes M)$

$D : \beta \rightarrow \gamma$, written here DC (note the conventional order of composition), is formed by replacing the hole in D by C and joining the arcs according to the common ports. (This is context composition, not horizontal action graph composition described earlier.) Composition is clearly associative, and there is an identity context $\mathsf{id}_\alpha = -_\alpha$ for each arity. An action graph $a : \alpha$ can be considered as a context $a : (0,0) \rightarrow \alpha$ whose hole has minimum arity. We shall use lower case letters $a, \ldots$ for action graphs.

We have thus associated with an action calculus a reactive system **C-Ixt**, whose objects are arities, with distinguished null arity $(0,0)$, and whose arrows are contexts, including action graphs.

5.3 Examples and a problem

In this section I give examples of specific RPOs in **C-Ixt**, illustrating several phenomena. I end with an example showing cases in which RPOs fail to exist; this motivates the strategy of defining a category "upstairs" (the domain of a functor with codomain **C-Ixt**) for which enough RPOs do exist.

Remember that **C-Ixt** is really a family of reactive systems arising from action calculi; each is determined by a control signature and a set of reaction rules.

Example 5.1 (arithmetic) I first illustrate how an RPO can determine a labelled transition, using an action calculus for elementary arithmetic having controls $\mathsf{0} : (0,1)$, $\mathsf{S} : (1,1)$, and $+ : (2,1)$. The reactive system is shown in Figure 5.2; it is an example of the sharing graphs of Hasegawa [Has99], which add sharing to the interaction nets of Lafont [Laf90]. Nodes represent subexpressions, and the forking of arcs allows these to be shared. The reaction rules are in the top diagram; the garbage collection rules allow unattached expressions to be incrementally destroyed.

The middle diagram shows an action graph a occurring in a larger one b', which also contains an occurrence of the redex l_1 of the rule for S. The contexts C' and D' correspond to the two occurrences, which overlap. Now what is the "just large enough" context C which extends a to contain l_1? It is not quite C', because C' has destroyed the possibility of sharing S which is offered by l_1. In fact it is C as shown in the lower diagram; it may not seem "smaller" than C', but it is indeed a factor of C', as witnessed by the context

E. (C' cannot be a factor of C; no context F surrounding C' can cause its S-node to be shared.) So our derived labelled transition system will admit the transition $a \xrightarrow{C} Dr_1$. We would expect to add further controls, e.g. for subtraction, before obtaining an interesting behavioural congruence.

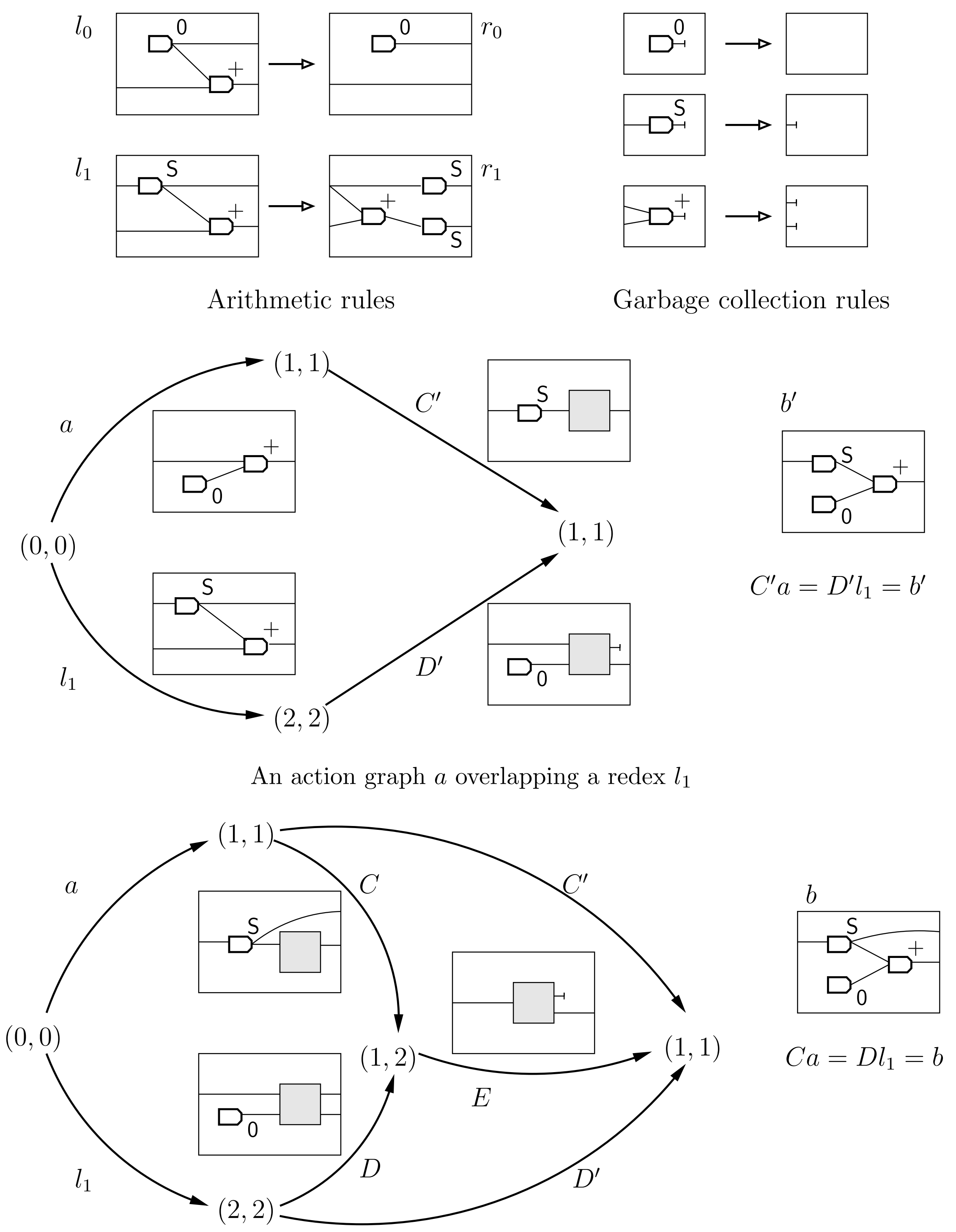
Example 5.2 (wiring) The preceding example used the forking and deletion of arcs to represent the sharing of components. This non-linearity is a pervasive feature in process calculi. CCS and the π-calculus depend heavily on it; witness the double occurrence of x in its reaction rule for CCS: $\bar{x}.a \mid x.b \longrightarrow a \mid b$ (and similarly for the π-calculus). The redex has no direct representation in the family of action calculi introduced in this section because we confine our attention to *shallow, closed* action graphs, i.e. ones without nesting of graphs (needed for prefixing) and free names. Without such limitations, the redex is exactly modelled by an action graph with a forked arc (representing the sharing of x) connected to two controls representing the output and input prefixes, containing inside a and b respectively. See [Mil96] for many examples.

Non-linearity can give rise to RPOs which are more complex than one might expect. Figure 5.3 shows two identical action graphs $a = b = K{\cdot}\mathsf{c}$, where $K : (0,1)$; using the identity contexts $C' = D' = -_{0,2}$ they are embedded in $K{\cdot}\mathsf{c}$. But the RPO C, D, E does not consist of identity contexts! A candidate might choose to identify t_0 in a with either t_2 or t_3 in b, and similarly for t_1. To be the "best" candidate, the C, D, E must handle all these pairings; to indicate this we have indexed its targets by pairs in the diagram. In fact we have

$$Ca = Db = K{\cdot}\mathsf{c}{\cdot}(\mathsf{c} \otimes \mathsf{c}) \ .$$

Example 5.3 (reflexion) A surprising phenomenon is how the presence of reflexion can affect the RPO. Let $K, N : (1,1)$, $L : (0,2)$ and $M : (2,0)$, and recall that i_1 is the identity of arity $(1,1)$ for action graph composition. Figure 5.4 shows $a = L{\cdot}(\mathsf{i}_1 \otimes K)$ and $b = (\mathsf{i}_1 \otimes K){\cdot}M$ embedded in $C'a = D'b = L{\cdot}(N \otimes K){\cdot}M$. The contexts C' and D' do not involve reflexion. In the RPO C, D, E shown we have $Ca = Db = (\mathsf{i}_1 \otimes L){\cdot}(\mathsf{p} \otimes K){\cdot}(\mathsf{i}_1 \otimes M)$; this extends a by only one control (M) in order to create an occurrence of b. The contexts C and D do not use reflexion, but E *does* use it. If reflexion is forbidden then the RPO C^+, D^+, E^+ is such that $C^+a = D^+b$ contains N; this would yield a more complex derived labelled transition relation.

These examples do not exhaust the phenomena which arise in finding RPOs in **C-Ixt**, but they indicate that the general construction will not be trivial. The reader may feel that, having coped informally with a number of phenomena, we are well on the way to finding RPOs in every case. However, they do not always exist in **C-Ixt**! Here is a counter-example.



Arithmetic rules

Garbage collection rules

An action graph a overlapping a redex l_1

An RPO for a and l_1 w.r.t. C' and D'

Figure 5.2: A reactive system for arithmetic (Example 5.1)

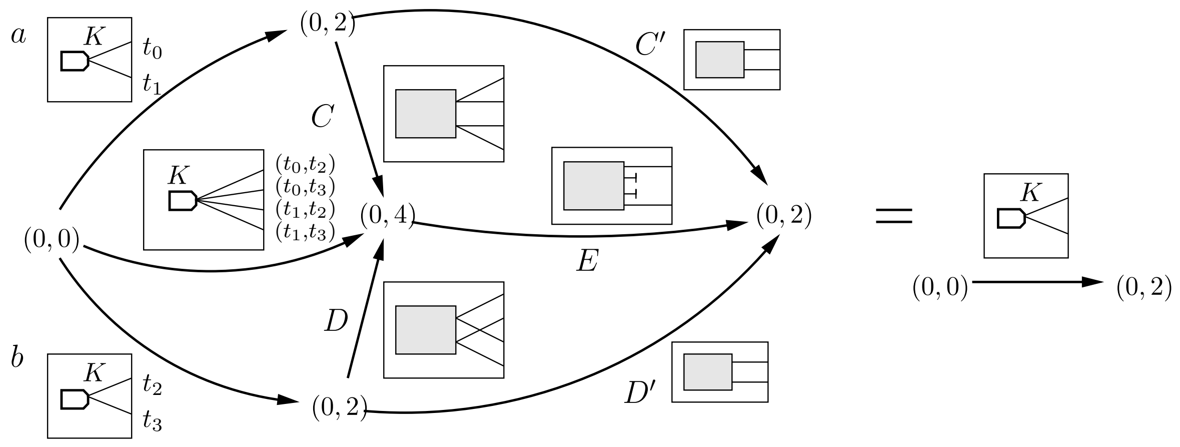


Figure 5.3: An RPO for copied wiring (Example 5.2)

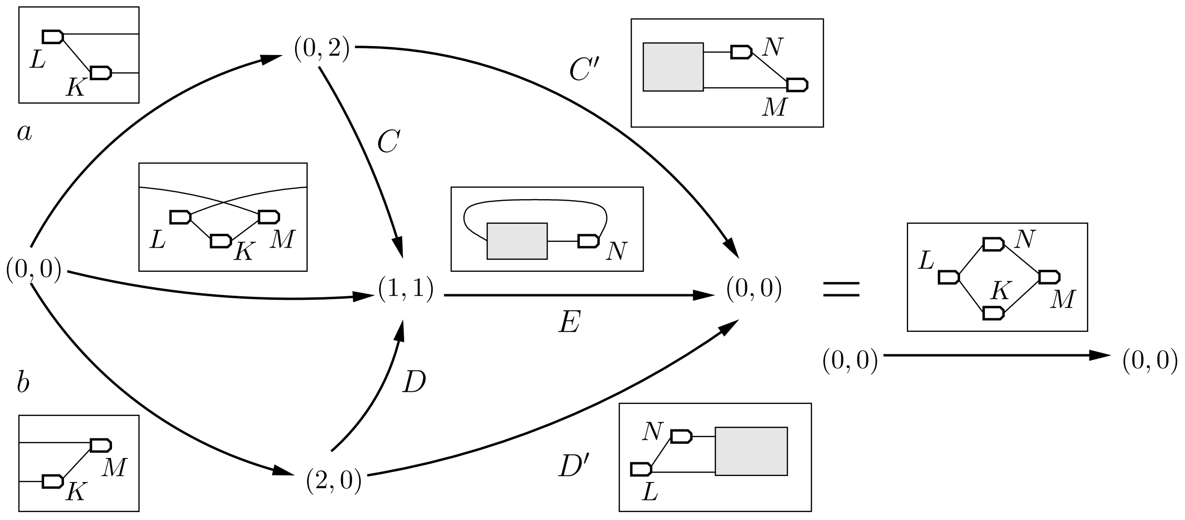
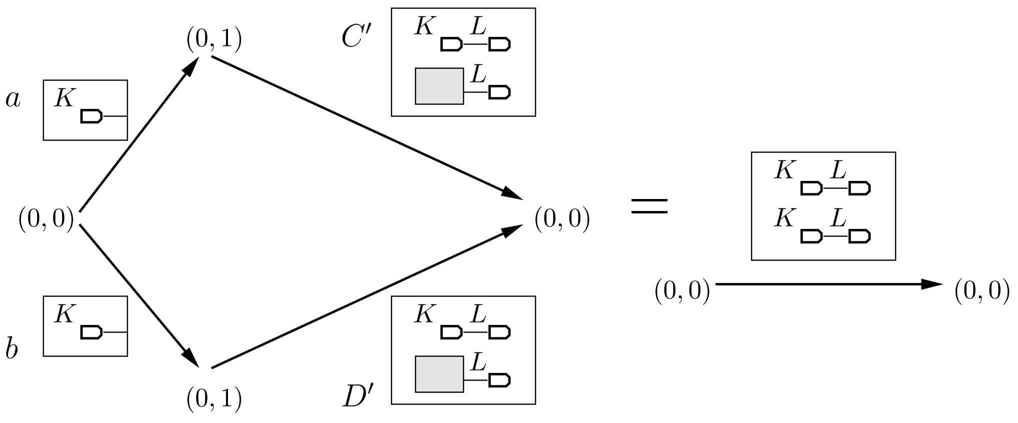


Figure 5.4: An RPO using reflexion (Example 5.3)



A context equation $C'a = D'b$

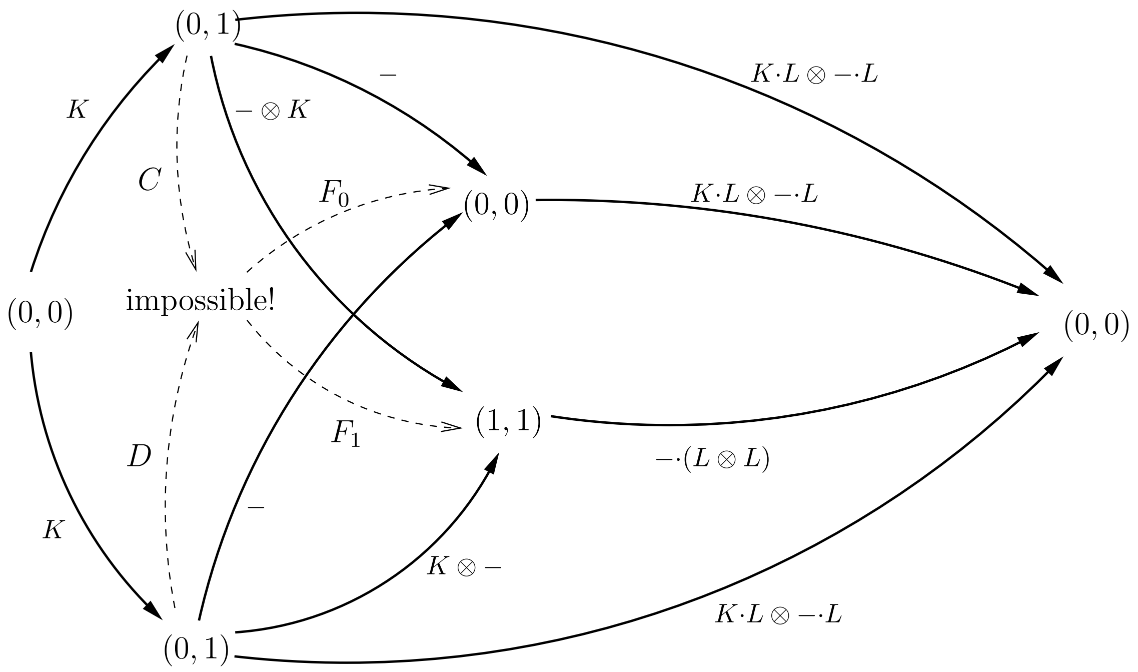


Figure 5.5: A missing RPO in **C-Ixt** (Example 5.4)

Example 5.4 (missing RPO) Let $K : (0,1)$ and $L : (1,0)$. Let $a = b = K$ and let $C' = D' = K{\cdot}L \otimes -_{0,1}{\cdot}L$ with arity $(0,1) \rightarrow (0,0)$. Then $C'a = D'b = K{\cdot}L \otimes K{\cdot}L$; this is shown in the upper diagram of Figure 5.5. The lower diagram shows two candidate RPOs, for which it is easy to verify commutation:

$$\begin{array}{rcccc} C_0, D_0, E_0 & = & -, & -, & K{\cdot}L \otimes -{\cdot}L \\ C_1, D_1, E_1 & = & - \otimes K, & K \otimes -, & -{\cdot}(L \otimes L)\ . \end{array}$$

But if there were an RPO, then contexts C, D, F_0, F_1 would exist as shown making the diagram commute, yielding a contradiction as follows: Neither C nor D can contain any control, since $F_0C = F_0D = -$. Hence from $CK = DK$ we deduce $C = D$, using the criterion for context equality stated above (since the control K appears in neither C nor D). Hence $- \otimes K = F_1C = F_1D = K \otimes -$, a contradiction.

This counter-example involves neither copying nor discard of arcs, so is applicable to *linear* action calculi too [LM00b] — those in which all source ports bear exactly one arc. Thus we cannot attribute the lack of RPOs to c and ω, even though they demand careful treatment as shown in Example 5.1 and Example 5.2.

The counter-example also illustrates *why* RPOs do not always exist in **C-Ixt**. The equations $C_0K = D_0K = K$ and $C_1K = D_1K = K \otimes K$ hold respectively for the two candidates; but if we “track” the two *occurrences* of $a = K$ and $b = K$ through these equations, we find that they correspond to the *same* occurrence of K in the first case, and to two *different* occurrences in the second case. This is a key to solving our problem; we seek a suitable refinement of **C-Ixt** that possesses the RPOs we need. We expect its contexts to track the occurrences of nodes. This is a similar idea to that of “colouring”, which has been suggested by Sewell to replace his dissection based definitions [Sew01].

The strategy is as follows. The next two sections are devoted to the construction of **A-Ixt**, a well-supported precategory of arities and raw contexts. Replaying the constructions of Chapter 4, we derive two categories from **A-Ixt** and a functor between them $\mathcal{F} : \hat{\mathbf{C}}\textbf{-Ixt} \rightarrow \mathbf{C}\textbf{-Ixt}$. The next chapter shows that sufficient RPOs exist in $\hat{\mathbf{C}}\textbf{-Ixt}$, as desired.

5.4 Closed shallow action graphs

In this section, after introducing some notation, I define a class of action graphs. These graphs are enriched in the next section to form *raw contexts* (graphs with a hole in them), the arrows of the precategory **A-Ixt**. No attempt is made to verify formally that the action graphs presented here correspond to action calculi terms quotiented by the action calculi axioms (which are not presented here). Such an undertaking represents future work and will be of more value when the tractable graph-theoretic features include free names and nesting.

Notation I write $[m]$ for the ordinal number $\{0, 1, \ldots, m-1\}$. The disjoint sum $\sum_{i \in I} X_i$ of a family of sets is taken to be the set $\bigcup_{i \in I}(\{i\} \times X_i)$. A particular case is when $I = [n]$; then the disjoint sum may be written, without parentheses, as $X_0 + X_1 + \cdots + X_{n-1}$. Examples in this chapter take the form $S = \sum_{v \in V} S_v + [m] + [n]$, a ternary disjoint sum, the first summand of which is itself a disjoint sum; S has elements of the form $(0, (v, s))$ for each $v \in V$ and $s \in S_v$, $(1, i)$ for each $i \in [m]$, and $(2, j)$ for each $j \in [n]$.

If the sets X and Y are disjoint, I often write their union as $X \uplus Y$. This notation is to be distinguished from a disjoint sum. In particular, $(X_1 \uplus X_2) \uplus X_3 = X_1 \uplus (X_2 \uplus X_3)$ and will often be written without parentheses; on the other hand, the disjoint sums $X_1 + X_2 + X_3$, $(X_1 + X_2) + X_3$ and $X_1 + (X_2 + X_3)$ are all distinct but in bijective correspondence. If $f : X \rightarrow Z$ and $g : Y \rightarrow Z$ are two functions with X disjoint from Y, then $f \uplus g : X \uplus Y \rightarrow Z$ is their union.

I use $f : X \rightarrowtail Y$, $f : X \twoheadrightarrow Y$ and $f : X \rightarrowtail\!\!\!\!\rightarrow Y$ for respectively injective, surjective and bijective functions, and $f : X \hookrightarrow Y$ for an injection which is an inclusion. Finally, $\circ$ denotes function composition, Id_X the identity function on the set X, and $\varnothing_X$ the empty function from $\varnothing$ to X.

Definition 5.5 (control signature) We fix a control signature $\mathcal{K}$, a set of *controls*, equipped with an *arity function* called $\mathit{arity} : \mathcal{K} \rightarrow \mathbb{N}^2$ and let $K, L, \ldots$ range over $\mathcal{K}$. For $\mathit{arity}(K) = (m, n)$ we write $K : (m, n)$; in this case, two functions extract the components of the pair: $\mathit{arin}(K) \mathrel{\hat{=}} m$ and $\mathit{arout}(K) \mathrel{\hat{=}} n$. ■

Definition 5.6 (action graphs) A *(closed, shallow) action graph* $G = (m, n, V, \mathit{contr}, \mathit{src})$ comprises an arity (m, n), a set V of *nodes*, called a *support*, a *control* map $\mathit{contr} : V \rightarrow \mathcal{K}$ assigning a control in $\mathcal{K}$ to each node in V, and a *source* map $\mathit{src} : T \rightarrow S$ assigning a *source (port)* in S to each *target (port)* in T, where

- the *source set* $S \mathrel{\hat{=}} \sum_{v \in V}[\mathit{arout}(\mathit{contr}(v))] + [m]$ comprises the *binding* sources for each $v \in V$ and the *input* sources indexed by $[m]$;

- the *target set* $T \mathrel{\hat{=}} \sum_{v \in V}[\mathit{arin}(\mathit{contr}(v))] + [n]$ comprises the *argument* targets for each $v \in V$ and the *output* targets indexed by $[n]$. ■

Nomenclature We may write a graph as $G = (V, \mathit{contr}, \mathit{src}) : (m, n)$, or just $G = (V, \mathit{contr}, \mathit{src})$ when the arity is understood. We denote the empty graph $(\varnothing, \varnothing_{\mathcal{K}}, \varnothing_{\varnothing}) : (0, 0)$ by $\mathbf{0}$. We shall abbreviate $\mathit{arin}(\mathit{contr}(v))$ to $\mathit{arin}(v)$ etc., when there is no ambiguity. We denote the injections induced by the disjoint sums S and T as

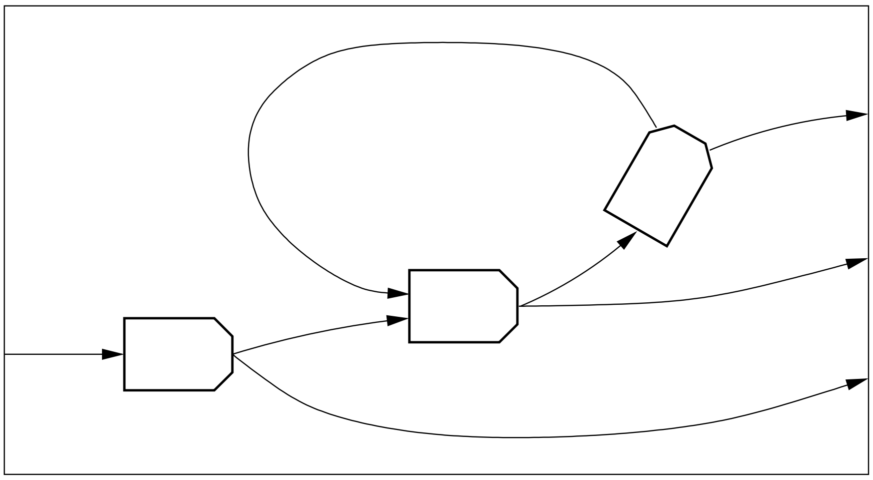


Figure 5.6: A closed shallow action graph

follows:

$bind(v) : [arout(v)] \rightarrowtail S$	for the binding sources of each $v \in V$;
$in : [m] \rightarrowtail S$	for the input sources;
$arg(v) : [arin(v)] \rightarrowtail T$	for the argument targets of each $v \in V$;
$out : [n] \rightarrowtail T$	for the output targets. ■

We shall write $bind(v, i)$ and $arg(v, j)$ for the ports $bind(v)(i)$ and $arg(v)(j)$. For any injection f into a set A we write A^f for the range of f; thus for example S^{in} is the set of all input sources and $T^{arg(v)}$ the set of argument targets of v. We shall also write for example S^{bind} for $\biguplus_{v \in V} S^{bind(v)}$. With this notation we can represent our partitions as

$$S = S^{in} \uplus S^{bind}$$
$$T = T^{out} \uplus T^{arg} \ .$$

An example of an action graph with arity $(1, 3)$ is shown in Figure 5.6, with node names and the control map omitted. The whole graph is in a rectangle, with input sources at the left and output targets at the right. Nodes are drawn as rectangles with two corners blunted to give orientation; we may want to tilt some of them, as here, or even turn them upside down. The three nodes have arities $(1, 1)$, $(2, 1)$ and $(1, 2)$. The arcs represent the source map, with arrows pointing from source to target. Ports could be drawn as blobs, but this is somewhat redundant; a target is always indicated by exactly one incoming arc, and we indicate a source with no outgoing arcs by a little aborted arc.

Cycles are allowed. The graphs studied here are for action calculi which are *closed*, meaning that free names such as $x, y, \ldots$ are not used as sources, and *shallow*, meaning that nodes do not contain action graphs nested inside. I study the closed, shallow action graphs in this dissertation as a preliminary to future work on a full graphical presentation of action calculi, since notions such as graph embedding and graph context are more clearly handled in this simpler setting.

Convention Action graphs are almost always denoted by G suitably subscripted. Rather than always explicitly list all their primary components V, contr, and src, or their derived components S, T, bind, in etc., we shall follow a firm convention that the names of these components are standard, decorated as necessary to connote the graph's name.

5.5 The well-supported precategory A-Ixt of arities and raw contexts

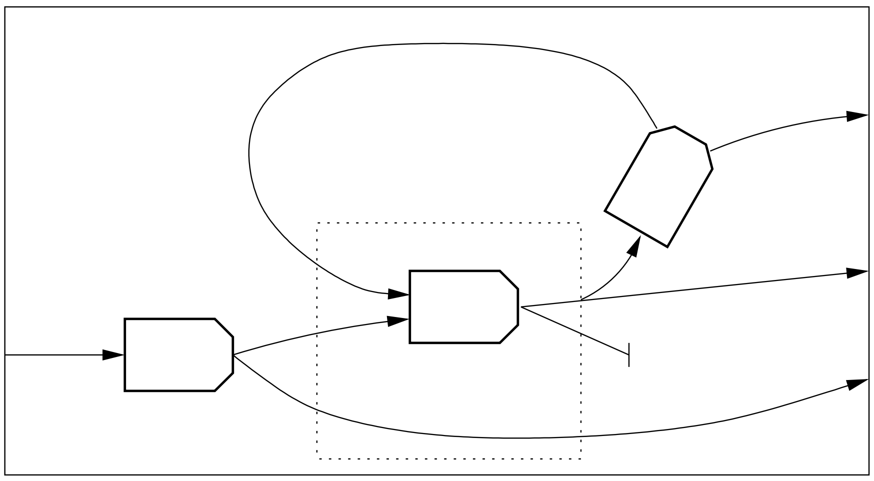
I proceed now to construct the well-supported precategory **A-Ixt** of raw contexts with a support function and support translation operation (Definition 4.2).

The intuition of "context" is well supplied by Figure 5.7; the graph G occurs *inside* the dotted rectangle in G', and we may think of the context in which G is placed as that part of G' lying *outside* the dotted rectangle. A context is therefore an action graph, but with a little more structure since it has an internal as well as an external interface. The internal interface is often called a *hole*. (I do not consider here contexts with more than one hole.)

The lower diagram shows the context C which results when G is excised from G'; in the notation of what follows, $G' = CG$ (C composed with G). Note the new targets and sources on respectively the left and right sides of C's internal interface; in particular, the middle internal source lacks any targets and therefore represents the discard of the corresponding output target of G — or of any other graph — when placed in the hole.

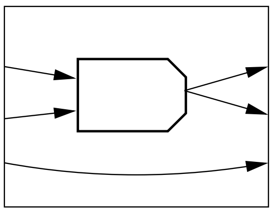
Definition 5.7 (raw context) A *(closed, shallow, loose) raw context* $A = (V, \mathit{contr}, \mathit{src})$ of arity (m', n') to (m, n), written $A : (m', n') \rightarrow (m, n)$, comprises a support V, which is a set of nodes, a *control* map $\mathit{contr} : V \rightarrow \mathcal{K}$ assigning a control in $\mathcal{K}$ to each node in V, and a *source* map $\mathit{src} : T \rightarrow S$ assigning a *source (port)* in S to each *target (port)* in T, where

- the *source set* $S \mathrel{\hat{=}} \sum_{v \in V} [\mathit{arout}(v)] + [m] + [n']$ comprises the *binding* sources for each $v \in V$, the *input* sources indexed by $[m]$, and the *upput* sources indexed by $[n']$;
- the *target set* $T \mathrel{\hat{=}} \sum_{v \in V} [\mathit{arin}(v)] + [n] + [m']$ comprises the *argument* targets for each $v \in V$, the *output* targets indexed by $[n]$, and the *downput* targets indexed by $[m']$.



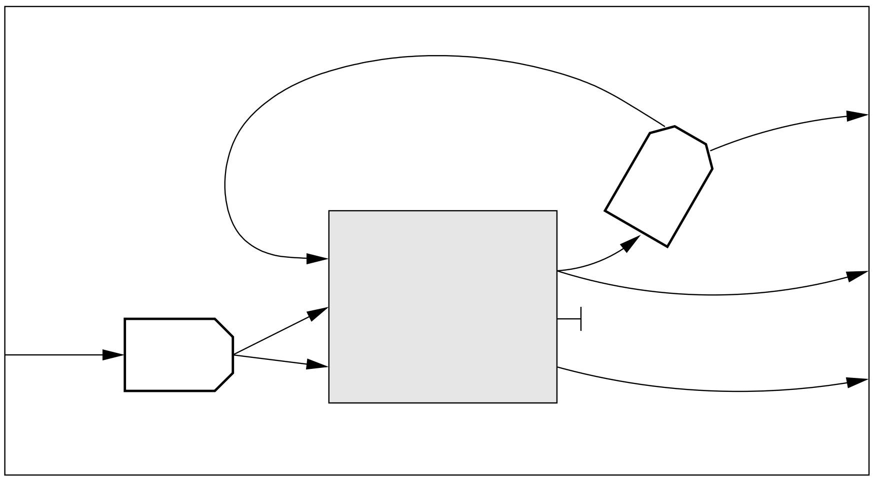
G'

G



The action graph G is a subgraph of G'

C



The corresponding context

Figure 5.7: The bottom context composed with the middle action graph yields the top action graph

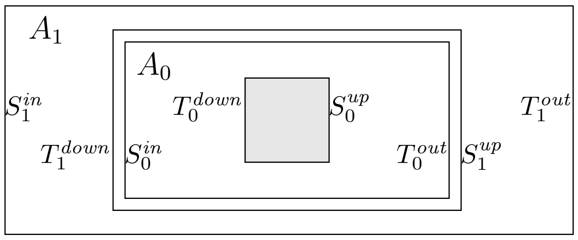


Figure 5.8: The composition of two raw contexts

Furthermore, we require that the looseness condition is satisfied (see nomenclature defined below):

$$src(T^{down}) \cap S^{up} = \varnothing \; . \qquad \textsc{Loose}$$

The looseness condition precludes a "tight" loop from the back of the hole to the front, such as is formed by reflexion in action calculi. It does not preclude such a loop proceeding via a control, which indeed occurs in Figure 5.7; thus the contexts permit only limited reflexion, which simplifies the definition of composition. (See Chapter 8 for further discussion.)

Finally, $|A|$ is the support of A, i.e. V in this case. ■

Note that an action graph G is just a raw context A as above in which $m' = n' = 0$.

Nomenclature As for action graphs, there are induced injections for raw contexts as follows:

$bind(v) : [arout(v)] \rightarrowtail S$	for the binding sources of each $v \in V$;
$in : [m] \rightarrowtail S$	for the input sources;
$up : [n'] \rightarrowtail S$	for the upput sources;
$arg(v) : [arin(v)] \rightarrowtail T$	for the argument targets of each $v \in V$;
$out : [n] \rightarrowtail T$	for the output targets;
$down : [m'] \rightarrowtail T$	for the downput targets.

With naming similar to that in action graphs for the partitions of S and T, we have

$$S = S^{bind} \uplus S^{in} \uplus S^{up}$$
$$T = T^{arg} \uplus T^{out} \uplus T^{down} \; .$$

Raw contexts cannot be composed when the supports of each operand in a composition intersect non-trivially. Composition *is* well-defined when the operands have disjoint supports— one of the required properties of a well-supported precategory (Definition 4.2).

Definition 5.8 (identity and composition for raw contexts) The *identity* raw context $\mathsf{id}_{(m,n)} = (\varnothing, \varnothing_{\mathcal{K}}, src) : (m,n) \rightharpoonup (m,n)$ has

$$S \mathrel{\hat{=}} \varnothing + [m] + [n] = S^{in} \uplus S^{up} \qquad\qquad src : \begin{cases} down(i) \mapsto in(i) & i \in [m] \\ out(j) \mapsto up(j) & j \in [n] \,. \end{cases}$$
$$T \mathrel{\hat{=}} \varnothing + [n] + [m] = T^{out} \uplus T^{down}$$

Let $A_i = (V_i, contr_i, src_i) : (m_i, n_i) \rightharpoonup (m_{i+1}, n_{i+1})$ be two raw contexts for $i = 0, 1$, with $V_0 \cap V_1 = \varnothing$. Their *composition*, illustrated in Figure 5.8 by nesting A_0 inside A_1, is

$$A_1 \oplus A_0 \mathrel{\hat{=}} A_2 = (V_2, contr_2, src_2) : (m_0, n_0) \rightharpoonup (m_2, n_2) \,,$$

where $V_2 \mathrel{\hat{=}} V_1 \uplus V_0$ and $contr_2 \mathrel{\hat{=}} contr_1 \uplus contr_0$ (thus determining $arin_2 = arin_1 \uplus arin_0$ and likewise $arout_2, bind_2, arg_2$), and

$$\begin{aligned} S_2 &\mathrel{\hat{=}} \textstyle\sum_{v \in V_2} [arout_2(v)] + [m_2] + [n_0] = (S_1^{bind} \uplus S_0^{bind}) \uplus S_1^{in} \uplus S_0^{up} \\ T_2 &\mathrel{\hat{=}} \textstyle\sum_{v \in V_2} [arin_2(v)] + [n_2] + [m_0] \;\; = (T_1^{arg} \uplus T_0^{arg}) \uplus T_1^{out} \uplus T_0^{down} \,. \end{aligned}$$

Note (see Figure 5.8) that T_1^{down} is in bijection with S_0^{in} and $[m_1]$, while S_1^{up} is in bijection with T_0^{out} and $[n_1]$. It remains to define the source function src_2; this is done in terms of two auxiliary functions $\sigma_i : S_i \rightharpoonup S_2$ $(i = 0, 1)$ which describe how sources of A_0 and A_1 "become" sources of A_2:

$$\sigma_0(s) \mathrel{\hat{=}} \begin{cases} s & \text{if } s \in S_0^{bind} \uplus S_0^{up} \\ src_1 \, down_1(i) & \text{if } s = in_0(i) \in S_0^{in} \end{cases}$$

$$\sigma_1(s) \mathrel{\hat{=}} \begin{cases} s & \text{if } s \in S_1^{bind} \uplus S_1^{in} \\ \sigma_0 src_0 \, out_0(j) & \text{if } s = up_1(j) \in S_1^{up} \end{cases}$$

$$src_2(t) \mathrel{\hat{=}} \begin{cases} \sigma_1 src_1(t) & \text{if } t \in T_1^{arg} \uplus T_1^{out} \\ \sigma_0 src_0(t) & \text{if } t \in T_0^{arg} \uplus T_0^{down} \end{cases} .$$

■

We have adopted the symbol "$\oplus$" for composition of raw contexts as a reminder that it is partial: the supports of the operands are required to be disjoint in order for a composition to be defined.

Proposition 5.9 (raw contexts form a precategory) If A_0 and A_1 are raw contexts with $|A_1| \cap |A_0| = \varnothing$ then $A_1 \oplus A_0$ is a raw context too. (In particular, $\oplus$ preserves the LOOSE condition.) Composition is associative and has identity id (in the way required of a precategory, see Definition 4.1).

Proof Follows immediately from Propositions 21 and 22 of [CLM00]. ■

By definition composition satisfies Supp-comp and Supp-id (Definition 4.2), i.e.

$$|A_1 \oplus A_0| = |A_1| \uplus |A_0| \qquad \text{Supp-comp}$$
$$|\mathsf{id}_m| = \varnothing\ . \qquad \text{Supp-id}$$

The only task remaining in order to show that **A-Ixt** is a well-supported precategory is the definition of a support translation operation and the verification of its properties.

Definition 5.10 (support translation for raw contexts) Given a raw context $A : (m_0, n_0) \rightarrow (m_1, n_1)$ and an injection ρ whose domain contains the support of A, i.e. $\mathsf{Dom}\,\rho \supseteq |A|$, we define the *support translation by ρ of A*, written $\rho{\bullet}A$, as follows: $\rho{\bullet}A \mathrel{\hat=} A'$, where:

$$\begin{aligned} V' &\mathrel{\hat=} \rho V \\ contr'(v) &\mathrel{\hat=} contr(\rho^{-1}(v)) \qquad \text{thus determining } arin, arout, bind, arg \\ S' &\mathrel{\hat=} \textstyle\sum_{v \in V'} [arout'(v)] + [m_1] + [n_0] \\ T' &\mathrel{\hat=} \textstyle\sum_{v \in V'} [arin'(v)] + [n_1] + [m_0] \end{aligned}$$

thus determining bijections $\rho^{\mathsf{S}} : S \rightarrowtail\!\!\!\!\twoheadrightarrow S'$ and $\rho^{\mathsf{T}} : T' \rightarrowtail\!\!\!\!\twoheadrightarrow T$ whence we define:

$$src' \mathrel{\hat=} \rho^{\mathsf{S}} \circ src \circ \rho^{\mathsf{T}}\ . \qquad ■$$

This definition satisfies the appropriate healthiness conditions:

Proposition 5.11 (support translation is well-defined) If A satisfies Loose then so does $\rho{\bullet}A$, thus $\rho{\bullet}(\cdot)$ is a well-defined operation on raw contexts.

Proof Immediate from $src' \mathrel{\hat=} \rho^{\mathsf{S}} \circ src \circ \rho^{\mathsf{T}}$, since ρ^{S} and ρ^{T} are the identity on upput sources and downput targets respectively. (Proposition 24 in [CLM00].) ■

Furthermore, support translation satisfies the remaining axioms in the definition of a well-supported precategory (Definition 4.2):

Proposition 5.12 (support translation properties) All of the following properties hold:

$$\rho{\bullet}\mathsf{id}_m = \mathsf{id}_m \qquad \text{Trans-id-r}$$
$$\rho{\bullet}(A_1 \oplus A_0) = \rho{\bullet}A_1 \oplus \rho{\bullet}A_0 \qquad \text{Trans-comp-r}$$
$$\mathsf{Id}_{|A|}{\bullet}A = A \qquad \text{Trans-id-l}$$
$$(\rho_1 \circ \rho_0){\bullet}A = \rho_1{\bullet}(\rho_0{\bullet}A) \qquad \text{Trans-comp-l}$$
$$\rho_0 \restriction |A| = \rho_1 \restriction |A| \ \text{ implies } \ \rho_0{\bullet}A = \rho_1{\bullet}A \qquad \text{Trans-res}$$
$$|\rho{\bullet}A| = \rho|A|\ . \qquad \text{Trans-supp}$$

Proof All of these except the last follow from Proposition 25 in [CLM00]. TRANS-SUPP follows immediately from Definition 5.10. ■

Thus:

Theorem 5.13 **A-Ixt** is a well-supported precategory. ■

5.6 Constructing a functorial reactive system

Having established that **A-Ixt** is a well-supported precategory, the rest of the results of Chapter 4 are immediately applicable. We let $\hat{\mathbf{C}}$**-Ixt** be the track of **A-Ixt** (Definition 4.3), a category of profiles and *insertion contexts*. The arrows are called "insertion contexts" to remind us that they "insert" the nodes of the domain profile into the codomain profile. Thus, if $(m_0, n_0, U_0) \xrightarrow{A} (m_1, n_1, U_1)$ is an arrow of $\hat{\mathbf{C}}$**-Ixt** then $U_0 \subseteq U_1$. A more complex category is considered in [CLM00] which allows the domain set to be injected (rather than just included) in the codomain set. This richness seems to be superfluous, so I ignore it here.

Now let **C-Ixt** be the support quotient (Definition 4.10) of **A-Ixt**. Finally let $\mathcal{F} : \hat{\mathbf{C}}\textbf{-Ixt} \rightarrow \mathbf{C}\textbf{-Ixt}$ be the support-quotienting functor (Definition 4.12). Then by Theorem 4.14, $\mathcal{F}$ lifts arrows by their domain, creates isos, creates compositions, and creates left inverses. (The last is used only in Appendix B.) By Theorem 4.15, $\mathcal{F}$ allows IPO sliding.

A-Ixt has a distinguished object $(0, 0)$: as stated immediately after Definition 5.7, a raw context with domain $(0, 0)$ is identical to an action graph. Since **C-Ixt** has the same objects as **A-Ixt** does, the distinguished object 0 of **C-Ixt** as a reactive system is just $(0, 0)$. Likewise, let $\boldsymbol{\varepsilon} \hat{=} (0, 0, \varnothing)$, the distinguished object for $\hat{\mathbf{C}}$**-Ixt**. Since $\mathcal{F}(\boldsymbol{\varepsilon}) = 0$, we have that $\mathcal{F}$ lifts agents, and thus:

Theorem 5.14 ($\mathcal{F}$ is functorial reactive system) The support-quotienting functor $\mathcal{F} : \hat{\mathbf{C}}\textbf{-Ixt} \rightarrow \mathbf{C}\textbf{-Ixt}$ is a functorial reactive system for any choice of $\mathbf{D}$ and Reacts. ■

Thus all the congruence results of Chapter 3 are applicable to $\mathcal{F}$ (except Theorem 3.33 since we are not dealing with multi-hole contexts) with one proviso: we wish to determine choices of $\mathbf{D}$ and Reacts such that $\mathcal{F}$ has all redex-RPOs. That is the subject of the next chapter which shows that an arbitrary choice of $\mathbf{D}$ and a wide variety of choices of Reacts (in particular of redexes) leads to sufficient RPOs.

Chapter 6

RPOs for action graph contexts

6.1 Introduction

$$\begin{tikzcd} \varepsilon \arrow[r, "G_0"] \arrow[d, "G_1"'] & p_0 \arrow[d, "A_0"] \\ p_1 \arrow[r, "A_1"'] & p \end{tikzcd} \quad 6.1$$

The goal of this chapter is to show that $\mathcal{F} : \hat{\mathbf{C}}\textbf{-Ixt} \to \mathbf{C}\textbf{-Ixt}$ has all redex-RPOs for a wide variety of reaction rules Reacts and reactive contexts $\mathbf{D}$. With regard to $\mathbf{D}$, the strongest result obtainable is with $\mathbf{D} = \mathbf{C}\textbf{-Ixt}$, i.e. with all contexts in $\mathbf{C}\textbf{-Ixt}$ reactive, so let us fix on this throughout this chapter. Recall from Definition 3.4, that $\mathcal{F}$ *has all redex-RPOs* if any square in $\hat{\mathbf{C}}\textbf{-Ixt}$, such as in Figure 6.1, has an RPO provided that there exists $r \in \mathbf{C}\textbf{-Ixt}$ such that $(\mathcal{F}(\varepsilon \xrightarrow{G_1} p_1), r) \in \mathsf{Reacts}$.

So the key question is: for which choice of redexes l (first components of the Reacts relation) do there exist RPOs in $\hat{\mathbf{C}}\textbf{-Ixt}$ for squares whose upper-left profile is $\varepsilon = (0, 0, \varnothing)$ and whose left leg is a preimage of l? Answering this question is the task of the present chapter.

As indicated in the previous chapter, constructing RPOs by the direct manipulation of contexts in $\hat{\mathbf{C}}\textbf{-Ixt}$ is difficult and not attempted here. An alternate strategy is to carry out a related construction (relative coproducts, to be explained later) in a related category $\mathbf{G}\textbf{-Inc}$ of action graphs and inclusion embeddings. That is the plan followed here.

This chapter is organised as follows. First I define $\mathbf{G}\textbf{-Inc}$ and relate it to $\hat{\mathbf{C}}\textbf{-Ixt}$. Second, I define *relative coproducts* and relate them to RPOs. Third, I show precise (necessary and sufficient) conditions for the existence of relative coproducts in $\mathbf{G}\textbf{-Inc}$. Fourth, I give conditions on the redexes in Reacts which guarantee that $\mathcal{F} : \hat{\mathbf{C}}\textbf{-Ixt} \to \mathbf{C}\textbf{-Ixt}$ has all redex-RPOs (Theorem 6.29) — the main applied result in this dissertation.

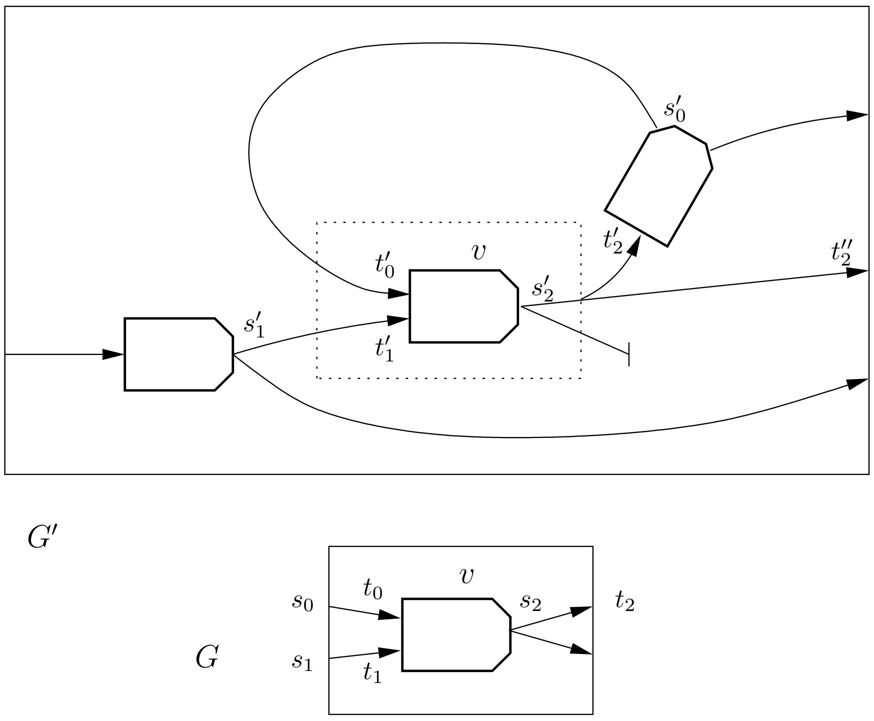


Figure 6.2: An inclusion embedding

6.2 The category G-Inc of action graphs and inclusion embeddings

I now define the category **G-Inc** in which the objects are action graphs and the arrows are inclusion embeddings. Our main task is to define what we mean by inclusion embeddings and to capture the meaning in a tractable list of axioms.

Intuition Two graphs are shown in Figure 6.2. G' is the graph of Figure 5.6, and G is a smaller graph which intuitively "occurs" in G'. The way it occurs is represented informally by the dotted rectangle. The input sources s_0 and s_1 of G are represented respectively by s'_0 and s'_1 of G', and the binding source s_2 by s'_2. The argument targets t_0 and t_1 are represented respectively by t'_0 and t'_1, and the output target t_2 by both t'_2 and t''_2. The other output target of G is not represented in G'; the aborted arc (not formally part of G') indicates that the lower arc from s_2 in G is discarded by the inclusion embedding into G'. Note that the action graph G here is simpler (by one less arc) than the one considered in Figure 5.7.

There is no reason why the node v in G has to be sent to v in G' and not to some v', however for simplicity, we confine our attention to *inclusion embeddings* — and usually omit the adjective "inclusion". The more general situation (**G-Emb**) for which embeddings inject (rather than include) the nodes of one graph in another is considered in detail in [CLM00]. I make no use of **G-Emb** here.

Of course several parts of G' do not represent G. Also our definition of inclusion embedding must allow that a target of G may have any number (including 0) of representatives in G'; but we shall insist that every source in G is represented exactly once in G'. Two sources may have the same representative, though this is not the case in our example.

We are therefore led to the following definition:

Definition 6.1 (inclusion embedding) Given graphs $G_i = (V_i, contr_i, src_i)$, $i = 0, 1$, a *(loose) inclusion embedding* $\eta : G_0 \rightarrow G_1$ *of* G_0 *in* G_1 consists of a pair of functions $(\eta^{\mathsf{S}}, \eta^{\mathsf{T}})$ where

$$\begin{array}{ll} \eta^{\mathsf{S}} : S_0 \rightarrow S_1 & \text{is a map of sources;} \\ \eta^{\mathsf{T}} : T_1 \rightharpoonup T_0 & \text{is a partial map of targets} \end{array}$$

satisfying the following axioms:

$$contr_1 \restriction V_0 = contr_0 \qquad \textsc{Inc-Contr}$$

$$\eta^{\mathsf{S}} \circ src_0 \circ \eta^{\mathsf{T}} \subseteq src_1 \qquad \textsc{Inc-Src}$$

$$\eta^{\mathsf{S}\,-1}\{bind_1(v,i)\} = \{bind_0(v,i)\} \qquad \textsc{Inc-Bind}$$

$$\eta^{\mathsf{T}\,-1}\{arg_0(v,j)\} = \{arg_1(v,j)\} \qquad \textsc{Inc-Arg}$$

$$src_1^{-1}\eta^{\mathsf{S}}(S_0^{bind}) \subseteq \mathsf{Dom}\,(\eta^{\mathsf{T}})\ . \qquad \textsc{Inc-Targs}$$

■

These axioms deserve a little explanation. Inc-Contr says that η respects the control function; Inc-Src says that it also respects the source function where η^{T} is defined. Inc-Bind says not only that η respects the binding source functions, but also that η^{S} never identifies (in G_1) a binding source of G_0 with any other source. This enforces the looseness constraint (see below). Inc-Arg is similar. Inc-Targs says that if a source s in G_1 represents a bound source of G_0, then all its targets represent targets of G_0.

The looseness constraint prevents an inclusion embedding for which G_1 creates a tight loop (such as is formed by reflexion in action calculi) from the back of the G_0 to the front. It does not preclude such a loop proceeding via a control, which indeed occurs in Figure 6.2; thus our inclusion embeddings will permit some but not all of the power of reflexion. To match full reflexion is more difficult and is not attempted here. This limitation on inclusion embeddings corresponds to the limitation on contexts (Loose) described in Definition 5.7. I shall usually omit the adjective "loose" as it applies to inclusion embeddings throughout.

Definition 6.2 (the category G-Inc of action graphs and inclusion embeddings) The objects of the category **G-Inc** are action graphs (Definition 5.6) and the arrows are (loose) inclusion embeddings (Definition 6.1). If $\eta : G \rightarrowtail G'$ and $\theta : G' \rightarrowtail G''$ are two inclusion embeddings, then their *composition* $\theta\eta : G \rightarrowtail G''$ is defined by

$$\theta\eta \mathrel{\hat{=}} (\theta^{\mathsf{S}} \circ \eta^{\mathsf{S}}, \eta^{\mathsf{T}} \circ \theta^{\mathsf{T}}) \ .$$

The *identity* inclusion embedding $\mathsf{id}_G : G \rightarrowtail G$ is defined by

$$\mathsf{id}_G \mathrel{\hat{=}} (\mathsf{Id}_S, \mathsf{Id}_T) \ . \qquad ■$$

Proposition 6.3 **G-Inc** is a category with an initial object **0** (the empty graph).

Proof Definition 16 and Proposition 7 in [CLM00]. ■

For any graph G we shall write $\mathbf{0}_G$ for the unique inclusion embedding $\mathbf{0} \rightarrowtail G$ whose components are both empty (partial) functions.

Immediately following Definition 5.7, we noticed that action graphs $(V, \mathit{contr}, \mathit{src}) : (m, n)$ correspond to raw contexts with domain $(0, 0)$ and support V, and thus to contexts in $\hat{\mathbf{C}}$**-Ixt** of the form $\varepsilon \rightarrow (m, n, V)$. The relationship between **G-Inc** and $\hat{\mathbf{C}}$**-Ixt** goes further, as witnessed by the following functor:

$$\mathcal{A} : \textbf{G-Inc} \rightarrowtail \varepsilon/\hat{\mathbf{C}}\textbf{-Ixt}$$

where $\varepsilon/\hat{\mathbf{C}}$**-Ixt** is a *coslice* category of $\hat{\mathbf{C}}$**-Ixt**; this is a simple general notion that is defined formally in Appendix A. In brief, the objects of $\varepsilon/\hat{\mathbf{C}}$**-Ixt** are contexts in $\hat{\mathbf{C}}$**-Ixt** with domain ε and the arrows $G \rightarrowtail G'$ are contexts $p \xrightarrow{A} p'$ in $\hat{\mathbf{C}}$**-Ixt** for which $\varepsilon \xrightarrow{G} p \xrightarrow{A} p' = \varepsilon \xrightarrow{G'} p'$, i.e. $A \oplus G = G'$. The functor is formally defined as follows:

Definition 6.4 ($\mathcal{A}$) The functor $\mathcal{A} : \textbf{G-Inc} \rightarrowtail \varepsilon/\hat{\mathbf{C}}\textbf{-Ixt}$ is defined on objects by $\mathcal{A}(G) \mathrel{\hat{=}} \varepsilon \xrightarrow{G} (m, n, V)$, where $G = (V, \mathit{contr}, \mathit{src}) : (m, n)$. Now let $\eta : G_0 \rightarrowtail G_1$ be an arrow (i.e. an embedding). Suppose

$$G_i = (V_i, \mathit{contr}_i, \mathit{src}_i) : (m_i, n_i) \qquad \text{for } i = 0, 1$$

with $V_0 \subseteq V_1$. Then

$$\begin{aligned} \mathcal{A}(\eta) &\mathrel{\hat{=}} (m_0, n_0, V_0) \xrightarrow{A} (m_1, n_1, V_1) \\ A &\mathrel{\hat{=}} (V_A, \mathit{contr}_A, \mathit{src}_A) \ , \end{aligned}$$

where $V_A \mathrel{\hat{=}} V_1 \setminus V_0$ and $\mathit{contr}_A \mathrel{\hat{=}} \mathit{contr}_1 \restriction V_A$ (thus determining $\mathit{arity}_A = \mathit{arity}_1 \restriction V_A$, etc.), and

$$\begin{aligned} S_A &\mathrel{\hat{=}} \textstyle\sum_{v \in V_A} [\mathit{arout}_A(v)] + [m_1] + [n_0] &= S_A^{\mathit{bind}} \uplus S_1^{\mathit{in}} \uplus S_1^{\mathit{up}} \\ T_A &\mathrel{\hat{=}} \textstyle\sum_{v \in V_A} [\mathit{arin}_A(v)] + [n_1] + [m_0] &= T_A^{\mathit{arg}} \uplus T_1^{\mathit{out}} \uplus T_1^{\mathit{down}} \ . \end{aligned}$$

Note that S_1^{up} and T_1^{down} are in bijection with T_0^{out} and S_0^{in} respectively. Finally we define src_A:

$$src_A(t) \mathrel{\hat{=}} \begin{cases} \begin{cases} up_1(j) & \text{if } \eta^{\mathsf{T}}(t) \text{ is defined, i.e. } \eta^{\mathsf{T}}(t) = out_0(j) \text{ for some } j \in [n_0] \\ src_1(t) & \text{if } \eta^{\mathsf{T}}(t) \text{ is undefined} \end{cases} \\ \qquad\qquad\qquad\qquad \text{if } t \in T_A^{arg} \uplus T_1^{out} \\ \eta^{\mathsf{S}} in_0(i) \qquad\qquad\quad \text{if } t = down_1(i) \in T_1^{down} \text{ for some } i \in [m_0] \end{cases}$$

$\mathcal{A}$ is a functor by Corollary 32 in [CLM00]. ■

This functor is full and faithful, as witnessed by its inverse on every homset

$$\mathcal{D}_{G_0,G_1} : \varepsilon/\hat{\mathbf{C}}\text{-}\mathbf{Ixt}(\mathcal{A}(G_0), \mathcal{A}(G_1)) \longrightarrow \mathbf{G}\text{-}\mathbf{Inc}(G_0, G_1) \ ,$$

which is formally defined as follows:

Definition 6.5 ($\mathcal{D}$) Let G_0 and G_1 be action graphs. We define a function $\mathcal{D}_{G_0,G_1} : \varepsilon/\hat{\mathbf{C}}\text{-}\mathbf{Ixt}(\mathcal{A}(G_0), \mathcal{A}(G_1)) \longrightarrow \mathbf{G}\text{-}\mathbf{Inc}(G_0, G_1)$ from contexts to embeddings.

Let $(p_0 \xrightarrow{A} p_1) : \mathcal{A}(G_0) \longrightarrow \mathcal{A}(G_1)$ be a context, where $G_i = (V_i, contr_i, src_i) : (m_i, n_i)$ for $i = 0, 1$. Recall that by definition $G_1 = A \oplus G_0$. Thus if $A = (V, contr, src)$ then the components of $G_1 = A \oplus G_0 = (V_1, contr_1, src_1)$ are as follows (a special case of raw context composition)

$$\begin{aligned} V_1 &= V \uplus V_0 \\ contr_1 &= contr \uplus contr_0 \\ S_1 &= (S^{bind} \uplus S_0^{bind}) \uplus S^{in} \\ T_1 &= (T^{arg} \uplus T_0^{arg}) \uplus T^{out} \\ src_1(t) &= \begin{cases} \sigma\ src(t) & \text{if } t \in T^{arg} \uplus T^{out} \\ \sigma_0\ src_0(t) & \text{if } t \in T_0^{arg} \end{cases} \end{aligned}$$

where

$$\sigma_0(s) = \begin{cases} src\ down(i) & \text{if } s = in_0(i) \in S_0^{in} \\ s & \text{if } s \in S_0^{bind} \end{cases}$$

$$\sigma(s) = \begin{cases} \sigma_0 src_0\, out(j) & \text{if } s = up(j) \in S^{up} \\ s & \text{if } s \in S^{bind} \uplus S^{in} \end{cases}$$

The components of the inclusion embedding $\mathcal{D}_{G_0,G_1}(p_0 \xrightarrow{A} p_1) \mathrel{\hat{=}} \theta : G_0 \longrightarrow G_1$ are then

$$\begin{tikzcd}
\cdot \arrow[r, "f_0"] \arrow[d, "f_1"'] & \cdot \arrow[d, "g_0"] \\
\cdot \arrow[r, "g_1"'] & m
\end{tikzcd}$$

6.3(1)

$$\begin{tikzcd}
\cdot \arrow[rr, "f_0"] \arrow[dd, "f_1"'] & & \cdot \arrow[dd, "g_0"] \arrow[dl, "h_0"'] \\
& \cdot \arrow[dr, "h"] & \\
\cdot \arrow[rr, "g_1"'] \arrow[ur, "h_1"] & & m
\end{tikzcd}$$

6.3(2)

$$\begin{tikzcd}
\cdot \arrow[rrr, "f_0"] \arrow[dd, "f_1"'] & & & \cdot \arrow[dd, "g_0"] \arrow[dll, "h_0"'] \arrow[dl, "h'_0"] \\
& \cdot \arrow[r, dotted, "k"] \arrow[drr, "h"'] & \cdot \arrow[dr, "h'"] & \\
\cdot \arrow[rrr, "g_1"'] \arrow[ur, "h_1"] \arrow[urr, bend right, "h'_1"'] & & & m
\end{tikzcd}$$

6.3(3)

Figure 6.3: Construction of an RPO

defined by

$$\begin{aligned}
&\theta^{\mathsf{V}} : V_0 \hookrightarrow V_1 \mathrel{\hat{=}} v \mapsto v \qquad (v \in V_0)\\
&\theta^{\mathsf{S}} : S_0 \to S_1 \mathrel{\hat{=}} \sigma_0\\
&\theta^{\mathsf{T}} : T_1 \rightharpoonup T_0 \mathrel{\hat{=}} t \mapsto \begin{cases} t & \text{if } t \in T_0^{arg} \\ \begin{cases} \text{undefined} & \text{if } src(t) \in S^{bind} \uplus S^{in} \\ out_0(j) & \text{if } src(t) = up(j) \in S^{up} \end{cases} & \text{if } t \in T^{arg} \uplus T^{out} \end{cases}
\end{aligned}$$

■

This function and is bijective on objects, so:

Proposition 6.6 $\mathcal{A}$: **G-Inc** $\to \varepsilon/\hat{\mathbf{C}}$**-Ixt** is an isomorphism of categories. (See Proposition 42 of [CLM00].) ■

Because of the notational overload of action graphs and contexts out of ε, we think of $\mathcal{A}$ as the identity on objects, i.e. $\mathcal{A}(G) = G$.

6.3 Relative coproducts

A consequence of $\mathcal{A}$: **G-Inc** $\to \varepsilon/\hat{\mathbf{C}}$**-Ixt** being an isomorphism of categories is that *relative coproducts* (defined below) in **G-Inc** correspond naturally to RPOs in $\hat{\mathbf{C}}$**-Ixt**. This is a crucial fact in deriving labelled transition systems.

I first recall RPOs, then define relative coproducts and prove a result relating them. For this pure category theory arrows are denoted by $f, g, h, \ldots$. When we come to use the results I shall revert to the applied notation.

Definition (RPO recalled; see Definition 2.4) In any category **C**, consider a commuting square (Figure 6.3(1)) consisting of $g_0 f_0 = g_1 f_1$. An *RPO* is a triple h_0, h_1, h satisfying two properties:

commutation: $h_0 f_0 = h_1 f_1$ and $h h_i = g_i$ for $i = 0, 1$ (Figure 6.3(2));

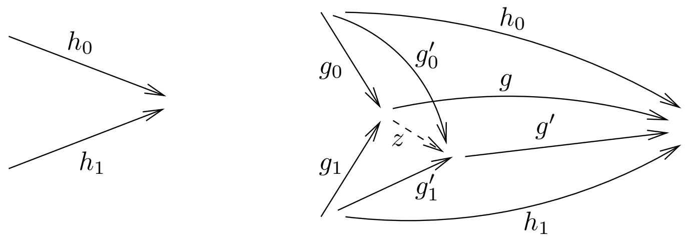


Figure 6.4: A relative coproduct g_0, g_1, g

universality: for any h'_0, h'_1, h' satisfying $h'_0 f_0 = h'_1 f_1$ and $h' h'_i = g_i$ for $i = 0, 1$, there exists a unique k such that $h'k = h$ and $kh_i = h'_i$ (Figure 6.3(3)). ■

Definition 6.7 (relative coproduct) In any category $\mathbf{C}$, let h_0, h_1 be a pair of arrows with a common codomain (see Figure 6.4). A triple g_0, g_1, g for which $gg_i = h_i$ $(i = 0, 1)$ is a *relative coproduct of* h_0, h_1 if, for any other triple g'_0, g'_1, g' such that $g' g'_i = h'_i$, there is an unique mediating arrow z such that $zg_i = g'_i$ and $g'z = g$. ■

Remark RPOs and relative coproducts are pushouts and coproducts in slice categories (the obvious dual notion to that of coslice, see Appendix A). I make no specific use of this observation.

Nomenclature If h_0, h_1 is a pair of arrows as in Figure 6.4, a triple g_0, g_1, g for which $gg_i = h_i$ is a *candidate (relative coproduct)* for h_0, h_1.

If X is an object of $\mathbf{C}$, relative coproducts in the coslice category $X/\mathbf{C}$ correspond to RPOs in $\mathbf{C}$ for pairs of arrows with domain X:

Proposition 6.8 Let $\mathbf{C}$ be a category. Let X, X_0, X_1, X_2 be four objects of $\mathbf{C}$, and $f_i : X \rightarrow X_i$, $h_i : X_i \rightarrow X_2$ $(i = 0, 1)$ four arrows of $\mathbf{C}$. If we regard f_0 and f_1 as objects of $X/\mathbf{C}$ and $h_i : f_i \rightarrow h_i f_i$ as arrows of $X/\mathbf{C}$, respectively, the triple g_0, g_1, g is a relative coproduct of h_0, h_1 iff it is an RPO in $\mathbf{C}$ for f_0, f_1 w.r.t. h_0, h_1.

Proof The proof is a trivial consequence of the definitions of relative coproduct and RPO applied to the categories $X/\mathbf{C}$ and $\mathbf{C}$ respectively. ■

This proposition relates RPOs in $\hat{\mathbf{C}}$**-Ixt** to relative coproducts in $(m, n, U)/\hat{\mathbf{C}}$**-Ixt**. In particular we are interested in RPOs for pairs of contexts out of the empty profile ε, so I focus on relative coproducts in $\varepsilon/\hat{\mathbf{C}}$**-Ixt**. As shown above, $\mathcal{A} :$ **G-Inc** $\rightarrow \varepsilon/\hat{\mathbf{C}}$**-Ixt** is an isomorphism of categories, so preserves relative coproducts.

These facts underpin Corollary 6.28 which shows how to derive RPOs for $\hat{\mathbf{C}}$**-Ixt** from relative coproducts in **G-Inc**. Let us look at an example of this derivation in the simple

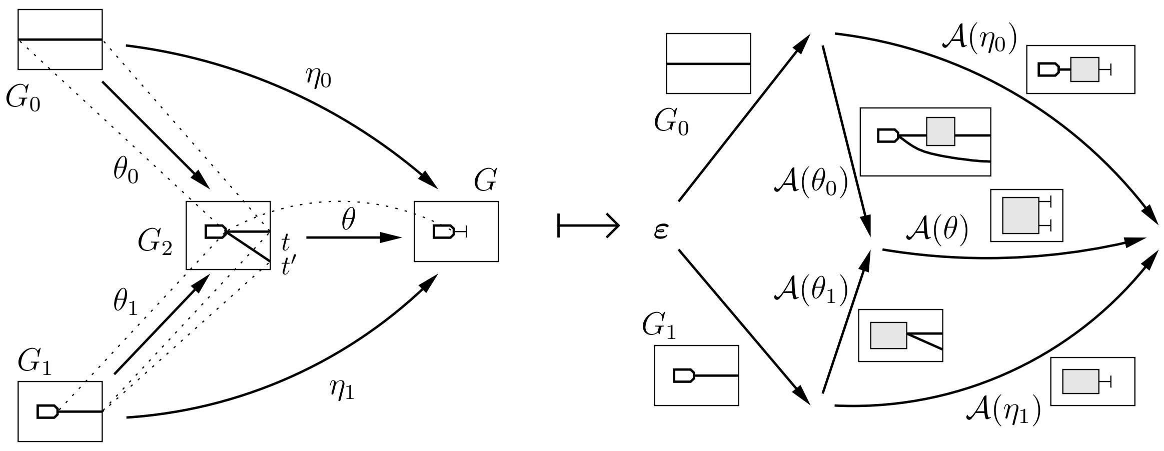


Figure 6.5: A relative coproduct in **G-Inc** becomes an RPO in $\hat{\mathbf{C}}$**-Ixt**

situation depicted in Figure 6.5. The left diagram shows a relative coproduct $\theta_0, \theta_1, \theta$ for η_0, η_1 in **G-Inc**, which becomes an RPO in $\hat{\mathbf{C}}$**-Ixt** via $\mathcal{A}$. (Recall that an arrow $\mathcal{A}(\eta)$ of $\varepsilon/\hat{\mathbf{C}}$**-Ixt** is also an arrow of $\hat{\mathbf{C}}$**-Ixt**.) Note that $\mathcal{A}(\theta_0)\,(\varepsilon \xrightarrow{G_0} p_0) = \mathcal{A}(\theta_1)\,(\varepsilon \xrightarrow{G_1} p_1) = (\varepsilon \xrightarrow{G_2} p_2)$ and $\mathcal{A}(\theta)\,\mathcal{A}(\theta_i) = \mathcal{A}(\theta\,\theta_i) = \mathcal{A}(\eta)$, thus the proposed RPO commutes correctly.

The source and target maps of θ_0, θ_1 and θ are shown as dotted lines. For example the embedding of G_0's single arc into G_2 is represented by the hole in the context $\mathcal{A}(\theta_0)$. Since G has no targets, the target map θ^{T} is empty; this corresponds to the discarding of the targets t and t', represented by the aborted arcs in $\mathcal{A}(\theta)$. Just as in the action graph b of Figure 5.2 in Example 5.1 (arithmetic), the extra arc of G_2 connecting the control node to t' may seem at first superfluous. But it is essential, since a competing candidate can have a target like t', sourced by the control node, that is mapped back to G_1 but not to G_0. This distinguishes it from t, which is mapped to both. By Inc-Targs there are no other possibilities (such as a target mapped to G_0 but not to G_1). Thus G_2 contains precisely the targets it needs to be "as good as" any candidate. The formal construction of G_2 is given later in this chapter.

6.4 Existence of relative coproducts

Our task now is to establish relative coproducts in **G-Inc**. They are not always present! Figure 6.6 shows a case with no controls — only "wiring" — in which there is no relative coproduct. G_0 is a graph with a single input source s_0 and output target t_0, with $s_0 = \mathit{src}_0(t_0)$; G_1 is similar. G has a single source but no target. H and H' are candidate relative coproducts— one with a single arc and one with two sources and no arcs. The

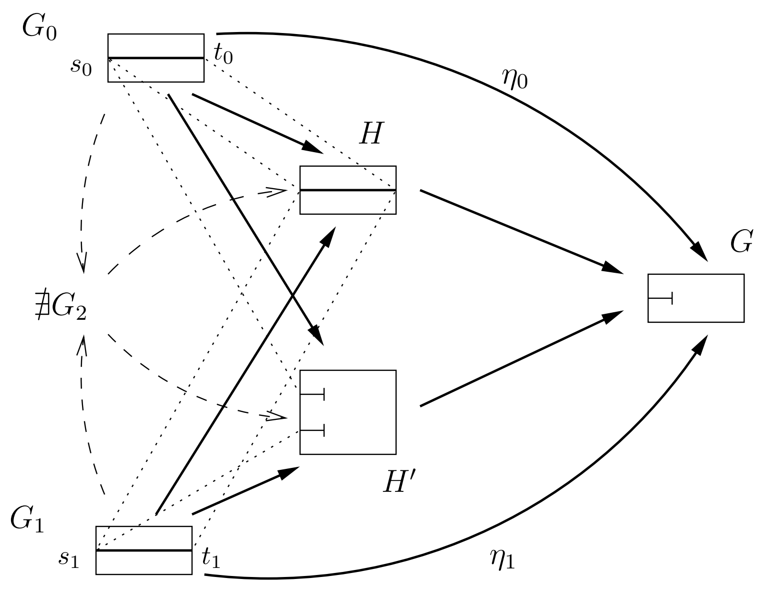


Figure 6.6: A case where no relative coproduct exists (and η_0, η_1 do not satisfy CHASTE)

source and targets maps of the inclusion embeddings into H and H' are shown as dotted lines. These maps are not shown for inclusion embeddings into G (they are simple: all sources go to the single source in G, and G has no target to map). Now any relative coproduct G_2 must possess inclusion embeddings from G_i and to H and H' as shown by dashed lines, making the diagram commute. But no such inclusion embeddings can exist. The reader may enjoy proving this.

However, relative coproducts exist for many pairs of inclusion embeddings. We define below a condition CHASTE, which characterises exactly these pairs, i.e. is a necessary and sufficient condition for the existence of relative coproducts.

First we define a key equivalence relation on which CHASTE depends.

Definition 6.9 (source coalescing ($\equiv$)) Let $\eta_i : G_i \rightarrow G$ $(i = 0, 1)$ be a pair of inclusion embeddings. The *source coalescing* of η_0, η_1 is the smallest equivalence relation $\equiv$ on $S_0 + S_1$ such that $src_0\eta_0^{\mathsf{T}}(t) \equiv src_1\eta_1^{\mathsf{T}}(t)$ for all $t \in \mathsf{Dom}\,\eta_0^{\mathsf{T}} \cap \mathsf{Dom}\,\eta_1^{\mathsf{T}}$. ■

An important property of $\equiv$ is that it relates sources that must be equated in any candidate, as shown by the following proposition:

Proposition 6.10 Let ζ_0, ζ_1, ζ be any candidate for η_0, η_1 and let s_i be a source of G_i for $i = 0, 1$. If $s_0 \equiv s_1$ then $\zeta_0^{\mathsf{S}}(s_0) = \zeta_1^{\mathsf{S}}(s_1)$. ■

Definition 6.11 (CHASTE) Let η_i be as in the previous definition. The CHASTE condition holds for η_0, η_1 if for all $s_i \in S_i^{in}$ and $t_i \in T_i^{out}$ $(i = 0, 1)$, if $s_i = src_i(t_i)$ and $\eta_0^{\mathsf{S}}(s_0) = \eta_1^{\mathsf{S}}(s_1)$

then $s_0 \equiv s_1$. ■

The reader may like to check that CHASTE fails for the pair η_0, η_1 in Figure 6.6. The motivation for the name CHASTE comes from the fact that it ensures that the inclusion embeddings do not equate "promiscuously" sources which they should not.

The CHASTE condition is implied by a striking property which is asymmetric between G_0 and G_1:

Definition 6.12 (OUTPUT-CONTROLLED) In any graph, a target is *controlled* if its source is a bound source. G is OUTPUT-CONTROLLED if all its output targets are controlled; formally:

$$src(T^{out}) \subseteq S^{bind} . \qquad \text{(OUTPUT-CONTROLLED)}$$

Since $\mathsf{Cod}\, src = S^{in} \uplus S^{bind}$, an equivalent formulation is:

$$src(T^{out}) \cap S^{in} = \varnothing . \qquad \text{(OUTPUT-CONTROLLED)}$$

■

Proposition 6.13 If G_1 is OUTPUT-CONTROLLED, then any pair of inclusion embeddings $\eta_i : G_i \rightarrowtail G$ $(i = 0, 1)$ satisfies CHASTE. ■

Our interest in this fact arises because we are mainly concerned to find relative coproducts, or RPOs, when one of the graphs is a redex. In all cases I have met, the required reaction relation $\longrightarrow\!\!\triangleright$ can be achieved with reaction rules whose redexes are OUTPUT-CONTROLLED. This point is taken up later in Chapter 7.

Let us fix a pair $\eta_i : G_i \rightarrowtail G$ $(i = 0, 1)$ of inclusion embeddings. To show that CHASTE is sufficient for η_0, η_1 to have a relative coproduct, we could give a direct construction. However, we also wish to show that CHASTE is necessary, and therefore require a second candidate to argue that no relative coproduct exists when CHASTE fails. To factor the work of constructing both candidates, the approach taken is to give a general way to lift a triple of partial maps $\theta_0^{\mathsf{T}}, \theta_1^{\mathsf{T}}, \theta^{\mathsf{T}}$ to a candidate $\theta_0, \theta_1, \theta$. Only those triples that satisfy certain conditions (given here) can be so lifted; we call them *target scaffolds (for η_0, η_1)*.

The drawback to the scaffold approach is that it builds up the relative coproduct incrementally, with many lemmas verifying the healthiness of the intermediate constructions interspersed. The reader may wish to look directly at Figure 6.8 which distills out the construction. The proof that the inclusion embeddings shown in Figure 6.8 *do* actually form a relative coproduct provided that CHASTE is satisfied is given in terms of scaffolds in the following pages.

Remark The construction of T_2 in Figure 6.8 appears complex, so the reader may wish to pattern-match it against the target sets of the RPO examples given in Section 5.3. As we will see shortly (Corollary 6.28) there is a tight correspondence as we now illustrate.

$$\begin{tikzcd}
G_0 \arrow[dr, "\eta_0"] & \\
& G \\
G_1 \arrow[ur, "\eta_1"'] &
\end{tikzcd}$$

6.7(1)

$$\begin{tikzcd}
T_0 & \\
T_2 \arrow[u, "\zeta_0^{\mathsf{T}}"] \arrow[d, "\zeta_1^{\mathsf{T}}"'] & T \arrow[ul, "\eta_0^{\mathsf{T}}"'] \arrow[l, "\zeta^{\mathsf{T}}"'] \arrow[dl, "\eta_1^{\mathsf{T}}"] \\
T_1 &
\end{tikzcd}$$

6.7(2)

$$\begin{tikzcd}
G_0 \arrow[d, "\zeta_0"'] \arrow[dr, "\eta_0"] & \\
G_2 \arrow[r, "\zeta"] & G \\
G_1 \arrow[u, "\zeta_1"] \arrow[ur, "\eta_1"'] &
\end{tikzcd}$$

6.7(3)

$$\begin{tikzcd}
G_0 \arrow[ddr, "\varphi_0"] \arrow[drr, "\eta_0"] & & \\
& & G \\
& G_3 \arrow[ur, "\varphi"] & \\
G_1 \arrow[ur, "\varphi_1"] \arrow[uurr, "\eta_1"'] & &
\end{tikzcd}$$

6.7(4)

$$\begin{tikzcd}
T_0 & & \\
T_2 \arrow[u, "\zeta_0^{\mathsf{T}}"] \arrow[dd, "\zeta_1^{\mathsf{T}}"'] & & T \arrow[ull, "\eta_0^{\mathsf{T}}"'] \arrow[ll, "\zeta^{\mathsf{T}}"'] \arrow[dl, "\varphi^{\mathsf{T}}"'] \arrow[ddll, "\eta_1^{\mathsf{T}}"] \\
& T_3 \arrow[uul, "\varphi_0^{\mathsf{T}}"] \arrow[ul, "\xi^{\mathsf{T}}"'] \arrow[dl, "\varphi_1^{\mathsf{T}}"] & \\
T_1 & &
\end{tikzcd}$$

6.7(5)

$$\begin{tikzcd}
G_0 \arrow[d, "\zeta_0"'] \arrow[ddr, "\varphi_0"] \arrow[drr, "\eta_0"] & & \\
G_2 \arrow[rr, "\zeta"] \arrow[dr, "\xi"'] & & G \\
& G_3 \arrow[ur, "\varphi"] & \\
G_1 \arrow[uu, "\zeta_1"] \arrow[ur, "\varphi_1"] \arrow[uurr, "\eta_1"'] & &
\end{tikzcd}$$

6.7(6)

Figure 6.7: Lifting a scaffold to a candidate relative coproduct

Example 5.1 (arithmetic): Consider the action graphs $G_0 = a$ and $G_1 = l_1$ in Figure 5.2, embedded in b'. The targets of $G_2 = b$, the RPO graph, correspond exactly to T_2; in particular, the extra arc in b connecting S to the top output target is generated by the set O_1 in the definition of T_2. Similarly, $t' \in O_1$ in Figure 6.5.

Example 5.2 (wiring): Consider the identical action graphs $G_0 = a$ and $G_1 = b$ in Figure 5.3, embedded in an identical graph. The targets $\{(t_0, t_2), \dots, (t_1, t_3)\}$ of K in $G = Ca = Db$ are generated by the set $O_{0,1}$ (a subset of the product of the targets from G_0 and G_1) in the definition of T_2.

The outline of the argument is as follows. Fix a pair of inclusion embeddings $\eta_i : G_i \rightarrowtail G$, $i = 0, 1$, as in Figure 6.7(1).

Section 6.5 I define the notion of target scaffold $(\zeta_0^{\mathsf{T}}, \zeta_1^{\mathsf{T}}, \zeta^{\mathsf{T}})$, which consists of a triple of partial maps (Figure 6.7(2)) satisfying certain properties. A scaffold can be lifted to form a candidate w.r.t. η_0, η_1 (Figure 6.7(3)).

Section 6.6 Fix any other candidate $\varphi_0, \varphi_1, \varphi$ (Figure 6.7(4)) and a partial map χ^{T} making the triangles in Figure 6.7(5) commute. Then χ^{T} can be lifted to a mediating inclusion embedding χ (Figure 6.7(6)). Under certain conditions χ is unique.

Section 6.7 I pick a particular scaffold $\theta_0^{\mathsf{T}}, \theta_1^{\mathsf{T}}, \theta^{\mathsf{T}}$ and lift it to a candidate $\theta_0, \theta_1, \theta$ using the technology of scaffold lifting. Provided that CHASTE holds, this candidate is a

The construction builds an action graph G_2 and inclusion embeddings $\theta_i : G_i \rightarrowtail G_2$, $i = 0, 1$ and $\theta : G_2 \rightarrowtail G$. It is convenient to first construct the target set T_2 of G_2 and the target components $\theta_i^{\mathsf{T}}, \theta^{\mathsf{T}}$ of the inclusion embeddings:

$$\begin{aligned}
T_2 &\mathrel{\hat{=}} A + O_0 + O_{0,1} + O_1 \\
A &\mathrel{\hat{=}} \{ arg(v,k) \ / \ v \in V_0 \cup V_1 \} \\
O_{0,1} &\mathrel{\hat{=}} \{ (t_0, t_1) \in T_0^{out} \times T_1^{out} \ / \ \eta_0^{\mathsf{S}} src_0(t_0) = \eta_1^{\mathsf{S}} src_1(t_1) \} \\
O_i &\mathrel{\hat{=}} \{ t_i \in T_i^{out} \ / \ \eta_i^{\mathsf{S}}\, src_i(t_i) \notin \eta_{1-i}^{\mathsf{S}}(S_{1-i}^{bind}) \} \\
\theta_i^{\mathsf{T}}(t) &\mathrel{\hat{=}} \begin{cases} \eta_i^{\mathsf{T}}(t) & \text{if } t \in A \\ t & \text{if } t \in O_i \\ t_i & \text{if } t = (t_0, t_1) \in O_{0,1} \end{cases} \\
\theta^{\mathsf{T}}(t) &\mathrel{\hat{=}} \begin{cases} t & \text{if } t \in A \\ \eta_i^{\mathsf{T}}(t) & \text{if } t \notin A \text{ and } t \in \mathsf{Dom}\,\eta_i^{\mathsf{T}} \setminus \mathsf{Dom}\,\eta_{1-i}^{\mathsf{T}} \\ (\eta_0^{\mathsf{T}}(t), \eta_1^{\mathsf{T}}(t)) & \text{if } t \notin A \text{ and } t \in \mathsf{Dom}\,\eta_0^{\mathsf{T}} \cap \mathsf{Dom}\,\eta_1^{\mathsf{T}} \ . \end{cases}
\end{aligned}$$

Now let $G_2 \mathrel{\hat{=}} (V_2, contr_2, src_2) : (m_2, n_2)$, where:

$$\begin{aligned}
V_2 &\mathrel{\hat{=}} V_0 \cup V_1 \\
contr_2(v) &\mathrel{\hat{=}} contr(v) && \text{for } v \in V_2 \\
arg_2(v,k) &\mathrel{\hat{=}} \theta^{\mathsf{T}} arg(v,k) && \text{for } v \in V_2 \\
n_2 &\mathrel{\hat{=}} \|T_2\| - \|\mathsf{Im}\ arg_2\| \ .
\end{aligned}$$

and m_2 is defined below. Take $\doteq$ to be the least equivalence relation on $S_0 + S_1$ satisfying the following conditions:

$$\begin{aligned}
s_0 &\doteq s_1 && \text{for } i = 0,1,\ s_i \in S_i^{bind},\ s_{1-i} \in S_{1-i}, \text{ and } \eta_0^{\mathsf{S}}(s_0) = \eta_1^{\mathsf{S}}(s_1) \\
src_0\theta_0^{\mathsf{T}}(t) &\doteq src_1\theta_1^{\mathsf{T}}(t) && \text{for } t \in \mathsf{Dom}\,\theta_0^{\mathsf{T}} \cap \mathsf{Dom}\,\theta_1^{\mathsf{T}} \ .
\end{aligned}$$

Let $[\ ]_{\doteq} : S_0 + S_1 \rightarrowtail (S_0 + S_1)/\doteq$ map sources to their equivalence classes. Finally, let the source set of G_2 and the source components of θ_i, θ be as follows:

$$\begin{aligned}
S_2 &\mathrel{\hat{=}} (S_0 + S_1)/\doteq \\
bind_2(v,k) &\mathrel{\hat{=}} [bind_i(v,k)]_{\doteq} && \text{if } v \in V_i \\
m_2 &\mathrel{\hat{=}} \|S_2\| - \|\mathsf{Im}\ bind_2\| \\
src_2(t) &\mathrel{\hat{=}} [src_i\theta_i^{\mathsf{T}}(t)]_{\doteq} && \text{if } t \in \mathsf{Dom}\,\theta_i^{\mathsf{T}} \\
\theta_i^{\mathsf{S}}(s) &\mathrel{\hat{=}} [s]_{\doteq} && \text{for } s \in S_i \\
\theta^{\mathsf{S}}([s]_{\doteq}) &\mathrel{\hat{=}} \eta_i^{\mathsf{S}}(s) && \text{for } s \in S_i \ .
\end{aligned}$$

This completes the construction.

Figure 6.8: The construction of a relative coproduct $\theta_0, \theta_1, \theta$ for $\eta_i : G_i \rightarrowtail G$, $i = 0, 1$ (without scaffolds)

relative coproduct, thus proving the sufficiency of CHASTE. I then exhibit a second target scaffold, $\hat{\theta}^{\mathsf{T}}_0, \hat{\theta}^{\mathsf{T}}_1, \hat{\theta}^{\mathsf{T}}$, lift it to a candidate, and show that the negation of CHASTE implies that no putative relative coproduct is better than both $\theta, \theta_0, \theta_1$ and $\hat{\theta}, \hat{\theta}_0, \hat{\theta}_1$, thus verifying the necessity of CHASTE.

6.5 Construction of a candidate from a scaffold

First I define the notion of *target scaffold*:

Definition 6.14 (target scaffold) A *target scaffold* consists of a set T_2 and a triple of maps $(\zeta^{\mathsf{T}}, \zeta^{\mathsf{T}}_0, \zeta^{\mathsf{T}}_1)$ such that $\zeta^{\mathsf{T}} : T \rightharpoonup T_2$, $\zeta^{\mathsf{T}}_i : T_2 \rightharpoonup T_i$ $(i = 0, 1)$, and such that several conditions hold:

$$\zeta^{\mathsf{T}}_i \circ \zeta^{\mathsf{T}} = \eta^{\mathsf{T}}_i \qquad \text{for } i = 0, 1 \qquad \textsc{Scaf-Comp}$$

$$\mathsf{Dom}\,\zeta^{\mathsf{T}}_0 \cup \mathsf{Dom}\,\zeta^{\mathsf{T}}_1 = T_2 \qquad \textsc{Scaf-Tot}$$

$$\eta^{\mathsf{S}}_0\,(src_0\,(\zeta^{\mathsf{T}}_0\,t_2)) = \eta^{\mathsf{S}}_1\,(src_1\,(\zeta^{\mathsf{T}}_1\,t_2)) \qquad \text{for } t_2 \in \mathsf{Dom}\,\zeta^{\mathsf{T}}_0 \cap \mathsf{Dom}\,\zeta^{\mathsf{T}}_1 \qquad \textsc{Scaf-S}$$

$$\zeta^{\mathsf{T}\,-1}_i\{arg_i(v_i, k)\} = \{\zeta^{\mathsf{T}}\,(arg(v_i, k))\} \qquad \text{for } v_i \in V_i \text{ and } i = 0, 1 \qquad \textsc{Scaf-Arg}$$

$$\zeta^{\mathsf{T}\,-1}_i\,(src^{-1}_i\,(\eta^{\mathsf{S}\,-1}_i\,(\eta^{\mathsf{S}}_j\,S^{bind}_j))) \subseteq \mathsf{Dom}\,\zeta^{\mathsf{T}}_j \qquad \text{for } i, j \in \{0, 1\} \qquad \textsc{Scaf-Targs}$$

■

In the presence of SCAF-COMP, the equality in SCAF-ARG can be replaced by $\subseteq$. SCAF-TARGS is equivalent to: "if $t_2 \in \mathsf{Dom}\,\zeta^{\mathsf{T}}_i$ and $\eta^{\mathsf{S}}_i\,(src_i\,(\zeta^{\mathsf{T}}_i\,t_2)) \in \eta^{\mathsf{S}}_{1-i}\,S^{bind}_{1-i}$ then $t_2 \in \mathsf{Dom}\,\zeta^{\mathsf{T}}_{1-i}$".

It is convenient to express the construction of G_2 up to bijection of its source and target sets, rather than in the exact form of Definition 5.6. Let $G_2 \mathrel{\hat{=}} (V_2, contr_2, src_2) : (m_2, n_2)$ where the parts are defined as follows. First we consider the vertices and targets of G_2:

$$\begin{aligned} V_2 &\mathrel{\hat{=}} V_0 \cup V_1 \\ contr_2(v) &\mathrel{\hat{=}} contr(v) \qquad \text{for } v \in V_2 \\ arg_2(v, k) &\mathrel{\hat{=}} \zeta^{\mathsf{T}} arg(v, k) \qquad \text{for } v \in V_2 \\ n_2 &\mathrel{\hat{=}} \|T_2\| - \|\mathsf{Im}\; arg_2\| \;. \end{aligned}$$

Notice that V_0 and V_1 are not necessarily disjoint: $V_0 \cup V_1$ contains the vertices of G_0 and G_1 with appropriate overlap.

Now we consider the sources and source maps. Take $\doteq$ to be the least equivalence relation on $S_0 + S_1$ satisfying the following conditions:

$$\begin{aligned} s_0 &\doteq s_1 \qquad \text{for } i = 0, 1,\ s_i \in S^{bind}_i,\ s_{1-i} \in S_{1-i}, \text{ and } \eta^{\mathsf{S}}_0(s_0) = \eta^{\mathsf{S}}_1(s_1) \\ src_0\zeta^{\mathsf{T}}_0(t) &\doteq src_1\zeta^{\mathsf{T}}_1(t) \qquad \text{for } t \in \mathsf{Dom}\,\zeta^{\mathsf{T}}_0 \cap \mathsf{Dom}\,\zeta^{\mathsf{T}}_1 \quad . \end{aligned}$$

Let $[\ ]_{\doteq} : S_0 + S_1 \rightarrow (S_0 + S_1)/\doteq$ map sources to their equivalence classes. Then, let:

$$\begin{aligned} S_2 &\mathrel{\hat{=}} (S_0 + S_1)/\doteq \\ bind_2(v,k) &\mathrel{\hat{=}} [bind_i(v,k)]_{\doteq} \qquad \text{if } v \in V_i \\ m_2 &\mathrel{\hat{=}} \|S_2\| - \|\mathsf{Im}\ bind_2\| \\ src_2(t) &\mathrel{\hat{=}} [src_i \zeta_i^{\mathsf{T}}(t)]_{\doteq} \qquad \text{if } t \in \mathsf{Dom}\,\zeta_i^{\mathsf{T}} \\ \zeta_i^{\mathsf{S}}(s) &\mathrel{\hat{=}} [s]_{\doteq} \qquad \text{for } s \in S_i \\ \zeta^{\mathsf{S}}([s]_{\doteq}) &\mathrel{\hat{=}} \eta_i^{\mathsf{S}}(s) \qquad \text{for } s \in S_i\ . \end{aligned}$$

Proposition 6.16 below shows that this construction yields a well-defined graph and well-defined inclusion embeddings which form a candidate for η_0, η_1. Before getting to this result, we first need a technical lemma used several times in the proposition.

Lemma 6.15 If for some $i, j \in \{0,1\}$, $s \in S_i$ and $s' \in S_j$ and $s \doteq s'$ then $\eta_i^{\mathsf{S}}\, s = \eta_j^{\mathsf{S}}\, s'$.

Proof For the first rule generating $\doteq$, the result follows by definition. For the second rule, the result follows by SCAF-S. ■

Proposition 6.16 (scaffold lifting) For the construction given above, G_2 is a graph, ζ_0, ζ_1, ζ are all well-defined inclusion embeddings, and together they form a candidate for η_0, η_1.

Proof

src_2 **is well-defined:** Suppose $t_2 \in \mathsf{Dom}\,\zeta_0^{\mathsf{T}} \cap \mathsf{Dom}\,\zeta_1^{\mathsf{T}}$; then $src_0\,(\zeta_0^{\mathsf{T}}\,t_2) \doteq src_1\,(\zeta_1^{\mathsf{T}}\,t_2)$ by definition of $\doteq$. By SCAF-TOT, src_2 is a total function.

arg_2 **is injective:** Suppose $arg_2(v,k) = arg_2(v',k')$, i.e. $\zeta^{\mathsf{T}}\,(arg(v,k)) = \zeta^{\mathsf{T}}\,(arg(v',k'))$. Suppose $v \in V_i$. By SCAF-COMP,

$$\begin{aligned} arg_i(v,k) = \eta_i^{\mathsf{T}}\,(arg(v,k)) &= \zeta_i^{\mathsf{T}}\,(\zeta^{\mathsf{T}}\,(arg(v,k))) \\ &= \zeta_i^{\mathsf{T}}\,(\zeta^{\mathsf{T}}\,(arg(v',k'))) = \eta_i^{\mathsf{T}}\,(arg(v',k')) \quad . \end{aligned}$$

By INC-ARG for η_i and the injectivity of arg, $v = v'$ and $k = k'$, as desired.

$bind_2$ **is injective:** Suppose $bind_2(v,k) = bind_2(v',k')$ where $v \in V_i$ and $v' \in V_j$. Then $bind_i(v,k) \doteq bind_j(v',k')$. By Lemma 6.15, $bind(v,k) = bind(v',k')$; hence $v = v'$ and $k = k'$, as desired.

ζ_i **is an inclusion embedding for** $i = 0,1$**:** We consider each non-trivial axiom.

INC-SRC: if $t_2 \in \mathsf{Dom}\,\zeta_i^{\mathsf{T}}$ then $\zeta_i^{\mathsf{S}}\,(src_i\,(\zeta_i^{\mathsf{T}}\,t_2)) = [src_i\,(\zeta_i^{\mathsf{T}}\,t_2)]_{\doteq} = src_2\,t_2$.

INC-BIND: for $v_i \in V_i$, ${\zeta_i^{\mathsf{S}}}^{-1}\{bind_2(v_i,k)\} = \{s_i \in S_i \ / \ s_i \doteq bind_i(v_i,k)\}$. If $s_i \in S_i$ and $s_i \doteq bind_i(v_i,k)$ then by Lemma 6.15, $\eta_i^{\mathsf{S}}\,s_i = \eta_i^{\mathsf{S}}\,(bind_i(v_i,k))$, hence by INC-BIND, $s_i = bind_i(v_i,k)$.

INC-ARG: SCAF-ARG.

INC-TARGS: Suppose $src_2\,t_2 = bind_2(v_i,k)$ for $v_i \in V_i$. By SCAF-TOT it is sufficient to assume $t_2 \in \mathsf{Dom}\,\zeta_{1-i}^{\mathsf{T}}$ and prove $t_2 \in \mathsf{Dom}\,\zeta_i^{\mathsf{T}}$. Thus, $src_{1-i}\,(\zeta_{1-i}^{\mathsf{T}}\,t_2) \doteq bind_i(v_i,k)$. By Lemma 6.15,

$$\eta_{1-i}^{\mathsf{S}}\,(src_{1-i}\,(\zeta_{1-i}^{\mathsf{T}}\,t_2)) = \eta_i^{\mathsf{S}}\,(bind_i(v_i,k))$$

By SCAF-TARGS, $t_2 \in \mathsf{Dom}\,\zeta_i^{\mathsf{T}}$.

ζ is an inclusion embedding: First note that ζ^{S} is well-defined by Lemma 6.15.

> INC-SRC: Suppose $t \in \mathsf{Dom}\,\zeta^{\mathsf{T}}$. By SCAF-TOT, there exists $i = 0, 1$ such that $\zeta^{\mathsf{T}}\, t \in \mathsf{Dom}\,\zeta^{\mathsf{T}}_i$. Hence, $\zeta^{\mathsf{S}}\,(src_2\,(\zeta^{\mathsf{T}}\,t)) = \eta^{\mathsf{S}}_i\,(src_i\,(\zeta^{\mathsf{T}}_i\,(\zeta^{\mathsf{T}}\,t))) = src\,t$ by SCAF-COMP and INC-SRC for η_i.
>
> INC-BIND: Let $v_i \in V_i$; then $\zeta^{\mathsf{S}}\,(bind_2(v_i,k)) = bind(v_i,k)$ by INC-BIND for η_i. If $\zeta^{\mathsf{S}}\,[s_i]_{\doteq} = bind(v_i,k)$ for $s_i \in S_i$ then by INC-BIND for η_i, $s_i = bind_i(v_i,k)$; if $\zeta^{\mathsf{S}}\,[s_{1-i}]_{\doteq} = bind(v_i,k)$ for $s_{1-i} \in S_{1-i}$ then $s_{1-i} \doteq bind_i(v_i,k)$.
>
> INC-ARG: By definition, $\zeta^{\mathsf{T}}\,(arg(v,k)) = arg_2(v,k)$. Also, by SCAF-COMP and INC-ARG for η^{T}, we have that $\zeta^{\mathsf{T}}\,t = arg_2(v,k)$ implies $t = arg(v,k)$.
>
> INC-TARGS: If $src\,t = bind(v_i,k)$ for $v_i \in V_i$ then $t \in \mathsf{Dom}\,\eta^{\mathsf{T}}_i$, hence $t \in \mathsf{Dom}\,\zeta^{\mathsf{T}}$ by SCAF-COMP.

$\zeta\zeta_i = \eta_i$: By SCAF-COMP we only need to show that $\zeta^{\mathsf{S}} \circ \zeta^{\mathsf{S}}_i = \eta^{\mathsf{S}}_i$. Observe that $\zeta^{\mathsf{S}}\,(\zeta^{\mathsf{S}}_i\,s_i) = \zeta^{\mathsf{S}}\,[s_i]_{\doteq} = \eta^{\mathsf{S}}_i\,s_i$ for all $s_i \in S_i$. ■

When ζ^{T} is surjective, the inclusion embeddings induced by the scaffold have an important property which we make use of later, namely ζ_0, ζ_1 do not identify more input sources than required by $\equiv$ (cf. Proposition 6.10).

Lemma 6.17 Suppose $\zeta^{\mathsf{T}}\,T = T_2$. Let $s_i \in S^{in}_i$ for $i = 0, 1$. If $s_0 \not\equiv s_1$ then $\zeta^{\mathsf{S}}_0\,s_0 \neq \zeta^{\mathsf{S}}_1\,s_1$.

Proof We show the contrapositive: $s_0 \doteq s_1$ implies $s_0 \equiv s_1$. Consider any derivation for the antecedent of minimal length:

$$s_0 = s^1_0 \doteq s^1_1 \doteq \cdots \doteq s^n_0 \doteq s^n_1 = s_1$$

where $s^j_i \in S_i$ for $i = 0, 1$, $1 \le j \le n$. Suppose for contradiction there exist i, j such that $s^j_i \in S^{bind}_i$. By Lemma 6.15, $\eta^{\mathsf{S}}_i\,s_i = \eta^{\mathsf{S}}_i\,s^j_i$, hence $\zeta^{\mathsf{S}}\,(\zeta^{\mathsf{S}}_i\,s_i) = \zeta^{\mathsf{S}}\,(\zeta^{\mathsf{S}}_i\,s^j_i)$; by INC-BIND $\zeta^{\mathsf{S}}_i\,s_i = \zeta^{\mathsf{S}}_i\,s^j_i$, contradicting the minimality of the derivation. Thus, there exist $t^1_2, \ldots, t^n_2 \in \mathsf{Dom}\,\zeta^{\mathsf{T}}_0 \cap \mathsf{Dom}\,\zeta^{\mathsf{T}}_1$ such that $s^j_i = src_i\,(\zeta^{\mathsf{T}}_i\,t^j_2)$. Because $\zeta^{\mathsf{T}}\,T = T_2$, there exist $t^1, \ldots, t^n \in T$ such that $\zeta^{\mathsf{T}}\,t^j = t^j_2$ for $1 \le j \le n$. Hence $s^j_i = src_i\,(\zeta^{\mathsf{T}}_i\,(\zeta^{\mathsf{T}}\,t^j)) = src_i\,(\eta^{\mathsf{T}}_i\,t^j)$. Thus $s_0 \equiv s_1$ as desired. ■

6.6 Construction of a mediating inclusion embedding

Let G_3 be a graph and $\varphi_0, \varphi_1, \varphi$ a triple of inclusion embeddings such that $\varphi_i : G_i \rightarrowtail G_3$ and $\varphi : G_3 \rightarrowtail G$ and $\eta_i = \varphi\varphi_i$, i.e. $\varphi_0, \varphi_1, \varphi$ is a relative coproduct candidate for η_0, η_1. Our aim to construct an inclusion embedding χ mediating between the ζ_0, ζ_1, ζ candidate (lifted from the candidate scaffold above) and the $\varphi_0, \varphi_1, \varphi$ candidate by lifting a mediating target map χ^{T} (a "mediating scaffold"). Formally, let $\chi^{\mathsf{T}} : T_3 \rightharpoonup T_2$ be a partial map satisfying $\chi^{\mathsf{T}} \circ \varphi^{\mathsf{T}} = \zeta^{\mathsf{T}}$ and $\zeta_i \circ \chi^{\mathsf{T}} = \varphi^{\mathsf{T}}_i$ for $i = 0, 1$. We wish to lift χ^{T} to an inclusion embedding χ that mediates between the two candidates. Construct $\chi^{\mathsf{S}} : S_2 \rightarrowtail S_3$ as follows:

$$\chi^{\mathsf{S}}\,[s_i]_{\doteq} = \varphi^{\mathsf{S}}_i\,s_i \qquad \text{for } s_i \in S_i \quad .$$

Now, χ has all the required properties, as shown in the following proposition, the only result that makes use of the CHASTE condition.

Proposition 6.18 (mediator lifting) χ^{S} is well-defined; χ is an inclusion embedding and mediates between the candidates ζ_0, ζ_1, ζ and $\varphi_0, \varphi_1, \varphi$.

Proof

χ^{S} **is well-defined:** We consider each case of the definition of $\doteq$.

Case $s_i \in S_i^{bind}$**,** $s_{1-i} \in S_{1-i}$**, and** $\eta_i^{\mathsf{S}}\, s_i = \eta_{1-i}^{\mathsf{S}}\, s_{1-i}$**:** Then $\varphi_i^{\mathsf{S}}\, s_i \in S_3^{bind}$ by Inc-Bind for φ_i; also $\varphi^{\mathsf{S}}\,(\varphi_i^{\mathsf{S}}\, s_i) = \varphi^{\mathsf{S}}\,(\varphi_{1-i}^{\mathsf{S}}\, s_{1-i})$ so $\varphi_i^{\mathsf{S}}\, s_i = \varphi_{1-i}^{\mathsf{S}}\, s_{1-i}$ by Inc-Bind for φ^{S}.

Case $t_2 \in \mathsf{Dom}\, \zeta_0^{\mathsf{T}} \cap \mathsf{Dom}\, \zeta_1^{\mathsf{T}}$**:** Let $s_i = src\,(\zeta_i^{\mathsf{T}}\, t_2)$ for $i = 0, 1$. By Lemma 6.15, $\eta_0^{\mathsf{S}}\, s_0 = \eta_1^{\mathsf{S}}\, s_1$.

Case $s_i \in S_i^{bind}$ **for some** $i = 0, 1$**:** By Inc-Bind, $\varphi_i^{\mathsf{S}}\, s_i \in S_3^{bind}$, hence $\varphi^{\mathsf{S}}\,(\varphi_0^{\mathsf{S}}\, s_0) = \varphi^{\mathsf{S}}\,(\varphi_1^{\mathsf{S}}\, s_1)$ implies that $\varphi_0^{\mathsf{S}}\, s_0 = \varphi_1^{\mathsf{S}}\, s_1$ by Inc-Bind for φ.

Case $s_i \in S_i^{in}$ **for** $i = 0, 1$**:** We distinguish two cases:

Case $t_2 \in T_2^{arg}$**:** There exists $t \in T^{arg}$ such that $\zeta^{\mathsf{T}}\, t = t_2$, by Inc-Arg for ζ. Thus, by Inc-Src for φ_i we have that for $i = 0, 1$:

$$\begin{aligned}\varphi_i^{\mathsf{S}}\, s_i &= \varphi_i^{\mathsf{S}}\,(src_i\,(\zeta_i^{\mathsf{T}}\, t_2)) = \varphi_i^{\mathsf{S}}\,(src_i\,(\zeta_i^{\mathsf{T}}\,(\zeta^{\mathsf{T}}\, t))) = \varphi_i^{\mathsf{S}}\,(src_i\,(\eta_i^{\mathsf{T}}\, t))\\ &= \varphi_i^{\mathsf{S}}\,(src_i\,(\varphi_i^{\mathsf{T}}\,(\varphi^{\mathsf{T}}\, t))) = src_3(\varphi^{\mathsf{T}}\, t)\end{aligned}$$

Case $t_2 \in T_2^{out}$**:** By Inc-Arg for ζ_i, $\zeta_i^{\mathsf{T}}\, t_2 \in T_i^{out}$ for $i = 0, 1$. By the Chaste condition, $s_0 \equiv s_1$. By Proposition 6.10, $\varphi_0^{\mathsf{S}}\, s_0 = \varphi_1^{\mathsf{S}}\, s_1$.

χ **is an inclusion embedding:**

Inc-Src: if $t_3 \in \mathsf{Dom}\, \chi^{\mathsf{T}}$, then by Scaf-Tot there exists $i = 0, 1$ such that $\chi^{\mathsf{T}}\, t_3 \in \mathsf{Dom}\, \zeta_i^{\mathsf{T}}$. Hence $\chi^{\mathsf{S}}\,(src_2\,(\chi^{\mathsf{T}}\, t_3)) = \chi^{\mathsf{S}}\,[src_i\,(\zeta_i^{\mathsf{T}}\,(\chi^{\mathsf{T}}\, t_3))]_{\doteq} = \chi^{\mathsf{S}}\,[src_i\,(\varphi_i^{\mathsf{T}}\, t_3)]_{\doteq} = \varphi_i^{\mathsf{S}}\,(src_i\,(\varphi_i^{\mathsf{T}}\, t_3)) = src_3\, t_3$ by Inc-Src for φ_i.

Inc-Bind: if $v_i \in V_i$ then $\chi^{\mathsf{S}}\,(bind_2(v_i, k)) = \chi^{\mathsf{S}}\,[bind_i(v_i, k)]_{\doteq} = \varphi_i^{\mathsf{S}}\,(bind_i(v_i, k)) = bind_3(v_i, k)$ by Inc-Bind for φ_i. Also, if $\chi^{\mathsf{S}}\,[s_j]_{\doteq} = bind_3(v_i, k)$ for $s_j \in S_j$ then $\varphi_j^{\mathsf{S}}\, s_j = bind_3(v_i, k)$, therefore $\eta_j^{\mathsf{S}}\, s_j = bind(v_i, k) = \eta_i^{\mathsf{S}}\,(bind_i(v_i, k))$ by Inc-Bind for φ.

Case $i = j$**:** By Inc-Bind, $s_j = bind_i(v_i, k)$.

Case $i \neq j$**:** Then $s_j \doteq bind_i(v_i, k)$.

In both cases $[s_j]_{\doteq} = bind_2(v_i, k)$.

Inc-Arg: for $v_i \in V_i$, $\chi^{\mathsf{T}}\, t_3 = arg_2(v_i, k)$ iff $\zeta_i^{\mathsf{T}}\,(\chi^{\mathsf{T}}\, t) = \zeta_i^{\mathsf{T}}\,(arg_2(v_i, k)) = arg_i(v_i, k)$ (by Inc-Arg for ζ_i) iff $\varphi_i^{\mathsf{T}}\, t = arg_i(v_i, k)$ iff $t = arg_3(v_i, k)$ (by Inc-Arg for φ_i).

Inc-Targs: if $src_3\, t_3 = bind_3(v_i, k)$ for $v_i \in V_i$, then $t_3 \in \mathsf{Dom}\, \varphi_i^{\mathsf{T}} = \mathsf{Dom}\,(\zeta_i^{\mathsf{T}} \circ \chi^{\mathsf{T}})$ by Inc-Targs for φ_i, hence $t_3 \in \mathsf{Dom}\, \chi^{\mathsf{T}}$.

$\chi\zeta_i = \varphi_i$ **and** $\varphi\chi = \zeta$**:** The equalities for the target maps are postulated.

For the source map equalities, observe that $\chi^{\mathsf{S}}\,(\zeta_i^{\mathsf{S}}\, s_i) = \chi^{\mathsf{S}}\,[s_i]_{\doteq} = \varphi_i^{\mathsf{S}}\, s_i$ for $s_i \in S_i$; also $\varphi^{\mathsf{S}}\,(\chi^{\mathsf{S}}\,[s_i]_{\doteq}) = \varphi^{\mathsf{S}}\,(\varphi_i^{\mathsf{S}}\, s_i) = \eta_i^{\mathsf{S}}\, s_i = \zeta^{\mathsf{S}}\,[s_i]_{\doteq}$. ■

Under certain conditions the mediator χ is unique. These conditions depend on the following general definition concerning partial maps.

Definition 6.19 (joint-injectivity) Given partial maps $f_i : X \rightharpoonup X_i$, $i = 0, 1$, we say that (f_0, f_1) are *jointly-injective* if two conditions hold:

- for $x, x' \in \mathsf{Dom}\, f_i \setminus \mathsf{Dom}\, f_{1-i}$, if $f_i\, x = f_i\, x'$ then $x = x'$, for $i = 0, 1$;

- for $x, x' \in \mathsf{Dom}\, f_0 \cap \mathsf{Dom}\, f_1$, if $(f_0\, x, f_1\, x) = (f_0\, x'e, f_1\, x')$ then $x = x'$. ■

Note that (f_0, f_1) are jointly-injective iff their product-pairing $\langle f_0, f_1\rangle : X \rightharpoonup X_0 + X_0 \times X_1 + X_1$ in the category of sets and partial maps is a monomorphism.

Proposition 6.20 (uniqueness of the mediator) If $(\zeta_0^{\mathsf{T}}, \zeta_1^{\mathsf{T}})$ are jointly-injective then χ is a unique mediator.

Proof Let $\xi : G_2 \rightarrowtail G_3$ be an inclusion embedding satisfying $\xi\zeta_i = \varphi_i$ for $i = 0, 1$ and $\varphi\xi = \zeta$, i.e. ξ is an another mediating inclusion embedding between the candidates $\varphi_0, \varphi_1, \varphi$ and ζ_0, ζ_1, ζ. Then $\chi = \xi$:

$\chi^{\mathsf{S}} = \xi^{\mathsf{S}}$**:** for $s_i \in S_i$, $\chi^{\mathsf{S}}\,[s_i]_{\doteq} = \chi^{\mathsf{S}}\,(\zeta_i^{\mathsf{S}}\, s_i) = \varphi_i^{\mathsf{S}}\, s_i = \xi^{\mathsf{S}}\,(\zeta_i^{\mathsf{S}}\, s_i) = \xi^{\mathsf{S}}\,[s_i]_{\doteq}$.

$\chi^{\mathsf{T}} = \xi^{\mathsf{T}}$**:** if $t_3 \in \mathsf{Dom}\,\xi^{\mathsf{T}}$ then by Scaf-Tot $\xi^{\mathsf{T}}\, t_3 \in \mathsf{Dom}\,\zeta_i^{\mathsf{T}}$ for some $i = 0, 1$; hence $t_3 \in \mathsf{Dom}\,(\zeta_i^{\mathsf{T}} \circ \xi^{\mathsf{T}}) = \mathsf{Dom}\,\varphi_i^{\mathsf{T}} = \mathsf{Dom}\,(\zeta_i^{\mathsf{T}} \circ \chi^{\mathsf{T}})$; thus $t_3 \in \mathsf{Dom}\,\chi^{\mathsf{T}}$. By symmetry, $t_3 \in \mathsf{Dom}\,\chi^{\mathsf{T}}$ iff $t_3 \in \mathsf{Dom}\,\xi^{\mathsf{T}}$, so $\mathsf{Dom}\,\chi^{\mathsf{T}} = \mathsf{Dom}\,\xi^{\mathsf{T}}$. Also, $\chi^{\mathsf{T}}\, t_3 \in \mathsf{Dom}\,\zeta_i^{\mathsf{T}}$ iff $\xi^{\mathsf{T}}\, t_3 \in \mathsf{Dom}\,\zeta_i^{\mathsf{T}}$, for $i = 0, 1$. Finally, $\zeta_i^{\mathsf{T}}\,(\xi^{\mathsf{T}}\, t_3) = \varphi_i^{\mathsf{T}}\, t_3 = \zeta_i^{\mathsf{T}}\,(\chi^{\mathsf{T}}\, t_3)$, for $i = 0, 1$. Hence $\xi^{\mathsf{T}} = \chi^{\mathsf{T}}$ by the joint-injectivity hypothesis for $(\zeta_0^{\mathsf{T}}, \zeta_1^{\mathsf{T}})$. ■

6.7 Construction of a relative coproduct

We now come to the main applied results of the dissertation which characterise precisely the conditions under which relative coproducts exist, and thus under which condition $\mathcal{F}$ has all redex-RPOs. The idea is to consider a particular scaffold, prove that it is well-defined, lift it to a candidate and then show that it is "better" than any other competing candidate, subject to condition Chaste. I discuss the sufficiency of Chaste later.

We exhibit a target scaffold consisting of a set T_2 and a triple $\theta_0^{\mathsf{T}}, \theta_1^{\mathsf{T}}, \theta^{\mathsf{T}}$, which we lift to a relative coproduct. Let:

$$\begin{aligned}
T_2 &\mathrel{\hat{=}} A + O_0 + O_{0,1} + O_1\\
A &\mathrel{\hat{=}} \{arg(v,k) \;\;/\;\; v \in V_0 \cup V_1\}\\
O_{0,1} &\mathrel{\hat{=}} \{(t_0, t_1) \in T_0^{out} \times T_1^{out} \;\;/\;\; \eta_0^{\mathsf{S}} src_0(t_0) = \eta_1^{\mathsf{S}} src_1(t_1)\}\\
O_i &\mathrel{\hat{=}} \{t_i \in T_i^{out} \;\;/\;\; \eta_i^{\mathsf{S}}\, src_i(t_i) \notin \eta_{1-i}^{\mathsf{S}}(S_{1-i}^{bind})\}\\
\theta_i^{\mathsf{T}}(t) &\mathrel{\hat{=}} \begin{cases} \eta_i^{\mathsf{T}}(t) & \text{if } t \in A\\ t & \text{if } t \in O_i\\ t_i & \text{if } t = (t_0, t_1) \in O_{0,1}\end{cases}\\
\theta^{\mathsf{T}}(t) &\mathrel{\hat{=}} \begin{cases} t & \text{if } t \in A\\ \eta_i^{\mathsf{T}}(t) & \text{if } t \notin A \text{ and } t \in \mathsf{Dom}\,\eta_i^{\mathsf{T}} \setminus \mathsf{Dom}\,\eta_{1-i}^{\mathsf{T}}\\ (\eta_0^{\mathsf{T}}(t), \eta_1^{\mathsf{T}}(t)) & \text{if } t \notin A \text{ and } t \in \mathsf{Dom}\,\eta_0^{\mathsf{T}} \cap \mathsf{Dom}\,\eta_1^{\mathsf{T}} \end{cases}.
\end{aligned}$$

In Proposition 6.22 below we verify that $\theta_0^{\mathsf{T}}, \theta_1^{\mathsf{T}}, \theta^{\mathsf{T}}$ is a target scaffold, so by Proposition 6.16 can be lifted to a candidate $\theta_0, \theta_1, \theta$. We first prove a technical lemma.

Lemma 6.21 Let G_3 be a graph and ψ_0, ψ_1, ψ a triple of inclusion embeddings such that $\psi_i : G_i \rightarrowtail G_3$ and $\psi : G_3 \rightarrowtail G$ and $\eta_i = \psi\psi_i$, i.e. ψ_0, ψ_1, ψ forms a candidate.

1. If $t_3 \notin \psi^{\mathsf{T}} A$ and $t_3 \in \mathsf{Dom}\, \psi_i^{\mathsf{T}} \setminus \mathsf{Dom}\, \psi_{1-i}^{\mathsf{T}}$ then $\psi_i^{\mathsf{T}}\, t_3 \in O_i$.

2. If $t_3 \notin \psi^{\mathsf{T}} A$ and $t_3 \in \mathsf{Dom}\, \psi_0^{\mathsf{T}} \cap \mathsf{Dom}\, \psi_1^{\mathsf{T}}$ then $(\psi_0^{\mathsf{T}}\, t_3, \psi_1^{\mathsf{T}}\, t_3) \in O_{0,1}$.

Proof Note that $\psi_i^{\mathsf{T}}\, t_3 \in T_i^{arg}$ iff for some k, i and $v_i \in V_i$, $t_3 = arg_3(v_i, k) = \psi^{\mathsf{T}}\,(arg(v_i, k)) \in \psi^{\mathsf{T}} A$, by INC-ARG for ψ_i and ψ. Thus $t_3 \notin \psi^{\mathsf{T}} A$ implies $\psi_i^{\mathsf{T}}\, t_3 \in T_i^{out}$ if $t \in \mathsf{Dom}\, \psi_i^{\mathsf{T}}$. We use this observation in each of the following parts.

1. By the observation in the head of this proof, $\psi_i^{\mathsf{T}}\, t \in T_i^{out}$; by INC-SRC, $\eta_i^{\mathsf{S}}\,(src_i\,(\psi_i^{\mathsf{T}}\, t)) = \psi^{\mathsf{S}}\,(src_3\, t_3)$; then $\psi^{\mathsf{S}}\,(src_3\, t_3) \in \eta_{1-i}^{\mathsf{S}}\, S_{1-i}^{bind} = \psi^{\mathsf{S}}\,(\psi_{1-i}^{\mathsf{S}}\, S_{1-i}^{bind})$ implies that $src_3\, t_3 \in \psi_{1-i}^{\mathsf{S}}\, S_{1-i}^{bind}$ by INC-BIND for ψ; by INC-TARGS for ψ_{1-i}, we have $t_3 \in \mathsf{Dom}\, \psi_{1-i}^{\mathsf{T}}$, which contradicts the hypothesis. Thus $\psi_i^{\mathsf{T}}\, t_3 \in O_i$ as desired.
2. By the observation in the head of this proof, $\psi_i^{\mathsf{T}}\, t_3 \in T_i^{out}$ for $i = 0, 1$. By INC-SRC, $(\psi_0^{\mathsf{T}}\, t_3, \psi_1^{\mathsf{T}}\, t_3) \in O_{0,1}$. ■

Proposition 6.22 (healthiness of relative coproduct scaffold) $(\theta^{\mathsf{T}}, \theta_0^{\mathsf{T}}, \theta_1^{\mathsf{T}})$ is a target scaffold.

Proof

SCAF-COMP: By Lemma 6.21 instantiated with $G_3 = G$, $\zeta = \mathsf{id}_G$, and $\zeta_i = \eta_i$, we have

$$\theta_i^{\mathsf{T}}\,(\theta^{\mathsf{T}}\, t) = \begin{cases} \eta_i^{\mathsf{T}}\, t & \text{if } t \in A \\ \eta_i^{\mathsf{T}}\, t & \text{if } t \notin A \text{ and } t \in \mathsf{Dom}\, \eta_i^{\mathsf{T}} \setminus \mathsf{Dom}\, \eta_{1-i}^{\mathsf{T}} \\ \eta_i^{\mathsf{T}}\, t & \text{if } t \notin A \text{ and } t \in \mathsf{Dom}\, \eta_0^{\mathsf{T}} \cap \mathsf{Dom}\, \eta_1^{\mathsf{T}} \end{cases}$$

Because the cases above are exhaustive, $\theta^{\mathsf{T}}\theta_i^{\mathsf{T}} = \eta_i^{\mathsf{T}}$.

SCAF-TOT: By definition.

SCAF-S: Let $t \in \mathsf{Dom}\, \theta_0^{\mathsf{T}} \cap \mathsf{Dom}\, \theta_1^{\mathsf{T}}$. If $t \in A$ then $\eta_i^{\mathsf{S}}\,(src_i\,(\theta_i^{\mathsf{T}}\, t_2)) = \eta_i^{\mathsf{S}}\,(src_i\,(\eta_i^{\mathsf{T}}\, t_2)) = src\, t_2$ by INC-SRC for $i = 0, 1$. If $t \in O_{0,1}$ then the result follows by definition.

SCAF-ARG: Since SCAF-COMP has already been verified, we need only show $\subseteq$. If $\theta_i^{\mathsf{T}}\, t_2 = arg_i(v_i, k)$ for some $v_i \in V_i$ then $t_2 \in A$. Hence $\eta_i^{\mathsf{T}}\, t_2 = \theta_i^{\mathsf{T}}\, t_2 = arg_i(v_i, k)$, so $t_2 = arg(v_i, k) = arg_2(v_i, k)$ by INC-ARG for η_i.

SCAF-TARGS: Suppose that $t_2 \in \mathsf{Dom}\, \theta_i^{\mathsf{T}}$ and $\eta_i^{\mathsf{S}}\,(src_i\,(\theta_i^{\mathsf{T}}\, t_2)) \in \eta_{1-i}^{\mathsf{S}}\, S_{1-i}^{bind}$. Then $t_2 \notin O_i$. If $t_2 \in A$ then by INC-SRC for η_i, $src\, t_2 = \eta_i^{\mathsf{S}}\,(src_i\,(\eta_i^{\mathsf{T}}\, t_2)) = \eta_i^{\mathsf{S}}\,(src_i\,(\theta_i^{\mathsf{T}}\, t_2)) \in \eta_{1-i}^{\mathsf{S}}\, S_{1-i}^{bind}$. By INC-TARGS for η_{1-i}, $t_2 \in \mathsf{Dom}\, \eta_{1-i}^{\mathsf{T}}$ hence $t_2 \in \mathsf{Dom}\, \theta_{1-i}^{\mathsf{T}}$ (since $t_2 \in A$). If $t_2 \notin A$ then $t_2 \in O_{0,1}$ hence $t_2 \in \mathsf{Dom}\, \theta_{1-i}^{\mathsf{T}}$. ■

Now, consider any other candidate: let G_3 be a graph and $(\varphi_0, \varphi_1, \varphi)$ a triple of inclusion embeddings such that $\varphi_i : G_i \rightarrowtail G_3$ and $\varphi : G_3 \rightarrowtail G$ and $\eta_i = \varphi\varphi_i$. Construct $\chi^{\mathsf{T}} : T_3 \rightharpoonup T_2$ as follows:

$$\chi^{\mathsf{T}}\, t_3 \mathrel{\hat{=}} \begin{cases} \varphi^{\mathsf{T}-1}\, t_3 & \text{if } t_3 \in \varphi^{\mathsf{T}} A \\ \varphi_i^{\mathsf{T}}\, t_3 & \text{if } t_3 \notin \varphi^{\mathsf{T}} A \text{ and } t_3 \in \mathsf{Dom}\, \varphi_i^{\mathsf{T}} \setminus \mathsf{Dom}\, \varphi_{1-i}^{\mathsf{T}} \\ (\varphi_0^{\mathsf{T}}\, t_3, \varphi_1^{\mathsf{T}}\, t_3) & \text{if } t_3 \notin \varphi^{\mathsf{T}} A \text{ and } t_3 \in \mathsf{Dom}\, \varphi_0^{\mathsf{T}} \cap \mathsf{Dom}\, \varphi_1^{\mathsf{T}} \end{cases}$$

Now we claim that χ^{T} satisfies all the conditions used by Proposition 6.18 needed for it to be lifted to a mediating inclusion embedding:

Proposition 6.23 (mediator healthiness) $\chi^\top$ is well-defined; $\chi^\top \circ \varphi^\top = \theta^\top$ and $\theta_i^\top \circ \chi^\top = \varphi_i^\top$ for $i = 0, 1$

Proof

$\chi^\top$ **is well-defined:** If $t_3 \in \varphi^\top A$ then by INC-ARG for φ there exists a unique $t \in A$ such that $\varphi^\top t = t_3$.

$\chi^\top \circ \varphi^\top = \theta^\top$**:** Expanding the LHS, we have:

$$\chi^\top(\varphi^\top t) = \begin{cases} t & \text{if } t \in \mathsf{Dom}\,\varphi^\top \text{ and } \varphi^\top t \in \varphi^\top A \\ \eta_i^\top t & \text{if } t \in \mathsf{Dom}\,\varphi^\top \text{ and } \varphi^\top t \notin \varphi^\top A \text{ and } \varphi^\top t \in \mathsf{Dom}\,\varphi_i^\top \setminus \mathsf{Dom}\,\varphi_{1-i}^\top \\ (\eta_0^\top t, \eta_1^\top t) & \text{if } t \in \mathsf{Dom}\,\varphi^\top \text{ and } \varphi^\top t \notin \varphi^\top A \text{ and } \varphi^\top t \in \mathsf{Dom}\,\varphi_0^\top \cap \mathsf{Dom}\,\varphi_1^\top \end{cases}$$

By INC-ARG for φ,

$$t \in \mathsf{Dom}\,\varphi^\top \text{ and } \varphi^\top t \in \varphi^\top A \qquad \text{iff} \qquad t \in A$$

Also, because $\varphi_i^\top \circ \varphi^\top = \eta_i^\top$ for $i = 0, 1$,

$$t \in \mathsf{Dom}\,\varphi^\top \text{ and } \varphi^\top t \in \mathsf{Dom}\,\varphi_i^\top \qquad \text{iff} \qquad t \in \mathsf{Dom}\,\eta_i^\top$$

Thus $\chi^\top \circ \varphi^\top = \theta^\top$.

$\theta_i^\top \circ \chi^\top = \varphi_i^\top$ **for** $i = 0, 1$**:** By Lemma 6.21, the LHS of the second equality expands to:

$$\theta_i^\top(\chi^\top t_3) = \begin{cases} \theta_i^\top(\varphi^{\top\,-1} t_3) & \text{if } t_3 \in \varphi^\top A \\ \varphi_i^\top t_3 & \text{if } t_3 \notin \varphi^\top A \text{ and } t_3 \in \mathsf{Dom}\,\varphi_i^\top \setminus \mathsf{Dom}\,\varphi_{1-i}^\top \\ \varphi_i^\top t_3 & \text{if } t_3 \notin \varphi^\top A \text{ and } t_3 \in \mathsf{Dom}\,\varphi_0^\top \cap \mathsf{Dom}\,\varphi_1^\top \end{cases}$$

Note that if $t_3 \in \varphi^\top A$ then $t_3 = \varphi^\top t$ for some $t \in A$; thus, the first case in the expansion is: $\theta_i^\top(\varphi^{\top\,-1} t_3) = \theta_i^\top t = \eta_i^\top t = \varphi_i^\top t_3$ by INC-ARG. Because the cases in the expansion above are exhaustive, $\theta_i^\top \circ \chi^\top = \varphi_i^\top$. ■

Moreover, by Proposition 6.20, the mediator χ is unique because $(\theta_0^\top, \theta_1^\top)$ are jointly-injective, as verified in Proposition 6.25 below after a technical lemma.

Lemma 6.24 If $t_2 \in \mathsf{Dom}\,\theta_i^\top \setminus \mathsf{Dom}\,\theta_{1-i}^\top$ then $t_2 \in O_i$ or $t_2 = arg_2(v_i, k)$ for some k and $v_i \in V_i$.

Proof If $t_2 \notin O_i$ then $t_2 \in A$, hence $t_2 = arg_2(v_j, k)$ for some k, j, $v_j \in V_j$. Since $t_2 \notin \mathsf{Dom}\,\theta_{1-i}^\top$, $i = j$. ■

Proposition 6.25 $(\theta_0^\top, \theta_1^\top)$ are jointly-injective.

Proof

Case $t_2, t_2' \in \mathsf{Dom}\,\theta_i^\top \setminus \mathsf{Dom}\,\theta_{1-i}^\top$ **and** $\theta_i^\top t_2 = \theta_i^\top t_2'$**:** By Lemma 6.24, there are two cases.

Case $t_2 \in O_i$**:** Then $t_2 = \theta_i^\top t_2'$. If $t_2' \notin O_i$ then by Lemma 6.24, $\theta_i^\top t_2' \in T_i^{arg}$, but $t_2 \in T_i^{out}$, a contradiction. Thus $t_2' \in O_i$ and $t_2 = t_2'$.

Case $t_2 = arg(v_i, k)$ **for some** $v_i \in V_i$ **and** k**:** then $arg_i(v_i, k) = \theta_i^\top t_2'$ thus $t_2' \notin O_i$; by Lemma 6.24, $t_2' = arg(v_i', k')$ for some $v_i' \in V_i$ and k'; but then $v_i = v_i'$ and $k = k'$, so $t_2 = t_2'$.

Case $t_2, t'_2 \in \mathsf{Dom}\,\theta_0^{\mathsf{T}} \cap \mathsf{Dom}\,\theta_1^{\mathsf{T}}$ and $(\theta_0^{\mathsf{T}}\,t_2, \theta_1^{\mathsf{T}}\,t_2) = (\theta_0^{\mathsf{T}}\,t'_2, \theta_1^{\mathsf{T}}\,t'_2)$: There are two subcases.

Case $t_2 \in O_{0,1}$: Then $(t_2, t_2) = (\theta_0^{\mathsf{T}}\,t'_2, \theta_1^{\mathsf{T}}\,t'_2)$, so $t'_2 \in O_{0,1}$ and $t_2 = t'_2$.

Case $t_2 = arg_2(v_i, k)$ for some $v_i \in V_i$ and k: Then $t'_2 \in A$ and $\eta_i^{\mathsf{T}}\,t_2 = \eta_i^{\mathsf{T}}\,t'_2$, hence $t_2 = t'_2$ by INC-ARG. ■

In summary, then:

Theorem 6.26 (sufficiency of CHASTE for relative coproducts) If a pair of inclusion embeddings $\eta_i : G_i \rightarrowtail G$ $(i = 0, 1)$ satisfies CHASTE then the constructed triple $\theta_0, \theta_1, \theta$ of inclusion embeddings forms a relative coproduct for η_0, η_1 in **G-Inc**. ■

We now turn to the necessity of CHASTE. We first construct a second candidate for η_0, η_1 as follows. Let

$$\begin{aligned} \hat{T}_2 &\mathrel{\hat{=}} \mathsf{Dom}\,\eta_0^{\mathsf{T}} \cup \mathsf{Dom}\,\eta_1^{\mathsf{T}} \\ \hat{\theta}^{\mathsf{T}} : T \rightharpoonup \hat{T} &\mathrel{\hat{=}} t \mapsto t \qquad \text{for } t \in \hat{T} \\ \hat{\theta}_i^{\mathsf{T}} : \hat{T} \rightharpoonup T_i &\mathrel{\hat{=}} \eta_i^{\mathsf{T}} \restriction \hat{T}\ . \end{aligned}$$

Then $\hat{\theta}_0^{\mathsf{T}}, \hat{\theta}_1^{\mathsf{T}}, \hat{\theta}^{\mathsf{T}}$ is a target scaffold, hence can be extended to a candidate $\hat{\theta}_0, \hat{\theta}_1, \hat{\theta}$. Now we proceed directly to the result:

Theorem 6.27 (necessity of CHASTE for relative coproducts) If a pair of inclusion embeddings $\eta_i : G_i \rightarrowtail G$ $(i = 0, 1)$ does not satisfy CHASTE then no relative coproduct exists for η_0, η_1 in **G-Inc**.

Proof Suppose that the CHASTE condition does not hold, i.e. there are $s_i \in S_i^{in}$ and $t_i \in T_i^{out}$ such that $src_i(t_i) = s_i$, $\eta_0^{\mathsf{S}}(s_0) = \eta_1^{\mathsf{S}}(s_1)$, and $s_0 \not\equiv s_1$. Suppose for contradiction that there is a graph G_3 and a relative coproduct triple ψ_0, ψ_1, ψ such that $\psi : G_3 \rightarrowtail G$ and $\psi_i : G_i \rightarrowtail G_3$ for $i = 0, 1$. By the definition of relative coproduct, there exist two mediating inclusion embeddings: $\chi : G_3 \rightarrowtail G_2$ for the candidate $\theta_0, \theta_1, \theta$ and $\xi : G_3 \rightarrowtail \hat{G}_2$ for the candidate $\hat{\theta}_0, \hat{\theta}_1, \hat{\theta}$.

By construction, $\hat{\theta}^{\mathsf{T}}\,T = \hat{T}$, so by Lemma 6.17, $\hat{\theta}_0^{\mathsf{S}}\,s_0 \neq \hat{\theta}_1^{\mathsf{S}}\,s_1$, hence $\xi^{\mathsf{S}}\psi_0^{\mathsf{S}}(s_0) \neq \xi^{\mathsf{S}}\psi_1^{\mathsf{S}}(s_1)$, hence $\psi_0^{\mathsf{S}}(s_0) \neq \psi_1^{\mathsf{S}}(s_1)$. However, $(t_0, t_1) \in O_{0,1}$, so $\psi_i^{\mathsf{S}}(s_i) = \psi_i^{\mathsf{S}} src_i(t_i) = \psi_i^{\mathsf{S}} src_i \theta_i^{\mathsf{T}}(t_0, t_1) = \psi_i^{\mathsf{S}} src_i \psi_i^{\mathsf{T}} \chi^{\mathsf{T}}(t_0, t_1) = src_3 \chi^{\mathsf{T}}(t_0, t_1)$ for $i = 0, 1$, by INC-SRC for ψ_i; so $\psi_0^{\mathsf{S}}(s_0) = \psi_1^{\mathsf{S}}(s_1)$, a contradiction. ■

6.8 Existence of RPOs

We are finally ready to deduce the existence of RPOs in the category $\hat{\mathbf{C}}$**-Ixt** of contexts, which is the goal set at the beginning of this chapter. Recall that in $\hat{\mathbf{C}}$**-Ixt** a context $\boldsymbol{\varepsilon} \xrightarrow{G} (m, n, U)$, whose domain is the empty profile $\boldsymbol{\varepsilon}$, is actually a graph $G : (m, n)$ with $|G| = U$.

Corollary 6.28 Let $C_0G_0 = C_1G_1 = G$ in $\mathbf{\hat{C}\text{-}Ixt}$, where G_1 is OUTPUT-CONTROLLED. Then there exists an RPO for G_0, G_1 w.r.t. C_0, C_1.

Proof Let $\eta_i = \mathcal{D}_{G_i,G}(C_i)$, for $i = 0, 1$. Since G_1 is OUTPUT-CONTROLLED the pair η_0, η_1 satisfies property CHASTE by Proposition 6.13, hence possesses a relative coproduct by Theorem 6.26. The iso functor $\mathcal{A} : \mathbf{G\text{-}Inc} \to \varepsilon/\mathbf{\hat{C}\text{-}Ixt}$ transforms any such relative coproduct into a relative coproduct of $C_0 : G_0 \to G$ and $C_1 : G_1 \to G$, and hence, by Proposition 6.8, into an RPO in $\mathbf{\hat{C}\text{-}Ixt}$ for G_0, G_1 with respect to C_0, C_1. ■

Finally, note that if G is OUTPUT-CONTROLLED then $\rho \bullet G$ is too for any injection ρ with $\mathsf{Dom}\,\rho \supseteq |G|$. Thus it is well-defined to say that a context with domain $(0, 0)$ in $\mathbf{C\text{-}Ixt}$ (the category formed by quotienting out support translations) is OUTPUT-CONTROLLED. Thus:

Theorem 6.29 ($\mathcal{F}$ has all redex-RPOs) The functorial reactive system $\mathcal{F} : \mathbf{\hat{C}\text{-}Ixt} \to \mathbf{C\text{-}Ixt}$ has all redex-RPOs provided that every redex is OUTPUT-CONTROLLED, i.e. provided that for all $(l, r) \in \mathsf{Reacts}$, l is OUTPUT-CONTROLLED. ■

From this all of the congruence results of Chapter 3 (except the one for multi-hole contexts) are immediately applicable to $\mathcal{F} : \mathbf{\hat{C}\text{-}Ixt} \to \mathbf{C\text{-}Ixt}$, as desired. The next chapter argues that it is tolerable to ask that every redex satisfies certain reasonable constraints, including OUTPUT-CONTROLLED.

Chapter 7

Expected properties of redexes

In this chapter, I consider the question of which constraints it is reasonable to ask a redex to satisfy in a reactive system. The motivation for this comes from two directions, the first from the applied work in Chapter 6 and the second from the categorical work in Chapter 3.

Firstly, the previous chapter's main result (Theorem 6.29) asserted that $\mathcal{F} : \hat{\mathbf{C}}\textbf{-Ixt} \longrightarrow \mathbf{C}\textbf{-Ixt}$ has all redex-RPOs provided that every redex is OUTPUT-CONTROLLED. But is this a reasonable constraint? Can we point to redexes that are not OUTPUT-CONTROLLED and show that they induce undesirable reactions?

Secondly, as shown in Section 3.4, id-labelled transitions and reactions coincide when *redexes have epi preimages* (Definition 3.14). Is there a concrete (graph theoretic) characterisation of this requirement? Is this characterisation a reasonable constraint? What form do redexes that do not have such constrained preimages take? Do they lead to undesirable reactions?

The first motivation is the stronger one: without RPOs, the edifice of congruence results collapses. And yet, the latter is worth considering in order to understand the space of possible redexes. Delightfully, there *is* a brief concrete characterisation of when a redex has epi preimages and, moreover, this characterisation consists of three simple conditions, one of which is OUTPUT-CONTROLLED.

As a result, the OUTPUT-CONTROLLED condition is on the path to understanding epis, so I argue about the former when it comes up in the course of examining each of the three epi conditions.

Let us start then with the problem of finding a concrete characterisation for the epi conditions. Instantiating the property *redexes have epi preimages* for $\mathcal{F} : \hat{\mathbf{C}}\textbf{-Ixt} \longrightarrow \mathbf{C}\textbf{-Ixt}$ yields the following: For all $G \in \hat{\mathbf{C}}\textbf{-Ixt}$, if there exists $(l, r) \in \mathbf{C}\textbf{-Ixt}$ such that $\mathcal{F}(G) = l$, then G is an epi. (As usual we do not distinguish between an action graph G and arrows in $\hat{\mathbf{C}}\textbf{-Ixt}$ with domain ε.)

Notice that this definition does not require that redexes themselves be epis, just that

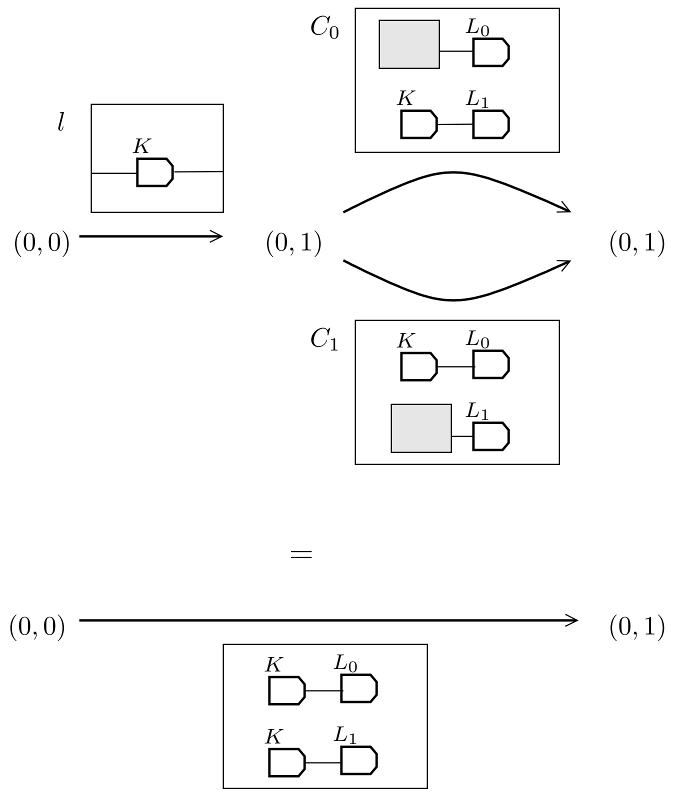


Figure 7.1: In **C-Ixt**, l is a not an epi

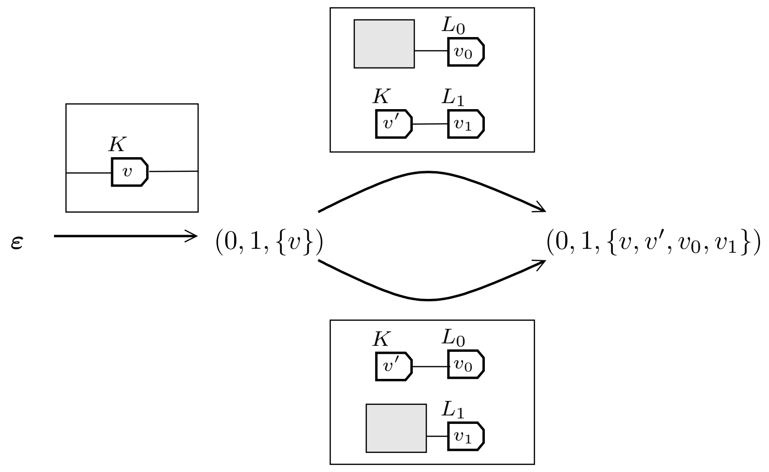


Figure 7.2: In $\hat{\mathbf{C}}$**-Ixt**, the tracking of node occurrences makes the left-hand graph an epi

their preimages be. To see why, consider two compositions in **C-Ixt** (Figure 7.1). The redex l containing a single control K is as simple and unobjectionable a redex as could be found. Yet it is not an epi in **C-Ixt** as shown: there are two contexts $C_0, C_1 \in$ **C-Ixt** for which $C_0 l = C_1 l$ but $C_0 \neq C_1$. (The latter follows because the hole is attached to L_i in C_i.)

However every preimage of l *is* an epi in $\hat{\mathbf{C}}$**-Ixt** (as proved in Proposition 7.3). Consider the situation in $\hat{\mathbf{C}}$**-Ixt** as shown in Figure 7.2. Then the two compositions are not equal because an arc links v to v_0 in the top one but to v_1 in the bottom.

Under which conditions is an action graph G an epi in $\hat{\mathbf{C}}$**-Ixt**? The answer is given by the following definition which characterises exactly (Proposition 7.3) the epis with domain $\boldsymbol{\varepsilon}$ of $\hat{\mathbf{C}}$**-Ixt**. The interesting part of the argument is carried out in terms of inclusion embeddings (Proposition 7.2) with the functors $\mathcal{A}$ and $\mathcal{D}$ being used to transfer constructions from **G-Inc** to $\hat{\mathbf{C}}$**-Ixt** and back. (This is reminiscent of the approach used to construct RPOs.)

Definition 7.1 (G has modest wiring) Say that an action graph G *has modest wiring* iff G satisfies the following properties:

$$src(T^{out}) \cap S^{in} = \varnothing \qquad \text{OUTPUT-CONTROLLED}$$

$$S^{in} \subseteq src(T) \qquad \text{NO-DISCARDED-INPUTS}$$

$$src \restriction T^{out} \text{ is injective .} \qquad \text{NO-CONTROLLED-FORKS}$$

■

The negation of each condition is illustrated by each G shown in Figure 7.6.

Note that each one of these conditions is concerned only with wiring and not nodes, so holds for G iff it holds for $\rho \bullet G$, where ρ is any injection with $\mathsf{Dom}\,\rho \supseteq |G|$. As was pointed out immediately after Corollary 6.28, it is thus well-defined to say that an agent, i.e. an arrow in **C-Ixt** with domain $(0, 0)$, has modest wiring. The reason is that an agent is an equivalence class of action graphs, all of which are a support translation of one another. Thus an agent is **C-Ixt** has modest wiring iff every (indeed any) preimage in $\hat{\mathbf{C}}$**-Ixt** out of $\boldsymbol{\varepsilon}$ has modest wiring.

Now, the interesting question is whether it is tolerable to demand that each redex has modest wiring. I believe yes, based on the absence of any example redex that does not have modest wiring. Of course, this is open to contradiction by future work. However, I now argue that the reactions obtained from a putative redex that does not have modest wiring are sufficiently strange that I feel safe in expunging such redexes for the moment. I consider each condition in turn. The first (OUTPUT-CONTROLLED) is indispensable for the existence of RPOs, as is explained at the beginning of this chapter:

OUTPUT-CONTROLLED: Consider the redex l in Figure 7.3 which does not satisfy OUTPUT-CONTROLLED because of the arc going across from an input source to an output

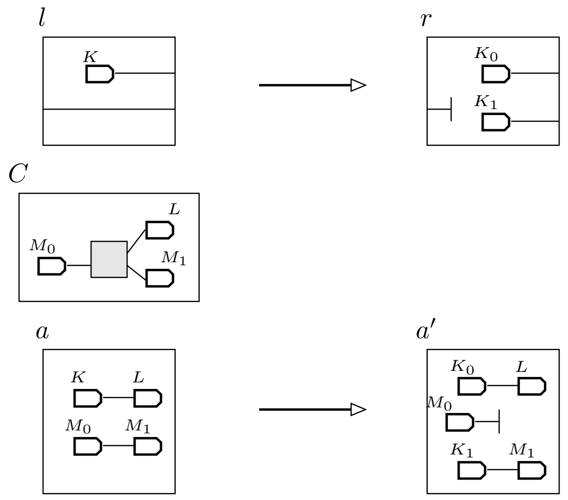


Figure 7.3: An undesirable reaction $a = Cl \longrightarrow\!\!\!\triangleright\, Cr = a'$ generated by a redex l that does not satisfy the condition OUTPUT-CONTROLLED

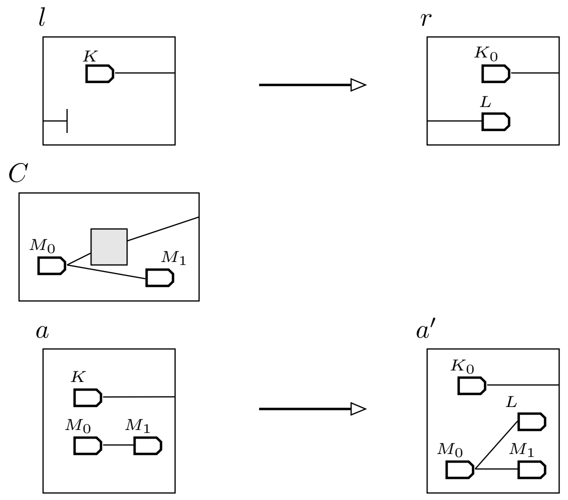


Figure 7.4: An undesirable reaction $a = Cl \longrightarrow\!\!\!\triangleright\, Cr = a'$ generated by a redex l that does not satisfy the condition NO-DISCARDED-INPUTS

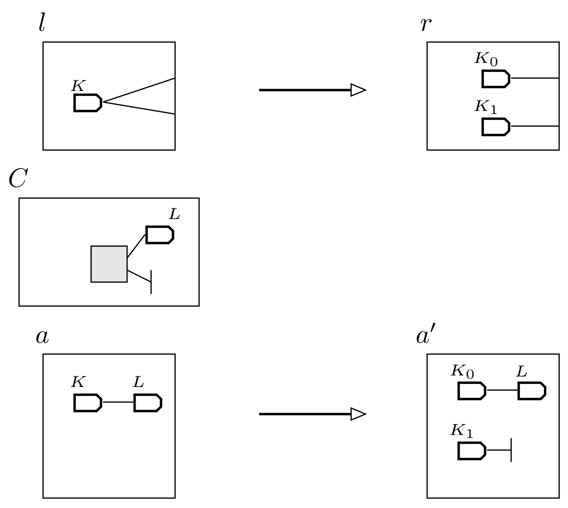


Figure 7.5: An undesirable reaction $a = Cl \longrightarrow\!\!\!\triangleright\, Cr = a'$ generated by a redex l that does not satisfy the condition NO-CONTROLLED-FORKS

target. This arc in l matches an arc connecting M_0 to M_1 in a (as witnessed by C). The induced reaction affects K (which seems reasonable given that K is in l), but also breaks M_0 from M_1 and connects a new control K_1 to M_1. The latter seem unreasonable because it is so highly nonlocal: the arc in l can match and then modify an arc *anywhere* in an agent, not necessarily near the site of K.

NO-DISCARDED-INPUTS: Consider the redex l in Figure 7.4 which does not satisfy NO-DISCARDED-INPUTS. It too has a nonlocal affect on a: the discarded input source of l is matched by the bound source of M_0 in a (as witnessed by C), resulting in a reaction that connects L to M_0.

NO-CONTROLLED-FORKS: Consider the redex l in Figure 7.5 which does not satisfy NO-CONTROLLED-FORKS. The bound source of K has a forked arc emanating from it. Despite this, the forked arc matches a non-forked arc in a (as witnessed by C) that connects K to L, resulting in a reaction that connects K_0 to L and leaves K_1 dangling.

Furthermore, in each of these cases, it seems that the redex could be replaced with a simpler one that does satisfy the condition in question. Consider the condition OUTPUT-CONTROLLED. Let (l, r) be a reaction rule whose redex l is not OUTPUT-CONTROLLED. Consider all arcs connecting an input source to an output target. If the contractum r possesses all these arcs, then it can be shown that the alternative rule (l', r') with all such arcs removed yields a superset of the reactions; the rule could be replaced by (l', r') for the purpose of deriving an LTS. If on the other hand r does not contain such an arc (as in Figure 7.3) then the rule (l, r) appears to generate non-deterministic behaviour of a kind not normally required.

I now prove that if G has modest wiring then G is an epi in $\hat{\mathbf{C}}$**-Ixt**. The first step is to show a related result for inclusion embeddings in **G-Inc** and then return to $\hat{\mathbf{C}}$**-Ixt** in Proposition 7.3.

Proposition 7.2 G has modest wiring iff for all inclusion embeddings $\eta_0, \eta_1 : G \rightarrowtail G_2$, we have that $\eta_0 = \eta_1$.

Proof

LHS implies RHS: Suppose that G has modest wiring. Let G_2 be an arbitrary graph and let $\eta_0, \eta_1 : G \rightarrowtail G_2$.

$\eta_0^{\mathsf{S}} = \eta_1^{\mathsf{S}}$**:** Consider $s \in S$. We distinguish two cases.

case $s \in S^{bind}$**:** By Inc-Bind, $\eta_0^{\mathsf{S}}(s) = \eta_1^{\mathsf{S}}(s)$.

case $s \in S^{in}$**:** By no-discarded-inputs and output-controlled, there exists $arg(v, j) \in T^{arg}$ such that $s = src\, arg(v, j) =^{\text{Inc-Arg}} src\, \eta_i^{\mathsf{T}}\, arg_2(v, j)$ for $i = 0, 1$. Thus

$$\eta_i^{\mathsf{S}}(s) = \eta_i^{\mathsf{S}}\, src\, \eta_i^{\mathsf{T}}\, arg_2(v, j) =^{\text{Inc-Src}} src_2\, arg_2(v, j)$$

for $i = 0, 1$, so $\eta_0^{\mathsf{S}}(s) = \eta_1^{\mathsf{S}}(s)$.

$\eta_0^{\mathsf{T}} = \eta_1^{\mathsf{T}}$**:** Consider $t \in \mathsf{Dom}\, \eta_0^{\mathsf{T}}$. We distinguish two cases:

case $\eta_0^{\mathsf{T}}(t) \in T^{arg}$**:** By Inc-Arg, $t \in \mathsf{Dom}\, \eta_1^{\mathsf{T}}$ and $\eta_0^{\mathsf{T}}(t) = \eta_1^{\mathsf{T}}(t)$.

case $\eta_0^{\mathsf{T}}(t) \in T^{out}$**:** By output-controlled, $src\, \eta_0^{\mathsf{T}}(t) \in S^{bind}$; then,

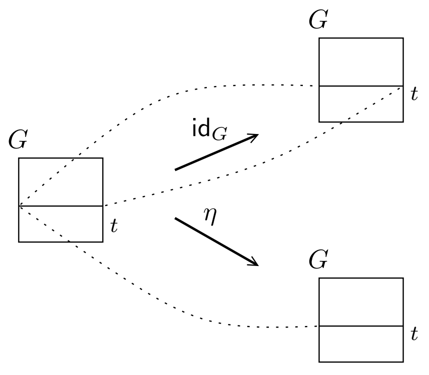
$$\eta_1^{\mathsf{S}}\, src\, \eta_0^{\mathsf{T}}(t) =^{\text{Inc-Bind}} \eta_0^{\mathsf{S}}\, src\, \eta_0^{\mathsf{T}}(t) =^{\text{Inc-Src}} src_2(t)\ .$$

By Inc-Targs, $t \in \mathsf{Dom}\, \eta_1^{\mathsf{T}}$. By Inc-Arg, $\eta_1^{\mathsf{T}}(t) \in T^{arg}$ implies $\eta_0^{\mathsf{T}}(t) \in T^{arg}$, which contradicts the case's hypothesis. Thus $\eta_1^{\mathsf{T}}(t) \in T^{out}$. By output-controlled, $src\, \eta_1^{\mathsf{T}}(t) \in S^{bind}$. By Inc-Src, $\eta_i^{\mathsf{S}}\, src\, \eta_i^{\mathsf{T}}(t) = src_2(t)$ for $i = 0, 1$, so by Inc-Bind, $src\, \eta_0^{\mathsf{T}}(t) = src\, \eta_1^{\mathsf{T}}(t)$. By no-controlled-forks, $\eta_0^{\mathsf{T}}(t) = \eta_1^{\mathsf{T}}(t)$.

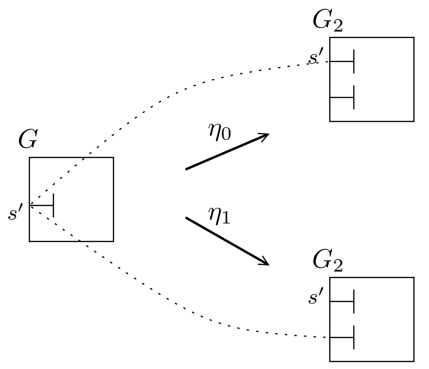
In both cases $\eta_0^{\mathsf{T}} \subseteq \eta_1^{\mathsf{T}}$. By symmetry, $\eta_0^{\mathsf{T}} = \eta_1^{\mathsf{T}}$.

RHS implies LHS (outline): Let G be a graph. We argue that each condition in Definition 7.1 is necessary in order for the RHS to be true.

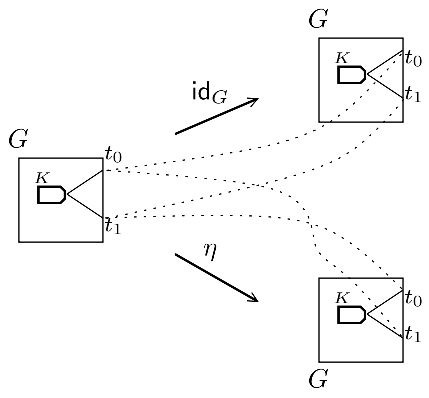
output-controlled**:** Suppose G does not satisfy output-controlled, i.e. there exists $t \in T^{out}$ such that $src(t) \in S^{in}$. We know that $\mathsf{id}_G : G \rightarrowtail G$ is one



7.6(1) Necessity of OUTPUT-CONTROLLED



7.6(2) Necessity of NO-DISCARDED-INPUTS



7.6(3) Necessity of NO-CONTROLLED-FORKS

Figure 7.6: Necessity of Definition 7.1

inclusion embedding. Let η be the same at id_G but with $t \mapsto t$ removed from η^{T} (a partial map). Then $\eta : G \rightarrowtail G$ is an inclusion embedding and $\mathsf{id}_G \neq \eta$. See Figure 7.6(1) for an example.

NO-DISCARDED-INPUTS: Suppose G does not satisfy NO-DISCARDED-INPUTS, i.e. there exists $s' \in S^{in}$ such that $s' \notin \mathit{src}(T)$. Suppose $G : (m, n)$. Let $G_2 : (m+1, n)$ be the same as G but with one extra discarded input source, i.e.

$$\begin{aligned} V_2 &\mathrel{\hat=} V \\ \mathit{contr}_2 &\mathrel{\hat=} \mathit{contr} \\ T_2 &\mathrel{\hat=} T \\ S_2 &\mathrel{\hat=} S \uplus \mathit{in}_2(m) \\ \mathit{src}_2 : T_2 \rightarrowtail S_2 &\mathrel{\hat=} t \mapsto \mathit{src}(t) \end{aligned}$$

Let $\eta_0 : G \rightarrowtail G_2$ be an inclusion embedding that is essentially the identity, i.e.

$$\begin{aligned} \eta_0^{\mathsf{S}} : S \rightarrowtail S_2 &\mathrel{\hat=} s \mapsto s \\ \eta_0^{\mathsf{T}} : T_2 \rightarrowtail T &\mathrel{\hat=} \mathsf{Id}_T \end{aligned}$$

Let $\eta_1 : G \rightarrowtail G_2$ be an inclusion embedding that differs only in its action on s':

$$\begin{aligned} \eta_1^{\mathsf{S}} : S \rightarrowtail S_2 &\mathrel{\hat=} \begin{cases} s \mapsto s & \text{if } s \neq s' \\ s \mapsto \mathit{in}_2(m) & \text{if } s = s' \end{cases} \\ \eta_1^{\mathsf{T}} : T_2 \rightarrowtail T &\mathrel{\hat=} \mathsf{Id}_T \end{aligned}$$

Then $\eta_0 \neq \eta_1$. See Figure 7.6(2) for an example.

NO-CONTROLLED-FORKS: Suppose G does not satisfy NO-CONTROLLED-FORKS, i.e. there exist $t_0, t_1 \in T^{out}$ such that $\mathit{src}(t_0) = \mathit{src}(t_1) \in S^{bind}$. We know that $\mathsf{id}_G : G \rightarrowtail G$ is one inclusion embedding. Let η be the same at id_G but with $\eta^{\mathsf{T}} : t_i \mapsto t_{1-i}$ for $i = 0, 1$. Then $\eta : G \rightarrowtail G$ is an inclusion embedding and $\mathsf{id}_G \neq \eta$. See Figure 7.6(3) for an example. ■

To make the following result easier to read, we take the liberty of omitting the usual decorations on arrows in $\hat{\mathbf{C}}$**-Ixt**.

Proposition 7.3 Let G be a context in $\hat{\mathbf{C}}$**-Ixt** with domain ε. Then G has modest wiring iff G is an epi in $\hat{\mathbf{C}}$**-Ixt**.

Proof Suppose G has modest wiring and that $G_2 \mathrel{\hat=} C_0 G = C_1 G$ in $\hat{\mathbf{C}}$**-Ixt** for some C_0, C_1. By Proposition 7.2, $\mathcal{D}_{G,G_2}(C_0) = \mathcal{D}_{G,G_2}(C_1) \in \mathbf{G\text{-}Inc}(G, G_2)$, so

$$C_0 = \mathcal{A}(\mathcal{D}_{G,G_2}(C_0)) = \mathcal{A}(\mathcal{D}_{G,G_2}(C_1)) = C_1$$

since $\mathcal{A}$ inverts $\mathcal{D}$.

Suppose that G does not have modest wiring. Then by Proposition 7.2, there exist a graph G_2 and $\eta_0, \eta_1 \in \mathbf{G\text{-}Inc}(G, G_2)$ such that $\eta_0 \neq \eta_1$. Then $\mathcal{A}(\eta_0) \neq \mathcal{A}(\eta_1)$ since $\mathcal{A}$ is faithful but $\mathcal{A}(\eta_0)G = G_2 = \mathcal{A}(\eta_1)G$. Thus G is not an epi in $\hat{\mathbf{C}}\text{-}\mathbf{Ixt}$. ■

This result shows that Definition 7.1 provides an exact characterisation of epis in $\hat{\mathbf{C}}\text{-}\mathbf{Ixt}$, whence we have the following, which rounds out the chapter:

Corollary 7.4 For $\mathcal{F} : \hat{\mathbf{C}}\text{-}\mathbf{Ixt} \to \mathbf{C}\text{-}\mathbf{Ixt}$, if every redex has modest wiring (Definition 7.1) then redexes have epi preimages (Definition 3.14). ■

The implications of this chapter are not that epis are important *per se* but that there is a space of possible constraints that may be imposed upon redexes. One can imagine requiring all the nodes in a redex to be connected by wiring. This would guarantee that the parts of the redex occur locally in an agent, precluding far-flung instantiations. It would be enlightening if this or other graph-theoretic constraints could be described categorically. Conversely, as we investigate richer graph structure (such as nesting, free names, binding, etc.) we can look specifically for which redexes satisfy known categorical properties, such as was done here for epi preimages.

If we can build up a library of useful categorical constraints, then it may be possible to provide a way of classifying redexes from different functorial reactive systems according to these constraints. An application of this might be a new way of disproving the existence of encodings between process calculi represented as functorial reactive systems: if one could show that any putative encoding preserves certain categorical properties that hold of the redexes in the source of the encoding but not of the redexes in the target, then one has a handle on disproving the encoding's existence.

It remains to be seen whether such classifications are possible or useful. Yet, it seems like a fruitful direction: the power of functorial reactive systems is they are abstract and thus provide a general setting in which to derive operational congruences. There is no reason why this power cannot be exploited for other ends as well, such as the classification of dynamics. This, however, is future work.

Chapter 8

Conclusions and future work

This final chapter summarises some of the accomplishments of the dissertation and points to areas that require further work.

The main motivation for my research is to provide mathematical tools that ease the construction of new process calculi. Every computational phenomenon (distributed hosts, failures, cryptography, etc.) that gives rise to a new calculus carries with it rules for describing how it works, how a computation evolves. These are often reaction rules, which capture the allowable state changes. The redexes are typically *composites* formed from atoms, so the following questions naturally arise. When an agent has parts of a redex, which atoms does the agent require to form a complete redex? Can we get a definition of labelled transition if we choose the labels to be those small *contexts* that complete redexes? These questions are not original with me: they are often used by the process calculus community as motivation and intuition for designing specific labelled transition relations. I believe, though, that the original contribution of my work is to give a general way of transforming reaction rules into tractable labelled transitions.

By tractable, I mean two things: (i) the labels are small, i.e. contain *only* those parts of a redex necessary to complete a whole redex in an agent; (ii) the labelled transitions yield operational equivalences and preorders that are congruences. The key ideas in trying to achieve both desiderata are that of a *relative pushout* (RPO) and an *idem pushout* (IPO). With respect to (ii) the results are good: I prove congruence for strong bisimulation, weak bisimulation, the traces preorder, and the failures preorder. For (i), the results are encouraging: IPOs ensure that the labels of transitions contain no junk, i.e. that no lesser label would do. There is, however, more work required in making the labels even simpler by exploiting the uniformity present in reaction rules that contain metavariables. This is discussed below in the paragraphs concerned with multi-hole redexes.

One of the most attractive qualities of the theory of RPOs is that it is *abstract*: it is applicable to all categories of contexts and therefore provides a unifying discipline. Before I understood RPOs, I spent several years thinking about *dissection results* for specific graph-like contexts: "If $C_0 a_0 = C_1 a_1$, then what part of a_i is present in C_{1-i} for $i = 0, 1$?" I could only get results about nodes (for example, which nodes of a_1 are supplied by C_0) but not about arcs. Trying to work in an *ad hoc* way with graphs was unsatisfying until the RPO theory arrived: it was only the discipline of a universal property in the category theory sense that sustained me in carrying out the constructions contained in Chapter 6. Sewell's *ad hoc* reasoning about dissection for term algebras with parallel composition [Sew01] was successful without employing RPOs, but his statement of dissection was complicated. To be fair, his dissections were for multi-hole redexes that RPOs do not neatly handle. I say more on this below. He is now recasting some of his results in terms of RPOs and other universal constructions.

RPOs thus support the goal of developing a shared theory that can produce labelled transitions and operational congruences for many process calculi. The task that we usually face for each new process calculus of defining labelled transitions and proving that equivalences and preorders are congruences is replaced by the task of choosing reactions and proving that RPOs exist. This is attractive leverage gained from RPOs.

Nonetheless, constructing RPOs is difficult, as witnessed by the daunting complexity in Chapter 6. It is therefore desirable that it not be done afresh for each new process calculus. This is the impetus for the leading example in this dissertation, namely the construction of RPOs for action calculi-like contexts. Recall that the functorial reactive system $\mathcal{F} : \hat{\mathbf{C}}\textbf{-Ixt} \to \mathbf{C}\textbf{-Ixt}$ is really a family with varying choices of control signature $\mathcal{K}$ and of reaction rules Reacts. The fragment of action calculi considered in Chapter 5 is thus a *framework*: *if* one can express states of a computational model using controls (for atoms) and arcs (for names and sharing) and reactions in terms of redexes that satisfy the condition OUTPUT-CONTROLLED *then* RPOs exist and hence Chapter 3 gives labelled transitions and operational congruences.

A framework is often deficient for two reasons: (i) it provides insufficient benefits to compensate for the effort of casting a specific example in the framework's form; (ii) it is not rich enough to cover all the phenomena that one wants. Criticism (ii) hits home. The fragment of action calculi presented in this dissertation is inadequate for most purposes: there is no nesting of action graphs (needed for prefixing), no free names, no binding, limited reflexion, no choice operator, and no multi-hole redexes (needed to represent faithfully metavariables in reaction rules). I describe future work below that addresses these issues. Criticism (i), however, is unjustified when applied to the action calculi shown here: getting "for free" a labelled transition system and a collection of operational congruences is a worthwhile benefit.

The reverse situation held for action calculi as originally presented by Milner [Mil96]: criticism (ii) was not such a problem but criticism (i) was: up until now there has been little tangible reward to using action calculi to present specific examples of process calculi.

The central goal of future work is to construct a framework with rich enough structure

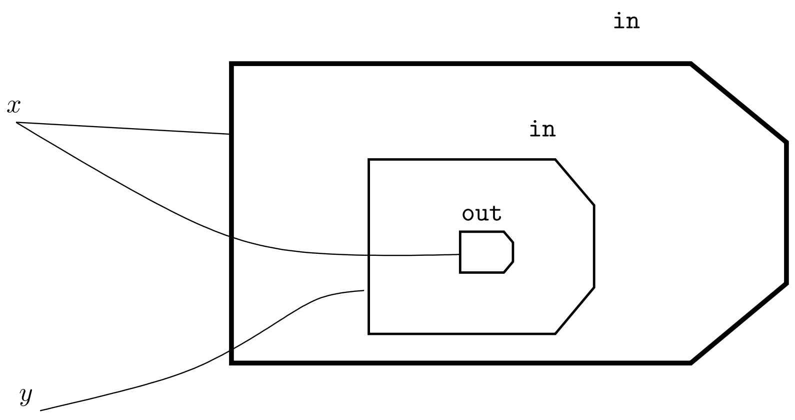


Figure 8.1: Nesting of nodes

(perhaps like Milner's original action calculi) to dodge criticism (ii) while still providing the rewards of labelled transitions and operational congruences.

To this end, it will be important to pursue several research directions:

nesting: The nesting of agents inside nodes is essential for representing prefixing. For example, in CCS the agent $x.y.\bar{x}$ contains three levels of nesting with the name x shared by the outermost and the innermost. In order to represent this with graph-like constructions, we need that each node packages up the continuation agent, e.g. as in Figure 8.1 which gives a possible representation of the CCS agent above.

At first glance, nesting is a trivial thing: term algebras, for instance, consists of just pure nesting structure with nothing else. It is straightforward to calculate RPOs for them. However the real challenge is the combination of wiring and nesting: Figure 8.1 has an arc sourced by x that is forked and has targets at multiple levels. In particular, even the notion of context composition is difficult: when composing contexts that fork an arc at different nesting levels, the forks have to be normalised in some way, either by pushing them in to the innermost level or pulling them out to the outermost.

As a stepping stone to handling nesting with rich wiring, Milner is currently looking at a simpler situation, namely the enhancement with nesting of work we did on *linear graphs* [LM00b]. In this case the normalisation of forked arcs is not an issue since arcs cannot be forked or discarded.

There are many possible representations for nested graphs. One that seems most promising comes out of the work by Milner referred to here: the idea is to treat a nested graph as a flat graph with an added parent function overlaid. It seems that RPOs can then be calculated by handling the wiring structure and the nesting structure orthogonally. Furthermore, this orthogonality appears to be robust when passing to more complex wiring.

Even without the full power of nesting, it may be possible to handle the π-calculus (or variants thereof) indirectly. Honda and Yoshida [HY94a] give a translation in terms of "concurrent combinators"; these encode prefixing in a flat structure by using triggers to point to parts of agents that may execute. A derived labelled transition relation for their combinator reaction rules might be too intensional since the labels could detect exactly the combinators present in an agent. It is worth trying this, though, to see what kind of equivalences would transpire.

free names and binding: It is straightforward to add free names to graphs: the idea is to enrich the set of source ports to include names. It is more subtle though to make precise how a context can *bind* the names of the thing that goes in its hole. It seems likely that the objects of the category of contexts need to contain name sets to ensure that RPOs exist (analogously to inclusion of node sets in $\hat{\mathbf{C}}$**-Ixt**), i.e. a context is an arrow of the form $(m, n, U, X) \rightarrow (m', n', U', X')$, meaning that its hole is expecting a context with support (nodes) U and names X. Nonetheless, it is not clear whether the downstairs category (the analogue of **C-Ixt**) should include these name sets. For example, in π-calculus we write the binding context $(\nu x)-$ without saying which names can fit in the hole. Perhaps the functor from upstairs to downstairs (like $\mathcal{F} : \hat{\mathbf{C}}\text{-}\mathbf{Ixt} \rightarrow \mathbf{C}\text{-}\mathbf{Ixt}$ in this dissertation) will allow two kinds of IPO sliding: the first with respect to nodes (as I have shown here) and the second with respect to names.

There has been recent work on modelling names and name binding in a categorical way [TFP99, GP99] and it would be exciting to join together those syntactic models with operational semantics, and in particular, the reactive systems shown here.

Name binding carries with it the problem of *scope extrusion* in labelled transitions. In π-calculus, for example, it is a subtle problem to handle the labelled transition of $(\nu x)(\bar{y}\langle x\rangle)$. There are many ways of handling this when looking at bisimulation, for example, but most involve augmenting the definition with freshness side conditions to cater explicitly for extrusion. An alternate approach [CS00] keeps track of extruded names as annotations of the agents themselves, thus gaining the power of extrusion but keeping the simplicity of bisimulation without added conditions. Adding these annotations seems almost identical to adding name sets in the objects of a context category (as outlined above).

multi-hole redexes: The basic reaction rule of the π-calculus is

$$\bar{x}\langle y\rangle.a \mid x(z).b \quad \longrightarrow \quad a \mid \{y/z\}b \; . \tag{8.1}$$

Imagine for a moment that the questions of nesting and free names are overcome. There is still a problem adequately representing (8.1) in a reactive system. Currently I require that the reaction rules Reacts consist of pairs of *agents*, not *agent contexts*. As a result, the only way to represent (8.1) is via an infinite family of rules, one for each possible instantiation of a and b. Thus there might be labelled transitions of the form

$$\bar{x}\langle y \rangle \xrightarrow{-|x(z).b} \tag{8.2}$$

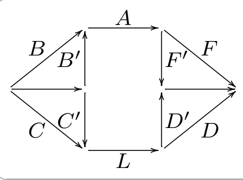
for all instantiations of b. But these are ungainly labels: the only important part of any label of the form $- \mid x(z).b$ is the top-level structure, namely $x(z)$; the b is irrelevant.

How can we winnow down the family of labels to a single one? The answer lies in using the uniformity present in (8.1). The a and b there are really metavariables and the rule can be expressed thus:

$$\bar{x}\langle y \rangle.-_1 \mid x(z).-_2 \quad \longrightarrow \quad -_1 \mid \{y/z\}-_2 \ . \tag{8.3}$$

This rule consists of a pair of 2-hole contexts which capture the uniformity present in it. If we relax the condition on Reacts and allow it to contain pairs of contexts (L, R) for which $L, R : m \rightarrow n$ are arbitrary contexts, then it is easy to generate the reaction relation $\longrightarrow$ (forgetting about the details of $\mathbf{D}$)

$$A \longrightarrow A' \quad \text{iff} \quad (\exists (L, R) \in \mathsf{Reacts}, C, D.\ A = DLC \ \&\ A' = DRC) \ .$$



8.2

Contrast this to Definition 2.2 for which the reaction rules are closed-up under context application on the outside, not also on the inside (as shown here). The hard problem is to define labelled transitions $A \xrightarrow{F} A'$. We cannot rely on IPOs anymore since the domain of A is potentially different from the domain of any redex L. A possible replacement for IPOs might be via *hexagon idem pushouts* (which are special kinds of *hexagon relative pushouts*) as suggested by Sewell. Notice how L and A are wrapped up in Figure 8.2 by arrows composed on either side. The hexagon idem pushout property guarantees that there is no lesser hexagon bounded by B', C', F', D'. It seems plausible that such hexagons can yield the single labelled transition $\bar{x}\langle y \rangle \xrightarrow{-_1|x(z).-_2}$ from (8.3), which is lighter than having the infinite family shown in (8.2).

In [Sew01], Sewell already exploited the uniformity present in reaction rules when looking at term algebras with parallel composition. He achieved lightweight labelled transitions, but made use of complex context dissection lemmas. He is now looking at the problem of recasting his results in terms of universal constructions (such as hexagons). In conclusion, there are two important lines of research related to multi-hole redexes which mirror what was done in my dissertation: (i) defining labelled transitions by universal constructions and proving congruence for a variety of equivalences and preorders; (ii) showing that the relevant universal constructions exist in categories of interest (term algebras, graphs, etc.).

sum: As discussed in Section 3.5, the proper treatment of summation (such as occurs in CCS and the π-calculus) requires care. It is not good enough for sum to be a free operator (just another control) if we want to describe its reactive behaviour, e.g.

$$\begin{aligned} (\bar{x}.a + b) \mid (x.a' + b') &\quad \longrightarrow\!\!\!\!\triangleright \quad a \mid a' \\ \tau.a + b &\quad \longrightarrow\!\!\!\!\triangleright \quad a\ . \end{aligned}$$

Consider the second reaction rule. In order for the sum to be rearranged so that $\tau.a$ is on the left and the rest is on the right, we require $+$ to be commutative, associative, and have the empty agent as an identity. I do not understand how to represent this in graphs.

The same problem surfaces for other operators that change the structural congruence (syntactic equality). In the π-calculus, for example, replication is handled by the axiom

$$\boldsymbol{!}\,a \equiv a \mid \boldsymbol{!}\,a\ . \tag{8.4}$$

This seems even harder to represent in terms of graphs since the RHS has, potentially, more nodes than the LHS. Even if the problems of finding graph theoretic representation of sum and replication can be overcome, it may be difficult to construct RPOs. A possible solution, at least for replication, is to disallow the structural equality in (8.4), and instead confine replication to be *input guarded* and to have a reaction rule of the form

$$\bar{x}\langle y\rangle \mid \boldsymbol{!}\,x(z).a \quad \longrightarrow\!\!\!\!\triangleright \quad \{y/z\}a \mid \boldsymbol{!}\,x(z).a\ .$$

This is a commonly done in asynchronous variations of π-calculus [HT91].

Another way of handling replication is via an encoding in terms of combinators (already mentioned above) [HY94b]. For input guarded summation, encoding [NP96] is also an option.

full reflexion: The contexts considered in Chapter 5 have only a limited form of reflexion, as enforced by the condition LOOSE that prevents tight loops linking a hole to itself. This simplifying assumption makes the composition of contexts and the composition of the associated embeddings easy. With full reflexion, the composition of contexts can create arcs that are formed from arbitrarily many segments of arcs present in the inner and outer context. Indeed, it is difficult to prove even that the composition of reflexive embeddings is associative [Lei01]. Better technology for handling reflexion has been developed in [LM00b] for graphs with linear wiring. Proving the existence of RPOs for full reflexion with non-linear wiring represents a future challenge.

inductive presentation of labelled transitions: This item is not related to computational phenomena, but rather the form in which derived labelled transitions are presented.

In this dissertation, the way of deriving labelled transitions from reactions yields a direct characterisation. This is in contrast to the inductive presentation usually found in the process calculus literature. Can the gap be bridged, though? In other words, can an inductive presentation of labelled transitions be derived from reaction rules? It seems plausible. For any particular reactive system whose syntax is generated from some constructors, one could instantiate Lemma 2.19 (recalled here) by substituting each constructor for C:

$$\dfrac{\begin{tikzcd} {} \arrow[r, "C"] \arrow[d, "F'"'] & {} \arrow[d, "F"] \\ {} \arrow[r, "C'"'] & {} \end{tikzcd} \text{ is an IPO} \qquad a \xrightarrow{F'} a'' \qquad C' \in \mathbf{D}}{Ca \xrightarrow{F} C'a''} \; .$$

There are several interesting questions that follow: Under what conditions can we derive an inductive presentation that generates precisely the same labelled transition relation as the direct presentation gives? Under what conditions would an inductive presentation satisfy any of the GSOS rule formats [GV92, Blo93]? If some set of the latter is satisfied, it would be enlightening to compare two different proofs of congruence for strong bisimulation, say — one based on the RPO theory shown in this dissertation and the other based on GSOS reasoning, particularly as provided by recent categorical treatments of the latter [TP97].

It is not surprising that some of these areas (e.g. free names and multi-hole redexes) require changes to the categorical abstractions of a reactive system, not just cleverer ways of constructing RPOs. This pattern is well-established in my dissertation. I started with reactive systems, then passed to functorial reactive systems when it became clear that RPOs needed to be calculated in a separate category, then considered functorial monoidal reactive system to cater explicitly for RPOs resulting in multi-hole contexts.

A test of the future work outlined above is to see what labelled transitions and operational congruences are generated for applications such as CCS, π-calculus, λ-calculus, and ambients. It would be exciting (though not likely) to get congruences that coincide exactly with well-known ones (open bisimulation for π-calculus, say). It should not be a cause for dejection if the derived congruences do not match exactly the historically established ones: π-calculus comes with a zoo of bespoke congruences, none of which is good for all applications.

The best outcome of this work will be if the mathematical tools that are proposed here ease the path for exploring new primitives and new patterns of behaviour inherent in distributed computing. The development of mathematical techniques for reasoning about hosts and host failure, agent mobility, and cryptography, for example, is critical for guiding infrastructure and language design. Proofs techniques for these models are critical for precise reasoning. Much work needs to be done before the tools of this dissertation are good enough to address these applications. Nonetheless, I see no limits as to what can be so achieved.

Appendix A

Review of category theory

I summarise here the main ideas of category theory used in this dissertation. For greater detail, the reader is referred to the classic work by Mac Lane [Mac71]. Another good reference is [BW01] (which is online).

Categories A category $\mathbf{C}$ consists of *objects* $X, Y, \ldots$ and of *arrows* $f, g, \ldots$ between objects. An arrow f has a *domain* object $\mathsf{Dom}\, f$ and a *codomain* object $\mathsf{Cod}\, f$. We write $f : X \to Y$ if $\mathsf{Dom}\, f = X$ and $\mathsf{Cod}\, f = Y$. The arrows from X to Y are a *homset*, denoted $\mathbf{C}(X, Y)$. The objects of $\mathbf{C}$ are denoted $\mathsf{obj}\, \mathbf{C}$. *Composition* is defined on all pairs of arrows between appropriate objects:

$$\frac{f : X \to Y \qquad g : Y \to Z}{gf : X \to Z}$$

and is associative: $h(gf) = (hg)f$. Every object X has a (unique) *identity* arrow $\mathsf{id}_X : X \to X$; for $f : X \to Y$, we have $\mathsf{id}_Y f = f = f \mathsf{id}_X$.

Special arrows An arrow $f : X \to Y$ is an *iso* iff there exists an arrow $g : Y \to X$ such that $gf = \mathsf{id}_X$ and $fg = \mathsf{id}_Y$. If $g' : Y \to X$ has the same properties as g then $g = g\mathsf{id}_Y = gfg' = \mathsf{id}_X g' = g'$, so the *inverse* of f is unique and is often written f^{-1}.

An arrow f is an *epi* iff it is right-cancellable, i.e. $g_0 f = g_1 f$ implies $g_0 = g_1$. A *split epi* f is an arrow with a right-inverse, i.e. there exists h such that $fh = \mathsf{id}$.

An arrow f is a *mono* iff it is left-cancellable, i.e. $fg_0 = fg_1$ implies $g_0 = g_1$. A *split mono* (also known as a *retraction*) f is an arrow with a left-inverse, i.e. there exists h such that $hf = \mathsf{id}$.

Subcategories A subcategory $\mathbf{D}$ of $\mathbf{C}$ is a category all of whose objects and arrows are objects and arrows of $\mathbf{C}$ and whose identities and compositions are the same as in $\mathbf{C}$.

Furthermore, $\mathbf{D}$ is a *full subcategory* if every homset in $\mathbf{D}$ is equal to the corresponding homset in $\mathbf{C}$, i.e. for all $X, Y \in \mathsf{obj}\,\mathbf{D}$, $\mathbf{D}(X,Y) = \mathbf{C}(X,Y)$. (Thus a full subcategory is determined entirely by its objects.)

Commutation The composition of arrows is depicted by concatenating them. For example, if $f : X \to Y$ and $g : Y \to Z$ then gf is shown as $X \xrightarrow{f} Y \xrightarrow{g} Z$ Often the object labels are omitted. A square *commutes* if every path with common start and end is equal. For example, the following square commutes iff $g_0 f_0 = g_1 f_1$:

$$\begin{tikzcd}
{} \arrow[r, "f_0"] \arrow[d, "f_1"'] & {} \arrow[d, "g_0"] \\
{} \arrow[r, "g_1"'] & {}
\end{tikzcd}$$

Coproducts Given two objects X_0, X_1, their *coproduct* consists of an object X (sometimes written $X_0 + X_1$) and a pair of arrows $f_i : X_i \to X$, $i = 0, 1$, so that for any other pair $f'_i : X_i \to X'$, there exists a unique mediating arrow $k : X \to X'$ (sometimes written $[f'_0, f'_1]$) such that the two triangles below commute:

$$\begin{tikzcd}
X_0 \arrow[d, "f_0"'] \arrow[dr, "f'_0"] & \\
X \arrow[r, dotted, "k"] & X' \\
X_1 \arrow[u, "f_1"] \arrow[ur, "f'_1"'] &
\end{tikzcd} .$$

Universal constructions and isos Coproducts are *unique up to isomorphism* in the following sense: if f_0, f_1 and f'_0, f'_1 are both coproducts then the k shown above is an iso. Conversely, if k is an iso and f_0, f_1 is a coproduct then f'_0, f'_1 is too.

This robustness with respect to isos is not accidental and is not limited to coproducts. In general, *universal constructions*, like coproducts, pushouts (see below), and relative pushouts (see Definition 2.4) are unique up to isomorphism. Moreover, if one replaces any object Y in such a construction with, say, Y' for which there is an iso $k : Y \to Y'$ and one composes k with all the arrows in the construction whose codomain is Y and composes k^{-1} with all the arrows whose domain is Y, then the diagram is still a universal construction. For example, we have the following property about IPOs (Definition 2.5):

$$\begin{tikzcd}
{} \arrow[r, "f_0"] \arrow[d, "f_1"'] & Y \arrow[d, "g_0"] \\
{} \arrow[r, "g_1"'] & {}
\end{tikzcd} \text{ is an IPO iff } \begin{tikzcd}
{} \arrow[r, "kf_0"] \arrow[d, "f_1"'] & Y' \arrow[d, "g_0 k^{-1}"] \\
{} \arrow[r, "g_1"'] & {}
\end{tikzcd} \text{ is too.}$$

Pushouts Give two arrows f_0, f_1 with common domain, a *pushout* is a pair g_0, g_1 such that $g_0 f_0 = g_1 f_1$ and for any other pair g'_0, g'_1 satisfying $g'_0 f_0 = g'_1 f_1$, there exists a unique k such that $k g_i = g'_i$ for $i = 0, 1$:

$$\begin{tikzcd}
{} \arrow[r, "f_0"] \arrow[d, "f_1"'] & {} \arrow[d, "g_0"] \arrow[ddr, bend left, "g'_0"] & \\
{} \arrow[r, "g_1"'] \arrow[drr, bend right, "g'_1"'] & {} \arrow[dr, dotted, "k"] & \\
& & {}
\end{tikzcd}$$

Slice and coslice categories Given a category $\mathbf{C}$ and an object of X, the *slice category* $\mathbf{C}/X$ is defined as follows. An object of $\mathbf{C}/X$ is an arrow in $\mathbf{C}$ with codomain X; an arrow $h : f \to g$ of $\mathbf{C}/X$ is an arrow of $\mathbf{C}$ such that $gh = f$:

$$\begin{tikzcd}
{} \arrow[rr, "h"] \arrow[dr, "f"'] & & {} \arrow[dl, "g"] \\
& X &
\end{tikzcd}$$

Composition in $\mathbf{C}/X$ is just given by composition in $\mathbf{C}$. Finally, id_f in $\mathbf{C}/X$ is $\mathsf{id}_{\mathsf{Cod}\, f}$ in $\mathbf{C}$.

The *coslice category* $X/\mathbf{C}$ is the dual notion: an object in $X/\mathbf{C}$ is an an arrow in $\mathbf{C}$ with domain X; an arrow $h : f \to g$ of $X/\mathbf{C}$ is an arrow of $\mathbf{C}$ such that $hf = g$:

$$\begin{tikzcd}
& X \arrow[dl, "f"'] \arrow[dr, "g"] & \\
{} \arrow[rr, "h"'] & & {}
\end{tikzcd}$$

Composition in $X/\mathbf{C}$ is just given by composition in $\mathbf{C}$. Finally, id_f in $X/\mathbf{C}$ is $\mathsf{id}_{\mathsf{Dom}\, f}$ in $\mathbf{C}$.

Strict monoidal categories A strict monoidal category $\mathbf{C}$ has a *tensor* $\otimes$, which is a binary operation on objects and arrows, and a *unit* object U. On objects, the tensor is associative and has identity U. On arrows, the tensor satisfies the following inference rule:

$$\frac{f_0 : X_0 \to Y_0 \qquad f_1 : X_1 \to Y_1}{f_0 \otimes f_1 : X_0 \otimes X_1 \to Y_0 \otimes Y_1}$$

and is associative and with identity id_U. Furthermore, compositions and tensors can be rearranged as follows,

$$(g_0 \otimes g_1)(f_0 \otimes f_1) = (g_0 f_0) \otimes (g_1 f_1)$$

provided that the arrows match up appropriately: $f_i : X_i \to Y_i$ and $g_i : Y_i \to Z_i$ for $i = 0, 1$.

Functors A *functor* $F : \mathbf{C}_0 \to \mathbf{C}_1$ maps objects and arrows of category $\mathbf{C}_0$ to those of $\mathbf{C}_1$, such that the following properties hold:

$$\frac{f : X \to Y}{F(f) : F(X) \to F(Y)} \qquad F(gf) = F(g)F(f) \qquad F(\mathsf{id}_X) = \mathsf{id}_{F(X)}$$

We say that F is *faithful* if it is injective on arrows; F is *full* if it maps surjectively the homeset $\mathbf{C}_0(X, Y)$ onto $\mathbf{C}_1(F(X), F(Y))$ for all objects X, Y in $\mathbf{C}_0$.

Appendix B

Labelled transitions via retractions

As promised in Section 3.4, this appendix presents a new definition of labelled transition, which is denoted by $\xrightarrow[\mathsf{r}]{\cdot}$. This definition has two properties: (i) it recovers the reaction relation, i.e. $\xrightarrow{\mathsf{id}} = \longrightarrow$; (ii) it does not involve case analysis (unlike $\xrightarrow[\mathsf{c}]{\cdot}$, see Definition 3.10).

The key idea is to make use of *retractions*, otherwise knows as *split monos* (see p. 22 in [BW01]). We say that an arrow $\bar{f}$ is a retraction if it has a left inverse, i.e. if there exists an arrow f such that $f\bar{f} = \mathsf{id}$.

The development follows closely that of Section 3.4: I present the definition of $\xrightarrow[\mathsf{r}]{\cdot}$; show how to recover the reaction relation; prove cutting and pasting results; prove that the induced strong bisimulation $\sim_{\mathsf{r}}$ is a congruence; and, finally, compare $\sim_{\mathsf{r}}$ to $\sim_{\mathsf{c}}$.

These results (specifically Lemma B.4 and Proposition B.6) assume an added property of functorial reactive systems not included in Definition 3.1: If $\mathcal{F}$ is a functorial reactive system then $\mathcal{F}$ *creates left inverses*, i.e. if $\mathsf{id} = C_1\mathcal{F}(\mathtt{C}_0)$ then there exists $\mathtt{C}_1 \in \hat{\mathbf{C}}$ such that $\mathsf{id} = \mathtt{C}_1\mathtt{C}_0$ and $\mathcal{F}(\mathtt{C}_1) = C_1$. This added property holds for all functors constructed in the form shown in Chapter 4, as proved in Theorem 4.14. As a result, the functor $\mathcal{F} : \hat{\mathbf{C}}\textbf{-Ixt} \to \mathbf{C}\textbf{-Ixt}$ (Chapter 5) creates left inverses, so the results presented here are applicable to graph contexts.

Definition B.1 (labelled transition by retractions; cf. Definition 3.10) $a \xrightarrow[\mathsf{r}]{F} a'$ iff there exists $\mathtt{a}, \mathtt{l}, \mathtt{F}, \mathtt{D}, \mathtt{R}, \bar{\mathtt{R}} \in \hat{\mathbf{C}}$ and $r \in \mathbf{C}$ such that Figure B.1 is an IPO in $\hat{\mathbf{C}}$ and

$$\begin{tikzcd} \cdot \arrow[r, "\mathtt{a}"] \arrow[d, "\mathtt{l}"'] & \cdot \arrow[d, "\bar{\mathtt{R}}\mathtt{F}"] \\ \cdot \arrow[r, "\mathtt{D}"'] & \cdot \end{tikzcd}$$
B.1

$$a' = \mathcal{F}(\mathtt{RD})r \qquad \mathcal{F}(\mathtt{D}) \in \mathbf{D} \qquad (\mathcal{F}(\mathtt{l}), r) \in \mathsf{Reacts} \qquad \mathtt{R}\bar{\mathtt{R}} = \mathsf{id}$$
$$\mathcal{F}(\mathtt{a}) = a \qquad \mathcal{F}(\mathtt{F}) = F\ .$$
■

It follows immediately from this definition that $\xrightarrow[\mathsf{r}]{\mathsf{id}} \subseteq \longrightarrow$. The interesting question is the converse:

Proposition B.2 (recovery of the reaction relation for $\xrightarrow[\mathsf{r}]{\cdot}$) Suppose $\mathcal{F} : \hat{\mathbf{C}} \to \mathbf{C}$ is a functorial reactive system and has all redex-RPOs. Then $a \longrightarrow a'$ implies $a \xrightarrow[\mathsf{r}]{\mathsf{id}} a'$.

Proof By Proposition 3.2, there exist $\mathtt{a}, \mathtt{l}, \mathtt{D} \in \hat{\mathbf{C}}$ and $r \in \mathbf{C}$ such that $\mathtt{a} = \mathtt{Dl}$ and

$$\begin{tikzcd} \bullet \arrow[rr, "\mathtt{a}"] \arrow[dd, "\mathtt{l}"'] & & \bullet \arrow[dl, "\bar{\mathtt{R}}"'] \arrow[dd, "\mathsf{id}"] \\ & \bullet \arrow[dr, "\mathtt{R}"] & \\ \bullet \arrow[ur, "\mathtt{D}'"] \arrow[rr, "\mathtt{D}"'] & & \bullet \end{tikzcd}$$

B.2

$$\begin{tikzcd} \bullet \arrow[r, "\mathtt{a}"] \arrow[d, "\mathtt{l}"'] & \bullet \arrow[d, "\bar{\mathtt{R}}\,\mathsf{id}"] \\ \bullet \arrow[r, "\mathtt{D}'"'] & \bullet \end{tikzcd}$$

B.3

$$a' = \mathcal{F}(\mathtt{D})r \qquad \mathcal{F}(\mathtt{D}) \in \mathbf{D} \qquad (\mathcal{F}(\mathtt{l}), r) \in \mathsf{Reacts}$$
$$\mathcal{F}(\mathtt{a}) = a\ .$$

Thus the outer square in Figure B.2 commutes. Since $\mathcal{F}$ has all redex-RPOs, we can construct $\bar{\mathtt{R}}, \mathtt{R}, \mathtt{D}'$ forming an RPO as shown in Figure B.2. Thus $a' = \mathcal{F}(\mathtt{D})r = \mathcal{F}(\mathtt{RD}')r$ and $\mathtt{R}\bar{\mathtt{R}} = \mathsf{id}$. Moreover, by Proposition 2.7, the square in Figure B.3 is an IPO. Hence $a \xrightarrow[\mathsf{r}]{\mathsf{id}} a'$ as desired. ∎

Lemma B.3 (portable IPO cutting; cf. Lemma 3.5) Suppose $\mathcal{F} : \hat{\mathbf{C}} \to \mathbf{C}$ is a functorial reactive system and has all redex-RPOs. If $Ca \xrightarrow[\mathsf{r}]{F} a'$ then there exist $a'' \in \mathbf{C}$, $\mathtt{R} \in \hat{\mathbf{C}}$ and an IPO square shown in Figure B.4 such that $a \xrightarrow[\mathsf{r}]{\mathcal{F}(\mathtt{F}')} a''$ and

$$\begin{tikzcd} \bullet \arrow[r, "\mathtt{C}"] \arrow[d, "\mathtt{F}'"'] & \bullet \arrow[d, "\bar{\mathtt{R}}\mathtt{F}"] \\ \bullet \arrow[r, "\mathtt{C}'"'] & \bullet \end{tikzcd}$$

B.4

$$a' = \mathcal{F}(\mathtt{RC}')a'' \qquad \mathcal{F}(\mathtt{C}') \in \mathbf{D} \qquad \mathtt{R}\bar{\mathtt{R}} = \mathsf{id}$$
$$\mathcal{F}(\mathtt{C}) = C \qquad \mathcal{F}(\mathtt{F}) = F\ .$$

Proof By the definition of $\xrightarrow[\mathsf{r}]{F}$ and the hypothesis that $\mathcal{F}$ creates compositions, there exist $\mathtt{a}, \mathtt{C}, \mathtt{l}, \mathtt{F}, \mathtt{R}, \bar{\mathtt{R}} \in \hat{\mathbf{C}}$ and $r \in \mathbf{C}$ such that the big rectangle in Figure B.5 is an IPO and

$$\begin{tikzcd} \bullet \arrow[r, "\mathtt{a}"] \arrow[dd, "\mathtt{l}"'] & \bullet \arrow[r, "\mathtt{C}"] \arrow[d, "\mathtt{F}'"] & \bullet \arrow[dd, "\bar{\mathtt{R}}\mathtt{F}"] \\ & \bullet \arrow[dr, "\mathtt{C}'"] & \\ \bullet \arrow[ur, "\mathtt{D}'"] \arrow[rr, "\mathtt{D}"'] & & \bullet \end{tikzcd}$$

B.5

$$a' = \mathcal{F}(\mathtt{RD})r \qquad \mathcal{F}(\mathtt{D}) \in \mathbf{D} \qquad (\mathcal{F}(\mathtt{l}), r) \in \mathsf{Reacts} \qquad \mathtt{R}\bar{\mathtt{R}} = \mathsf{id}$$
$$\mathcal{F}(\mathtt{a}) = a \qquad \mathcal{F}(\mathtt{C}) = C \qquad \mathcal{F}(\mathtt{F}) = F\ .$$

Because $\mathcal{F}$ has all redex-RPOs, there exist $\mathtt{F}', \mathtt{D}', \mathtt{C}'$ forming an RPO in $\hat{\mathbf{C}}$, as in Figure B.5. Then $\mathcal{F}(\mathtt{C}') \in \mathbf{D}$ since $\mathcal{F}(\mathtt{C}')\mathcal{F}(\mathtt{D}') = \mathcal{F}(\mathtt{D}) \in \mathbf{D}$. By Proposition 2.7, the small left-hand square of Figure B.5 is an IPO. Because $\mathcal{F}$ has all redex-RPOs, Proposition 2.9 implies that the small right-hand square is an IPO too. By definition, $a \xrightarrow[\mathsf{r}]{\mathcal{F}(\mathtt{F}')} a''$ and $a' = \mathcal{F}(\mathtt{RC}')a''$, where $a'' \mathrel{\hat{=}} \mathcal{F}(\mathtt{D}')r$, as desired. ∎

Lemma B.4 (portable IPO pasting; cf. Lemma 3.7) Suppose $\mathcal{F} : \hat{\mathbf{C}} \to \mathbf{C}$ is a functorial reactive system and has all redex-RPOs. If $a \xrightarrow[\mathsf{r}]{\mathcal{F}(\mathtt{F}')} a'$, Figure B.6 is an IPO with $\mathcal{F}(\mathtt{C}') \in \mathbf{D}$, and there exists $\mathtt{R}$ such $\mathtt{R}\bar{\mathtt{R}} = \mathsf{id}$ then $\mathcal{F}(\mathtt{C})a \xrightarrow[\mathsf{r}]{\mathcal{F}(\mathtt{F})} \mathcal{F}(\mathtt{RC}')a'$.

$$\begin{tikzcd} \bullet \arrow[r, "\mathtt{C}"] \arrow[d, "\mathtt{F}'"'] & \bullet \arrow[d, "\bar{\mathtt{R}}\mathtt{F}"] \\ \bullet \arrow[r, "\mathtt{C}'"'] & \bullet \end{tikzcd}$$

B.6

Proof Since $a \xrightarrow[\mathsf{r}]{\mathcal{F}(\mathtt{F}')} a'$, there exist $\mathtt{a}, \mathtt{l}, \mathtt{F}'_0, \mathtt{D}, \mathtt{T}_0, \bar{\mathtt{T}}_0 \in \hat{\mathbf{C}}$ and $r \in \mathbf{C}$ such that Figure B.7 is an IPO in $\hat{\mathbf{C}}$ and

$$\begin{tikzcd} \bullet \arrow[r, "\mathtt{a}"] \arrow[d, "\mathtt{l}"'] & \bullet \arrow[d, "\bar{\mathtt{T}}_0\mathtt{F}'_0"] \\ \bullet \arrow[r, "\mathtt{D}"'] & \bullet \end{tikzcd}$$

B.7

$$\begin{tikzcd} \bullet \arrow[r, "\mathtt{C}_0"] \arrow[d, "\mathtt{F}'_0"'] & \bullet \arrow[d, "\bar{\mathtt{R}}_0\mathtt{F}_0"] \\ \bullet \arrow[r, "\mathtt{C}'_0"'] & \bullet \end{tikzcd}$$

B.8

$$a' = \mathcal{F}(\mathtt{T}_0\mathtt{D})r \qquad \mathcal{F}(\mathtt{D}) \in \mathbf{D} \qquad (\mathcal{F}(\mathtt{l}), r) \in \mathsf{Reacts}$$
$$\mathtt{T}_0\bar{\mathtt{T}}_0 = \mathsf{id} \qquad \mathcal{F}(\mathtt{a}) = a \qquad \mathcal{F}(\mathtt{F}'_0) = \mathcal{F}(\mathtt{F}')\ .$$

Since $\mathcal{F}$ allows IPO sliding and creates compositions, there exist $\mathtt{C}_0, \bar{\mathtt{R}}_0, \mathtt{F}_0, \mathtt{C}'_0 \in \hat{\mathbf{C}}$ such that Figure B.8 is an IPO and

$$\mathcal{F}(\mathtt{C}_0) = \mathcal{F}(\mathtt{C}) \qquad \mathcal{F}(\bar{\mathtt{R}}_0) = \mathcal{F}(\bar{\mathtt{R}}) \qquad \mathcal{F}(\mathtt{F}_0) = \mathcal{F}(\mathtt{F}) \qquad \mathcal{F}(\mathtt{C}'_0) = \mathcal{F}(\mathtt{C}') \ .$$

Since $\mathsf{id} = \mathcal{F}(\mathsf{id}) = \mathcal{F}(\mathtt{R})\mathcal{F}(\bar{\mathtt{R}}) = \mathcal{F}(\mathtt{R})\mathcal{F}(\bar{\mathtt{R}}_0)$ and $\mathcal{F}$ creates left inverses, there exists $\mathtt{R}_0$ such that $\mathsf{id} = \mathtt{R}_0\bar{\mathtt{R}}_0$ and $\mathcal{F}(\mathtt{R}_0) = \mathcal{F}(\mathtt{R})$.

Pasting Figure B.8 vertically to the commuting square shown in Figure B.9 and then pasting horizontally the composite to Figure B.7 results in the outer rectangle shown in Figure B.10. Because $\mathcal{F}$ has all redex-RPOs, there exist $\mathtt{K}_0, \mathtt{K}_1, \mathtt{K}$ forming an RPO inside, as shown. By Proposition 2.7, the rectangle in Figure B.11 is an IPO.

$$\begin{tikzcd}
{} \arrow[r, "\mathtt{C}'_0"] \arrow[d, "\bar{\mathtt{T}}_0"'] & {} \arrow[d, "\mathsf{id}"] \\
{} \arrow[r, "\mathtt{C}'_0\mathtt{T}_0"'] & {}
\end{tikzcd}$$

B.9

$$\begin{tikzcd}
{} \arrow[r, "\mathtt{a}"] \arrow[dd, "\mathtt{1}"'] & {} \arrow[r, "\mathtt{C}_0"] & {} \arrow[dl, "\mathtt{K}_0"'] \arrow[dd, "\bar{\mathtt{R}}_0\mathtt{F}_0"] \\
& {} \arrow[dr, "\mathtt{K}"] & \\
{} \arrow[ur, "\mathtt{K}_1"] \arrow[r, "\mathtt{D}"'] & {} \arrow[r, "\mathtt{C}'_0\mathtt{T}_0"'] & {}
\end{tikzcd}$$

B.10

$$\begin{tikzcd}
{} \arrow[r, "\mathtt{a}"] \arrow[d, "\mathtt{1}"'] & {} \arrow[r, "\mathtt{C}_0"] & {} \arrow[d, "\mathtt{K}_0"] \\
{} \arrow[rr, "\mathtt{K}_1"'] & & {}
\end{tikzcd}$$

B.11

Since $\mathcal{F}$ has all redex-RPOs and Figure B.7 is an IPO, Proposition 2.8 implies that Figure B.12 is an RPO. But $\mathtt{K}_0\mathtt{C}_0, \mathtt{K}_1, \mathtt{K}$ is a candidate, so there exists a unique $\mathtt{J}$ as shown in Figure B.13. Now

$$\begin{tikzcd}
{} \arrow[rr, "\mathtt{a}"] \arrow[dd, "\mathtt{1}"'] & & {} \arrow[dl, "\bar{\mathtt{T}}_0\mathtt{F}'_0"'] \arrow[dd, "\mathtt{C}'_0\mathtt{F}'_0"] \\
& {} \arrow[dr, "\mathtt{C}'_0\mathtt{T}_0"] & \\
{} \arrow[ur, "\mathtt{D}"] \arrow[rr, "\mathtt{C}'_0\mathtt{T}_0\mathtt{D}"'] & & {}
\end{tikzcd}$$

B.12

$$\begin{tikzcd}
{} \arrow[rrr, "\mathtt{a}"] \arrow[dd, "\mathtt{1}"'] & & & {} \arrow[dll, "\bar{\mathtt{T}}_0\mathtt{F}'_0"'] \arrow[dl, "\mathtt{K}_0\mathtt{C}_0"] \arrow[dd, "\mathtt{C}'_0\mathtt{F}'_0"] \\
& {} \arrow[r, dotted, "\mathtt{J}"] \arrow[drr, "\mathtt{C}'_0\mathtt{T}_0"] & {} \arrow[dr, "\mathtt{K}"] & \\
{} \arrow[ur, "\mathtt{D}"] \arrow[urr, bend right, "\mathtt{K}_1"] \arrow[rrr, "\mathtt{C}'_0\mathtt{T}_0\mathtt{D}"'] & & & {}
\end{tikzcd}$$

B.13

$$\mathcal{F}(\mathtt{K})\mathcal{F}(\mathtt{J})\mathcal{F}(\bar{\mathtt{T}}_0) \ = \ \mathcal{F}(\mathtt{K}\mathtt{J}\bar{\mathtt{T}}_0) \ =^{\text{Figure B.13}} \ \mathcal{F}(\mathtt{C}'_0) \ = \ \mathcal{F}(\mathtt{C}') \ \in \ \mathbf{D}$$

so $\mathcal{F}(\mathtt{J}) \in \mathbf{D}$.

Thus, $\mathtt{K}_0, \mathtt{J}\bar{\mathtt{T}}_0, \mathtt{K}$ is a candidate for Figure B.7, an IPO, as shown in Figure B.14. Thus there exists a unique $\bar{\mathtt{K}}$ such that $\mathtt{K}\bar{\mathtt{K}} = \mathsf{id}$, $\bar{\mathtt{K}}\bar{\mathtt{R}}_0\mathtt{F}_0 = \mathtt{K}_0$, and $\bar{\mathtt{K}}\mathtt{C}'_0 = \mathtt{J}\bar{\mathtt{T}}_0$. We redraw Figure B.11, an IPO, with equivalent arrows in Figure B.15. Recall that $\mathcal{F}(\mathtt{J}) \in \mathbf{D}$, so $\mathcal{F}(\mathtt{JD}) \in \mathbf{D}$. Thus:

$$\begin{tikzcd}
{} \arrow[rr, "\mathtt{C}_0"] \arrow[dd, "\mathtt{F}'_0"'] & & {} \arrow[dl, "\mathtt{K}_0"'] \arrow[dd, "\bar{\mathtt{R}}_0\mathtt{F}_0"] \\
& {} \arrow[dr, "\mathtt{K}"'] & \\
{} \arrow[ur, "\mathtt{J}\bar{\mathtt{T}}_0"] \arrow[rr, "\mathtt{C}'_0"'] & & {} \arrow[ul, dotted, bend right, "\bar{\mathtt{K}}"']
\end{tikzcd}$$

B.14

$$\begin{tikzcd}
{} \arrow[r, "\mathtt{a}"] \arrow[d, "\mathtt{1}"'] & {} \arrow[r, "\mathtt{C}_0"] & {} \arrow[d, "\bar{\mathtt{K}}\bar{\mathtt{R}}_0\mathtt{F}_0"] \\
{} \arrow[rr, "\mathtt{JD}"'] & & {}
\end{tikzcd}$$

B.15

$$\mathcal{F}(\mathtt{C})a \ = \ \mathcal{F}(\mathtt{C}_0)a \ \xrightarrow[\mathsf{r}]{\mathcal{F}(\mathtt{F}_0)} \ \mathcal{F}(\mathtt{R}_0\mathtt{KJD})r \ =^{\text{Figure B.13}} \ \mathcal{F}(\mathtt{RC}')\mathcal{F}(\mathtt{T}_0\mathtt{D})r \ = \ \mathcal{F}(\mathtt{RC}')a' \ ,$$

so $\mathcal{F}(\mathtt{C})a \xrightarrow[\mathsf{r}]{\mathcal{F}(\mathtt{F})} \mathcal{F}(\mathtt{RC}')a'$, as desired. ■

Theorem B.5 (congruence for $\sim_{\mathsf{r}}$; cf. Theorem 3.12) Let $\mathcal{F} : \hat{\mathbf{C}} \longrightarrow \mathbf{C}$ be a functorial reactive system which has all redex-RPOs. Then $\sim_{\mathsf{r}}$ is a congruence, i.e. $a \sim_{\mathsf{r}} b$ implies $Ca \sim_{\mathsf{r}} Cb$ for all $C \in \mathbf{C}$ of the required domain.

Proof By symmetry, it is sufficient to show that the following relation is a strong simulation:

$$\mathcal{S} \mathrel{\hat{=}} \{(Ca, Cb) \ / \ a \sim_{\mathsf{r}} b \text{ and } C \in \mathbf{C}\} \ .$$

Suppose that $a \sim_{\mathsf{r}} b$ and $C \in \mathbf{C}$, and thus $(Ca, Cb) \in \mathcal{S}$. Suppose $Ca \xrightarrow[\mathsf{r}]{F} a'$.

$$\begin{tikzcd} \cdot \arrow[r, "\mathtt{C}"] \arrow[d, "\mathtt{F}'"'] & \cdot \arrow[d, "\bar{\mathtt{R}}\mathtt{F}"] \\ \cdot \arrow[r, "\mathtt{C}'"'] & \cdot \end{tikzcd}$$

B.16

By Lemma B.3, there exist $a'' \in \mathbf{C}$, $\mathtt{R} \in \hat{\mathbf{C}}$ and an IPO square shown in Figure B.16 such that $a \xrightarrow[\mathsf{r}]{\mathcal{F}(\mathtt{F}')} a''$ and

$$a' = \mathcal{F}(\mathtt{R}\mathtt{C}')a'' \qquad \mathtt{R}\bar{\mathtt{R}} = \mathsf{id} \qquad \mathcal{F}(\mathtt{C}') \in \mathbf{D}$$
$$\mathcal{F}(\mathtt{C}) = C \qquad \mathcal{F}(\mathtt{F}) = F \; .$$

Since $a \sim_{\mathsf{r}} b$, there exists b'' such that $b \xrightarrow[\mathsf{r}]{\mathcal{F}(\mathtt{F}')} b''$ and $a'' \sim_{\mathsf{r}} b''$. By Lemma B.4, $Cb \xrightarrow[\mathsf{r}]{F} \mathcal{F}(\mathtt{R}\mathtt{C}')b''$. Hence $(\mathcal{F}(\mathtt{R}\mathtt{C}')a'', \mathcal{F}(\mathtt{R}\mathtt{C}')b'') \in \mathcal{S}$ since $a'' \sim_{\mathsf{r}} b''$, as desired. ■

Proposition B.6 ($\sim_{\mathsf{c}} \subseteq \sim_{\mathsf{r}}$; cf. Proposition 3.13) Let $\mathcal{F} : \hat{\mathbf{C}} \to \mathbf{C}$ be a functorial reactive system which has all redex-RPOs. Then $\sim_{\mathsf{c}} \subseteq \sim_{\mathsf{r}}$.

Proof We show that $\sim_{\mathsf{c}}$ is a strong bisimulation w.r.t. the labelled transition relation $\xrightarrow[\mathsf{r}]{\cdot}$. By symmetry, it is sufficient to shown that $\sim_{\mathsf{c}}$ is a strong simulation over $\xrightarrow[\mathsf{r}]{\cdot}$. Consider any a, b for which $a \sim_{\mathsf{c}} b$. Suppose $a \xrightarrow[\mathsf{r}]{F} a'$. By definition, there exists $\mathtt{F}, \mathtt{R}, \bar{\mathtt{R}} \in \hat{\mathbf{C}}$ and $a'' \in \mathbf{C}$ such that $a \xrightarrow[\mathsf{c}]{\mathcal{F}(\bar{\mathtt{R}}\mathtt{F})} a''$ and

$$a' = \mathcal{F}(\mathtt{R})a'' \qquad \mathtt{R}\bar{\mathtt{R}} = \mathsf{id} \qquad \mathcal{F}(\mathtt{F}) = F \; .$$

Since $a \sim_{\mathsf{c}} b$, there exists b'' such that $b \xrightarrow[\mathsf{c}]{\mathcal{F}(\bar{\mathtt{R}}\mathtt{F})} b''$ and $a'' \sim_{\mathsf{c}} b''$. We now distinguish two cases:

case $\mathcal{F}(\bar{\mathtt{R}}\mathtt{F})$ is an iso: In this case, $\mathcal{F}(\bar{\mathtt{R}}\mathtt{F})b \longrightarrow b''$. Since $\mathcal{F}(\mathtt{R})\mathcal{F}(\bar{\mathtt{R}}) = \mathsf{id} \in \mathbf{D}$, we have that $\mathcal{F}(\mathtt{R}) \in \mathbf{D}$, thus $\mathcal{F}(\mathtt{F})b \longrightarrow \mathcal{F}(\mathtt{R})b''$. Since $\mathcal{F}$ creates isos, $\bar{\mathtt{R}}\mathtt{F}$ is an iso in $\hat{\mathbf{C}}$, so has an inverse $\mathtt{K}$. Thus $\mathtt{K}\bar{\mathtt{R}}$ is an inverse for $\mathtt{F}$, so $\mathtt{F}$ is an iso, so $\mathcal{F}(\mathtt{F}) = F$ is an iso. Thus $b \xrightarrow[\mathsf{c}]{F} \mathcal{F}(\mathtt{R})b''$. Since $\sim_{\mathsf{c}}$ is a congruence, $a' = \mathcal{F}(\mathtt{R})a'' \sim_{\mathsf{c}} \mathcal{F}(\mathtt{R})b''$, as desired.

$$\begin{tikzcd} \cdot \arrow[r, "\mathtt{b}"] \arrow[d, "\mathtt{l}"'] & \cdot \arrow[d, "\bar{\mathtt{R}}_0\mathtt{F}_0"] \\ \cdot \arrow[r, "\mathtt{D}"'] & \cdot \end{tikzcd}$$

B.17

case $\mathcal{F}(\bar{\mathtt{R}}\mathtt{F})$ is not an iso: In this case, $b \xrightarrow{\mathcal{F}(\bar{\mathtt{R}}\mathtt{F})} b''$. Since $\mathcal{F}$ creates compositions, there exist $\mathtt{b}, \mathtt{l}, \mathtt{F}_0, \bar{\mathtt{R}}_0, \mathtt{D} \in \hat{\mathbf{C}}$ and $r \in \mathbf{C}$ such that Figure B.17 is an IPO in $\hat{\mathbf{C}}$ and

$$b'' = \mathcal{F}(\mathtt{D})r \qquad \mathcal{F}(\mathtt{D}) \in \mathbf{D} \qquad (\mathcal{F}(\mathtt{l}), r) \in \mathsf{Reacts}$$
$$\mathcal{F}(\mathtt{b}) = b \qquad \mathcal{F}(\mathtt{F}_0) = F = \mathcal{F}(\mathtt{F}) \qquad \mathcal{F}(\bar{\mathtt{R}}_0) = \mathcal{F}(\bar{\mathtt{R}}) \qquad .$$

Since $\mathsf{id} = \mathcal{F}(\mathsf{id}) = \mathcal{F}(\mathtt{R})\mathcal{F}(\bar{\mathtt{R}}) = \mathcal{F}(\mathtt{R})\mathcal{F}(\bar{\mathtt{R}}_0)$ and $\mathcal{F}$ creates left inverses, there exists $\mathtt{R}_0$ such that $\mathsf{id} = \mathtt{R}_0\bar{\mathtt{R}}_0$ and $\mathcal{F}(\mathtt{R}_0) = \mathcal{F}(\mathtt{R})$. Thus $b \xrightarrow[\mathsf{r}]{F} \mathcal{F}(\mathtt{R}_0\mathtt{D})r = \mathcal{F}(\mathtt{R})b''$. Since $\sim_{\mathsf{c}}$ is a congruence, $a' = \mathcal{F}(\mathtt{R})a'' \sim_{\mathsf{c}} \mathcal{F}(\mathtt{R})b''$, as desired. ■

Bibliography

Curly braces enclose pointers back to the pages in this dissertation that cite the work.

[AG97] M. Abadi and A. D. Gordon. A calculus for cryptographic protocols: the spi calculus. In *Proc. 4th ACM Conf. on Computer and Communications Security, Zürich*, pages 36–47. ACM Press, 1997. {6}

[Bar84] H. P. Barendregt. *The Lambda Calculus: Its Syntax and Semantics.* North Holland, revised edition, 1984. {7, 8}

[BB90] G. Berry and G. Boudol. The chemical abstract machine. In *Proc. 17th Annual Symposium of Principles of Programming Languages*, pages 81–94. ACM Press, 1990. {6}

[BB92] G. Berry and G. Boudol. The chemical abstract machine. *Theoretical Computer Science*, 96(1):217–248, 1992. {6}

[BF00] M. Bunge and M. P. Fiore. Unique factorisation lifting functors and categories of linearly-controlled processes. *Mathematical Structures in Computer Science*, 10(2):137–163, 2000. {69}

[Blo93] B. Bloom. Structural operational semantics for weak bisimulations. Technical Report TR-93-1373, Department of Computer Science, Cornell University, August 1993. {14, 45, 129}

[BW01] M. Barr and C. F. Wells. Toposes, triples and theories. Version 1.1. Available from: http://www.cwru.edu/artsci/math/wells/pub/ttt.html, 2001. {131, 135}

[CG98] L. Cardelli and A. D. Gordon. Mobile ambients. In *Foundations of Software Science and Computation Structure, First International Conference, FoSSaCS '98, Held as Part of the European Joint Conferences on the Theory and Practice of Software, ETAPS '98, Lisbon, Portugal, March 28 – April 4, 1998, Proceedings*, volume 1378 of *Lecture Notes in Computer Science*. Springer-Verlag, 1998. {7}

[CLM00] G. L. Cattani, J. J. Leifer, and R. Milner. Contexts and embeddings for closed shallow action graphs. Technical Report 496, Computer Laboratory, University of Cambridge, July 2000. {16, 74, 87, 88, 89, 93, 94, 95, 96}

[CM91] A. Corradini and U. Montanari. An algebra of graphs and graph rewriting. In *4th Biennial Conference on Category Theory and Computer Science, Proceedings*, volume 530 of *Lecture Notes in Computer Science*, pages 236–260. Springer-Verlag, 1991. {15}

[Con72] F. Conduché. Au sujet de l'existence d'adjoints à droite aux foncteurs "image réciproque" dans la catégorie des catégories. *Comptes rendus de l'Académie des sciences A*, pages 891–894, 1972. {69}

[CS00] G. L. Cattani and P. Sewell. Models for name-passing processes: interleaving and causal. In *15th Annual IEEE Symposium on Logic in Computer Science, 26–29 June 2000, Santa Barbara, California, USA*, pages 322–332. IEEE Press, 2000. {126}

[DH84] R. De Nicola and M. C. B. Hennessy. Testing equivalences for processes. *Theoretical Computer Science*, 34:83–133, 1984. {14}

[Ehr79] H. Ehrig. Introduction to the algebraic theory of graph grammar. In *Proc. first international Workshop on Graph Grammars and their application to Computer Science and Biology*, volume 73 of *Lecture Notes in Computer Science*, pages 1–69. Springer-Verlag, 1979. {15}

[FF86] M. Felleisen and D. P. Friedman. Control operators, the SECD-machine and the λ-calculus. In M. Wirsing, editor, *Formal Description of Programming Concepts III*, pages 193–217. North Holland, 1986. {7, 21}

[FG98] C. Fournet and G. Gonthier. A hierarchy of equivalences for asynchronous calculi. In *Automata, Languages and Programming, 25th International Colloquium, ICALP '98, Aalborg, Denmark, July 13–17, 1998, Proceedings*, volume 1443 of *Lecture Notes in Computer Science*, pages 844–855. Springer-Verlag, 1998. {14}

[FGL$^+$96] C. Fournet, G. Gonthier, J.-J. Lévy, L. Maranget, and D. Rémy. A calculus of mobile agents. In *CONCUR '96, Concurrency Theory, 7th International Conference, Pisa, Italy, August 26–29, 1996, Proceedings*, volume 1119 of *Lecture Notes in Computer Science*, pages 406–421. Springer-Verlag, 1996. {7}

[Fou98] C. Fournet. *The Join-Calculus: a Calculus for Distributed Mobile Programming.* PhD thesis, École Polytechnique, November 1998. {14}

[GP99] M. J. Gabbay and A. M. Pitts. A new approach to abstract syntax involving binders. In *14th Annual Symposium on Logic in Computer Science, 2–5 July, 1999, Trento, Italy*, pages 214–224. IEEE Press, 1999. {126}

[GV92] J. F. Groote and F. W. Vaandrager. Structural operational semantics and bisimulation as a congruence. *Information and Computation*, 100(2):202–260, 1992. {14, 129}

[Has99] M. Hasegawa. *Models of sharing graphs: a categorical semantics of let and letrec.* BCS Distinguished Dissertation Series. Springer-Verlag, 1999. {76}

[Hoa78] C. A. R. Hoare. Communicating sequential processes. *Communications of the ACM*, 21(8):666–677, August 1978. {4}

[Hoa85] C. A. R. Hoare. *Communicating Sequential Processes.* Prentice-Hall, 1985. {4, 47, 50}

[Hon00] K. Honda. Elementary structures for process theory (1): sets with renaming. *Mathematical Structures in Computer Science*, 10(5):617–663, October 2000. {64}

[HT91] K. Honda and M. Tokoro. An object calculus for asynchronous communication. In *ECOOP '91: European Conference on Object-Oriented Programming, Geneva, Switzerland, July 15–19, 1991, Proceedings*, volume 512, pages 133–147. Springer-Verlag, 1991. {128}

[HY94a] K. Honda and N. Yoshida. Combinatory representation of mobile processes. In *POPL '94: 21st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, Portland, Oregon, January 17–21, 1994*, pages 348–360. ACM Press, 1994. {126}

[HY94b] K. Honda and N. Yoshida. Replication in concurrent combinators. In *Theoretical Aspects of Computer Software, International Conference TACS '94, Sendai, Japan, April 19–22, 1994, Proceedings*, volume 789, pages 786–805. Springer-Verlag, 1994. {128}

[HY95] K. Honda and N. Yoshida. On reduction-based process semantics. *Theoretical Computer Science*, 152(2):437–486, November 1995. {14}

[Joh99] P. Johnstone. A note on discrete Conduché fibrations. *Theory and Application of Categories*, 5(1):1–11, 1999. {69}

[JR99] A. Jeffrey and J. Rathke. Towards a theory of bisimulation for local names. In *14th Annual Symposium on Logic in Computer Science, 2–5 July, 1999, Trento, Italy*, pages 56–66. IEEE Press, 1999. {15}

[JSV96] A. Joyal, R. Street, and D. Verity. Traced monoidal categories. *Mathematical Proceedings of the Cambridge Philosophical Society*, 119(3):425–446, 1996. {75}

[Kön99] B. König. Generating type systems for process graphs. In *CONCUR '99: Concurrency Theory, 10th International Conference, Eindhoven, The Netherlands, August 24–27, 1999, Proceedings*, volume 1664, pages 352–367. Springer-Verlag, 1999. {15}

[Laf90] Y. Lafont. Interaction nets. In *Conference Record of the Seventeenth Annual ACM Symposium on Principles of Programming Languages, San Francisco, California, January 1990*, pages 95–108. ACM Press, 1990. {76}

[Lei01] J. J. Leifer. A category of action graphs and reflexive embeddings. Technical report, Computer Laboratory, University of Cambridge, 2001. To appear. {129}

[Lév78] J.-J. Lévy. *Réductions correctes et optimales dans le lambda calcul.* PhD thesis, Université Paris VII, 1978. {8}

[LM00a] J. J. Leifer and R. Milner. Deriving bisimulation congruences for reactive systems. In C. Palamidessi, editor, *CONCUR 2000 - Concurrency Theory, 11th International Conference, University Park, PA, USA, August 22–25, 2000, Proceedings*, volume 1877 of *Lecture Notes in Computer Science*, pages 243–258. Springer-Verlag, 2000. {16}

[LM00b] J. J. Leifer and R. Milner. Shallow linear action graphs and their embeddings. Technical Report 508, Computer Laboratory, University of Cambridge, November 2000. {81, 125, 129}

[Low96] G. Lowe. Breaking and fixing the Needham-Schroeder public-key protocol using FDR. In T. Margaria and B. Steffen, editors, *Tools and Algorithms for Construction and Analysis of Systems, Second International Workshop, TACAS '96, Passau, Germany, March 27–29, 1996, Proceedings*, volume 1055 of *Lecture Notes in Computer Science*, pages 147–166. Springer-Verlag, 1996. {4, 50}

[Mac71] S. Mac Lane. *Categories for the Working Mathematician.* Springer-Verlag, 1971. {131}

[Mil80] R. Milner. *A Calculus of Communicating Systems*, volume 92 of *Lecture Notes in Computer Science.* Springer-Verlag, 1980. {4}

[Mil88] R. Milner. *Communication and Concurrency.* Prentice-Hall, 1988. {4, 34, 44, 45, 47}

[Mil90] R. Milner. Functions as processes. Technical Report RR-1154, INRIA, Sophia Antipolis, February 1990. {6, 21}

[Mil92] R. Milner. Functions as processes. *Mathematical Structures in Computer Science*, 2(2):119–141, 1992. {41}

[Mil94] R. Milner. Action calculi V: reflexive molecular forms. Third draft; with an appendix by O. Jensen. Available from: ftp://ftp.cl.cam.ac.uk/users/rm135/ac5.ps.Z, 1994. {75}

[Mil96] R. Milner. Calculi for interaction. *Acta Informatica*, 33(8):707–737, 1996. {10, 74, 77, 124}

[Moo56] E. F. Moore. Gedanken-experiments on sequential machines. In C. E. Shannon and J. McCarthy, editors, *Automata Studies*, pages 129–153. Princeton University Press, 1956. {3}

[MPW89] R. Milner, J. Parrow, and D. Walker. A calculus of mobile processes, parts I and II. Technical Report ECS-LFCS-89-85 and ECS-LFCS-89-86, Laboratory for the Foundations of Computer Science, University of Edinburgh, 1989. {6}

[MPW92] R. Milner, J. Parrow, and D. Walker. A calculus of mobile processes, parts I and II. *Information and Computation*, 100(1):1–77, September 1992. {6}

[MS92] R. Milner and D. Sangiorgi. Barbed bisimulation. In *Automata, Languages and Programming, 19th International Colloquium, ICALP '92, Vienna, Austria, July 13–17, 1992, Proceedings*, volume 623 of *Lecture Notes in Computer Science*. Springer-Verlag, 1992. {14}

[MSS00] P. Mateus, A. Sernadas, and C. Sernadas. Precategories for combining probabilistic automata. In M. Hofmann, G. Rosolini, and D. Pavlovic, editors, *Proc. CTCS '99*, volume 29 of *Electronic Notes in Theoretical Computer Science*. Elsevier Science, 2000. {63}

[NP96] U. Nestmann and B. C. Pierce. Decoding choice encodings. In *CONCUR '96, Concurrency Theory, 7th International Conference, Pisa, Italy, August 26–29, 1996, Proceedings*, volume 1119 of *Lecture Notes in Computer Science*. Springer-Verlag, 1996. {128}

[Par81] D. Park. Concurrency and automata on infinite sequences. In P. Duessen, editor, *Proc. 5th GI Conference*, volume 104 of *Lecture Notes in Computer Science*, pages 167–183. Springer-Verlag, 1981. {5, 31}

[Pau98] L. C. Paulson. The inductive approach to verifying cryptographic protocols. *J. Computer Security*, 6:85–128, 1998. {47}

[Plo75] G. D. Plotkin. Call-by-name, call-by-value, and the λ-calculus. *Theoretical Computer Science*, 1(2):125–159, December 1975. {21}

[Plo81] G. D. Plotkin. A structural approach to operational semantics. Technical Report DAIMI-FN-19, Department of Computer Science, University of Aarhus, 1981. {4}

[Rei85] W. Reisig. *Petri Nets: An Introduction.* EATCS Monographs on Theoretical Computer Science. Springer-Verlag, 1985. {4}

[Ros94] A. W. Roscoe. Model-checking CSP. In A. W. Roscoe, editor, *A Classical Mind: Essays in Honour of C. A. R. Hoare*, pages 353–378. Prentice-Hall, 1994. {4, 50}

[Ros98] A. W. Roscoe. *The Theory and Practice of Concurrency.* Prentice-Hall, 1998. {3, 4, 48, 50}

[Sew98] P. Sewell. Global/local subtyping and capability inference for a distributed pi-calculus. In *Automata, Languages and Programming, 25th International Colloquium, ICALP '98, Aalborg, Denmark, July 13–17, 1998, Proceedings*, volume 1443 of *Lecture Notes in Computer Science.* Springer-Verlag, 1998. {7}

[Sew00] P. Sewell. Applied π — A brief tutorial. Technical Report 498, Computer Laboratory, University of Cambridge, August 2000. {41}

[Sew01] P. Sewell. From rewrite rules to bisimulation congruences. *Theoretical Computer Science*, 272(1–2), 2001. {10, 15, 21, 55, 60, 81, 124, 127}

[SV99] P. Sewell and J. Vitek. Secure compositions of insecure components. In *Proc. 12th Computer Security Foundations Workshop.* IEEE Press, June 1999. {7}

[SV00] P. Sewell and J. Vitek. Secure composition of untrusted code: wrappers and causality types. In *Proc. 13th Computer Security Foundations Workshop.* IEEE Press, July 2000. {47}

[SW01] D. Sangiorgi and D. Walker. *The π-calculus: a Theory of Mobile Processes.* Cambridge University Press, 2001. {7}

[TFP99] D. Turi, M. P. Fiore, and G. D. Plotkin. Abstract syntax and variable binding. In *14th Annual Symposium on Logic in Computer Science, 2–5 July, 1999, Trento, Italy*, pages 193–202. IEEE Press, 1999. {126}

[TP97] D. Turi and G. Plotkin. Towards a mathematical operational semantics. In *12th Annual IEEE Symposium on Logic in Computer Science, Warsaw, Poland, June 29 – July 2, 1997*, pages 280–291. IEEE Press, 1997. {14, 129}

[WN95] G. Winskel and M. Nielsen. Models for concurrency. In D. M. Gabbay, S. Abramsky, and T. S. E. Maibaum, editors, *Handbook of Logic in Computer Science*, volume 4, pages 1–148. Oxford University Press, 1995. {4}