Basic Research in Computer Science

Deriving Process Congruences from Reaction Rules

Paweł Sobociński

BRICS Dissertation Series **DS-04-6**

ISSN 1396-7002 **December 2004**

See back inner page for a list of recent BRICS Dissertation Series publications. Copies may be obtained by contacting:

BRICS
Department of Computer Science
University of Aarhus
Ny Munkegade, building 540
DK–8000 Aarhus C
Denmark
Telephone: +45 8942 3360
Telefax: +45 8942 3255
Internet: BRICS@brics.dk

BRICS publications are in general accessible through the World Wide Web and anonymous FTP through these URLs:

`http://www.brics.dk`
`ftp://ftp.brics.dk`
This document in subdirectory `DS/04/6/`

Deriving process congruences from reaction rules

Paweł Sobociński

PhD Dissertation

Department of Computer Science
University of Aarhus
Denmark

Deriving process congruences from reaction rules

A Dissertation
Presented to the Faculty of Science
of the University of Aarhus
in Partial Fulfilment of the Requirements for the
PhD Degree

by
Paweł Sobociński
August 31, 2004

Abstract

This thesis is concerned with the development of a theory which, given a formalism with a reduction semantics, allows the derivation of a canonical labelled transition system on which bisimilarity as well as other other equivalences are congruences; provided that the contexts of the formalism form a category which has certain colimits.

We shall begin by extending Leifer and Milner's theory of reactive systems to a 2-categorical setting. This development is motivated by the common situation in which the contexts of a reactive system contain non-trivial algebraic structure with an associated notion of context isomorphism. Forgetting this structure often leads to problems and we shall show that the theory can be extended smoothly, retaining this useful information as well as the congruence theorems.

Technically, the generalisation includes defining the central notion of groupoidal-relative-pushout (GRPO) (categorically: a bipushout in a pseudo-slice category), which turns out to provide a suitable generalisation of Leifer and Milner's relative pushout (RPO). The congruence theorems are then reproved in this more general setting. We shall also show how previously introduced alternative solutions to the problem of forgetting the two-dimensional structure can be reduced to the 2-categorical approach.

Secondly, we shall construct GRPOs in settings which are general enough to allow the theory to be applied to useful, previously studied examples. We shall begin by constructing GRPOs in a category whose arrows correspond closely to the contexts a simple process calculus, and extend this construction further to cover the category of bunch contexts, studied previously by Leifer and Milner. The constructions use the structure of extensive categories.

We shall argue that cospans provide an interesting notion of "generalised contexts". In an effort to find a natural class of categories which allows the construction of GRPOs in the corresponding cospan bicategory, we shall introduce the class of adhesive categories. As extensive categories have

well-behaved coproducts, so adhesive categories have well-behaved pushouts along monomorphisms. Adhesive categories also turn out to be a useful tool in the study and generalisation of the theory of double-pushout graph transformation systems, indeed, such systems have a rich rewriting theory when defined over adhesive categories.

Armed with the theory of adhesive categories, we shall present a construction of GRPOs in input-linear cospan bicategories. As an immediate application, we shall shed light on as well as extend the theory of rewriting via borrowed contexts, due to Ehrig and König. Secondly, we shall examine the implications of the construction for Milner's bigraphs.

Acknowledgements

First, thanks to my supervisor, Mogens Nielsen, who started me off and was always helpful. Thanks to my coauthors: Bartek Klin, Steve Lack and Vladimiro Sassone from whom I have learned much and with whom it was a lot of fun to learn to do research.

Thanks to Vincent Danos for the invitation to spend an extremely useful and stimulating week in Paris in March 2004, and to Thomas Hildebrandt in Copenhagen, Fabio Gadducci in Pisa and Robin Milner in Cambridge for their interest and enthusiasm.

Thanks to my officemate in Ny Munkegade, Mikkel Nygaard Ravn, for helping me to begin to understand Denmark and the Danes, and for his amazing proofreading skills. Thanks to the international crew of (now mostly ex) students at BRICS who made it a special place to be: Saurabh Agarwal, Jesús Almansa, Claus Brabrand, Mario Cáccamo, Marco Carbone, Federico Crazzolarra, Riko Jacob, Bartek Klin, Giuseppe Milicia, Kirill Morozov, Oliver Müller, Paulo Oliva, Jiří Srba, Frank Valencia, and my nemesis, Daniele Varacca. A special thanks to Janne Christensen, whose smileys are well-known amongst the BRICS alumni.

Thanks to everyone in opgang 86 of Skjoldhøjkollegiet, it was a great two years, from the tour de chambres to the spring cleaning.

My cumulative year in Brighton was incredibly productive and enjoyable. Thanks to the people who made it such a friendly and inspirational environment: Guy McCusker, Jim Laird, Cédric Lhoussaine, Julian Rathke, Bernhard Reus and Vladimiro Sassone; and to the students for making the theory lab so much fun: Jan Schwinghammer, Adrian Francalanza, Matt Wall, Damiano Macedonio (and Simona, for teaching us the art of tiramisu.)

Thanks to Mike Shuter in Cambridge for housing me whenever I dropped by and for not leaving me behind in the Asturias. Thanks to the Basque girls, Itziar and Arantza for a great first year in Denmark, and for the *Virgen Blanca* in Vitoria.

Thank you to Lea for putting up with me throughout the last 2 years.

Thanks to the extended Gerner family: Erik, Bodil, Heidi, Christian and Anja for Val Thorens and two wonderful Danish christmases, dancing around the christmas tree included.

A big thanks to the people I've managed to convince into reading parts of the thesis and who've really helped its presentation: Mikkel, Bartek, Julian and Daniele.

Finally, thanks to my mum, dad and Agnieszka in Sydney, and to my grandmother in Bydgoszcz, for her advice and her stories.

Paweł Sobociński,
Odense, August 31, 2004.

Contents

Chapter 1

Introduction

1.1 Overview

This thesis develops general mathematical technology for the study of the behavioural theory of computational formalisms with underlying reduction-based operational semantics. Such formalisms include both syntactic models, such as functional programming languages and process-calculi, as well as graphical models such as Petri nets or bigraphs.

The basic technical idea is very simple and can be expressed fairly concisely within a single paragraph: a formalism is equipped with a labelled transition system (lts) semantics where the labels on the transitions out of any particular state are the smallest contexts which, when instantiated with the term corresponding to that state, can reduce. If the notion of "smallest" is well-behaved enough, the resulting synthesised lts is very well-behaved – for instance many popular lts-based equivalences are congruences.

Our main source of inspiration shall be the field of process calculus, which is concerned with foundations of concurrent and mobile computation. The field has enjoyed wide popularity over the last 20 years, with several successful depth-first research programs. The usual approach has been to define a relatively simple (compared to industrial programming languages such as ML, Java or C) syntax-based process languages, sometimes referred to as a process algebras or process calculi. These calculi are designed so that they exhibit some fundamental aspect of computation, and research is then devoted to the study of the calculus' behavioural theory, its "expressivity" and decidability aspects. The theory of such calculi is often complicated, perhaps because of the various design decisions involved in the design of a calculus. This fragmented picture makes it difficult to extract generalised

principles which are robust, meaning that they apply in several different formalisms. As a result, the field has been described as being in a state of *flux* [66]. Thus, while the behaviour of concurrent, distributed and mobile systems can be difficult to understand, and techniques such as testing, while prevalent in industry today, have no chance of scaling up to large complex systems with concurrent and mobile aspects without the *future* development of a science which sheds light on these issues; the science which *has been* developed in order to study these problems is in itself in many places difficult, mysterious and interesting from a theoretical point of view.

The approach taken in this thesis is breadth-first, in the sense that we shall not be directly interested in such notions as synchronisation or mobility of code. Rather, we shall focus on developing a mathematical theory that can, to some extent, cover several basic concepts which have some role to play in many process calculi. Such an approach can be criticised for being too artificial; we shall, after all be concerned with "man-made" things like process-calculi, and not "natural" things such as concurrency or mobility. However, while most of the benefits of the (future) full development of the theory presented in this thesis are at the meta level (with process-calculus *designers* perhaps benefiting from the insight derived from a general treatment several basic issues common to many calculi) it could be argued that such a general approach may help in isolating robust common principles of important sub-concepts under the umbrella of concurrency or mobility.

In this sense, the approach of this thesis is related to the development of a domain theory for concurrency [81, 78], which advocates the use of mathematics to guide the design of process calculi [79, 80], instead of the, more common, reverse methodology of expending much effort on understanding particular ad-hoc process languages with the use of mathematics. Similarly, the research presented in this thesis shares the idea of finding an underlying formalism in which one can study some of the issues which occur in existing process languages with Milner's work on action calculi [71] and bigraphs [73], as well as with Gadducci and Montanari's work on tile models [35]. Differently from the first two of these, we do not introduce a monolithic model into which we find encodings of other formalisms. The idea is rather to build from bottom-up instead of top-down, i.e. start with basic structures and study their theory instead of starting with a powerful model which is capable of subsuming other formalisms via encodings. In this facet, the approach taken in this thesis is consistent with mathematical tradition of simplifying complex situations into a simple yet rich structure which is amenable to systematic study.

The material presented in this thesis is intended as a contribution in the

field of concurrency theory. Since much of it relies on using the language and technology of category theory, parts of it may be considered to be in the field of applied category theory. The category theory contained used and assumed is standard and well-studied: these concepts include 2-categories [54], bicategories [7], bicolimits [100, 53] and extensive categories [12]. This is largely the point: by finding the right mathematical structures to model concurrent (and other) computational phenomena one can use well-understood and elegant tools to solve problems, instead of developing specialised ad-hoc models from scratch. Indeed, it can be argued that research which applies well-known mathematics to problems in computer science is more valuable to computer science in the long run than the development of novel, specialised mathematics in order to tackle them. We shall present a little novel category theory within this thesis, namely the classes of *adhesive* and *quasiadhesive* categories [59] and argue that they are both natural and useful.

Reaction semantics. By a reaction[1] semantics we mean an unlabelled transition system, usually generated by closing a small set of *reaction rules* under *reactive* (evaluation) contexts. An agents p *reacts* into an agent q when there has been an interaction (specific to the calculus) inside p which, after its application, results in the agent q. The actual technical mechanism of performing a reaction can be seen as an instance of term rewriting; at least in examples where terms are syntactic and not quotiented by exotic structural congruences.

The basic setup involving contexts (which organise themselves as a category, with substitution as composition), rules and reactive contexts corresponds to a mathematical structure: Leifer and Milner's notion of *reactive system* [66]. A reactive system, thus, consists of an underlying category $\mathbf{C}$ with a chosen object 0 and a collection $\mathbf{D}$ of arrows of $\mathbf{C}$ called reactive contexts[2]. The arrows with domain 0 are usually called terms or agents, other arrows are contexts. The reaction rules are of the form $\langle l, r \rangle$, where $l : 0 \to C$ is the redex and $r : 0 \to C$ is the reactum. Notice that the rules are *ground* in that they are terms and do not take parameters. One generates a *reaction relation* $\longrightarrow\!\!\triangleright$ by closing the reaction rules under all reactive contexts; we have $p \longrightarrow\!\!\triangleright q$ if, for some $d \in \mathbf{D}$, we have $p = dl$ and $q = dr$. The advantage of a theory at least partly based in the language of

[1] Many authors use the term 'reduction' instead of 'reaction'. We shall use 'reaction' because the word 'reduction' is related to the concept of termination, and termination is usually not an interesting notion in concurrency theory.

[2] There are some additional constraints on the set of reactive contexts which we do not specify here.

category theory is that the constructions and proofs are performed on an abstract level, meaning that they are portable across a range of models.

In many cases, modern presentations of well-known process calculi have their semantics formalised in terms of an underlying rewriting system. This includes the more recent incarnations of CCS [70,72][3], the Pi-calculus [74,72, 89][4] and the Ambient Calculus [13][5]. These calculi are all syntax based, but have non-trivial structural congruences associated with the syntax. Taking the terms and contexts up to structural congruence clearly results in a setting where substitution is associative. Moreover, they all have specialised notions of reactive contexts; in CCS for instance, any context which has its hole under a prefix does *not* preserve reaction and thus, in our terminology, is *not* reactive. Thus, all of these calculi can be seen as instances of reactive systems. Much of the theoretical development which is presented in this thesis can be seen as an effort to extend Leifer and Milner's original theory.

Process equivalence. There have been various attempts at defining process equivalences starting with the reaction semantics. The notion of process equivalence is of fundamental importance, both theoretically and for practical reasons. For theorists, a natural contextual process equivalence is a starting point in the development of bisimulation-based proof techniques, logical characterisations, model checking of restricted classes and so forth. More practically, process equivalence may be used, for instance, to check that a program adheres to its specification; assuming an a priori encoding of both the program and the specification into a chosen formalism.

The idea of generating a process equivalence using contextual reasoning goes back to the definitions of Morris-style process equivalences of the simply typed and the untyped variants of the lambda calculus [5], as well as other functional formalisms. In the field of process calculus and process algebra, such equivalences are sometimes called *testing* equivalences [40].

We shall now discuss some of developments in the quest of finding general techniques for generating equivalences from reaction rules which are relatively robust in that they are not specialised to a single process calculus. The first is the notion of *barbed congruence* by Milner and Sangiorgi [75]. In that article, the authors first study *reduction bisimulation* which involves comparing the internal evolutions of processes. The equivalence this gives is very coarse, and in order to obtain something sensible, one has to close

[3] fundamental notion: synchronisation on names.

[4] fundamental notion: name passing, with the associated notion of scope extrusion. Early exploratory work in this field was done by Engberg and Nielsen [28].

[5] fundamental notion: spatial mobility of process code.

contextually (in one of two possible ways, as we shall discuss later). Milner and Sangiorgi do this in CCS, obtaining *reduction congruence*. The resulting process equivalence is coarser than bisimilarity on the standard labelled transition system semantics, but the correspondence is close. The reason for the mismatch is, essentially, that a congruence built up from reactions does not distinguish certain processes with infinite internal behaviour. To fix the congruence, Milner and Sangiorgi proposed adding an extra ad-hoc notion of observable based on the underlying syntax of CCS. This extra notion of observable is known as a *barb*. Their work has proven very influential and can be repeated for other calculi [13, 105, 14, 39], with the notion of barb chosen ad-hoc in each calculus, using calculus-specific intuition.

An important study which develops a process equivalence based purely on reactions is by Honda and Yoshida [43] who, based on intuitions from the λ-calculus, build equational theories directly from rewrites requiring no a priori specification of observables. They achieve this by using reduction and contextual closure as well as the equating of *insensitive* terms. These are terms which can never interact with their environment or, in other words, can never contribute to a reaction with a context. This elegant characterisation of a useful equivalence which is robust across many formalisms and relies only on the underlying reaction semantics is close in spirit to the aims of this thesis. The full investigation of how the theory presented within this thesis relates with Honda and Yoshida's approach is left as important future work.

Given a reduction bisimulation equivalence, one can obtain a sensible congruence in two possible ways. First, Honda and Yoshida [43] advocate obtaining a congruence by considering the largest congruence contained in bisimilarity which is also a bisimulation (or, equivalently, postulating congruence in the definition of a bisimulation relation and then considering the resulting bisimilarity). Similarly, an earlier work by Montanari and Sassone [76] obtains a congruence from bisimilarity[6] by considering the largest congruent bisimulation. Alternatively, Milner and Sangiorgi's barbed congruence is defined as follows: two processes p and q are barbed congruent if, given any context c, $c[p]$ and $c[q]$ are barbed bisimilar. This yields the largest congruence contained in bisimilarity. The first approach gives, in general, a finer congruence. This is because any relation which is both a congruence and a barbed bisimulation is clearly included in barbed congruence. On the other hand, the reverse direction is not true in general as barbed congruence may not be a barbed bisimulation.

[6] More precisely, weak bisimilarity on the lts semantics of CCS.

Fournet and Gonthier [32] have confirmed that barbed congruence in the style of Milner and Sangiorgi coincides with the barbed congruence in the style of Honda and Yoshida (usually called reduction equivalence) in the setting of the Pi-calculus. In other process calculi, the situation is less clear. Because of the close correspondence with theories for functional programming languages and the elegance and canonicity of the definition, we shall consider only congruences in the style of Honda and Yoshida, and Montanari and Sassone.

Equivalences which are based on an underlying reduction system and are generated contextually have both advantages and disadvantages. Their chief advantage is their naturality, in the sense that it is often relatively easy to justify their correctness and appropriateness as notions of equivalence. A disadvantage of barbed congruence in particular, is that the barbs, or observables, are a usually of a rather ad-hoc syntactic nature, specific to each calculus. An important common problem of contextually defined equivalences is that it is often very difficult to prove directly that two process terms are equivalent. The main complication follows from the quantification over all contexts, usually an infinite number. Thus, in order to prove equivalence directly, one has to construct a proof based on structural induction; this, when possible, is usually a tedious and a complicated procedure.

We should note that contextually based equivalences based on reduction rules naturally come in *strong* and *weak* variants. A strong equivalence allows one to distinguish processes which vary only in how they react internally, while weak equivalences aim to abstract away from internal reaction. Although weak equivalences are more suitable as a notion of observational equivalence, we shall concentrate our theoretical development on strong equivalences. We shall discuss weak equivalences further at a later point in this introduction.

Labelled transition systems. An elegant solution to the problem of universal quantification over the usually infinite set of contexts is to endow a process calculus with an appropriate labelled transition system (lts) semantics. Before we explain what is meant by 'appropriate' in this setting, we shall recall some of the basic theory behind lts semantics. Labelled transition systems have been a very popular tool in theoretical computer science, not least because of their origins in classical automata theory. Indeed, some process calculi, including the earlier variants of the well known CCS [70], have their semantics *a priori* formalised in terms of an lts; the use of reduction based semantics and structural congruence only becoming fashionable after

Berry and Boudol's influential work [9] on the chemical abstract machine.

A labelled transition system consist of a set of states S and a set of labelled transitions T. A transition has a domain state, a codomain state and a label from some, usually fixed, set A of "actions". Technically, the set of transitions is usually considered to be a subset of the cartesian product $S \times A \times S$ which brings with it the usual restriction of there being at most one transition with label a between any two states. Although the intuition may vary between applications, it is often the case that a transition with label a from state s to state s' means that s can participate in an interaction which the symbol a represents, and by doing so, evolve into s'. Although our use of the term "interaction" is intentionally meant to be vague, when there is an underlying reduction semantics such an interaction could be represented by a reaction.

Labelled transition system semantics facilitate a large number of equivalences which vary depending on how much branching structure is taken into consideration. Thus, one of the coarsest (relates most) is the trace preorder and associated equivalence because no branching is taken into consideration. Park's notion of *bisimilarity* [82], adapted for labelled transition systems by Milner [70], is at the other end of the spectrum [104], meaning that it examines all branching structure and is the finest (relates least) of such equivalences. Bisimilarity is often denoted $\sim$.

The notion of bisimilarity has stimulated much research because it is canonical from a number of perspectives. Firstly, it has a elegantly simple coinductive definition, meaning that in order to prove that two states of an lts are bisimilar, it is enough to construct a bisimulation which contains them. Secondly, it has an elegant game-theoretic characterisation in terms of the so-called bisimulation game. Thirdly, there is an elegant and simple logical characterisation in terms of the well-known Hennessy-Milner logic [41]. Finally, there are two, so far largely unrelated general approaches to bisimilarity. The first is usually known as the coalgebraic approach, where a bisimulation is sometimes defined as a spans of coalgebra morphisms for some functor [88]. This is a very general approach which recovers the notion of ordinary bisimulation for a particular endofunctor on the category of sets, namely $\mathcal{P}(A \times X)$ where A is the set of labels of the lts and $\mathcal{P}$ is the power set. Usually one restricts to the finite power set $\mathcal{P}_f$ (which corresponds to the technical assumption of requiring the lts to be *finitely branching*) in order for the final coalgebra to exist [3, 6]. Observational equivalence, when final coalgebras exist, is sometimes taken to mean equality under the unique mapping to the final coalgebra. Span bisimilarity and observational equivalence via the map to the final coalgebra yield the same equivalence under

certain assumptions on the underlying endofunctor. The second general approach to bisimulation is the open map approach [48], where a bisimulation is taken as a span of so called open maps in a category of transition systems and simulations. Open maps are taken with respect to an ad-hoc underlying subcategory of open maps, which led to the study of presheaf categories where such path categories are canonical via the Yoneda embedding.

While all of the above form an impressive body of theory on bisimilarity, they all start off with the following assumption - a predefined set of actions A over which the labelled transition systems are built in some, usually unspecified way. Indeed, even the fact that the states of the lts correspond to the terms of some formalism is usually abstracted away.

A work in the general area of combining lts semantics with some notion of syntax is the seminal paper by Turi and Plotkin [102] which combines the coalgebraic approach with structural operational semantics [84] (and in particular the GSOS [10] format) in a comprehensive theory known as *bialgebraic semantics*. Similar ideas have recently been pursued by Corradini, Heckel and Montanari [17], who used a coalgebraic framework to define labelled transition systems on algebras.

The area of bialgebraic semantics is an exciting field with ongoing research into extending the basic theory with the generation of new names [31, 30] and equivalences other than bisimilarity [56,55]. Such developments yield insights into labelled transition systems and subformats of GSOS which guarantee congruence properties in such settings. However, even in bialgebraic semantics, the labels of the lts are assumed to come from some fixed ad-hoc set of observable behaviours which one is meant to provide a priori for each setting.

Labelled transition systems for reactive systems. Returning to the question of what constitutes an appropriate labelled transition system for a formalism with an underlying reaction semantics, certainly bisimilarity on such an lts should be at least *sound* with respect to the contextually-defined equivalence, meaning that to prove that two terms are contextually equivalent it is enough to show that they are bisimilar. In some cases, bisimilarity is also *complete* (or fully-abstract) with respect to the contextually-defined equivalence, meaning that the two notions of process equivalence – bisimilarity and contextually-defined equivalence – actually coincide, and one can always, in principle, find a bisimulation for any two contextually equivalent processes.

Thus the chief advantage of such a suitable lts is that, in order to prove

the equivalence of two processes, one can use the power of coinduction and construct a bisimulation which includes the two processes. This task is usually more attractive and easier then the messy structural inductions involved in proving contextual equivalence defined using quantification over an infinite set of contexts.

There has been much research concerned with finding suitable labelled transition system semantics for different reaction-based formalisms. Unfortunately, from a theoretical point of view, the labels of such a semantics, if it exists, may seem ad-hoc; they need to be tailored and locally optimised for each process language under consideration. Indeed, the task of identifying a "natural" lts for a particular calculus is often far from obvious, even when its semantics is well understood. On the contrary, labelled transition systems are often intensional: they aim at describing observable behaviours in a compositional way and, therefore, their labels may not be immediately justifiable in operational terms. For example there are two alternative labelled transition system semantics for the Pi-calculus [74], the early and the late version, each giving a different bisimulation equivalence.

An additional benefit of full abstraction and a property of considerable importance in its own right is *compositionality* of lts bisimilarity (and of other useful lts preorders and equivalences). A relation is compositional, in other words a *congruence*, if whenever we have tRu then we have $c[t]Rc[u]$ for any context $c[-]$ of the underlying language. It can be argued that congruence should be a required property of any reasonable notion of observational equivalence – if we prove that a and b are indistinguishable then they certainly should behave equivalently in any given environment.

Compositionality and coinduction work together: compositionality allows one to use modular reasoning to simplify coinductive proofs. Indeed, compositionality is highly desirable because it usually makes equivalence proofs considerably simpler. In particular, it allows the familiar methods of equational reasoning, such as substituting "equals for equals", sound. As an example, consider two nontrivial systems, each of which can be expressed as a parallel composition of two smaller systems, in symbols $p \equiv q \parallel r$ and $p' \equiv q' \parallel r'$. To show that $p \sim p'$, using compositionality it is enough to show that $q \sim q'$ and $r \sim r'$.

It is a serious problem, then, that given an lts designed ad-hoc for a particular calculus, bisimilarity is not automatically a congruence. Even when it *is* a congruence, proving that it is can be a very difficult and technical task. For example, the well-known Howe's method [44] is a technique for proving that lts bisimilarity is a congruence for certain languages with higher-order features. In the field of process calculus, such proofs usually involve finding

a close connection between the labels of an lts and the syntactic contexts of the calculus.

Interestingly, from a historical perspective, labelled transition systems as a way of formalising semantics of process calculi actually were used *before* reaction semantics. In particular, the original presentation [70] of Milner's CCS formalised the semantics with a labelled transition system presented with SOS-style rules. An early paper by Larsen [61] identified the importance of congruence results for lts based process equivalences. Starting with an lts, Larsen introduced the notion of a context (itself an lts) which is capable of consuming the actions of a state in the lts. By adding constructors (action prefix and nondeterministic choice) to the set of contexts, he proved a congruence theorem for bisimilarity. This early work can be seen as related to CCS-like calculi, since Larsen's environments can be otherwise understood as ordinary CCS contexts (with input-actions changed to output-actions and vice-versa) – with the consumption of lts labels by the context being handled by CCS interaction. Even in the basic setting of CCS, it quickly became apparent that the labelled transition systems is not the ideal technology with which to define notions of observational equivalence. For instance, weak bisimilarity in CCS is *not* a congruence. Because, as we have demonstrated, compositionality is a very useful property, Montanari and Sassone [77, 76] considered the largest congruent bisimulation contained in weak bisimilarity. Alternatively, weak observational congruence [70] considers the largest congruence contained in bisimilarity (the difference is similar to the difference between Milner and Sangiorgi's and Honda and Yoshida's approaches). These approaches became for some time accepted techniques for obtaining satisfactory notions of observational equivalence in calculi. The advent of reaction semantics and congruences obtained from reactions have since arguably replaced these approaches as "canonical" methods of obtaining an observational equivalence.

Weak equivalences. Another yardstick to measure the appropriateness of an lts for a formalism with reactions is how the lts simulates internal reduction within terms. For example, in CCS and many other calculi, there are "silent" transitions; traditionally labelled τ. Such τ transitions usually correspond closely to the underlying reaction semantics.

Having τ labels as part of an lts allows one to define a notion of *weak* bisimulation and the resulting equivalence: *weak* bisimilarity. Roughly, weak bisimilarity does not distinguish processes which differ only in internal behaviour as represented by the τ-labelled transitions. Such equivalences are

considered to be more useful from a practical point of view since it can be argued that any reasonable notion of observational equivalence should not take internal behaviour into consideration.

There are a number inequivalent ways [103] to define precisely what is meant to be a weak equivalence and the appropriateness to any particular application depends on the ad-hoc design of the particular lts. The techniques involved are usually not specialised to bisimilarity and thus one may easily define a notion of weak trace equivalence or a weak failures equivalence. One popular definition pioneered by Milner [70] is allow a (non-τ) label a to be matched by a "weak" a, which means a (possibly empty) sequence of τ labels followed by a and followed again by a (possibly empty) sequence off τs. A τ label is normally allowed to be matched by any (possibly empty) string of τs. Weak bisimilarity in CCS is *not* a congruence.

Weak equivalences have traditionally been difficult to handle in general categorical settings. Indeed, there is still no general approach based on coalgebras, although there has recently been an attempt [69] to develop the theory in this direction. The theory has been developed to a more satisfactory level in the field of open maps [29], yet the general approach advocated there is, arguably, quite technical. Surprisingly, the theory of weak bisimulation seems to be quite easily and elegantly handled in the theory of reactive systems, see Jensen's upcoming PhD thesis [45].

Deriving bisimulation congruences. We have discussed attempts by Milner and Sangiorgi [75] and by Honda and Yoshida [43] to identify general techniques at arriving at a reasonable notion of process congruence through contextual means. We have also discussed some of the problems inherent in contextual definitions and discussed one solution to the difficulties involved in quantifying over an infinite set of contexts, finding a *suitable* labelled transition system. A third development, which has led in a direct line to the the theory developed in this thesis, is by Sewell [98]. Sewell's idea is to *derive* a labelled transition system directly from the reaction semantics so that useful lts based equivalences, including bisimilarity, are automatically congruences.

Sewell's approach involved a new way of obtaining a labelled transition: the labels of transitions from a particular term should be the contexts which allow the term to react (that is, a rewrite of the term inside the context should be possible in the underlying rewriting semantics). Moreover, the labels should be, in some sense, the *smallest* such contexts. The notion of smallest was elegantly expressed in categorical terms by Leifer and

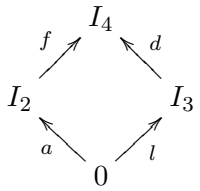


Figure 1.1: Redex square.

Milner [66].

Leifer and Milner's characterisation of the notion of smallest context utilises the fact that contexts can be organised in a category as part of a reactive system. First, the notion that a term a can be instantiated in a context f and react can be summed up by giving a commutative *redex* square, as illustrated in Figure 1.1, where d is some *reactive* context and l is the redex or a reaction rule.

Using Leifer and Milner's characterisation, the context f is the smallest such context when the diagram is an *idem pushout* (IPO). Categorically, it means that it is a pushout in the slice category over I_4. Starting with an arbitrary redex square, one obtains an IPO by constructing a *relative pushout* (RPO), which amounts to constructing a pushout in the relevant slice category.

The advantages of such a definition is that we have the universal properties of such contexts at our disposal. Indeed, Leifer and Milner [66] showed that a labelled transition system with labels being precisely the contexts which come from IPOs is very well behaved. In particular, bisimilarity is a congruence. In his PhD dissertation, Leifer [64] complemented this result by showing that trace equivalence and failures equivalence are also congruences. In the examples treated by Sewell, Leifer and Milner, bisimilarity on the labelled transition semantics obtained using this approach have corresponded closely to the expected process equivalences.

A 2-categorical approach. When applied naively, Leifer and Milner's theory has proven inadequate in reactive systems where contexts have non-trivial algebraic structure. In some cases, IPOs do not give the expected labels in the lts [93], while in others, they do not exist [92]. The troublesome contexts often exhibit non-trivial automorphisms, which naturally form a part of a 2-dimensional structure on the underlying category $\mathbf{C}$. It is important to notice that such situations are the norm, rather than the

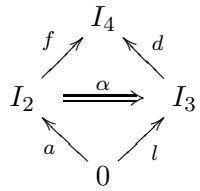


Figure 1.2: Redex square in a 2-category.

exception. Context isomorphisms arise naturally already in simple process calculi, where terms are up to structural congruence.

In more accessible terms, whereas Leifer and Milner consider categories where the objects are "holes" and arrows are contexts, we shall consider *2-categories* where the intuition for the objects and arrows is the same as for Leifer and Milner, but there is additional structure, the 2-cells. The suggested intuition that the 2-cells is a term isomorphism, in a loose sense, a "derivation" or "proof" of structural congruence. To give a redex square in this setting, it is not enough to say that a in the context of f *equals* a redex l in a reactive context d, one needs to provide an explicit isomorphism α, as illustrated in Figure 1.2. It turns out that this 2-dimensional structure is crucial and solves many of the problems involved in Leifer and Milner's original theory. The idea of using 2-cells as part of the theory of reactive systems was independently proposed by Sewell [99].

We shall demonstrate suitable generalisations of IPO and RPO (dubbed GIPO and GRPO) to this 2-dimensional setting. The associated categorical notion is no longer a pushout in a slice category but rather a bipushout [53, 100] in a pseudo-slice category. It turns out, however, that these extra complications do not detract from the good behaviour of the resulting lts; bisimilarity as well as trace and failures equivalences are congruences.

Leifer and Milner, aware of the problems which arise as a consequence of discarding the 2-dimensional structure, have also introduced technology in order to deal with these issues. The main developments have centered around Leifer's *functorial reactive systems* and Milner's S-*precategories* [46], also known as well-supported precategories and S-categories. These solutions have a similar flavour: decorate the contexts by so-called "*support sets*" which identify elements of the contexts so as to keep track of them under arrow composition. This eliminates any confusion about which automorphism to choose since diagrams can now be commutative in only one way. Unfortunately, such supported structures no longer form categories –

arrow composition is partial – which has the effect of making the theory laborious and based in part on set theoretical reasoning and principles.

We shall present a general translation which maps reactive systems on precategories to reactive systems on 2-categories in a way which ensures that the lts generated using the 2-categorical approach is the same as the lts generated using the technology functorial reactive systems or S-precategories. The translation derives a notion of isomorphism, specific to the particular structure in hand, from the precategory's support information. Such isomorphisms constitute the 2-cells of the derived 2-category. We shall argue that this yields an approach mathematically more elegant and considerably simpler than precategories. Moreover, while subsuming the previous theories, it appears that the 2-categorical theory is more general: there is no obvious way of reversing the translation and obtaining an S-precategory from a general 2-category.

One of the contributions of this thesis which follows directly from the 2-categorical approach to reactive systems and structural congruence is a "categorisation" of bigraphs. Bigraphs were introduced by Milner in his conference presentation [73] and later in the comprehensive technical report by Jensen and Milner [46]. They aim at modelling systems with two orthogonal modes of connectivity. The first mode is a physical link structure, which may for instance correspond to a physical nesting of systems similar to the nesting of process terms in the ambient calculus [13], or Alastair living next door to Beatrice. The second mode of connectivity is a logical link structure, which may correspond to processes knowing a reference to a resource of an another process, as, for example a process in the Pi-calculus [74] knowing a free name of another process, or Alastair knowing Beatrice's email address. The two sorts of connectivity are orthogonal in the sense that the physical separation of processes should not have an effect on the ability to maintain logical links. Bigraphs are algebraic structures with an underlying carrier set. Thus, in this sense they are like ordinary graphs, groups, modules, fields and many other well-studied mathematical objects. However, bigraphs, unlike the aforementioned structures, do not have an associated notion of homomorphism. F. W. Lawvere has written in [63]: "The crystallized philosophical discoveries which still propel our subject include the idea that a category of objects of thought is not specified until one has specified the category of maps which transform these objects into one another and by means of which they can be compared and distinguished." When applying our translation from the precategory of bigraphs to a 2-category we obtain a natural notion of bigraph isomorphisms as the 2-cells.

There have been several applications of the theory of 2-categories to

computer science, see for example [8, 97, 101, 34, 16]. The 2-dimensional structure has been typically used to model a small-step reduction relation, say in the simply-typed lambda calculus. As in our examples, the objects of the 2-categories are types and the arrows are terms. However, for us the 2-dimensional structure consists of isomorphisms between terms, in other words, structural congruence, and the rewrite relation is external to the 2-category. Indeed, there is a fundamental problem in modelling the rewrite relation as 2-cells in our examples, if we allow non-reactive contexts (as, say, prefix in CCS or lambda abstraction in the lazy lambda calculus) as arrows in the category. This is because the axioms of 2-categories ensure that all arrows preserve reaction through horizontal composition with identity 2-cells; otherwise known as "whiskering". In symbols, if $\alpha : f \Rightarrow g : X \to Y$ is a 2-cell then for any $h : Y \to Z$ we have that $h\alpha : hf \Rightarrow hg : X \to Z$ is a 2-cell.

Adhesive categories. In order to understand constructions on structures like bigraphs at a general level, we shall require a natural class of categories which includes many different notions of graphical structures used in computer science and at the same time has enough structure which allows us to prove useful results. This leads to the novel category theory introduced within the thesis: the classes of adhesive and quasiadhesive categories. We shall develop enough of their theory in order to illustrate some of their structure, as well as to justify their use in the fields of graph transformation systems. They have been developed as joint work together with S. Lack, see the conference paper [59].

As is the case with the well-known class of extensive [63, 96, 12] categories, adhesive categories have a simple axiomatic definition as well as an elegant "equivalence" of categories definition. Indeed, the idea behind the development of adhesive categories was to find a class of categories in which pushouts along monomorphisms are "well-behaved" – meaning they satisfy some of the properties of such pushouts in the category of sets and functions **Set** – in much the same way as coproducts are "well-behaved" in extensive categories.

An example of the good behaviour of such pushouts is that they are stable under pullback (the dual notion to pushout). The idea is analogous to that of extensive categories [12], which have well-behaved coproducts in a similar sense. Since coproducts can be obtained with pushouts and an initial object, and an initial object is "well-behaved" if it is strict, one might expect that adhesive categories with a strict initial object would be extensive, and

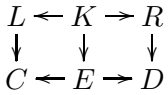


Figure 1.3: Double pushout.

this indeed turns out to be the case.

Adhesive categories include as examples many of the graphical structures used in computer science. This includes ordinary directed graphs, typed graphs [4] and hypergraphs [23], amongst others. The structure of adhesive category allows us to derive useful properties. For instance, the union of two subobjects is calculated as the pushout over their intersection, which corresponds well with the intuition of pushout as generalised union.

One can develop a rich *general* theory of double-pushout (dpo) rewriting [27] within adhesive categories. Dpo *graph* rewriting was first introduced in order to formalise a way of performing rewriting on graphs. It has been widely studied and the field can be considered relatively mature [87, 20, 26].

In dpo rewriting, a rewrite rule is given as a span $L \leftarrow K \rightarrow R$. Roughly, the intuition is that L forms the left-hand side of the rewrite rule, R forms the right-hand side and K, common to both L and R, is the sub-structure to be unchanged as the rule is applied. To apply the rule to a structure C, one first needs to find a match $L \rightarrow C$ of L within C. The rule is then applied by constructing the missing parts (E, D and arrows), as illustrated in Figure 1.3, in a way which ensures that the two squares are pushout diagrams. Once such a diagram is constructed we may deduce that $C \Longrightarrow D$, that is, C rewrites to D.

Dpo rewriting is formulated in categorical terms and is therefore portable to structures other than directed graphs. There have been several attempts [23, 21] to isolate classes of categories in which one can perform dpo rewriting and in which one can develop the rewriting theory to a satisfactory level. In particular, several axioms were put forward in [23] in order to prove a local Church-Rosser theorem for such general rewrite systems. Additional axioms were needed to prove a general version of the so-called concurrency theorem [57].

We shall define *adhesive grammars* which are dpo rewrite systems on adhesive categories and show that the resulting rewriting theory is satisfactory by proving the local Church-Rosser theorem and the concurrency theorem without the need for extra axioms. Indeed, we shall argue that adhesive categories provide a natural general setting for dpo rewriting. We also ex-

$$I_1 \xrightarrow{\iota} C \xleftarrow{o} I_2$$

Figure 1.4: Cospan from I_1 to I_2.

amine how adhesive categories fit within the previously conceived general frameworks for rewriting [23, 21]. Many of the axioms put forward in [23] follow elegantly as lemmas from the axioms of adhesive categories.

Cospans. Several constructions of RPOs have been proposed in the literature for particular categories of models. For example, Leifer [64] constructed RPOs in a category of action graphs, while Jensen and Milner did so in the (pre)category of bigraphs [73]. A construction of (G)RPOs in a general setting has so far been missing.

A general construction, provided that it covers several different models and the techniques used are robust, is quite useful. The reasons for this include:

- it provides a general intuition of how to construct GRPOs in many different settings, without having to provide model-specific constrictions and proofs;
- it allows the relating of different models as subcases of a more general setting;
- it allows one to vary the model within the specified constraints and retain the construction.

Throughout the thesis we shall deal with categories where the arrows represent contexts. An interesting question thus arises; what is a reasonable general notion of context which nonetheless has more structure than an arrow of an arbitrary category? We shall argue that the notion of cospan is suitable. Given objects I_1 and I_2 of some category $\mathbf{C}$, a cospan from I_1 to I_2 is simply a diagram in $\mathbf{C}$, as illustrated in Figure 1.4, where C is an object of $\mathbf{C}$ and the arrows are arbitrary. We shall refer to $\iota : I_1 \to C$ and $o : I_2 \to C$ as, respectively, the *input* and *output interface* of the cospan. Note that, as it stands, the notion of cospan is symmetric, and the same diagram forms a cospan from I_2 to I_1 with o forming the input interface and ι the output interface.

The rough intuition is that C corresponds to a "black box" computational environment, with some of its parts available through I_1 to its sub-

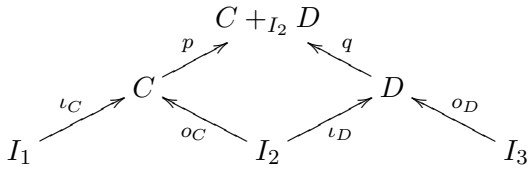


Figure 1.5: Composition in Cospan($\mathbf{C}$)

components, or variables; and others available publicly through I_2, which can be used to embed C in a larger system.

Given two cospans, $I_1 \xrightarrow{\iota_C} C \xleftarrow{o_C} I_2$ and $I_2 \xrightarrow{\iota_D} D \xleftarrow{o_D} I_3$, one can compose them to obtain a cospan from I_1 to I_3 by constructing the pushout, as illustrated in Figure 1.5, and letting the input interface be $p\iota_C$ and the output interface be qo_D. Such composition has an identities, the identity cospan on I_1 is $I_1 \xrightarrow{\text{id}} I_1 \xleftarrow{\text{id}} I_1$.

Cospans in $\mathbf{C}$ actually organise themselves as arrows of another category, or more accurately, the *bi*category Cospan($\mathbf{C}$). This bicategory has the same objects as $\mathbf{C}$ but the arrows from I_1 to I_2 are cospans and the 2-cells are cospan isomorphisms - isomorphisms $f : C \to C'$ of $\mathbf{C}$ which preserve input and output interfaces, that is $f\iota = \iota'$ and $fo = o'$.

A bicategory [7] can be described roughly as a 2-category where the horizontal composition is associative and has identities up to an isomorphic 2-cell. Composition of cospans is not associative on the nose because composition uses the pushout construction which is defined up to isomorphism. The associativity and identity isomorphisms are required to satisfy the so-called coherence conditions (including the famous Mac Lane pentagon for associativity [7,68]). It turns out that the canonical isomorphisms obtained using the universal property of pushouts do satisfy these conditions.

As an example of these concepts, consider the simple model of a coffee vending machine, illustrated by the leftmost diagram of Figure 1.6. It has an output interface consisting of two nodes, \$ and C, which one can think of as a money slot and the coffee out-tray. These are the parts of the coffee machine accessible to the environment, the internal components, represented by S, are invisible. The middle diagram represents a coffee drinker. He expects to see a money slot and a coffee out-tray, which are his input interfaces. As the output interface of the coffee machine and the input interface of the coffee drinker match, one may compose them and obtain the system pictured in the rightmost diagram. (The input interface of the vending machine and the output interface of the coffee drinker have been omitted.)

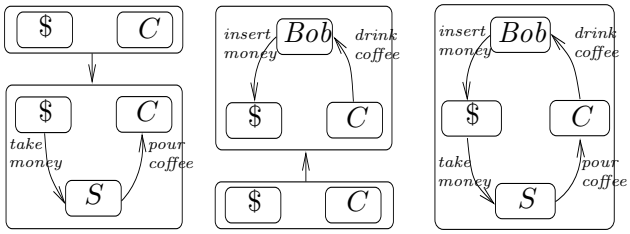


Figure 1.6: Example of a contextual system.

Construction of GRPOs. The central contribution of this thesis is the construction of GRPOs, our generalisation of RPOs to a 2-categorical setting, in input-linear cospan bicategories over adhesive categories. By an input linear cospan, we mean a cospan as in Figure 1.4 but where the input interface ι is mono. Observe that this breaks the symmetry of cospans: to give an input-linear cospan from I_1 to I_2 is not the same thing as to give an input-linear cospan from I_2 to I_1. When $\mathbf{C}$ is an adhesive category, the composition of two input-linear cospans in $\mathbf{C}$ gives an input-linear cospans: they form the bicategory ILC($\mathbf{C}$).

Although technical in nature, the linearity condition does have an intuitive account. As alluded in the coffee drinker example, one can consider a cospan as a "black box," with an input interface and an output interface. The environment cannot see the internals of the system and only interacts with it through the output interface. The fact that the output interface need not be linear means that the system is free to connect the output interface arbitrarily to its internal representation. For example, the coffee machine could have two extra buttons in its output interface; the "café latte" button and the "cappuccino" button. The machine internals could connect both these buttons to the same internal trigger for coffee with milk; the point is that the system controls its output interface and is able to equate parts of it. On the other hand, the system cannot control what is plugged into one of its holes. Thus, an assumption of input-linearity is essentially saying that the system does not have the right to assume that two components coming in through the input interface are equal.

The construction arose from an effort to understand the structure of GRPOs in categories of contexts where the contexts have graphical structure. Incidentally, it is the non-trivial algebraic structure of such contexts that makes it essential to consider 2-dimensional structure in of such categories; it is not enough to deal with the "abstract" versions (where the contexts are

quotiented by isomorphism) and consider RPOs. The construction is the first construction of GRPOs for general class of models.

We shall demonstrate two applications of the construction. Firstly, using an insight of Gadducci and Heckel [33] we notice that dpo graph rewriting systems can be seen as certain rewriting systems on cospan categories over the category of directed graphs and homomorphisms **Graph**, and thus can be seen as reactive systems. Since **Graph** is an adhesive category, we are able to derive labelled transition systems for a general class of dpo graph rewriting systems. One of the advantages of this technology is that it facilitates a transfer of concepts between the theories and technologies of process algebra and graph rewriting. Indeed, it becomes possible to think of graph rewriting systems as certain calculi, with cospans providing a notion of context. Interestingly, the construction of labelled transition systems captures and extends the borrowed context approach of Ehrig and König [25] who also derive labelled transition systems for double-pushout graph rewriting systems. Indeed, it becomes possible to see their work as part of the framework of reactive systems and GRPOs. The transfer of technology is in both directions, using Ehrig and König's characterisation of labels, we are able to provide a pleasantly simple characterisation of GIPOs in our setting.

Our second application shall consider Milner's bigraphs [73]. We shall see how bigraphs can be presented as cospan bicategories (which, incidentally, gives an automatic notion of bigraph *homomorphism*). It turns out that we do not capture Milner's theory of RPOs for bigraphs precisely, indeed, in order to apply our construction we need to deal with input-linear cospans. Requiring input-linearity corresponds to taking a slightly different notion of bigraph then the one treated by Milner. Indeed, it turns out that the category of bigraphs in Milner's sense is actually isomorphic to a certain bicategory of output-linear cospans over an adhesive category. As a consequence, it shall be interesting to investigate whether a general construction of GRPOs can be given for output-linear bicategories.

Cospans as well as spans have been used in computer science before. As previously mentioned, Gadducci and Heckel [33] have used cospans to shed light on connections between dpo graph rewriting and standard rewriting theory. In an effort to study a general notion of "partial map", Robinson and Rosolini investigated a particular class of span bicategories in [85]. Spans have also been studied by Katis, Sabadini and Walters [50, 51] in an effort to generalise ordinary automata theory in a modular way using spans. Moreover, using the technology of traced monoidal categories [49], they were able to include a "feedback" operation. Thus, as our cospans can be thought of as generalised contexts, their spans can be thought of as generalised au-

tomata. It is unclear at this stage what connection can be made between the two theories.

The π-calculus and the λ-calculus. The theory developed in this thesis aims at the further development of a general theory which deals uniformly with formalisms which have an underlying reduction semantics. This, on the surface, includes both the λ-calculus and the more recent work on the π-calculus. However, we do not deal with either of these fundamental calculi in any great detail within the thesis. The main reason for this is that the theory of the calculi, while sharing some essential features, are specialised for their particular domains – thus it is perhaps unreasonable to expect one theory to cover the semantics of both the calculi in a canonical way, in the sense that there are no questionable encodings of the calculi within a third calculus. However, the theory is in an early stage and future developments may suggest canonical approaches to cover the specialised theories of the two calculi to a greater extent.

1.2 Structure of the thesis

We shall start with section 1.3 which introduces some very basic but useful lemmas which we shall rely on throughout. Chapter 2 is a self-contained introduction to Leifer and Milner's theory of reactive systems [66] and its extension to 2-dimensional categories [91, 93].

Chapter 3 begins with of an introduction to bicolimits, which are used throughout the thesis under the guise of GRPOs. Secondly, several technical lemmas used for the congruence theorems presented in Chapter 2 are proved in full detail.

In Chapter 4 we shall provide constructions of GRPOs in for the simple yet illustrative examples studied in Chapter 2, thus demonstrating that the 2-categories can be used to successfully solve the problematic issues associated with RPOs. The chapter starts with an introduction to extensive categories [12], which are used in the construction. Finally, we show how previously introduced theory by Leifer and Milner (which includes the concepts of functorial reactive systems and precategories) can be translated into the 2-categorical setting [92]. Because the 2-categorical setting with the associated congruence theorems appears to be more general, and is arguably more elegant than the other approaches, we shall argue that this theory subsumes these previous efforts.

In Chapter 5 we shall introduce adhesive [59] and quasiadhesive categories and argue that they are natural, robust (under typical categorical constructions) and useful classes. Adhesive categories can be described roughly as categories with well-behaved pushouts along monos, while quasiadhesive categories have well-behaved pushouts allow regular monos. After presenting the definition, we shall develop several useful properties of adhesive categories. Finally we show that one can derive a rich dpo theory within adhesive categories by proving some of the more well-known theorems such as local Church-Rosser and the concurrency theorem.

Chapter 6 combines much of the theory presented in the other parts of the thesis and presents the main result of the thesis: GRPOs exist in a general class of cospan bicategories - the class of input-linear cospans over adhesive categories. While the construction and the proof are done at the level of pure category theory, there are several applications. Firstly, we are able to relate the fields of reactive systems (with the associated process calculus heritage) and graph-transformation – facilitating a transfer of concepts and introducing a "contextual" point of view to graph rewriting. This is because any dpo graph rewriting system corresponds to a reactive system on a cospan bicategory [33]. The (concrete) labelled transition system derived using the technology presented in this thesis in for a reactive system over the input-linear cospan bicategory over the (adhesive) category of directed graphs is *equal* to the labelled transition system obtained independently by Ehrig and König in their rewriting via borrowed contexts [25].

Much of the content of chapters 2 and 3 appeared first in the workshop paper [91] and its journal version [93] as joint work with Vladimiro Sassone. The main results of chapter 4 appeared as the conference paper [92] and shall appear in the upcoming journal version [95], also joint work with Vladimiro Sassone. Much of the development in chapter 5 has appeared in the conference paper [59] and is joint work with Stephen Lack. Finally, the results of chapter 6 have appeared in the technical report [94] as joint work with Vladimiro Sassone.

1.3 Preliminaries

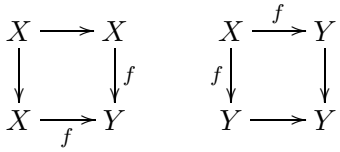
While a basic knowledge of category theory is assumed, standard categorical concepts like 2-categories and bicolimits shall be introduced in some detail within the body of the thesis. Here we recall a few basic categorical results which shall be used in many places throughout the thesis.

We shall follow the convention of not labelling identity morphisms in

diagrams. All composition shall be written in the "function" order and usually denoted by juxtaposition, so that the composition of $f : X \to Y$ and $g : Y \to Z$ shall be written gf.

Pushouts and pullbacks First, we recall a basic characterisation of monos (arrows $g : Y \to Z$ such that, for any pair $f_1, f_2 : X \to Y$, if $gf_1 = gf_2$ then f_1 and f_2) and, dually, epis.

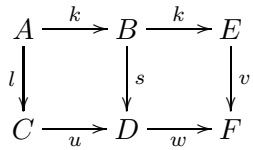
Proposition 1.3.1. An arrow $f : X \to Y$ is mono if and only if diagram (i) is a pullback. Dually, $f : X \to Y$ is epi if and only if diagram (ii) is a pushout.



(i) (ii)

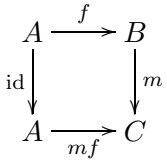
We shall repeatedly use basic properties of pushouts and pullbacks, and in particular, the following well-known lemma, sometimes referred to as the pasting lemma. Since pushouts and pullbacks are dual, there are two versions of the lemma.

Lemma 1.3.2. Given a commutative diagram:



- *Pullback version* - If the right square is a pullback then the left square is a pullback iff the whole rectangle is a pullback.
- *Pushout version* - If the left square is a pushout then the right square is a pushout iff the whole rectangle is a pushout.

Lemma 1.3.3. If $m : B \to C$ is mono then for any $f : A \to B$ the following diagram is a pullback.



Proof. Given an arbitrary object X and morphisms $\alpha : X \to A$ and $\beta : X \to B$ such that $mf\alpha = m\beta$, we use the fact that m is mono to get $f\alpha = \beta$ which gives immediately that α is the required unique mediating morphism. □

We say that a pullback diagram is a pullback *along* a morphism $f : A \to B$ if it is a pullback of $f : A \to B$ and some $g : C \to B$ as illustrated below:

$$\begin{array}{ccc} P & \xrightarrow{q} & C \\ {\scriptstyle p}\downarrow & & \downarrow{\scriptstyle g} \\ A & \xrightarrow[f]{} & B. \end{array}$$

Similarly, if the diagram above is a pushout, then it is a pushout along p. It is a pushout along a mono if (at least) one of p and q are mono.

The following is a very simple lemma which relates pushouts and coproducts. It can actually be seen as a consequence of the more general principle of pushouts and coproducts "commuting", since they are both just certain colimits.

Lemma 1.3.4. Suppose that the diagram (i), below, is a pushout. Then diagram (ii) is also a pushout.

$$\begin{array}{ccc} A & \xrightarrow{g} & C \\ {\scriptstyle f}\downarrow & & \downarrow{\scriptstyle f'} \\ B & \xrightarrow[g']{} & D \end{array} \qquad (i)$$

$$\begin{array}{ccc} A+E & \xrightarrow{g+\mathrm{id}_E} & C+E \\ {\scriptstyle f+\mathrm{id}_E}\downarrow & & \downarrow{\scriptstyle f'+\mathrm{id}_E} \\ B+E & \xrightarrow[g'+\mathrm{id}_E]{} & D+E \end{array} \qquad (ii)$$

Proof. Suppose there exist $\varphi : B + E \to X$ and $\psi : C + E \to X$ so that $\varphi(f + \mathrm{id}_E) = \psi(g + \mathrm{id}_E)$. This implies that $\varphi i_1 f = \psi i_1 g$ and $\varphi i_2 = \psi i_2$. Using the fact that the left square is a pushout, there exists a unique map $h : D \to X$ such that $hg' = \varphi i_1$ and $hf' = \psi i_1$. Then $[h, \varphi i_2] : D + E \to X$ is the required unique morphism. □

Chapter 2

Reactive systems and GRPOs

Perhaps the majority of the formalisms which have been developed in order to express different kinds (sequential, concurrent, probabilistic, etc.) of computation can trace their ancestry to the untyped lambda calculus. In this chapter we shall isolate several primitive notions common to many different formalisms and study a technique of deriving well-behaved labelled transition system semantics in this general setting.

The untyped lambda calculus can be considered as a simple formal term rewriting system, modulo an equivalence relation on terms, known as α-equivalence. The application of a rewriting step in such a term rewriting system is often referred to as a *reduction*. The main "computation" in the setting of untyped lambda is performed by the so-called *β-reduction rule*

$$(\lambda x.M)N \longrightarrow\!\!\triangleright\, M[N/x].$$

Indeed, languages such as PCF [83], lazy lambda calculus [1] and other developments in the field of functional languages have a similar style underlying reduction semantics. An *operational semantics* for such functional languages is then a *reduction strategy* which is, essentially, a way of choosing where to perform a reduction in a given term at a given time during the evaluation.

In modern presentations of process calculi with concurrency and mobility, there is usually also an underlying *reduction semantics* which is often defined as a rewriting system on the set of terms modulo a structural congruence. For example, in CCS [70], there is a reduction rule

$$a!.P + P' \mid a?.Q + Q' \longrightarrow\!\!\triangleright\, P \mid Q, \tag{2.1}$$

which expresses that two processes can synchronise along a channel with name a and continue. For readers familiar with the original presentation

of the semantics of CCS via a labelled transition system, the reduction semantics corresponds exactly to the silent $\xrightarrow{\tau}$ action.

There are several concepts which reappear in many different formalisms. The first is the notion of term. A term is a syntactic entity (perhaps modulo some equivalence) which represents, or models, a process. Second is a reduction system, a relation on the set of terms which captures computation, the behaviour of the term. Since reduction is a term with syntactic connotations, and we shall be interested in treating also non-syntactic examples, we shall henceforth refer to reductions as *reactions*. Third is a notion of context. The classic intuitive description of a context is that it is a "term with a hole" $C[-]$. The idea is that "plugging a term" a into a context $C[-]$ yields a term $C[a]$. One can learn about a by plugging it into various contexts and studying the behaviour of the resulting term. Indeed, this style of reasoning leads to what is normally called a Morris-style contextual equivalence for functional languages, which usually aim to model finite sequential computations. For more sophisticated languages, it is not enough to observe just the reactions, even if one is able to observe branching structure. Indeed, one has to add extra observational power, such as barbs [75].

In this chapter we shall explore a technique, first due to Sewell [98], of obtaining a congruent process equivalence with an associated coinductive proof method. This involves deriving a labelled transition systems with particular contexts forming the labels. We shall begin in section 2.1 by recalling Leifer and Milner's theory of reactive systems and the associated categorical tool, relative pushouts [66]. In section 2.2 we shall motivate and introduce an extension of the theory to a 2-categorical setting. Finally, in section 2.3 we shall show that the labelled transition systems derived using this technology are well-behaved in the sense that bisimilarity, trace equivalence and failures equivalence are congruences.

2.1 Reactive systems

In this section we shall recall Leifer and Milner's framework of reactive systems [66]. After motivating a general approach, in §2.1.1 we shall present the definition of reactive systems and give several examples. In §2.1.2, after recalling some of the basic theory associated with the notion of labelled transition system, we shall consider the idea of using contexts as labels. Finally, in §2.1.3 we shall recall the definition of relative pushouts, a categorical tool used by Leifer and Milner to characterise when a context is the smallest which allows a reaction to occur.

2.1.1 Definition and examples

Although the notion of context is specific to each formalism, there are some aspects of the notion which hold in greater generality. For example, we would certainly expect substitution to be associative, that is, plugging a into C and obtaining a term $C[a]$ which is then plugged into $D[-]$ should result in the same term as plugging $C[-]$ into $D[-]$ to obtain a context $D[C[-]]$ and then plugging in a – $D[C[a]]$. We would also expect an identity context, say $-$, so that plugging in a yields a back again.

It is also useful to consider what structure can be expected of the "hole". Firstly, the hole may have attached type information, so that only certain terms may be plugged in. Secondly, it may be reasonable to extend the definition of context to allow more than one hole.

The algebraic structure described so far is a description of a category $\mathbf{C}$ with objects being the holes, arrows the contexts and composition substitution. Such a category could perhaps be extended with extra structure in order to allow operations on the holes and the terms. Thus an arrow $I_1 \xrightarrow{c} I_2$ is a context with I_1 giving a description of its holes. The context itself may be plugged into a context with I_2-holes.

Let's consider such a category of holes and contexts. The next question to consider is: what are terms? One answer to this question is to consider a term as a sort of singular context with no hole, then one may pick an object $0 \in \mathbf{C}$ which describes the "empty hole" and consider the terms as being arrows with domain 0.

As an example, consider the (meaningless) programming language expression

$$(\, -_1 + \texttt{charToAscii}(-_2)\,) * \texttt{0.1}$$

which can be interpreted as a context $\texttt{int} \times \texttt{char} \to \texttt{float}$ and can be, for example, composed with the expression $\texttt{sin}(-_1)$ which is a context $\texttt{float} \to \texttt{float}$. To obtain a term, one could substitute in $\langle \texttt{0}, \texttt{a} \rangle : 0 \to \texttt{int} \times \texttt{char}$ to obtain the expression $\texttt{sin}((\texttt{0} + \texttt{charToAscii}(\texttt{a})) * \texttt{0.1})$ of type $\texttt{float}$, in other words, an arrow $0 \to \texttt{float}$

The next step is describing the reaction relation. As is the case with the formalisms described previously, we would like to take relatively simple generating rules and generate the reaction relation by closing the rules under context substitution.

Here there is one complication. In many formalisms there exist contexts which do not preserve reaction, that is, we may have that

$$a \longrightarrow\!\!\triangleright\, b, \text{ but } c[a] \not\longrightarrow\!\!\triangleright\, c[b] \text{ for some } c.$$

For example, in the lazy lambda calculus, reaction is not performed under a lambda abstraction, that is the context $\lambda x.[-]$ does not preserve reaction. Similarly, in CCS, the prefix operation does not preserve reaction: for instance, we have $a \mid \overline{a} \longrightarrow\!\!\triangleright\, 0$ yet $b.(a \mid \overline{a}) \not\longrightarrow\!\!\triangleright$. Thus, one should identify a subclass of contexts which do preserve reaction and form the reaction relation by closing the reaction rules under these *reactive* contexts.

We shall make the first of a number of simplifications in order to study the theory we have identified so far. Firstly, we shall only consider *ground* reaction rules, that is, reaction rules which do not have any process variables. Thus, for example, we shall not allow a rule of the form (2.1). To study a system like CCS, we shall first have to instantiate the variables by all required processes. The extension of the framework to cover reaction rules with process variables is the subject of ongoing work.

We shall now introduce the central concept of reactive system. It appeared first in Leifer and Milner's seminal paper [66]. We shall first state the definition and then discuss its components.

Definition 2.1.1 (Reactive system). A reactive system $\mathbb{C}$ consists of:

(i) a category $\mathbf{C}$;

(ii) a distinguished object $0 \in \mathbf{C}$;

(iii) a composition-reflecting subcategory $\mathbf{D}$ of *reactive contexts*;

(iv) a set or pairs $\mathcal{R} \subseteq \bigcup_{C \in \mathbf{C}} \mathbf{C}(0, C) \times \mathbf{C}(0, C)$ of *reaction rules*.

By composition-reflecting we mean that $dd' \in \mathbf{D}$ implies d and $d' \in \mathbf{D}$. As discussed before, the reactive contexts are those contexts inside which evaluation may occur.

Notice that the two components of a rule $\langle l : 0 \to C, r : 0 \to C \rangle$ in $\mathcal{R}$ are required to have the same codomain. This is necessary in order to define a *reaction relation* by closing the rules under reactive contexts: if there is a (reactive) context c into which one can plug l, obtaining a term cl, then we should be able to replace l by r and obtain cr.

Definition 2.1.2 (Reaction relation). Given a reactive system $\mathbb{C}$ with reactive contexts $\mathbf{D}$ and reaction rules $\mathcal{R}$, the *reaction relation* $\longrightarrow\!\!\triangleright$ is defined as follows:

$$t \longrightarrow\!\!\triangleright\, u \quad \text{iff} \quad t = dl,\ u = dr \text{ for some } d \in \mathbf{D} \text{ and } \langle l, r \rangle \in \mathcal{R}.$$

In the remainder of this section we shall illustrate the concepts presented so far with the aid of several examples. In particular, the simple process calculus of Example 2.1.7 and the category of bunches and wires of Definition 2.1.13 shall serve as the motivation and as test cases for much of the theory presented in this thesis.

Term rewriting. For syntactic examples, it shall be useful to consider a canonical way of obtaining a category where the arrows are terms and contexts.

Definition 2.1.3 (Lawvere Theory). Consider a syntactic signature Σ with an associated arity function $\Sigma \to \mathbb{N}$. The (free) Lawvere theory for Σ [62], denoted as $\mathsf{Th}(\Sigma)$, is a category with objects natural numbers and morphisms $t\colon m \to n$ being n-tuples of m-holed terms. Composition is substitution of terms, which is associative. The category is Identities $n \to n$ are $\langle -_1, -_2, \ldots, -_n \rangle$. cartesian, with 0 the terminal object and n being the product of 1 with itself n times. The theory is free in the sense that there are no equations between composite terms, apart from those imposed by the cartesian structure.

Example 2.1.4. Consider a signature $\Sigma = \{\mathsf{0}, s, +\}$ where $\mathsf{0}$ has arity 0, s has arity 1 and $+$ has arity 2.

We shall give several examples of arrows in $\mathsf{Th}(\Sigma)$. The term $\mathsf{0}$ denotes an arrow $0 \to 1$ (it is a 1-tuple with no holes), the term $s(-_1)$ is an arrow $1 \to 1$; composing the two yields $s(\mathsf{0}) : 0 \to 1$. The product of $s(\mathsf{0})$ with itself yields $\langle s(\mathsf{0}), s(\mathsf{0}) \rangle : 0 \to 2$. The term $(-_1) + (-_2)$ denotes an arrow $2 \to 1$ of $\mathsf{Th}(\Sigma)$; composing it with $\langle s(\mathsf{0}), s(\mathsf{0}) \rangle$ yields $s(\mathsf{0}) + s(\mathsf{0}) : 0 \to 1$.

Thus, given a signature Σ, the arrows of $\mathsf{Th}(\Sigma)$ can be thought of as contexts. However, the contexts are non-linear in the sense that the holes may be appear any number of times. For instance $\mathsf{0}$ also defines an arrow $1 \to 1$, it is a context which ignores its input (the hole does not appear in the term). It can be defined by taking the product of $\mathsf{0} : 0 \to 1$ with the unique arrow $1 \to 0$. Thus, for example, $(\mathsf{0} : 1 \to 1) \circ (s(\mathsf{0}) : 0 \to 1) = \mathsf{0} : 0 \to 1$. As well as ignore their "inputs" the contexts can also duplicate them, for example $s(-_1) + (-_1)$ is an arrow $1 \to 1$ and $s(-_1) + (-_1) \circ \mathsf{0} = s(\mathsf{0}) + (\mathsf{0})$.

The non-linear arrows of $\mathsf{Th}(\Sigma)$ can be argued to be unreasonable contexts for many applications. Thus, for our purposes, it makes sense to restrict our attention to the subcategory of linear contexts.

Definition 2.1.5 ($\mathbf{C}_\Sigma$). An arrow $t\colon m \to n$ of $\mathsf{Th}(\Sigma)$ is said to be linear if each of the m "holes" is used exactly once in t. Let $\mathbf{C}_\Sigma$ denote the subcategory of $\mathsf{Th}(\Sigma)$ with objects being the objects of $\mathsf{Th}(\Sigma)$ (natural numbers), but the arrows being just the linear morphisms. Notice that this restriction makes sense, since all the identities are linear and linearity is preserved by composition. For a signature Σ, we shall refer to $\mathbf{C}_\Sigma$ as the *term category*.

Example 2.1.6 (Ground term rewriting system). A term rewriting system can be given as a set $\mathcal{R}$ of pairs $\langle l, r\rangle$ where $l, r : n \to 1$ are arrows of $\mathbf{C}_\Sigma$. The rewrite relation $\longrightarrow\!\!\triangleright$ is derived from $\mathcal{R}$ by substitution under contexts and parameters, that is $a \longrightarrow\!\!\triangleright a'$ if $a = clp$, $a' = crp$ for some $c, p \in \mathbf{C}_\Sigma$. [1] A *ground* term rewriting system has $\mathcal{R}$ consisting only of pairs $\langle l, r\rangle$ with $l, r : 0 \to 1$. Then any ground term rewriting system defines a reactive system, with $\mathbf{C}_\Sigma$ being the underlying category, $\mathcal{R}$ being the reaction rules and $\mathbf{D} = \mathbf{C}_\Sigma$, that is, all contexts are reactive.

Simple Process Calculus. We shall now introduce a very simple process calculus which can be considered as an almost trivial subset of CCS. While it is simple, it does exhibit some interesting features which shall guide us in the development of the theory.

Example 2.1.7. Assume some set of channel names N and consider the language generated by the following specification:

$$P \ ::= \ \mathtt{0} \ \mid \ a \ \mid \ \overline{a} \ \mid \ P \mid P' \qquad\qquad \text{where } a \in N.$$

The signature consists of a set of input and output constants parametrised over the channel names, the null process constant and a binary operator, that is $\Sigma = \{a, \overline{a}\}_{a\in N} \cup \{\mathtt{0}, -_1 \mid -_2\}$.

The intuition for the dynamics is that an occurrence of a and $\overline{a}$ can interact and disappear, that is, rewrite to the null process $\mathtt{0}$; in symbols $a \mid \overline{a} \longrightarrow\!\!\triangleright \mathtt{0}$.

Our first attempt at building the reactive system to capture this calculus is to let the underlying category be $\mathbf{C}_\Sigma$ and let all contexts be reactive. We let the reaction relation be the closure of the relation $\{\, \langle a \mid \overline{a}, \mathtt{0}\rangle \mid a \in N \,\}$ under all contexts.

There are several problems with the reactive system described in Example 2.1.7. First, notice that the term $(b \mid a) \mid \overline{a}$ cannot react. This is because

[1] For many applications it is reasonable to expect only l to be linear. Since we shall concentrate exclusively on ground term rewriting (and all arrows with domain 0 are trivially linear), we do not elaborate here.

there is nothing in the theory which makes sure that the parallel composition operator should be associative. If we assume associativity, there are further problems: $\overline{a} \mid a$ and $a \mid b \mid \overline{a}$ cannot react. One solution would be to include extra rewrite rules to consider all these extra subcases, but this solution isn't elegant and feels like a hack since the basic interaction $a \mid \overline{a}$ is already described by one rewrite rule – and the problematic cases we've identified can all be seen as special cases of this basic reduction. An elegant solution to this sort of problem was first proposed by Berry and Boudol in their work on the Chemical Abstract Machine (CHAM) [9] – they argued that one one could think of the basic components of a parallel composition as molecules in a chemical solution, they are free to move around and interact with any other molecule.

Their work has led to the development of the idea of a *structural congruence* relation $\equiv$ which relates processes which are syntactically different but which are trivially semantically equivalent, as for example $a \mid \overline{a}$ and $\overline{a} \mid a$ or $a \mid \mathtt{0}$ and a are. Then the terms of a process calculus are not really syntactic any more, they are equivalence classes of syntactic terms with respect to $\equiv$. A reduction semantics (as well as labelled transition systems) are given "up to" structural congruence. This means, essentially, that reduction semantics have an implicit or explicit rule of the form

$$\frac{P' \equiv P \quad P \longrightarrow\!\triangleright\, Q \quad Q \equiv Q'}{P' \longrightarrow\!\triangleright\, Q'.}$$

Structural congruence has been a very influential concept and the majority of modern calculi come equipped with such a relation.

At this point it is useful to formalise exactly what is meant by a congruence on the terms and the contexts of a reactive system.

Definition 2.1.8 (Congruence). Given a reactive system $\mathbb{C} = \{\mathbf{C}, 0, \mathbf{D}, \mathcal{R}\}$ a relation $R \subseteq \bigcup_{C \in \mathbf{C}} \mathbf{C}(0, C) \times \mathbf{C}(0, C)$ on the terms of $\mathbf{C}$ is a *congruence* if the following condition is satisfied: whenever $\langle a : 0 \to C,\, b : 0 \to C \rangle \in R$ then for any arrow $c : C \to D$, we have that $\langle ca : 0 \to D,\, cb : 0 \to D \rangle \in R$.

Thus, if a congruence relates two terms a and b then it relates the terms ca and cb which result from substituting a and b into an arbitrary context c. Notice that we shall only consider relations which respect the codomain of the terms.

Returning to the simple calculus of Example 2.1.7, the expected structural congruence should include the associativity, commutativity and identity (ACI) laws of the parallel composition operator.

Example 2.1.9. Consider the term category $\mathbf{C}_\Sigma$. We would like to consider a category where the terms are quotiented by structural congruence. The structural congruence relation $\equiv$ is the smallest congruence which ensures that for any terms p, q and r, $p \mid (q \mid r) = (p \mid q) \mid r$ (associativity), $p \mid q = q \mid p$ (commutativity), and $p \mid \mathtt{0} = p$ (identity). The relation can be extended in the obvious way to contexts (arrows $m \to n$ of $\mathbf{C}_\Sigma$ where $m \neq 0$) – so for example $-_1 \mid -_2 \equiv -_2 \mid -_1 : 2 \to 1$ and $-_1 \mid \mathtt{0} \equiv -_1 : 1 \to 1$. Let $\mathbf{C}_\Sigma^{\equiv}$ be the category with objects that of $\mathbf{C}_\Sigma$, but with arrows $[f] : m \to n$ in $\mathbf{C}_\Sigma^{\equiv}$ being equivalence classes of arrows $f : m \to n$ in $\mathbf{C}_\Sigma$ with respect to $\equiv$. It is easy to check that the composition operator is well defined and that $\mathbf{C}_\Sigma^{\equiv}$ is indeed a category.

Defining reaction rules as in Example 2.1.7 and taking all contexts to be reactive yields a reactive system. It is easy to see that the reaction relation is as expected.

Bunch contexts. In the examples examined so far, terms and contexts have been syntactic. The following is a simple example of an underlying category **Bun** of a class of reactive systems where the contexts have an algebraic definition. This category shall be the focus of Chapter 4. The study of this category shall be useful in order to treat more sophisticated examples such as the bicategory of cospans of graphs of Chapter 6.

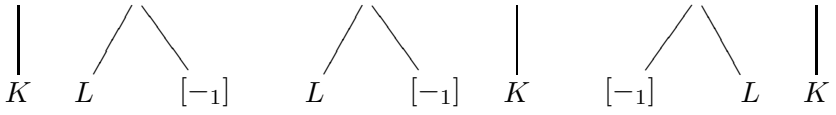
The category of 'bunches and wires' was introduced by Leifer and Milner in [66] as a skeletal algebra of shared wirings, abstracting over the notion of *names* in, for example, the π-calculus. It is related to various models studied in the theory of action calculi [71].

Note that the definition we give here is the natural definition of such an algebraic structure; the definition which appears in [66] has extra information, the so-called *trailing*, which is an ad-hoc way of avoiding some of the problems posed by the structural congruence of bunches.

A bunch context of type $m_0 \to m_1$ consists of an ordered set of m_1 trees of depth one containing exactly m_0 holes. Leaves are unordered and labelled from an alphabet $\mathcal{K}$. These data represent m_1 bunches of unspecified controls (the roots), together with m_0 places (the holes) where bunch contexts can be plugged into.

Diagrams (*i*), (*ii*) and (*iii*) are pictures of three bunches $1 \to 2$, that is,

they



(i) (ii) (iii)

have one hole and two roots. Diagrams (i) and (ii) represent two different bunches, because roots are ordered; conversely Diagrams (ii) and (iii) represent the same bunch, because leaves are not.

We would like to capture this structure with a formal definition. We start by defining concrete bunch contexts.

Definition 2.1.10 (Concrete bunch contexts). Let m_0 and m_1 be finite ordinals. A *concrete bunch context* $c : m_0 \to m_1$ is a tuple

$$c = \langle X,\, \mathrm{ch} : X \to \mathcal{K},\, \mathrm{rt} : m_0 + X \to m_1 \rangle\,,$$

where X is a finite *carrier* set, $\mathrm{rt}\colon m_0 + X \to m_1$ is a surjective function linking leaves (X) and holes (m_0) to their roots (m_1), and $\mathrm{ch}\colon X \to \mathcal{K}$ is a leaf labelling function.

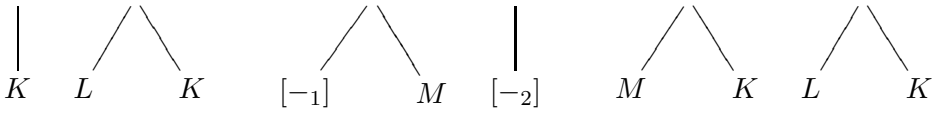
Given concrete bunch contexts $c_0\colon m_0 \to m_1$ and $c_1\colon m_1 \to m_2$, we can compose them to obtain a concrete bunch context $c_1 c_0\colon m_0 \to m_2$. Roughly, this involves 'plugging' the m_1 trees of c_0 orderly into m_1 holes of c_1; leaves and holes of c_0 are 'wired' to the roots of c_1, alongside c_1's leaves. Formally, $c_1 c_0$ is $\langle X, \mathrm{rt}, \mathrm{ch}\rangle$ with

$$X = X_0 + X_1, \qquad \mathrm{rt} = \mathrm{rt}_1(\mathrm{rt}_0 + \mathrm{id}_{X_1}), \qquad \mathrm{ch} = [\mathrm{ch}_0, \mathrm{ch}_1],$$

where $[-,\,-]$ is copairing. The root function of the composite is perhaps best illustrated by writing its components individually:

$$m_0 + X_0 + X_1 \xrightarrow{\mathrm{rt}_0 + X_1} m_1 + X_1 \xrightarrow{\mathrm{rt}_1} m_2.$$

Diagrams (iv), (v) and (vi) illustrate the concept of composition of concrete



$a : 0 \to 2$ $b : 2 \to 2$ $ba : 0 \to 2$

(iv) (v) (vi)

bunch contexts. Diagram (*iv*) represents a concrete bunch context $a : 0 \to 2$ where $X = 3$, $\mathrm{ch}(0) = \mathrm{ch}(2) = K$, $\mathrm{ch}(1) = L$, $\mathrm{rt}(0) = 0$ and $\mathrm{rt}(1) = \mathrm{rt}(2) = 1$. Diagram (*v*) represents a concrete bunch context $b : 2 \to 2$ where b has $X = \{*\}$, $\mathrm{ch}(*) = M$, $\mathrm{rt}(0) = \mathrm{rt}(*) = 1$ and $\mathrm{rt}(1) = 1$. Finally, diagram (*vi*) represents $ba : 0 \to 2$, the result of composing a and b.

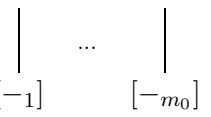
Remark 2.1.11. Notice that this composition is not associative "on the nose" because the coproduct (disjoint union) of finite sets is not; it is associative up to a canonical bijection. This is not an important problem because in order to obtain associativity one can work in a category in which it is easy to define a coproduct which *is* associative, such as the category of finite ordinals. Here we shall ignore the details and deal with this technical issue in Chapter 4.

A *homomorphism* of concrete bunch contexts $\alpha : c \Rightarrow c' \colon m_0 \to m_1$ is a function $\alpha \colon X \to X'$ which respects rt and ch, i.e. $\mathrm{rt}'(\mathrm{id}_{m_0} + \alpha) = \mathrm{rt}$ and $\mathrm{ch}' \, \alpha = \mathrm{ch}$. An isomorphism is a bijective homomorphism.

When reasoning about bunch contexts, we are interested in the structure, not in the set theoretical identity of the underlying carrier sets. It makes sense, therefore, to identify bunches which are related by an isomorphism – that is, a bijection on the underlying carrier sets which preserves all structure. The notion of isomorphism of concrete bunch contexts is thus the right notion of structural congruence in this setting.

Definition 2.1.12 (Abstract bunch contexts). An abstract bunch context $[c] : m_0 \to m_1$ is an isomorphism class of concrete bunch contexts. Thus, $[c] = [c']$ if and only if $c \cong c'$. Abstract bunch contexts may be composed, given $[c_0] : m_0 \to m_1$ and $[c_1] : m_1 \to m_2$ we let $[c_1] \circ [c_0] = [c_1 \circ c_0]$. It is easy to check that this composition is well-defined. It *is* associative (see Remark 2.1.11) since we are dealing with isomorphism classes.

The composition of abstract bunch contexts also has identities. Indeed, given an ordinal m_0, let id_{m_0} denote the concrete bunch context with empty carrier set, the unique character function and the root function $m_0 + 0 \to m_0$ being the identity on the first coproduct injection. Graphically, such a bunch context looks as follows:



and it is easy to check that for any abstract bunch context $[c] : m_0 \to m_1$ we have $[\mathrm{id}_{m_1}] \circ [c] = [c] = [c] \circ [\mathrm{id}_{m_0}]$.

Definition 2.1.13 ($\mathbf{Bun}_0$). The category of abstract bunch contexts $\mathbf{Bun}_0$ has

1. objects the finite ordinals, written as $m_0, m_1, \ldots$;
2. arrows from m_0 to m_1 are abstract bunch contexts $[a] \colon m_0 \to m_1$.

2.1.2 Contexts as labels

Reduction semantics is often relatively simple and intuitive. Moreover, the underlying mechanisms are general and are based on term rewriting, which is a well-studied area of computer science. It is, therefore, an important question whether an interesting process equivalence can be already be generated from the reaction rules without the need for extra "semantic" technology, such as labelled transition systems or barbs. Moreover, such a process equivalence should be generated in a "general" way; that is, without resorting to the ad-hoc syntactic structure of the underlying formalism. Clearly, if such a theory could be developed to a satisfactory level, it would allow the study of semantic issues across a wide number of formalisms, without the need to redevelop constructions and proofs from scratch.

Here we shall study the idea of Sewell [98], who proposed to generate a labelled transition system (see Definition 2.1.14) from reaction rules in a way which would ensure that the resulting bisimilarity is a congruence. The basic idea is that the labels of such an lts should be the smallest contexts which allow reaction. Leifer and Milner [66] later proposed relative pushouts, a categorical way of making precise this intuition. The work in this thesis continues in this direction.

The advantage of a contextually defined process equivalence based on reductions is simplicity and elegance. Such equivalences usually intuitive, in that it is often relatively easy to argue why such an equivalence is "natural". In fact, barbed congruence is usually considered to be *the* canonical process equivalence. Thus, barbed congruence can be considered as a Morris-style process equivalence for process calculi. Just as Morris-style equivalences, barbed congruence suffers from the quantification over all contexts in its definition. Thus, many important contributions in the field of process calculus have been about the development of methods for proving that processes are barbed congruent. Usually, this involves the development of a labelled transition system semantics on which bisimilarity is sound with respect to barbed congruence. Bisimilarity is even more valuable when it is also complete with respect to barbed congruence; the two equivalences then coincide.

We recall the basic definitions of labelled transition systems and bisimilarity below.

Definition 2.1.14 (Labelled Transition System). A labelled transition system (lts) is a triple $\langle S, L, T \rangle$, where:

- S is a set of *states* ranged over by $a, b, \ldots$;
- L is a set of *labels* ranged over by $f, g, \ldots$;
- a set of transitions $T \subseteq S \times L \times S$.

A transition $\langle a, f, a' \rangle \in T$ shall usually be written $a \xrightarrow{f} a'$.

Labelled transitions systems can be generated in a number of ways. A popular and influential framework for the derivation of labelled transition systems is called *structural operational semantics* or SOS [84]. An SOS style presentation of an lts involves presenting a number of natural-deduction like proof rules for each of the elements of a syntactic signature. This style of semantics has attracted much research, see for example the handbook chapter [2].

Example 2.1.15. Recall the simple calculus of Example 2.1.7. A labelled transition system can be generated, in a way which resembles SOS, by the following rules,

$$a \xrightarrow{a} \mathsf{0} \qquad \overline{a} \xrightarrow{\overline{a}} \mathsf{0} \qquad a \mid \overline{a} \xrightarrow{\tau} \mathsf{0}$$

$$\frac{P \xrightarrow{x} P'}{Q \mid P \xrightarrow{x} Q \mid P'} \qquad \frac{P \equiv P' \quad P \xrightarrow{x} Q \quad Q' \equiv Q}{P' \xrightarrow{x} Q'}$$

where $\equiv$ is the structural congruence relation introduced in Example 2.1.9.

Bisimilarity in its modern coinductive presentation was first introduced by Park [82]. In the setting of labelled transition systems it was introduced by Milner [70]. It is interesting for several reasons, firstly, it is the finest possible "reasonable" equivalence on a labelled transition system, where by reasonable we mean that the equivalence can only take into account the labels and the branching structure of the transitions out of a state. Secondly, it has several elegant characterisations: in terms of games, logics [41] and general categorical approaches [48, 88].

Definition 2.1.16 (Bisimulation). A relation R on the states of a labelled transition system T is a bisimulation if the following symmetric conditions are satisfied:

1. if $\langle a, b\rangle \in R$ and $a \xrightarrow{f} a'$ then there exists b' such that $b \xrightarrow{f} b'$ and $\langle a', b'\rangle \in R$;

2. if $\langle a, b\rangle \in R$ and $b \xrightarrow{f} b'$ then there exists a' such that $a \xrightarrow{f} a'$ and $\langle a', b'\rangle \in R$.

It is an easy exercise to show that an arbitrary union of bisimulations is a bisimulation. Thus, there exists a largest bisimulation.

Definition 2.1.17 (Bisimilarity). Given a labelled transition system, let *bisimilarity*, denoted by $\sim$, be the largest bisimulation.

It is easy to verify that bisimilarity is an equivalence relation. Because of the definition of $\sim$, in order to verify that $a \sim b$, it suffices to find a bisimulation R such that aRb.

As well as bisimilarity, labelled transition systems lend themselves to other equivalences and preorders. Some of the more important ones include the *trace preorder* (and the corresponding *trace equivalence*, obtained by a symmetric closure) and the *failures preorder/equivalence*. For a comprehensive survey see the paper by Van Glabbeek [104].

Bisimilarity, and other preorders/equivalences are most useful when they are congruences (Definition 2.1.8). This is because, as well as the technique of coinduction, one can use the technique of equational reasoning (commonly referred to as "substituting equals-for-equals"). In particular, suppose that we need to prove that a system consisting of a parallel composition $p \mid q$ is bisimilar to $p' \mid q'$ and we have bisimulations relating p and p' and q and q'. If bisimilarity is a congruence then $p \mid q \sim p' \mid q$ and $p' \mid q \sim p' \mid q'$ and therefore $p \mid q \sim p' \mid q'$, since bisimilarity is transitive.

In this chapter, and in the thesis in general, we shall restrict out investigation to strong equivalences. That is, the internal behaviour of processes *is* distinguished. Of course, weak equivalences are more useful for practical purposes, as normally one wants to equate processes that differ only in their internal behaviour. As discussed in Chapter 1, weak bisimilarity has traditionally been problematic for categorical approaches, however, ongoing research by Jensen [45] suggests that many of the technical developments presented within this thesis are *orthogonal* to whether one is interested in a strong or weak equivalence.

We shall now introduce a notion of reaction congruence in the style of Honda and Yoshida [43]. This differs from the reduction congruence of Milner and Sangiorgi [75] in a subtle point: whereas Milner and Sangiorgi consider the largest congruence contained within reaction bisimilarity, Honda and Yoshida consider the greatest congruent bisimulation. Thus, Honda and Yoshida's approach gives, in general, a finer equivalence.

Definition 2.1.18 (Reaction congruence). Given a reactive system $\mathbb{C}$, we let $\simeq_R$ be the largest symmetric relation on the terms of $\mathbb{C}$ such that

$$\text{if } a \simeq_R b \text{ then } \forall c \in \mathbf{C} : ca \longrightarrow\!\!\triangleright\, a' \quad \Rightarrow \quad \exists b'\, cb \longrightarrow\!\!\triangleright\, b' \text{ and } a' \simeq_R b', \quad (2.2)$$

where $\mathbf{C}$ is the underlying category of the reactive system. Notice that the $\simeq_R$ relates only terms which are arrows with the same codomain in $\mathbf{C}$.

A moment's thought confirms that an arbitrary union of relations which satisfy equation (2.2) satisfies the equation. In particular, to prove that two processes are reaction congruent it suffices to show that they are in a relation which satisfies (2.2). Thus, for example, $\simeq_R$ is reflexive, since the identity relation clearly satisfies (2.2). Also, it is easy to check that $\simeq_R$ is transitive; if a relation satisfies (2.2), its transitive closure satisfies it also.

Proposition 2.1.19. *The relation $\simeq_R$ is an equivalence.*

It is trivial to check that $\simeq_R$ defines a congruence, it suffices to show that

$$\{\, \langle ca,\, cb\rangle \mid a \simeq_R b \,\}$$

satisfies (2.2). Indeed, if $c'(ca) \longrightarrow\!\!\triangleright\, b$ then $(c'c)a \longrightarrow\!\!\triangleright\, b$ and as $a \simeq_R b$, there exists a b' such that $c'cb \longrightarrow\!\!\triangleright\, b'$ and $a' \simeq_R b'$.

There is a simply-defined labelled transition system which captures reaction congruence, in the sense that bisimilarity on the labelled transition system is fully abstract with respect to it.

Definition 2.1.20 (Reaction labelled transition system). Define a labelled transition system $RLTS(\mathbb{C})$ as follows

- States: arrows $a : 0 \to I$ in $\mathbf{C}$, where I is arbitrary and not fixed;
- Transitions: $a \xrightarrow{c} b$ if $ca \longrightarrow\!\!\triangleright\, b$.

Denote bisimilarity on $RLTS(\mathbb{C})$ by $\sim_R$.

Lemma 2.1.21.

$$a \simeq_R b \quad \textit{iff} \quad a \sim_R b$$

Proof. $\simeq_R \subseteq \sim_R$: $a \xrightarrow{c} b \Leftrightarrow ca \longrightarrow\!\triangleright\, b \Rightarrow ca' \longrightarrow\!\triangleright\, b'$ and $b \simeq_R b'$. But then, by definition $a' \xrightarrow{c} b'$ and so $\simeq_R$ is a bisimulation. $\sim_R \subseteq \simeq_R$: Suppose that $ca \longrightarrow\!\triangleright\, b$. Then $a \xrightarrow{c} b$ and so $a' \xrightarrow{c} b'$ ($ca' \longrightarrow\!\triangleright\, b'$) with $b \sim_R b'$. Clearly, $\sim_R$ satisfies equation 2.2. □

It should be mentioned that we are, in a way, cheating. Labelled transition systems, on which bisimilarity is fully abstract with respect to a certain contextually-defined process equivalence, are useful because, in practice, constructing bisimulations is easier than quantifying over all possible contexts. The labelled transition system of Definition 2.1.20 would usually be infinitely branching, since the labels are all contexts which allow reaction. As a consequence, the labelled transition system would not be very useful in practice.

A problem associated with taking all contexts allowing reaction as labels is that we admit labels which are redundant. For example, consider the term a of simple CCS (Example 2.1.7). The environment can learn about the term by putting it in the context $\overline{a} \mid -$ and observe a reaction. This corresponds to the label $a \xrightarrow{\overline{a}\mid -} 0$ of the reaction labelled transition system. However, we also have $a \xrightarrow{c\mid\overline{a}\mid -} c$ as a transition, yet c contributes nothing to the reaction. In particular, it would seem that we could add more power to the environment by only allowing labels which contain just the right amount of information in order to interact with the term.

Indeed, in certain examples $\simeq_R$ does not give the expected process equivalence.

Example 2.1.22 ($\simeq_R$ in simple CCS). We shall show that $a \mid \overline{a} \sim_R b \mid \overline{b}$. Suppose that $a \mid \overline{a} \xrightarrow{c} d$. Then $c(a \mid \overline{a}) \longrightarrow\!\triangleright\, d$. If the reaction is already in the term ($a \mid \overline{a} \longrightarrow\!\triangleright\, 0$), then $d = c0$ and we can use the reaction $b \mid \overline{b} \longrightarrow\!\triangleright\, 0$ to obtain a label $b \mid \overline{b} \xrightarrow{c} d$. Clearly, if the reaction is within c then we can match it with the same reaction. On the other hand, suppose that $a \mid \overline{a}$ nontrivially reacts with c to produce d. Without loss of generality, suppose that $c[-] = c' \mid \overline{a} \mid [-]$ and the $\overline{a}$ from c reacts with the a from our term. Then $d = c' \mid \overline{a}$. We can match the reaction by again reducing $b \mid \overline{b} \longrightarrow\!\triangleright\, 0$ (without using the context) to obtain $c(b \mid \overline{b}) \longrightarrow\!\triangleright\, c[0] = c' \mid \overline{a} = d$ and so $b \mid \overline{b} \xrightarrow{c} d$.

Remark 2.1.23. As an aside, the problem demonstrated in Example 2.1.22 is not possible in full CCS because the contexts there have enough power to distinguish between $a \mid \overline{a}$ and $b \mid \overline{b}$. Indeed, it suffices to consider a context $\overline{a}.d$. Then $a \mid \overline{a} \xrightarrow{\overline{a}.d} \overline{a} \mid d$ but the only possible transition with that label for $b \mid \overline{b}$ is $b \mid \overline{b} \xrightarrow{\overline{a}.d} \overline{a}.d$; clearly $\overline{a} \mid d \not\cong_R \overline{a}.d$.

We have presented a natural congruent process equivalence $\simeq_R$ built up only with the use of reactions and contexts, and we have demonstrated that it may be defined using bisimilarity on a certain labelled transition system. In the course of the development, we have identified two problems with $\simeq_R$. Firstly, the labelled transition system, as presented, is large, and contains redundant information. Secondly, in several examples the resulting process equivalence does not match with the expected process equivalence. In order to fix these problems, we shall restrict the labels of the labelled transition system only to the *smallest* contexts which allow reaction.

The idea of using such smallest contexts is originally due to Sewell [98] and was later taken up by Leifer and Milner in their pioneering paper [66]. It was later used by Leifer in his PhD thesis [64] and by Jensen and Milner in their work on bigraphs [46, 47]

Leifer and Milner's original contribution [66] was to identify a possible way of formalising the notion of "smallest context allowing reaction" in terms of a universal property in the language of category theory. They defined *relative-pushouts*, or RPOs, of which *idem-relative-pushouts*, or IPOs are a special case. Relative-pushouts can be described concisely as pushouts in slice categories. One may define a well-behaved labelled transition system using IPOs. It is well-behaved because bisimilarity, as well as some other equivalences in the van Glabbeek spectrum [104] are congruences.

Let us now present an intuitive account of the idea of what should constitute a smallest context which allows a reaction. In the previous sections we have settled on using categories where arrows correspond to contexts and where composition corresponds to context substitution.

The following definition formalises what it means for a reactive system context to allow reaction.

Definition 2.1.24 (Redex square). Suppose that $\mathbb{C}$ is a reactive system and $a : 0 \to I_2$ is some arrow. A *redex square* (see below) consists of a left hand side $l : 0 \to I_3$ of a reaction rule $\langle l : 0 \to I_3, r : 0 \to I_3 \rangle \in \mathcal{R}$, a context $f : I_2 \to I_4$ and a reactive context $d : I_3 \to I_4$ so that $fa = dl$.

$$\begin{array}{ccccc} & & I_4 & & \\ & {\scriptstyle f}\nearrow & & \nwarrow{\scriptstyle d} & \\ I_2 & & & & I_3 \\ & {\scriptstyle a}\nwarrow & & \nearrow{\scriptstyle l} & \\ & & 0 & & \end{array} \tag{2.3}$$

A redex square tells us that a in the context of f can react. Thus, another way of defining the labelled transition system of Definition 2.1.20 would be to say that $a \xrightarrow{f} dr$ iff diagram (2.3) is a redex square.

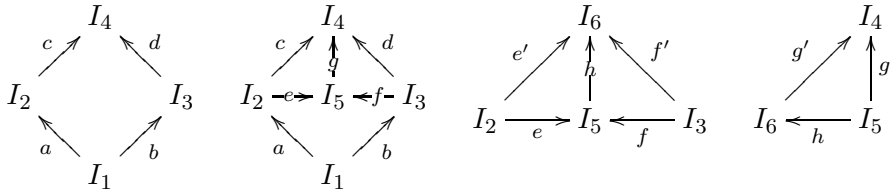
Of course, as mentioned previously, a redex square does not tell us about which parts of f are really needed for for the reaction and which parts are not necessary. As demonstrated in Example 2.1.22 the congruence derived by using the labels which correspond to such contexts may be too coarse.

In the following section we recall an elegant characterisation, due to Leifer and Milner [66], of the redex squares which correspond to such smallest contexts.

2.1.3 Relative-pushouts

In this section we give a brief review of the theory of RPOs, a more complete presentation may be found in [64].

Definition 2.1.25 (RPO). Let $\mathbf{C}$ be a category and (i) a commutative diagram.



(i) (ii) (iii) (iii)

Any tuple $\langle I_5, e, f, g\rangle$ which makes (ii) commute is called a *candidate* for (i). A relative-pushout is the "smallest" such candidate. More formally, it satisfies the universal property that given any other candidate $\langle I_6, e', f', g'\rangle$, there exists a *unique* mediating morphism $h\colon I_5 \to I_6$ such that (iii) and (iv) commute.

Another way of viewing RPOs is as ordinary pushouts in a slice-category. Indeed, the commuting square (i) above is simply a span

$$\langle I_2, c\rangle \xleftarrow{a} \langle I_1, ca\rangle \xrightarrow{b} \langle I_3, d\rangle$$

in the slice category $\mathbf{C}/I_4$. It is straightforward to verify that to give a relative-pushout of diagram (i) above is to give a pushout of the span in $\mathbf{C}/I_4$.

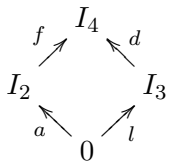
The following notion of idem-relative-pushout, or IPO, is central in the derivation of labelled transition systems. While RPOs allow us to calculate a minimal candidate for a particular redex square, IPOs characterise such minimal candidates. Lemma 2.1.28, due to Leifer and Milner [66] formally explains the relationship between RPOs and IPOs.

Definition 2.1.26 (IPO). A commuting square like diagram (i) of Definition 2.1.25 is a idem-relative-pushout (IPO) if $\langle I_4, c, d, \mathrm{id}_{I_4}\rangle$ is its RPO.

Equipped with a notion of IPO, we are now ready to define a labelled transition system with labels being the smallest contexts.

Definition 2.1.27 (LTS). Let $\mathbb{C}$ be a reactive system with an underlying category $\mathbf{C}$, distinguished object 0, a suitable collection of reactive contexts $\mathbf{D}$ and a set of reaction rules $\mathcal{R}$. We define a labelled transition system LTS($\mathbb{C}$) as follows:

- the states of LTS($\mathbb{C}$) are arrows $a : 0 \to I_2$ of $\mathbf{C}$, where I_2 is arbitrary and not fixed;
- there is a transition from $a : 0 \to I_2$ to $dr : 0 \to I_4$ with label $f : I_2 \to I_4$, written $a \xrightarrow{f} dr$ if the following condition holds: there exists $\langle l, r\rangle \in \mathcal{R}$ and $d \in \mathbf{D}$ such that $fa = dl$ and the redex square below

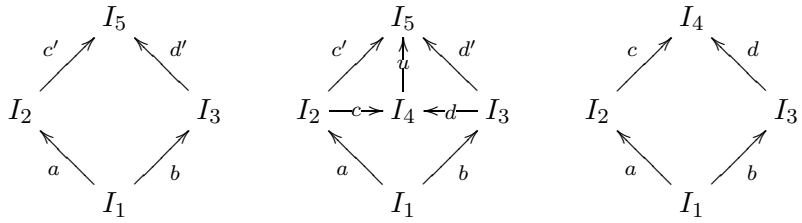


  is an IPO.

In other words, if insertion in context f makes a match l in context d (commutation of the diagram), where l is a redex, and f is the "smallest" such context (IPO condition), then a moves to dr with label f, where r is the reduct of l.

A reactive system $\mathbb{C}$ is said to *have redex RPOs* if every redex square (Definition 2.1.24) has an RPO. One of Leifer and Milner's central results is that if this condition is satisfied then $\sim$, bisimilarity on LTS($\mathbb{C}$), is a congruence [66] (Theorem 2.1.30). The congruence theorem relies on an important property of IPOs, namely that they compose and decompose in a manner similar to pushouts. We first recall a simple result which relates the definitions of IPOs and RPOs.

Lemma 2.1.28 (RPOs are IPOs; IPOs are RPOs).



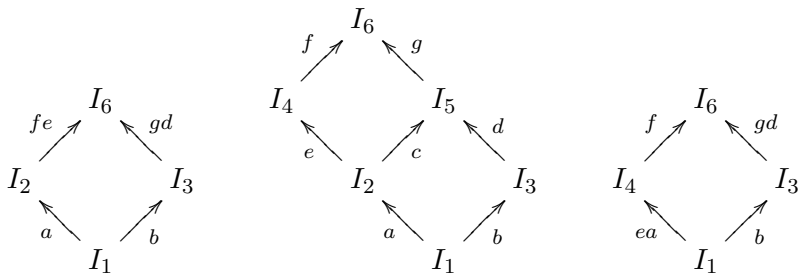
(i) (ii) (iii)

In an arbitrary category, the following hold:

1. (RPOs to IPOs) if $\langle I_4, c, d, u\rangle$ is an RPO for diagram (i), as illustrated in diagram (ii), then diagram (iii) is an IPO;
2. (IPOs to RPOs) if diagram (iii) is an IPO, diagram (i) has an RPO, and $\langle I_4, c, d, u\rangle$ is a candidate for it as shown in diagram (ii), then $\langle I_4, c, d, u\rangle$ is an RPO for diagram (i).

In section 2.2 we shall prove Lemma 2.2.19, which is a more general version of Lemma 2.1.28. The following lemma can be found in [66] and can also be seen as a special case of Lemma 2.2.20 about the properties of a 2-categorical generalisation of IPOs.

Lemma 2.1.29 (Composition and decomposition). Suppose that diagram (i) below has an RPO. Then:



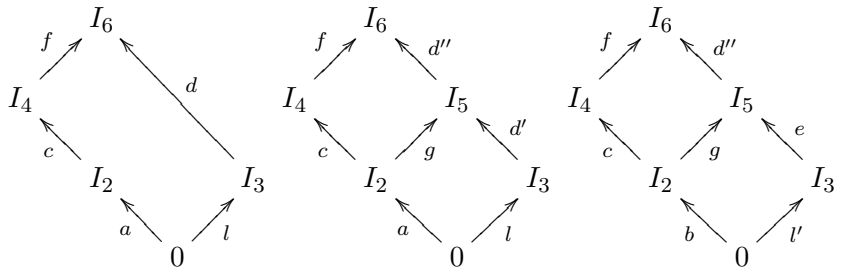
(i) (ii) (iii)

1. (Composition.) if both squares in diagram (ii) are IPOs then the exterior (diagram (iii)) is also an IPO;
2. (Decomposition.) if the lower square and the exterior (diagram (iii)) of diagram (ii) are IPOs then so is the upper square.

We are now ready to recall and prove Leifer and Milner's congruence theorem for bisimilarity [66]. It shall be useful for us to recall the proof because the proofs of the more general results in section 2.3 shall follow a very similar underlying proof strategy.

Theorem 2.1.30 (Leifer and Milner [66]). Suppose that $\mathbb{C}$ has redex RPOs. Then bisimilarity $\sim$ on LTS($\mathbb{C}$) is a congruence.

Proof.



We need to show that, assuming $a \sim b$ and an arbitrary $c \in \mathbf{C}$, we have $ca \sim cb$. It is enough to show that the contextual closure S of bisimilarity

$$S = \{\, \langle ca, cb \rangle \mid a \sim b,\, c \in \mathbf{C} \,\}$$

is a bisimulation. We can do this directly: suppose that $ca \xrightarrow{f} a'$. Then for some $\langle l, r \rangle \in \mathcal{R}$ and $d \in \mathbf{D}$ we have that diagram (i) is an IPO and $a' = dr$. Because $\mathbb{C}$ has redex GRPOs, we are able to construct a GRPO as illustrated in diagram (ii). Using the first part of Lemma 2.1.28 we are able to conclude that the lower square is an IPO. Now using the fact that IPOs decompose (second part of Lemma 2.1.29 we are able to conclude that the upper square is an IPO also.

Since the lower square is an IPO, we have that $a \xrightarrow{g} d'r$. Using our assumption, $b \xrightarrow{g} er'$, for some $e \in \mathbf{D}$ and $\langle l', r' \rangle \in \mathcal{R}$. Moreover, $d'r \sim er'$ and the lower square of diagram (iii) is an IPO. But then composing this IPO with the upper square of diagram (ii) yields an IPO – the exterior of diagram (iii) – using the first part of Lemma 2.1.29. We get that $cb \xrightarrow{f} d''er'$. But since $dr = d''d'r$ and since $d'r \sim er'$ we have that $\langle dr, d''er' \rangle \in S$, which completes the proof. □

This congruence theorem is a special case of a more general theorem, stating the bisimilarity on the labelled transition system generated using a

generalisation of the theory developed by Leifer and Milner to 2-dimensional categories. We shall motivate and present this development in the next section and prove the congruence theorem in Section 2.3.

2.2 Structural congruence and isomorphisms

We shall now motivate and develop an extension to the theory of reactive systems and RPOs – the ability to consider structural congruence/isomorphism as an integral part of the theory. This idea of generalising reactive systems to a 2-categorical setting was proposed independently by Sewell [99]. The technical tools used are 2-categories and a certain notion of colimit for bicategories.

We shall start in §2.2.1 by motivating the extension. After a brief introduction to the technical concepts involved, in §2.2.2 we shall introduce the main technical contributions of this chapter: G-reactive systems and GRPOs. Finally, in §2.2.3 we shall state several important properties of GRPOs.

2.2.1 Limitations of RPOs

In this subsection we shall discuss the motivation for a notion of relative-pushout in a 2-categorical setting. Indeed, it turns out that in many natural examples RPOs either do not exist or do not give the expected congruence. We shall illustrate these problems using our two running examples, the simple process calculus of Example 2.1.7 and abstract bunch contexts of Definition 2.1.12.

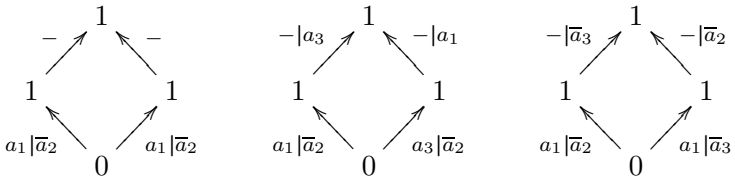
Simple Process Calculus. Recall the simple calculus introduced in Example 2.1.7 and its associated reactive system introduced in Example 2.1.9. Recall that the underlying category of the reactive system is $\mathbf{C}_{\Sigma}^{\equiv}$, the arrows of which are contexts of the process calculus modulo structural congruence.

Example 2.2.1. We shall consider what happens when we use RPOs to generate a labelled transition system for this calculus. Consider the term $a \mid \overline{a}$. Using the operational semantics introduced in Example 2.1.15 we should expect three transitions,

$$a \mid \overline{a} \xrightarrow{a} \overline{a}, \qquad a \mid \overline{a} \xrightarrow{\overline{a}} a \qquad \text{and} \qquad a \mid \overline{a} \xrightarrow{\tau} \mathtt{0}.$$

Consider the three squares in $\mathbf{C}_{\Sigma}^{\equiv}$ below, where we use subscripts to distinguish different occurrences of the term a (that may float around in larger

terms because of $\equiv$). Observe that such distinction is for the sake of exposition only: arrows in $\mathbf{C}_\Sigma^{\equiv\!\!\equiv}$ up to structural congruence, and therefore individual occurrences of terms are not discernible.

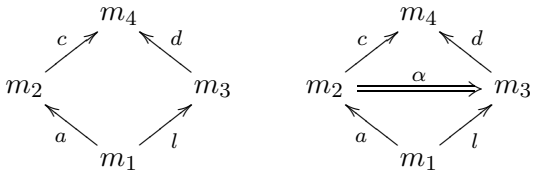


Only the left one could possibly be an IPO, and it is easy to see that it is a *candidate* for the middle and the right squares. Indeed, since the term and the left hand side of the reaction rule are identical, the identity context is clearly the smallest upper bound.

However, also the upper bounds given in the middle and the right square are in some sense *minimal*. Indeed, if we keep track of the place in the term where the reaction occurs, then the middle square is the smallest upper bound whose redex $(a_3 \mid \overline{a}_2)$ *only* uses $\overline{a}$ (as opposed to both a and $\overline{a}$) from the term. Similarly, in the right square the redex created by insertion into a context $(\overline{a}_3 \mid -)$ only uses a. It is precisely the fact that terms in $\mathbf{C}_\Sigma^{\equiv\!\!\equiv}$ are quotiented by $\equiv$ that makes it impossible to place reaction within a term.

For many applications it makes sense to have the extra power of locating reaction. Indeed, the reader may verify that the lts generated using RPOs on $\mathbf{C}_\Sigma^{\equiv\!\!\equiv}$ generates the same set of labels for the terms $a \mid \overline{a}$ and $b \mid \overline{b}$ and thus no operational equivalence can distinguish between these two terms. That is, against the intuition, $a \mid \overline{a}$ and $b \mid \overline{b}$ would be bisimilar. However, when reaction can be located within the redex, bisimulation equivalence coincides with the standard one. We shall demonstrate this for our running process calculus example in Examples 2.2.4 and 2.2.13.

At this point it is important to focus on what exactly is a commuting square in $\mathbf{C}_\Sigma^{\equiv\!\!\equiv}$. In order to verify that a diagram like (i) below is commutative, one has to exhibit a proof of structural congruence.

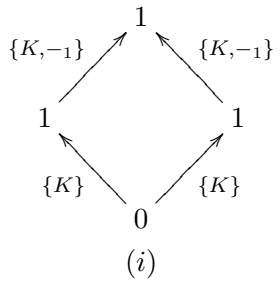


(i) (ii)

Different proofs can be chosen to exhibit commutativity, corresponding to different ways of rearranging the redex using structural congruence. Such "proofs" will for us be appropriate isomorphisms which preserve the structure of the terms, they shall be represented as 2-cells and used to give a 2-categorical structure on $\mathbf{C}_\Sigma$.

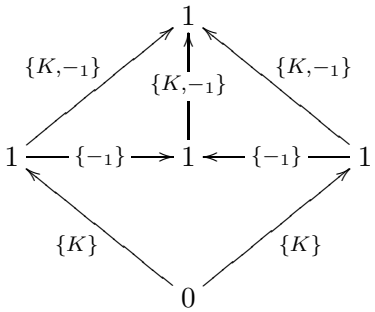
Abstract bunch contexts. Our second example of the theory of RPOs failing concerns the category $\mathbf{Bun}_0$ of abstract bunch contexts, introduced in Definition 2.1.13. Alternatively from the graphical presentation used previously, an abstract bunch context $[c]\colon m_0 \to m_1$ can be depicted as a string of m_1 nonempty multisets on $\mathcal{K} + m_0$ (the bunches of leaves and holes connected to the same root), with the proviso that elements m_0 must appear exactly once in the string. In the examples, we represent elements of m_0 as numbered holes $-_i$.

Example 2.2.2. The category $\mathbf{Bun}_0$ of abstract bunch contexts does not have RPOs. For a simple counterexample, consider diagram (i) below.

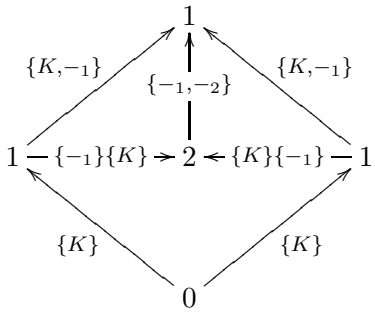


(i)

Diagrams (ii) and (iii) illustrate two candidates for diagram (i). We shall demonstrate that a common 'lower bound' candidate does not exist.

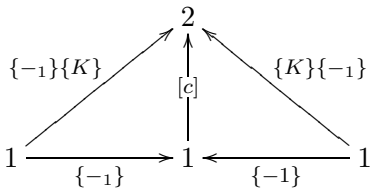


(ii)



(iii)

Indeed, it easily checked that the only candidate which factors the candidate illustrated in diagram (ii) is the trivial identity candidate. This is because $\{-_1\} : 1 \to 1$ has an empty carrier set, and it is the only such arrow with domain 1. Thus it remains to show that there doesn't exist an an abstract bunch context $[c] : 1 \to 2$ such that diagram (iv) commutes.



(iv)

Clearly, c has to have a singleton carrier set with the singleton having label K. Then the only two possibilities are $[c] = \{-_1\}\{K\}$ and $[c] = \{K\}\{-_1\}$, neither of which works.

The point here is that by taking the arrows of $\mathbf{Bun}_0$ up to isomorphism we lose information about *how* diagrams of bunch contexts commute. Diagram (i), for instance, can be commutative in two different ways: the K in the bottom left part may correspond either to the one in the bottom right or to the one in the top right, according to whether we read $\{K, -_1\}$ or $\{-_1, K\}$ for the top rightmost arrow. The point is therefore exactly which *occurrences* of K correspond to each other. The fundamental contribution of this section is to equip our structures of interest with an explicit structure to track such correspondences and to extend the machinery of RPOs to such richer settings.

2.2.2 2-categories, G-categories, GRPOs and GIPOs

Throughout this thesis, we shall be concerned with reactive systems with underlying categories which have 2-dimensional structure, that is "arrows between arrows" or *2-cells*. Moreover, in all of our examples, the 2-cells shall all be isomorphisms. One may think of the 2-dimensional structure as providing information about the structural congruence between terms. As demonstrated by the examples in § 2.2.1, it is important to keep this information for two reasons. Firstly, as demonstrated in 2.2.1, forgetting the 2-cells sometimes results in a labelled transition system on which bisimilarity is too coarse. Secondly, and more importantly, as demonstrated in Example 2.2.2, in categories like the category of bunches and contexts RPOs do not exist. In this subsection we shall recall the standard categorical tech-

nology we shall need, and introduce one of the main technical contributions of this thesis, the notion of groupoidal-relative-pushout (GRPO).

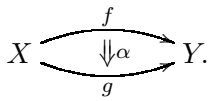
2-categories. We shall start by recalling the definition of 2-categories. For a more thorough introduction we refer the reader to [67, 101, 54]. A *2-category* $\mathbf{C}$ is a category where every homset (that is the collections of arrows between any pair of objects X and Y) is the class of objects of some category $\mathbf{C}(X, Y)$ and, correspondingly, whose composition "functions"

$$\mathbf{C}(Y, Z) \times \mathbf{C}(X, Y) \to \mathbf{C}(X, Z)$$

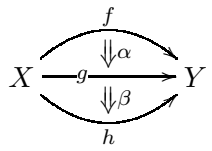
are functors. Explicitly, a 2-category $\mathbf{C}$ consists of the following.

(i) A class of *objects* $X, Y, Z, \ldots$.

(ii) For any $X, Y \in \mathbf{C}$, a category $\mathbf{C}(X, Y)$. The objects $\mathbf{C}(X, Y)$ are called *1-cells*, or simply arrows, and denoted by $f\colon X \to Y$. Its morphisms are called *2-cells*, are written $\alpha\colon f \Rightarrow g\colon X \to Y$ and drawn as



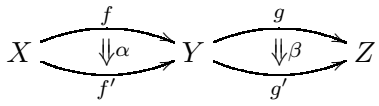
Composition in $\mathbf{C}(X, Y)$ is denoted by $\bullet$ and referred to as '*vertical*' composition. Identity 2-cells are denoted by $1_f\colon f \Rightarrow f$. Isomorphic 2-cells are occasionally denoted as $\alpha\colon f \cong g$. As an example of vertical composition, consider 2-cells $\alpha : f \Rightarrow g$ and $\beta : g \Rightarrow h$ as below.



They can be composed, yielding $\beta \bullet \alpha : f \Rightarrow h$.

(iii) For each X, Y, Z there is a functor $\circ\colon \mathbf{C}(Y, Z) \times \mathbf{C}(X, Y) \to \mathbf{C}(X, Z)$, the so-called '*horizontal*' composition, which we often denote by mere juxtaposition. Horizontal composition is associative and admits 1_{id_X}

as identities. As an example, consider 2-cells $\alpha : f \Rightarrow f'$ and $\beta : g \Rightarrow g'$, as illustrated below.



They can be composed horizontally, obtaining $\beta\alpha : g \circ f \Rightarrow g' \circ f'$.

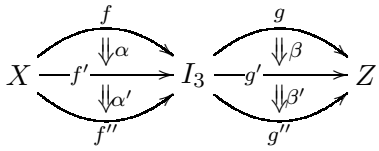
Moreover, we require that horizontal composition of 2-cells is associative and has two sided identities: given $\alpha : f \Rightarrow g : X \to Y$, we have $1_{\mathrm{id}_Y}\alpha = \alpha = \alpha 1_{\mathrm{id}_X}$. This is where 2-categories differ from *bicategories*; in bicategories horizontal composition is associative and has identities only up to coherent isomorphisms (see Chapter 3 for further details).

As a convenient notation, we shall write $\alpha \circ f$ and $g \circ \alpha$ for, respectively, $\alpha \circ 1_f$ and $1_g \circ \alpha$. We follow the convention that horizontal composition binds tighter than vertical composition.

The fact that horizontal composition $\circ$ is a functor can be otherwise said as follows:

(i) for any $f\colon X \to Y$ and $g\colon Y \to Z$ we have that $1_g \circ 1_f = 1_{gf}$

(ii) the *middle-four interchange law*: for $f, f', f''\colon X \to Y$ and $g, g', g''\colon Y \to Z$ and $\alpha\colon f \Rightarrow f'$, $\alpha'\colon f' \Rightarrow f''$, $\beta\colon g \Rightarrow g'$ and $\beta'\colon g' \Rightarrow g''$, as illustrated by

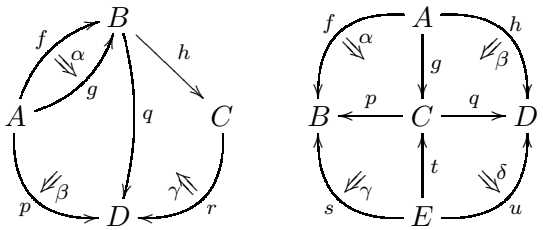


we have

$$\beta'\alpha' \bullet \beta\alpha = (\beta' \bullet \beta)(\alpha' \bullet \alpha).$$

In 2-categories, the order of composition of 2-cells is not important. More precisely, as a consequence of the middle-four interchange law, it can be shown that any diagram of 2-cells defines at most one composite 2-cell; that is, all the possible different ways to combine together vertical and horizontal composition, yield the same composite 2-cell. This primitive operation is referred to as *pasting*.

In order to illustrate the notion of pasting, we shall consider the following diagrams.



The left diagram features 2-cells $\alpha : f \Rightarrow g$, $\beta : qg \Rightarrow p$ and $\gamma : rh \Rightarrow q$. They can be pasted together uniquely to obtain a 2-cell $rhf \Rightarrow p$. This 2-cell can be written as either $\beta \bullet q\alpha \bullet \gamma f : rhf \Rightarrow p$, or equally, $\beta \bullet \gamma g \bullet rh\alpha : rhf \Rightarrow p$. Now consider the right diagram with 2-cells $\alpha : f \Rightarrow pg$, $\beta : h \Rightarrow qg$, $\gamma : pt \Rightarrow s$ and $\delta : qt \Rightarrow u$. There is no way of composing these 2-cells.

The canonical example of a 2-category is **Cat**, the 2-category of categories, functors and natural transformations. The fact that natural transformations satisfy the middle-four interchange property was first identified by Godement [38].

Two objects C, D of a 2-category $\mathbf{C}$ are said to be *equivalent* when there are arrows $f : C \to D$, $g : D \to C$ and isomorphic 2-cells $\alpha : \mathrm{id}_C \Rightarrow gf$, $\beta : fg \Rightarrow \mathrm{id}_D$. We refer to f and g as equivalences.

G-categories. Since the role of 2-cells in our approach is to represent (proofs of) structural congruences, we shall usually consider 2-categories whose 2-cells are all *isomorphisms*. As the categories all of whose morphisms are iso are commonly known as *groupoids*, our 2-categories are precisely the *groupoid-enriched* categories, or G-categories. See the book by Kelly [52] for a thorough treatment of enriched category theory.

Definition 2.2.3 (G-Category). A *G-category* is a category enriched over $\mathbf{G}$, the category of groupoids. In other words, it is a 2-category whose 2-cells are all isomorphisms.

The two examples which we have introduced thus far can presented as G-categories. Firstly, in Example 2.2.4, we shall present a G-category for the simple process calculus of Example 2.1.7. Secondly, in Definition 2.2.5, we present the G-category of concrete bunch contexts, introduced in Definition 2.1.10.

Example 2.2.4. Consider the subset of CCS introduced in Example 2.1.7 and its structural congruence relation introduced in Example 2.1.9.

Instead of studying GRPOs in a two-dimensional version of $\mathbf{C}^{\equiv\equiv}_{\Sigma}$ introduced in Example 2.1.9 and used in Example 2.2.1 we shall simplify the setting down to a one object category. This is done in order to simplify the constructions of GRPOs as far as possible without losing the "flavour of their two dimensionality".

Let $\mathbf{M}_{\Sigma}$ be the G-category with:

- a single object $\bullet$;
- arrows strings $a_0 \mid a_1 \mid \ldots \mid a_{n-1}$, $a_i \in N$ with composition by juxtaposition (e.g. $(a_0 \mid a_1)(a_0 \mid a_1) = a_0 \mid a_1 \mid a_2 \mid a_3$) and the empty string denoted by $\mathtt{0}$ serving as the identity;
- 2-cells permutations; namely, each arrow $a_0 \mid a_1 \mid \ldots \mid a_{n-1}$ is the source of $n!$ 2-cells determined by the permutations $\varphi\colon n \to n$, where $n = \{0, 2, \ldots, n-1\}$. Each such φ determines a 2-cell

  $$\varphi_{a_0,a_1,\ldots,a_{n-1}}\colon a_0 \mid a_1 \mid \cdots \mid a_{n-1} \Rightarrow a_{\varphi^{-1}(0)} \mid a_{\varphi^{-1}(1)} \mid \cdots \mid a_{\varphi^{-1}(n-1)}.$$

  For clarity we will usually leave out the subscripts. So, for example, there are two 2-cells $a \mid a \Rightarrow a \mid a$: the identity, and the permutation that "swaps" the two as. Vertical composition is via composition of permutations, horizontal composition is via juxtaposition, i.e. for $\varphi\colon m \to m$ and $\psi\colon n \to n$, we define $\psi\varphi\colon m+n \to m+n$ by $(\psi\varphi)(i) = \varphi(i)$ for $i < m$ and $(\psi\varphi)(i) = m + \psi(i-m)$ for $i \geq m$.

It should be clear to the reader that $p \equiv q$ iff there exists a 2-cell $\alpha\colon p \Rightarrow q$.

Our second example of a G-category concerns concrete bunch contexts introduced in Definition 2.2.5.

Definition 2.2.5 (Bun). The G-category of bunch contexts, **Bun** has:

1. objects: finite ordinals m_0, m_1, ...;
2. arrows: concrete bunch contexts $c : m_0 \to m_1$;
3. 2-cells: isomorphisms of concrete bunch contexts.

Recall that there is a small problem with associativity; the naive formulations of a two-dimensional category with objects ordinals, arrows concrete bunch contexts and 2-cells bunch context isomorphisms actually yields a bicategory. However, as mentioned in Remark 2.1.11, this problem is easily

fixed by working in a category where the operation of coproduct is associative. We shall treat this in more detail in Chapter 4.

There is a functor $(-)_0$ from the category G-**Cat** of of G-categories and 2-functors to the category **Cat** of categories and functors[2]. Given a G-category $\mathbf{C}$, $\mathbf{C}_0$ is the category with the same objects as $\mathbf{C}$, but arrows $C \to D$ are equivalence classes $[f]$ of arrows $f : C \to D$, where f and f' are in the same equivalence class iff there exists $\alpha : f \Rightarrow f'$. Composition in $\mathbf{C}_0$ is defined by letting $[g][f] = [gf]$. It is easy to check that the composition operator is well-defined, if $\alpha' : g' \Rightarrow g$ and $\alpha : f' \Rightarrow f$ then $(\alpha' f) \bullet (g'\alpha) : g'f' \Rightarrow gf$. It is easy to check that the identity and associativity laws hold.

A 2-functor $F : \mathbf{C} \to \mathbf{D}$ is taken to the functor $F_0 : \mathbf{C}_0 \to \mathbf{D}_0$ which acts as F on objects, but sends $[f] : C \to D$ to $[Ff] : FC \to FD$. This is well defined because 2-functors respect 2-structure, that is if $\alpha : f' \Rightarrow f$ then $F\alpha : Ff' \Rightarrow Ff'$.

As an example, applying the functor $(-)_0$ to the G-category **Bun** of Definition 2.2.5 yields the category $\mathbf{Bun}_0$ of Definition 2.1.13.

G-reactive systems and GRPOs. We can generalise the notion of reactive system (Definition 2.1.1) to the setting of G-categories.

Definition 2.2.6 (G-reactive system). A *G-reactive system* $\mathbb{C}$ consists of

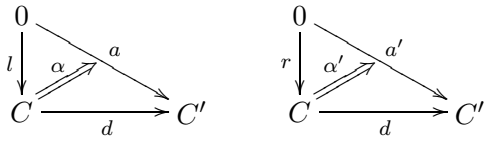
1. a G-category $\mathbf{C}$,
2. a collection $\mathbf{D}$ of arrows of $\mathbf{C}$ which shall be referred to as the *reactive contexts*; it is required to be closed under 2-cells and reflect composition,
3. a distinguished object $0 \in \mathbf{C}$,
4. a set of pairs $\mathcal{R} \subseteq \bigcup_{C \in \mathbf{C}} \mathbf{C}(0, C) \times \mathbf{C}(0, C)$ called the *reaction rules*.

The reactive contexts are those contexts inside which evaluation may occur. By composition-reflecting we mean that $dd' \in \mathbf{D}$ implies $d \in \mathbf{D}$ and $d' \in \mathbf{D}$, while the closure property means that given $d \in \mathbf{D}$ and $\alpha\colon d \Rightarrow d'$ in $\mathbf{C}$ implies $d' \in \mathbf{D}$. The reaction relation $\longrightarrow\!\!\triangleright$ is defined by taking

$$a \longrightarrow\!\!\triangleright a' \quad \text{if there exists } \langle l, r\rangle \in \mathcal{R},\ d \in \mathbf{D} \text{ and } \alpha\colon dl \Rightarrow a,\ \alpha'\colon a' \Rightarrow dr$$

[2]In fact, $(-)_0$ is a 2-functor from the 2-category G-**Cat** of G-categories, 2-functors and pseudo-natural transformations to the 2-category **Cat** of categories, functors and natural transformations.

As illustrated by the diagram below, this represents the fact that, up to structural congruence (as witnessed by α), a is the left-hand side l of a reaction rule in a reactive context d, while a' is, up to structural congruence (witness α'), the corresponding right-hand side r of the reaction rule in the reactive context d.



The set $\mathcal{R}$ of reaction rules is, therefore, a set of base rules with which one generates the reaction relation $\longrightarrow\!\!\triangleright$ by closure under suitable contexts. For pragmatic reasons, we choose not to stipulate that $\mathcal{R}$ is to be closed under structural congruence; that is, in our formalism, under 2-cells. More precisely, we do not require that $\langle l', r' \rangle \in \mathcal{R}$ if there exist $\langle l, r \rangle \in \mathcal{R}$ and 2-cells $\alpha : l \Rightarrow l'$, $\beta : r \Rightarrow r'$. Indeed, modern process calculi often have very simple reaction rules and the closure under structural congruence comes at the point of defining the reaction relation. For example, the standard textbook definition of CCS [72] lists the single reaction rule, for each name a, without listing, additionally, all

$$\overline{a.P + P' \mid \overline{a}.Q + Q' \longrightarrow\!\!\triangleright P \mid Q}$$

of its structurally congruent variants. It is easy to check that, if we did choose to impose this condition ($\mathcal{R}$ closed under 2-cells) then the reaction relation $\longrightarrow\!\!\triangleright$, as well as the canonical labelled transition system (Definition 2.2.16) would remain unchanged.

There is a canonical way of getting from a G-reactive system to a reactive system, via a canonical way of getting from a G-category to a category - the functor $(-)_0$. Intuitively, this involves quotienting all arrows by isomorphism. From a process calculus point of view, this means quotienting the set of terms and contexts by structural congruence.

Definition 2.2.7 (G-reactive systems to reactive systems). Given a G-reactive system $\mathbb{C}$ with underlying G-category $\mathbf{C}$, reactive contexts $\mathbf{D}$ and reaction rules $\mathcal{R}$, let $\mathbb{C}_0$ denote the reactive system with underlying category $\mathbf{C}_0$, reactive contexts $\mathbf{D}_0$ and reaction rules $\mathcal{R}_0 = \{\, \langle [l], [r] \rangle \mid \langle l, r \rangle \in \mathcal{R} \,\}$

This translation preserves the relevant structure of reactive systems, namely the reaction relation.

Lemma 2.2.8. Suppose that $\mathbb{C}$ is a G-reactive system. Then

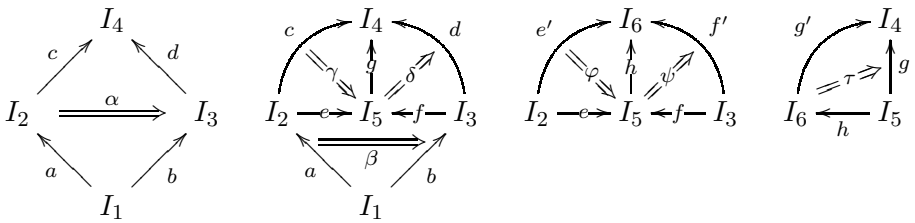
$$a \longrightarrow\!\!\!\!\triangleright\, a' \text{ in } \mathbb{C} \quad \text{iff} \quad [a] \longrightarrow\!\!\!\!\triangleright\, [a'] \text{ in } \mathbb{C}_0$$

Proof. If $a \longrightarrow\!\!\!\!\triangleright\, a'$ in $\mathbb{C}$ then there exists $\langle l, r\rangle$, $d \in \mathbf{D}$ and $\alpha : dl \Rightarrow a$, $\alpha' : a' \Rightarrow dr$, then we have $\langle [l], [r]\rangle \in \mathcal{R}_0$, $[d] \in \mathbf{D}$ and $[a] = [dl] = [d][l]$, $[a'] = [dr] = [d][r]$ which gives $[a] \longrightarrow\!\!\!\!\triangleright\, [a']$ in $\mathbb{C}_0$.

On the other hand, if $[a] \longrightarrow\!\!\!\!\triangleright\, [a']$ in $\mathbb{C}_0$ then $[a] = [d][l]$ and $[a'] = [d][r]$ for some $\langle [l], [r]\rangle \in \mathcal{R}_0$, and $[d] \in \mathbf{D}_0$. Then, we have $\beta : dl \Rightarrow a$ and $\beta' : a' \Rightarrow dr$ for some β and β'. By the closure property on $\mathbf{D}$, we can conclude that $d \in \mathbf{D}$. By the definition of $\mathcal{R}_0$ we know that there exist $\langle l', r'\rangle \in \mathcal{R}$ such that $\gamma : l' \Rightarrow l$ and $\gamma' : r \Rightarrow r'$ for some γ and γ'. Now $\beta \bullet d\gamma : dl' \Rightarrow a$ and $d\gamma' \bullet \beta' : a' \Rightarrow dr'$ which gives $a \longrightarrow\!\!\!\!\triangleright\, a'$. □

We shall now present a generalisation of the notion of RPO to G-categories.

Definition 2.2.9 (GRPO). Let $\mathbf{C}$ be a G-category. A *candidate* for square (i) below



(i) (ii) (iii) (iv)

is a tuple $\langle I_5, e, f, g, \beta, \gamma, \delta\rangle$ such that $\delta b \bullet g\beta \bullet \gamma a = \alpha$. In other words, the 2-cells γ, β and δ, illustrated in diagram (ii), paste together to give α.

A *groupoidal-relative-pushout* (GRPO) for (i) is a candidate which satisfies a universal property, namely, for any other candidate as specified below,

$$\langle I_6,\, e' : I_2 \to I_6,\, f' : I_3 \to I_6,\, g' : I_6 \to I_4,\\ \beta' : e'a \Rightarrow f'b,\, \gamma' : c \Rightarrow g'e',\, \delta' : g'f' \Rightarrow d\rangle$$

there exists a *mediating morphism*: a quadruple

$$\big\langle h : I_5 \to I_6,\, \varphi : e' \Rightarrow he,\, \psi : hf \Rightarrow f',\, \tau : g'h' \Rightarrow g\big\rangle$$

illustrated in diagrams (iii) and (iv). The equations that need to be satisfied are:

1. $\tau e \bullet g' \varphi \bullet \gamma' = \gamma$;
2. $\delta' \bullet g' \psi \bullet \tau^{-1} f = \delta$;
3. $\psi b \bullet h \beta \bullet \varphi a = \beta'$.

Such a mediating morphism must be *essentially unique*, namely, for any other mediating morphism $\langle h', \varphi', \psi', \tau' \rangle$ there must exist a unique 2-cell $\xi \colon h \Rightarrow h'$ which makes the two mediating morphisms compatible, i.e.:

1. $\xi e \bullet \varphi = \varphi'$;
2. $\psi \bullet \xi^{-1} f = \psi'$;
3. $\tau' \bullet g' \xi = \tau$.

We shall sometimes refer to such ξ as a *modification* between two mediating morphisms.

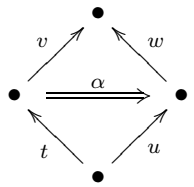
Observe that whereas RPOs are defined up to isomorphism, GRPOs, since they are certain bicolimits (see Section 3.1), are defined up to equivalence. It is easy to show that GRPOs directly generalise RPOs: if one considers a category as a discrete 2-category (the only 2-cells are identities) then a GRPO is an RPO.

Lemma 2.2.10 (GRPOs in $\mathbf{M}_\Sigma$). The 2-category $\mathbf{M}_\Sigma$ of Example 2.2.4 has GRPOs.

We shall delay a proof of Lemma 2.2.10 to Chapter 4. Instead, we shall now present the central concept of GIPO and show a simple characterisation of GIPOs in $\mathbf{M}_\Sigma$.

Definition 2.2.11 (GIPO). Diagram (i) of Definition 2.2.9 is said to be a G-idem-pushout (GIPO) if $\langle I_4, c, d, \mathrm{id}_{I_4}, \alpha, 1_c, 1_d \rangle$ is its GRPO.

Lemma 2.2.12 (Characterisation of GIPOs in $\mathbf{M}_\Sigma$). A diagram

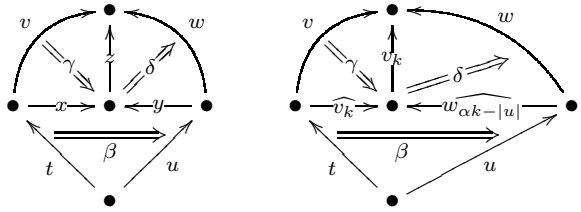


in the 2-category $\mathbf{M}_\Sigma$ of Example 2.2.4 is a GIPO iff condition

$$\alpha(i + |t|) < |u| \text{ for all } 0 \leq i < |v| \tag{2.4}$$

is satisfied.

Proof. Suppose that condition (2.4) is satisfied and that diagram (i) below

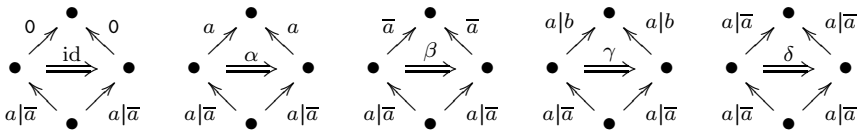


(i) (ii)

is a candidate. It is easy to see that $g = \mathtt{0}$, if this wasn't the case then $\alpha(\gamma^{-1}(|x| + 1) + |t|) \geq |u|$, which contradicts condition (2.4). Then the components of $\langle \mathtt{0}, \gamma^{-1}, \delta^{-1}, \mathrm{id} \rangle$ form a mediating morphism. This mediating morphism is unique (since there is only one automorphism $\mathtt{0} \to \mathtt{0}$) and therefore essentially unique.

On the other hand, suppose that the condition doesn't hold, namely, there exists $0 \leq k < |v|$ such that $\alpha(k + |t|) \geq |u|$. Here we introduce some notation, for an arrow w of $\mathbf{M}_\Sigma$ and $0 \leq l < |w|$ we let $\widehat{w_l}$ denote $w_0 \mid \ldots \mid w_{l-1} \mid w_{l+1} \mid \ldots \mid w_{|w|}$. Consider diagram (ii) where β, γ and δ are defined in the obvious way so that they compose to give α. Clearly $\langle \widehat{v_k}, \widehat{w_{\alpha k-|u|}}, v_k, \beta, \gamma, \delta \rangle$ is a non-trivial candidate, which means that the exterior is not a GIPO. □

Example 2.2.13. Consider the category $\mathbf{M}_\Sigma$ from Example 2.2.4 and the problem of RPOs not giving "enough" smallest contexts in Example 2.2.1. By remembering the position of the redex in the composite term with the help of 2-cells, the first three of the following diagrams



where $\alpha(0) = 2$, $\alpha(1) = 1$ and $\alpha(2) = 0$, and $\beta(0) = 0$, $\beta(1) = 2$ and $\beta(2) = 1$ are GIPOs. Informally, the two copies of a and $\overline{a}$ are swapped, respectively, in the second and third diagram. One can check that the diagrams are GIPOs by using the characterisation of GIPOs given in Lemma 2.2.12.

On the other hand, one may also use the characterisation to show that for any $b \neq a, \overline{a}$ and any γ, the fourth diagram cannot be a GIPO as either b or $a \mid b$ can be factored out.

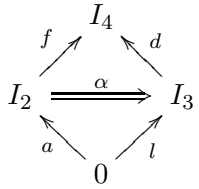
The final diagram *is* a GIPO if we take $\delta(0) = 2$, $\delta(1) = 3$, $\delta(2) = 0$ and $\delta(3) = 1$. That is, the entire redex is given by the context with our term playing no part in the reduction. Such contexts clearly do not give any information about the term, and thus should be avoided. Jensen and Milner, in their report on bigraphs [46], refer to such transitions as *not engaged*. We conjecture that adding a tensor product structure on the category of contexts would allow one to characterise such labels in a general way and thus allow one to avoid them. However, it is clear that such labels do not change the congruence because every term is equipped with them. We plan to pursue this problem in more detail as future work.

Recall from Example 2.2.2 that RPOs do not exist in the category $\mathbf{Bun}_0$ of abstract bunches. On the other hand, GRPOs *do* exist in the G-category **Bun**. The general construction is beyond the scope of this chapter and shall be presented in Chapter 4.

Labelled transition systems. We are now ready to define a labelled transition system generated with help of GRPOs.

Definition 2.2.14 (Concrete lts). For $\mathbb{C}$ a reactive system whose underlying category $\mathbf{C}$ is a G-category, define CTS($\mathbb{C}$) as follows:

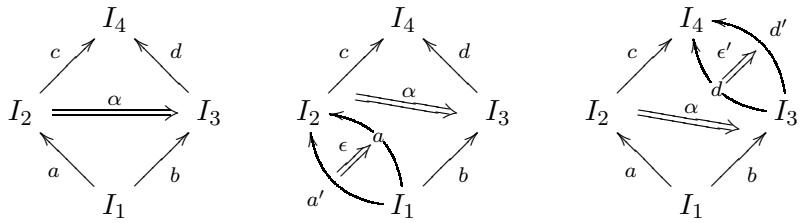
- the states CTS($\mathbb{C}$) are arrows $a\colon 0 \to I_2$ in $\mathbf{C}$, where I_2 is arbitrary and not fixed;
- there is a transition $a \xrightarrow{f} a'$ iff there exists $\langle l, r\rangle \in \mathcal{R}$, $d \in \mathbf{D}$ and a 2-cell $\alpha : fa \Rightarrow dl$ such that the redex square below is a GIPO and $a' = dr$.



The labelled transition system of Definition 2.2.14 is not satisfactory because the states and the labels are not quotiented by isomorphism. For reasoning purposes it is more useful to consider an *abstract* lts, this is the usual case in process calculi where terms in the lts are assumed to be taken up to structural congruence. Thus the states of the final lts should be taken up to isomorphism, since intuitively, structurally congruent terms should allow the same interactions with the environment. Similarly, intuition tells

us that isomorphic contexts provide "the same" information for the term in order to interact, thus it is an overkill to allow two isomorphic labels. This intuition is supported by the theory of GIPOs, as demonstrated by Lemma 2.2.15

Lemma 2.2.15. Suppose that diagram (i) below is a GIPO and than $\epsilon\colon a' \Rightarrow a$, $\epsilon'\colon d \Rightarrow d'$ are isomorphisms.



Then:

1. the exterior of diagram (ii) is a GIPO;
2. the exterior of diagram (iii) is a GIPO.

Proof. (1). Suppose that C, as defined below, is a candidate for the exterior of diagram (ii).

$$C = \langle I_5,\, e,\, f,\, g,\, \beta : ea' \Rightarrow fb,\, \gamma : c \Rightarrow ge,\, \delta : gf \Rightarrow d \rangle$$

Then by definition we have that $\delta b \bullet g\beta \bullet \gamma a' = \alpha \bullet c\epsilon$, and thus

$$\alpha = \delta b \bullet g\beta \bullet \gamma a' \bullet c\epsilon^{-1} = \delta b \bullet g\beta \bullet ge\epsilon^{-1} \bullet \gamma a' = \delta b \bullet g(\beta \bullet e\epsilon^{-1}) \bullet \gamma a'$$

which means that $C' = \langle I_5,\, e,\, f,\, g,\, \beta \bullet e\epsilon^{-1},\, \gamma,\, \delta \rangle$ is a candidate for diagram (i). We obtain a mediating morphism

$$\langle h : I_4 \to I_5,\, \varphi : e \Rightarrow hc,\, \psi : hd \Rightarrow f,\, \tau : gh \Rightarrow \mathrm{id}_{I_4} \rangle \quad (*)$$

between $\langle I_4,\, c,\, d,\, \mathrm{id}_{I_4},\, \alpha,\, 1_c,\, 1_d \rangle$ and C'. In particular,

$$\tau c \bullet g\varphi \bullet \gamma = 1_c,\ \delta \bullet g\psi \bullet \tau^{-1} d = 1_d \text{ and } \psi b \bullet h\alpha \bullet \varphi a = \beta \bullet e\epsilon^{-1}.$$

If we precompose the arrow described by the last equation with $e\epsilon^{-1}$ and use the middle-four interchange, we obtain $\psi b \bullet h(\alpha \bullet c\epsilon) \bullet \varphi a = \beta$, which means

that $(*)$ is also a mediating morphism between $\langle I_4,\, c,\, d,\, \mathrm{id}_{I_4},\, \alpha \bullet c\epsilon,\, 1_c,\, 1_d\rangle$ and C. Conversely, any such mediating morphism translates into a mediating morphism between $\langle I_4,\, c,\, d,\, \mathrm{id}_{I_4},\, \alpha,\, 1_c,\, 1_d\rangle$ and C', and thus essential uniqueness follows.

(2). If $\langle I_5,\, e,\, f,\, g,\, \beta : ea \Rightarrow fb,\, \gamma : c \Rightarrow ge,\, \delta : gf \Rightarrow d'\rangle$ is a candidate for the exterior of diagram (*iii*), then $\delta b \bullet g\beta \bullet \gamma a = \epsilon' b \bullet \alpha$ and by simple rearranging it follows that $(\epsilon'^{-1} \bullet \delta)b \bullet g\beta \bullet \gamma a = \alpha$ which in turn means that $\big\langle I_5,\, e,\, f,\, g,\, \beta,\, \gamma,\, \epsilon'^{-1} \bullet \delta\big\rangle$ is a candidate for (i). Hence there is a mediating morphism

$$\langle h : I_4 \to I_5,\, \varphi : e \Rightarrow hc,\, \psi : hd \Rightarrow f,\, \tau : gh \Rightarrow \mathrm{id}_{I_4}\rangle$$

satisfying

$$\tau c \bullet g\varphi \bullet \gamma = 1_c,\ (\epsilon'^{-1} \bullet \delta) \bullet g\psi \bullet \tau^{-1} d = 1_d \text{ and } \psi b \bullet h\alpha \bullet \varphi a = \beta.$$

The second equation transforms to $\delta \bullet g(\psi \bullet h\epsilon'^{-1}) \bullet \tau^{-1} d' = 1_{d'}$ and it follows that $\big\langle h,\, \varphi,\, \psi \bullet h\epsilon'^{-1},\, \tau\big\rangle$ is a mediating morphism for the region. Essential uniqueness follows for the same reason as in case (1). □

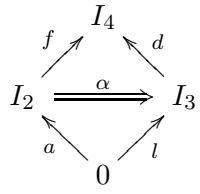
We shall now introduce the definition of the abstract labelled transition systems. Here, the states as well as the transitions are quotiented by structural congruence. This reduces the size of the labelled transition system and is the usual procedure in the theory or process calculi. Intuitively, the 2-dimensional information, while crucial in order to determine where the redex is in the term for an individual interaction, is not important when considering all the possible interactions, which is the function of the labelled transition system.

Definition 2.2.16 gives the definition of the abstract labelled transition system. The central result about the abstract transition system is Lemma 2.2.17, which utilises the conclusions of Lemma 2.2.15 to show that we retain the elegant characterisation of labels in terms of GIPOs, as in Definition 2.2.14.

Definition 2.2.16 (Abstract lts). For $\mathbb{C}$ a reactive system whose underlying category $\mathbf{C}$ is a G-category, define ATS($\mathbb{C}$) as follows:

- the states ATS($\mathbb{C}$) are iso-classes of arrows $[a]\colon 0 \to I_2$ in $\mathbf{C}$, where I_2 is arbitrary and not fixed;
- there is a transition $[a] \xrightarrow{[f]} [a']$ if there exists a transition $a \xrightarrow{f} a'$ in CTS($\mathbb{C}$).

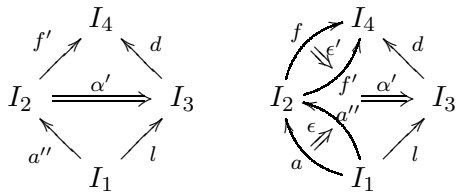
Lemma 2.2.17. There is a transition $[a] \xrightarrow{[f]} [a']$ in ATS($\mathbb{C}$) if and only if there exists $\langle l, r\rangle \in \mathcal{R}$, $d \in \mathbf{D}$ and a 2-cell $\alpha : fa \Rightarrow dl$ such that



is a GIPO and $a' \cong dr$.

Proof. The "if" part is easy, if the diagram is a GIPO then $a \xrightarrow{f} dr$ in CTS($\mathbb{C}$) and so $[a] \xrightarrow{[f]} [dr]$ in ATS($\mathbb{C}$). But $a' \cong dr$ which means that $[a'] = [dr]$ and so $[a] \xrightarrow{[f]} [a']$.

Suppose that $[a] \xrightarrow{[f]} [a']$. Then, for some $a'' \cong a$, $f' \cong f$ and $a''' \cong a'$ $(*)$ we have $a'' \xrightarrow{f'} a'''$. Then there exists $\langle l, r\rangle\, \mathcal{R}$, $d \in \mathbf{D}$ and a 2-cell $\alpha' : f'a'' \Rightarrow dl$ such that diagram (i) below



(i) (ii)

is a GIPO and $a''' = dr$ $(**)$. Let $\epsilon : a \Rightarrow a''$ and $\epsilon' : f \Rightarrow f'$ be arbitrary isomorphisms, and let $\alpha = \alpha' \bullet \epsilon' a'' \bullet f\epsilon$, illustrated in diagram (ii). Using the two parts of Lemma 2.2.15 allows us to conclude that diagram (ii) is a GIPO, and $(*)$ together with $(**)$ give us $a' \cong dr$, as required. □

As a companion to the previous lemma, we state a way of relating the transitions of the concrete and the abstract labelled transition systems.

Lemma 2.2.18. If there is a transition $[a] \xrightarrow{[f]} [a']$ in ATS($\mathbb{C}$) then there exists a'' such that $a'' \cong a'$ and a transition $a \xrightarrow{f} a''$ in CTS($\mathbb{C}$).

Proof. The if direction is trivial and follows from the definition of the abstract labelled transition system.

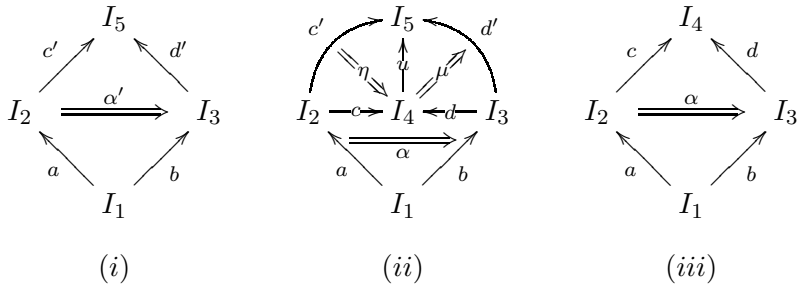
If $[a] \xrightarrow{[f]} [a']$, using Lemma 2.2.17 we obtain a redex square which implies that $a \xrightarrow{f} dr$ and $a' \cong dr$. □

2.2.3 Composition and decomposition

In order to show the good behaviour of the labelled transition system of Definition 2.2.16 we shall need to develop the theory of GRPOs and GIPOs further. In this section we state the main technical results needed for the congruence theorems in this chapter. We defer the proofs until Chapter 3.

The following lemma explains the relationship between GIPOs and GRPOs.

Lemma 2.2.19 (GRPOs are GIPOs; GIPOs are GRPOs). Given diagrams (i), (ii) and (iii)



in an arbitrary G-category, the following hold:

1. (GRPOs to GIPOs) if (2.5)

$$\langle I_4,\, c,\, d,\, u,\, \alpha : ca \Rightarrow db,\, \eta : c' \Rightarrow uc,\, \mu : ud \Rightarrow d' \rangle \qquad (2.5)$$

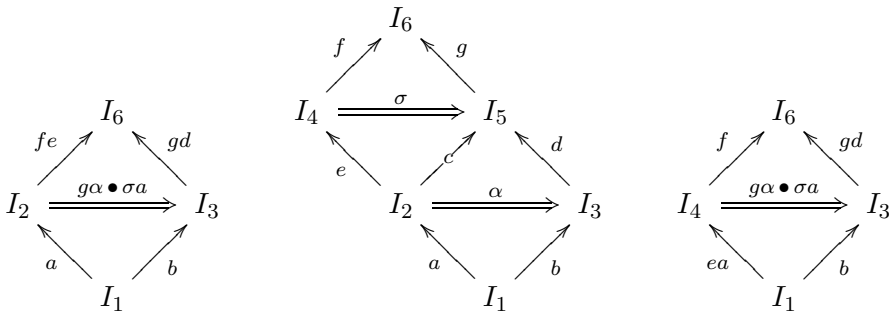
   is a GRPO for diagram (i), as illustrated in diagram (ii), then diagram (iii) is a GIPO;

2. (GIPOs to GRPOs) if diagram (iii) is a GIPO, diagram (i) has a GRPO, and (2.5) is a candidate for it as shown in diagram (ii), then (2.5) is a GRPO for diagram (i).

The following lemma states that, if the underlying category has GRPOs then GIPOs behave somewhat like ordinary pushouts (see Lemma 1.3.2). The lemma generalises Lemma 2.1.29.

Lemma 2.2.20 (Composition and decomposition). Suppose that dia-

gram (i) below has a GRPO. Then:



(i) (ii) (iii)

1. if both squares in diagram (ii) are GIPOs then the exterior (diagram (iii)) is also a GIPO;

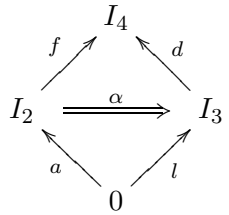
2. if the lower square and the exterior (diagram (iii)) of diagram (ii) are GIPOs then so is the upper square.

2.3 Congruence theorems

In this section we formulate and prove our central Theorem 2.3.6 which states that bisimilarity on the labelled transition system derived using the procedure outlined in Definition 2.2.16 is a congruence, provided that sufficient GRPOs exist. Following a proof strategy similar to that of [64], the proof depends on Lemmas 2.2.19 and 2.2.20.

The following notion is the only assumption made on the reactive system considered in order to prove the congruence theorems.

Definition 2.3.1 (Redex GRPOs). A G-reactive system $\mathbb{C}$ is said to have *redex GRPOs* if in its underlying G-category $\mathbf{C}$ there exist GRPOs for all redex squares (Definition 2.1.24). Recall that a redex square is a diagram



where l is the left-hand side of a reaction rule $\langle l, r\rangle \in \mathcal{R}$, and $d \in \mathbf{D}$ and f, a and α are arbitrary.

Example 2.3.2. Recall that in Example 2.2.13 we showed that $\mathbf{M}_\Sigma$ of Example 2.2.4 has all GRPOs. In particular, this implies that $\mathbf{M}_\Sigma$ has redex GRPOs for any choice of $\mathcal{R}$.

Observe that this means that there exists a GRPO for each possible interaction between a term and a context. We are therefore able to determine a 'smallest' label f to capture each of them in ATS($\mathbb{C}$).

In addition to proving that bisimilarity is a congruence in §2.3.1, we shall show that the trace and failures preorders are compositional in §2.3.2 and §2.3.3.

2.3.1 Bisimilarity

We first observe that it is enough to prove that bisimilarity is a congruence on the concrete lts of Definition 2.2.14. In this section we assume that $\mathbb{C}$ is an arbitrary reactive system.

Definition 2.3.3. Recall that bisimilarity refers to the largest bisimulation relation. We shall refer to bisimilarity on the concrete labelled transition system CTS($\mathbb{C}$) as *concrete bisimilarity* and denote it by $\sim_c$.

Bisimilarity on the abstract labelled transition system ATS($\mathbb{C}$) shall be referred to as *abstract bisimilarity* or simply *bisimilarity*. It shall be denoted by $\sim$.

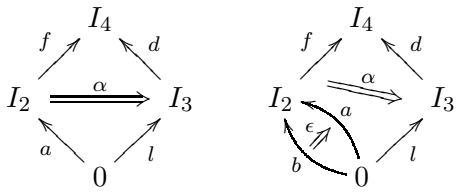
First, we shall show that structurally congruent terms are bisimilar over CTS($\mathbb{C}$).

Lemma 2.3.4. If $a \cong b$ then $a \sim_c b$.

Proof. It suffices to show that

$$\{\, (a, b) \mid a \cong b \,\}$$

is a bisimulation. Indeed, suppose that $\epsilon : b \Rightarrow a$ is an isomorphism and there is a transition $a \xrightarrow{f} a'$, then there exists a GIPO, illustrated in diagram (i)



(i) (ii)

where $\langle l, r\rangle \in \mathcal{R}$, $d \in \mathbf{D}$ and $a' = dr$. But then, using the conclusion of Lemma 2.2.15, diagram (ii) is also a GIPO and thus $b \xrightarrow{f} a'$. Clearly $a' \cong a'$ and we are finished. □

The following lemma relates the two bisimilarities.

Lemma 2.3.5.

$$a \sim_c b \quad \text{iff} \quad [a] \sim [b]$$

Proof. The proof is very simple because of the similarity of the label derivation procedures of Definitions 2.2.14 and the characterisation of the labels of the abstract transition system of Lemma 2.2.17.

First we shall show the "only if" part, that is $a \sim_c b$ implies $[a] \sim [b]$. It suffices to show that

$$S = \{\, \langle [a], [b]\rangle \mid a \sim_c b \,\}$$

is a bisimulation over ATS($\mathbb{C}$). Indeed, suppose that $[a] \xrightarrow{[f]} [a']$. Using the conclusion of Lemma 2.2.18, there exists a'' such that $a \xrightarrow{f} a''$ and $a' \cong a''$ in CTS($\mathbb{C}$). Using Lemma 2.3.4, $a' \sim_c a''$ $(*)$. Using $a \sim_c b$ we get $b \xrightarrow{f} b'$ with $a'' \sim_c b'$ $(**)$. By definition there is a transition $[b] \xrightarrow{[f]} [b']$ But $([a'], [b']) \in S$ since $(*)$ and $(**)$ hold and bisimilarity is transitive.

The "if" part of the proof is similar. Indeed, it suffices to prove that

$$S' = \{\, \langle a, b\rangle \mid [a] \sim [b] \,\}$$

is a bisimulation on CTS($\mathbb{C}$). Suppose that $a \xrightarrow{f} a'$, then $[a] \xrightarrow{[f]} [a']$, and therefore, $[b] \xrightarrow{[f]} [b']$ such that $[a'] \sim [b']$ $(*)$. Using the conclusion of Lemma 2.2.18, there exists b'' such that $b' \cong b''$ and $b \xrightarrow{f} b''$. Then $[b'] = [b'']$ and using $(*)$ we obtain $[a'] \sim [b'']$ which means that $\langle a', b''\rangle \in S'$. □

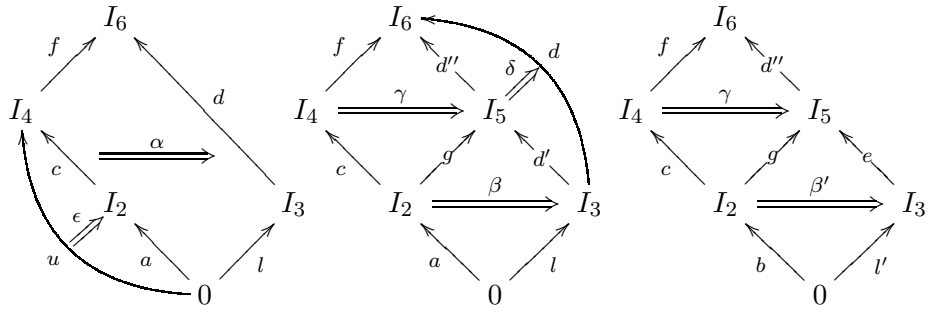
The close correspondence between bisimilarities on the concrete transition system and and the abstract transition system shall be useful in order to prove the congruence theorem for bisimilarity. In particular, it is enough to prove the theorem for concrete bisimilarity, the congruence of abstract bisimilarity follows as an easy corollary.

Theorem 2.3.6. *Let $\mathbb{C}$ be a reactive system whose underlying G-category $\mathbf{C}$ has redex GRPOs. Then bisimilarity $\sim$ on* CTS($\mathbb{C}$) *is a congruence.*

Proof. It suffices to show that

$$S = \{\, \langle u,\, v\rangle \mid u \cong ca,\, v \cong cb,\, c \text{ in } \mathbf{C} \text{ and } a \sim_c b \,\}$$

is a bisimulation. Suppose that $a \sim_c b$, $\epsilon : u \Rightarrow ca$ and $u \xrightarrow{f} a'$. Then



(i) (ii) (iii)

there exists $\langle l,\, r\rangle \in \mathcal{R}$, $d\colon I_3 \to I_6$ and $\alpha \bullet f\epsilon\colon fu \Rightarrow dl$ such that the exterior of diagram (i) is a GIPO and $a' = dr$. Using the first part of Lemma 2.2.15 we obtain that also the rectangle of diagram (i) is a GIPO. Since $\mathbb{C}$ has redex-GRPOs, there exists a GRPO

$$\big\langle I_5,\, g,\, d',\, d'',\, \beta : ga \Rightarrow d'l,\, \gamma : fc \Rightarrow d''g,\, \delta : d''d' \Rightarrow d\big\rangle$$

as shown in diagram (ii). By the first part of Lemma 2.2.19, the lower square in diagram (ii) is a GIPO.

Thus $a \xrightarrow{g} d'r$, and since $a \sim_c b$, $b \xrightarrow{g} b'$ where $b' \sim_c d'r$. By definition, there is a pair $\langle l',\, r'\rangle \in \mathcal{R}$, an arrow $e\colon I_3 \to I_5$ and a two-cell $\beta'\colon gb \Rightarrow el'$ so that the lower square of diagram (iii) is a GIPO and $b' = er'$.

The second part of Lemma 2.2.15 implies that the composite of the two squares in diagram (ii) is a GIPO and therefore, since the lower square is also a GIPO, applying part 2 of Lemma 2.2.20 allows us to conclude that the upper square is a GIPO. Since we have deduced that both the squares in diagram (iii) are GIPOs, part 1 of Lemma 2.2.20 ensures that the entire region is a GIPO and therefore $cb \xrightarrow{f} d''er'$. Since $d'r \sim_c er'$ and $d''d'r \cong dr$ we conclude that $\langle dr,\, d''er'\rangle \in S$, which finishes the proof. □

The following easy corollary is the central result of this chapter.

Corollary 2.3.7. *Abstract bisimilarity $\sim$ on* ATS($\mathbb{C}$) *is a congruence.*

Proof. Suppose that $[a] \sim [b]$. First, by Lemma 2.3.5 $a \sim_c b$. By Theorem 2.3.6, $ca \sim_c cb$, for any context c, and finally, again using Lemma 2.3.5, $[ca] \sim [cb]$. □

The next two subsections complement this result by proving congruence theorems for trace and failure equivalence. Theorems and proofs in this section follow closely those for RPOs [64].

2.3.2 Traces preorder

Trace semantics [86] is a simple notion of equivalence which equates processes if they can engage in the same sequences of actions. Even though it lacks the fine discriminating power of branching time equivalences, such as bisimilarity, it is nevertheless interesting because many safety properties can be expressed as conditions on sets of traces.

We say that a sequence $f_1 \cdots f_n$ of labels of a labelled transition system is a trace of a state a if

$$a \xrightarrow{f_1} a_1 \cdots a_n \xrightarrow{f_n} a_{n+1}$$

for some states $a_1, \ldots, a_{n+1}$. We shall also use the notation $a \xrightarrow{f_1 \cdots f_n} a_{n+1}$ as abbreviation for such a sequence of transitions. The trace preorder $\lesssim_{\mathrm{tr}}$ is then defined as follows: $a \lesssim_{\mathrm{tr}} b$ if all traces of a are also traces of b.

We say that a preorder is a congruence if $a \lesssim_{\mathrm{tr}} b$ implies that $ca \lesssim_{\mathrm{tr}} cb$ for all contexts c. Some authors refer to this property as *precongruence* and reserve the term congruence for the contextual property of an equivalence relation. We shall prove that $\lesssim_{\mathrm{tr}}$ is a congruence in on both the concrete and the abstract lts. Alternatively to the way we dealt with bisimilarity, we first prove the congruence theorem for the abstract lts (Theorem 2.3.8) – the result for the concrete lts follows as Corollary 2.3.11.

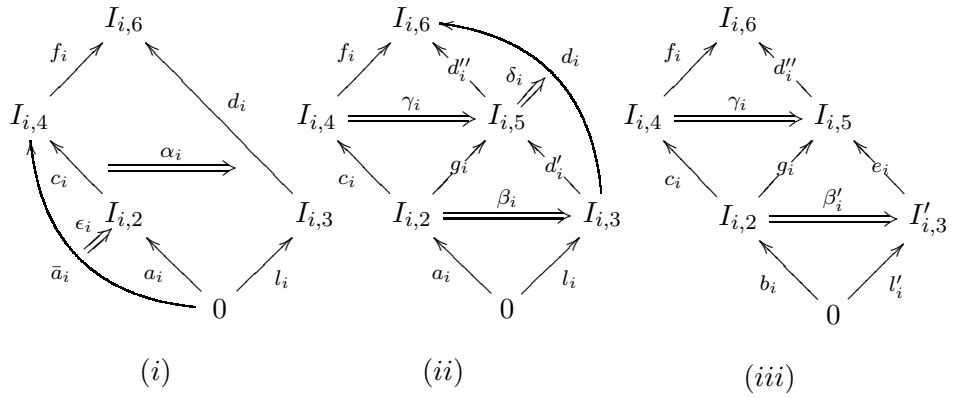
Theorem 2.3.8 (Trace congruence). $\lesssim_{\mathrm{tr}}$ *is a congruence on* ATS($\mathbb{C}$).

Proof. Assume $[a] \lesssim_{\mathrm{tr}} [b]$. We shall prove that $[ca] \lesssim_{\mathrm{tr}} [cb]$ for all contexts $[c] \in \mathbf{C}$. Suppose that $[ca]$ has a trace $[f_1] \cdots [f_n]$. Letting $\bar{a}_1 = ca$, we have

$$[\bar{a}_1] \xrightarrow{[f_1]} [\bar{a}_2] \cdots [\bar{a}_n] \xrightarrow{[f_n]} [\bar{a}_{n+1}].$$

for some $\bar{a}_2, \ldots, \bar{a}_{n+1}$.

We shall first show that, for $i = 1, \ldots, n$, there exists a sequence of reaction rules $\langle l_i, r_i \rangle \in \mathcal{R}$, diagrams (i) and (ii)



(i) (ii) (iii)

where $a_1 = a$, $c_1 = c$, $\bar{a}_i \cong c_i a_i$, $c_{i+1} = d''_i$, $a_{i+1} = d'_i r_i$ and each square of diagrams (i) and (ii) is a GIPO. The ith induction step proceeds as follows. Since $[\bar{a}_i] \xrightarrow{[f_i]} [\bar{a}_{i+1}]$, using Lemma 2.2.17 implies that there exists $\langle l_i, r_i \rangle \in \mathcal{R}$, $\alpha'_i \colon f_i \bar{a}_i \Rightarrow d_i l_i$, for some $\langle l_i, r_i \rangle \in \mathcal{R}$ and $d_i \in \mathbf{D}$, with $\bar{a}_{i+1} \cong d_i r_i$ and the resulting redex square a GIPO. Since $\bar{a}_i \cong c_i a_i$, we can write $\alpha'_i = \alpha_i \bullet f_i \epsilon_i$ for some $\alpha_i : f_i c_i a_i \Rightarrow d_i l_i$ and $\epsilon_i : \bar{a}_i \Rightarrow c_i a_i$ (see diagram (i)). Moreover, using the first part of Lemma 2.2.15, the interior redex square is a GIPO.

Since $\mathbb{C}$ has redex GRPOs (Definition 2.3.1), using the procedure described in the proof of Theorem 2.3.6 this can be split in two GIPOs: $\beta_i \colon g_i a_i \Rightarrow d'_i l_i$ and $\gamma_i \colon f_i c_i \Rightarrow d''_i g_i$ with $\delta_i : d''_i d'_i \Rightarrow d_i$, as illustrated in diagram (ii). Now $\bar{a}_{i+1} \cong d_i r_i \cong d''_i d'_i r_i = c_{i+1} a_{i+1}$ and so the induction hypothesis is maintained.

We obtain a trace

$$[a] = [a_1] \xrightarrow{[g_1]} [a_2] \cdots [a_n] \xrightarrow{[g_n]} [a_{n+1}]$$

and, by assumption, $a \lesssim_{\mathrm{tr}} b$ must be matched by a corresponding trace of b. This means that, for $i = 1, \ldots, n$, there exist GIPOs $\beta'_i \colon g_i b_i \Rightarrow e_i l'_i$, for some $\langle l'_i, r'_i \rangle \in \mathcal{R}$ and $e_i \in \mathbf{D}$, once we take $b_{i+1} \cong e_i r'_i$. We can then paste each of such GIPOs together with the corresponding $\gamma_i \colon f_i c_i \Rightarrow d''_i g_i$ obtained before, as illustrated in diagram (iii). Now using the first part of Lemma 2.2.20, we conclude that there exist GIPOs $f_i c_i b_i \Rightarrow d''_i e_i l'_i$, which means that $[c_i b_i] \xrightarrow{[f_i]} [d''_i e_i r'_i]$. As $[cb] = [c_1 b_1]$, in order to construct a trace $[cb] = [\bar{b}_1] \xrightarrow{[f_1]} \cdots \xrightarrow{[f_n]} [\bar{b}_{n+1}]$ and complete the proof, we only need to

verify that for $i = 1, \ldots, n$, we have that $d_i'' e_i r_i' \cong c_{i+1} b_{i+1}$. This follows at once, as $c_{i+1} \cong d_i''$ and $b_{i+1} \cong e_i r_i'$. □

Before we show that $\lesssim_{\mathrm{tr}}$ is also a congruence on CTS($\mathbb{C}$), it shall be convenient to extend the conclusion of Lemma 2.2.18 to traces.

Lemma 2.3.9. If $[a] \xrightarrow{[f_1]\cdots[f_n]} [a']$ in ATS($\mathbb{C}$) then there exists a'' such that $a' \cong a''$ and $a \xrightarrow{f_1 \cdots f_n} a''$ in CTS($\mathbb{C}$).

Proof. Easy induction on the length of trace. For $n = 1$, the result follows after an application of Lemma 2.2.18. Assuming that the result holds for all $1 \leq k < n$, we have $[a] \xrightarrow{[f_1]\cdots[f_{n-1}]} [a_n] \xrightarrow{[f_n]} [a_{n+1}]$ and using the induction hypothesis, $a \xrightarrow{f_1 \cdots f_{n-1}} a_n'$ such that $a_n \cong a_n'$. Then $[a_n] = [a_n']$ and so $[a_n'] \xrightarrow{[f_n]} [a_{n+1}]$; using Lemma 2.2.18 yields $a_n' \xrightarrow{f_n} a_{n+1}'$ such that $a_{n+1}' \cong a_{n+1}$. □

We can now relate the abstract and the concrete trace preorders.

Lemma 2.3.10.

$$a \lesssim_{\mathrm{tr}} b \quad \text{iff} \quad [a] \lesssim_{\mathrm{tr}} [b]$$

Proof. If $a \lesssim_{\mathrm{tr}} b$ and $[a]$ has a trace $[f_1] \cdots [f_n]$ then using Lemma 2.3.9 allows us to conclude that a has a trace $f_1 \cdots f_n$. By assumption, b has a trace $f_1 \cdots f_n$ and by the definition of the abstract transition system, $[b]$ has a trace $[f_1] \cdots [f_n]$ and thus $[a] \lesssim_{\mathrm{tr}} [b]$.

If $[a] \lesssim_{\mathrm{tr}} [b]$ and a has a trace $f_1 \cdots f_n$ then $[a]$ has a trace $[f_1] \cdots [f_n]$, by assumption $[b]$ has a trace $[f_1] \cdots [f_n]$ and using Lemma 2.3.9, b has a trace $f_1 \cdots f_n$; we conclude that $a \lesssim_{\mathrm{tr}} b$. □

Corollary 2.3.11. *Trace preorder $\lesssim_{\mathrm{tr}}$ is a congruence on* CTS($\mathbb{C}$).

Proof. Suppose that $a \lesssim_{\mathrm{tr}} b$, then using Lemma 2.3.10 yields $[a] \lesssim_{\mathrm{tr}} [b]$ and by Theorem 2.3.8, $[ca] \lesssim_{\mathrm{tr}} [cb]$ for any c. Suppose that ca has a trace $f_1 \cdots f_n$. Then $[ca]$ has a trace $[f_1] \cdots [f_n]$. Using the fact that $[ca] \lesssim_{\mathrm{tr}} [cb]$, $[cb]$ has a trace $[f_1] \cdots [f_n]$ and using Lemma 2.3.9, cb has a trace $f_1 \cdots f_n$. Since the trace $f_1 \cdots f_n$ was arbitrary, this means that $ca \lesssim_{\mathrm{tr}} cb$. □

2.3.3 Failures preorder

Failure semantics [42] enhances trace semantics with limited branch-inspecting power. More precisely, failure sets allow to test when processes do not have

the capability of engaging in certain actions after performing any particular trace.

Formally, for a a state of a labelled transition system over a reactive system, a *failure* of a is a pair $\langle f_1 \cdots f_n, X \rangle$, where $f_1 \cdots f_n$ and X are respectively a nonempty sequence and a set of labels, such that:

- $f_1 \cdots f_n$ is a trace of a, $a \xrightarrow{f_1} \cdots \xrightarrow{f_n} a_{n+1}$;
- a_{n+1}, the final state of the trace, is *stable*, i.e. $a_{n+1} \not\longrightarrow\!\!\triangleright$;
- a_{n+1} *refuses* X, i.e. $a_{n+1} \not\xrightarrow{x}$ for all $x \in X$.

The failure preorder $\lesssim_{\mathrm{f}}$ is defined as $a \lesssim_{\mathrm{f}} b$ if all failures of a are also failures of b.

As we did with the trace preorder, we shall prove that the failures preorder is a congruence on the abstract labelled transition system $\mathrm{ATS}(\mathbb{C})$ and use it derive that it is a congruence on the concrete labelled transition system $\mathrm{CTS}(\mathbb{C})$ as a corollary.

If X is some set of contexts of a G-reactive system $\mathbb{C}$, let $[X] = \{\, [x] \mid x \in X \,\}$ be the corresponding set of isomorphism classes of contexts in the underlying reactive system $\mathbb{C}_0$ (see Definition 2.2.7).

Theorem 2.3.12 (Failures congruence). *$\lesssim_{\mathrm{f}}$ is a congruence.*

Proof. Assume $[a] \lesssim_{\mathrm{f}} [b]$; we shall prove that $[ca] \lesssim_{\mathrm{f}} [cb]$ for all contexts $c \in \mathbf{C}$. The following proof extends the proof of Theorem 2.3.8.

Let $\langle [f_1] \cdots [f_n], [X] \rangle$, $n > 0$, be a failure of $[ca]$. Thus starting with such a failure

$$[ca] \xrightarrow{[f_1]} [\bar{a}_2] \cdots [\bar{a}_n] \xrightarrow{[f_n]} [\bar{a}_{n+1}]$$

we proceed exactly as in the proof of Theorem 2.3.8 to obtain a trace $[a] = [a_1] \xrightarrow{[g_1]} [a_2] \cdots [a_n] \xrightarrow{[g_n]} [a_{n+1}]$.

First, we claim that $[a_{n+1}]$ is stable. In fact, were it not, it would follow from $c_{n+1} \in \mathbf{D}$ (which equals d'_n) that also $[\bar{a}_{n+1}] = [c_{n+1} a_{n+1}] \longrightarrow\!\!\triangleright$. But this is impossible, since $[\bar{a}_{n+1}]$ is stable. Secondly, $[a_{n+1}]$ refuses both

$$Y = \{g \mid \text{there exists a GIPO } \delta_g \colon x c_{n+1} \Rightarrow dg, \text{ for } x \in X,\, d \in \mathbf{D}\} \text{ and}$$
$$Z = \{g \mid \text{there exists a 2-cell } \epsilon_g \colon c_{n+1} \Rightarrow dg, \text{ for } d \in \mathbf{D}\},$$

which can be seen as follows. If $[a_{n+1}] \xrightarrow{[g]}$ for $[g] \in [Y]$, then there exists a GIPO $\alpha \colon g a_{n+1} \Rightarrow d' l$, for some rule $\langle l, r \rangle$, which could be pasted together with δ_g to yield, using the first part of Lemma 2.2.20, a GIPO $x c_{n+1} a_{n+1} \Rightarrow$

$dd'l$, which is impossible since it means that $[\bar{a}_{n+1}] \xrightarrow{[x]}$, for $[x] \in [X]$. Similarly, if $[a_{n+1}] \xrightarrow{[g]}$ for $[g] \in [Z]$, pasting the corresponding GIPO with ϵ_g, we see that $\bar{a}_{n+1} \longrightarrow\!\!\triangleright$, contradicting the hypothesis that $\bar{a}_{n+1}$ is stable.

If follows from $a \lesssim_{\mathrm{f}} b$ that there exists a trace $[b_1] \xrightarrow{g_1} [b_2] \cdots [b_n] \xrightarrow{g_n} [b_{n+1}]$ and that b_{n+1} is stable and refuses $[Y \cup Z]$. It is then easy to complete the proof by transferring stability and $[X]$-refusal to $[\bar{b}_{n+1}]$. First, suppose that $[\bar{b}_{n+1}] \longrightarrow\!\!\triangleright$. This means that there exists a 2-cell $dl \Rightarrow \bar{b}_{n+1}$. Since $\mathbb{C}$ has redex-GRPOs, we can factor c_{n+1} out and obtain from this a GRPOs $\alpha\colon gb_{n+1} \Rightarrow d'l$ together with a 2-cell $d''g \Rightarrow c_{n+1}$. But this would mean that $[b_{n+1}] \xrightarrow{[g]}$, for $[g] \in [Z]$, which is a contradiction.

Suppose finally that $[\bar{b}_{n+1}] \xrightarrow{[x]}$, for $[x] \in [X]$. Again, by definition of the transition relation, and exploiting the existence of redex-GRPOs, we find GRPOs $[xc_{n+1}] \Rightarrow [d''g]$ and $gb_{n+1} \Rightarrow d'l$, which mean that $[b_{n+1}] \xrightarrow{[g]}$, for $[g] \in [Y]$. □

One should notice that our restriction to failures $\langle [f_1] \ldots [f_n],\, X\rangle$ where $[f_1] \ldots [f_n]$ is a *nonempty* trace is crucial for the proof to work. Indeed, the proof relies on the fact that the final state, after performing the trace, is of the form $[c_{n+1}a_{n+1}]$, where c_{n+1} is a reactive context. An empty trace results in $[ca]$ where c is arbitrary.

There is, again, a very close relation between the concrete and the abstract versions of the failure preorder. We investigate this relationship by proving Lemmas 2.3.13 and 2.3.14.

Lemma 2.3.13. $\langle [f_1] \cdots [f_n],\, [X]\rangle$ is a failure of $[a]$ iff $\langle f_1 \cdots f_n,\, X\rangle$ is a failure of a.

Proof. Suppose that $\langle [f_1] \cdots [f_n],\, [X]\rangle$ is a failure of $[a]$. Let $[a] \xrightarrow{[f_1]\cdots[f_n]} [a']$ be a witness for the failure. Using Lemma 2.3.9, we get a $a \xrightarrow{f_1 \cdots f_n} a''$ such that $a' \cong a''$. It is easy to show that a'' is stable and rejects any x' such that $x' \cong x \in X$. Similarly, if $\langle f_1 \cdots f_n,\, X\rangle$ is a failure of a, with witness $a \xrightarrow{f_1 \cdots f_n} a'$ then $[a] \xrightarrow{[f_1]\cdots[f_n]} [a']$ is a trace of $[a]$. It is easy to see that it is stable and rejects any $[x] \in [X]$. □

Lemma 2.3.14.

$$a \lesssim_{\mathrm{f}} b \quad \text{iff} \quad [a] \lesssim_{\mathrm{f}} [b]$$

Proof. Suppose that $a \lesssim_{\mathrm{f}} b$ and that $\langle [f_1] \cdots [f_n],\, [X]\rangle$ is a failure of $[a]$. Using Lemma 2.3.13, $\langle f_1 \cdots f_n,\, X\rangle$ is a failure of a, and by assumption, a

failure of b; using Lemma 2.3.13 again, $\langle [f_1] \cdots [f_n],\, [X] \rangle$ is a failure of $[b]$. The other direction is similar. □

Corollary 2.3.15. Failures preorder $\lesssim_{\mathrm{f}}$ on CTS($\mathbb{C}$) is a congruence.

Proof. Suppose that $a \lesssim_{\mathrm{f}} b$, then $[a] \lesssim_{\mathrm{f}} [b]$ by Lemma 2.3.14. Suppose that $\langle f_1 \cdots f_n,\, X \rangle$ is a failure of ca, for some c. Then, using Lemma 2.3.13 allows us to derive $\langle [f_1] \cdots [f_n],\, [X] \rangle$ as a failure of $[ca]$. Using Theorem 2.3.12, $\langle [f_1] \cdots [f_n],\, [X] \rangle$ is a failure of $[cb]$ and, again using Lemma 2.3.13, $\langle f_1 \cdots f_n,\, X \rangle$ is a failure of cb. □

Chapter 3

Bicolimits

The results of this chapter offer a more complete picture of GRPOs, from a mathematical point of view. In particular, we shall recall some of the basic concepts of 2-dimensional category theory and show that GRPOs are an instance of a more general theory of bicolimits. We shall also provide detailed proofs of several of the basic properties of GRPOs which we have relied on in Chapter 2.

In section 3.1 we shall recall some basic theory behind (co)limits in 2-categories and bicategories. We shall be especially interested in the notion of *bicolimit*. Indeed, we shall show that as RPOs are pushouts in a slice category so GRPOs are *bipushouts* (Definition 3.1.12) in a *pseudo-slice* category (Definition 3.1.1). The fact that GRPOs are certain bicolimits is helpful, for example we are able to conclude "for free" that GRPOs are defined up to an equivalence (see Definition 3.1.5). Also, the fact that GRPOs are an instance of a natural categorical construction helps to justify their definition, as stated in Chapter 2.

In section 3.2 we shall prove, in detail, the key lemmas stated in Chapter 2 which were crucial for the arguments used in the congruence theorems of section 2.3.

3.1 Bicolimits

Bicolimits were introduced by [100] in the context of *bicategories* [7], which consist of the same basic data as 2-categories where associativity and identity laws for 1-cells hold *only up to* coherent isomorphisms. Any 2-category may be thought of as a special kind of bicategory (where the coherent isomorphisms are all identities), and indeed the notion of bicolimit applies to

2-categories as well. Bicolimits are also briefly discussed by [53] along with other notions of 2-categorical (co)limits.

We shall begin this section with §3.1.1 where we recall some basic 2-categorical concepts such as 2-functors, pseudo-natural transformations and modifications, necessary for a satisfactory development of notions of colimits in 2-categories. Secondly, in §3.1.2 we shall present a general definition of bicolimits, and derive an elementary presentation. Finally, in §3.1.3 we shall present a particular type of bicolimit, the bipushout, and show that to give a GRPO is to give a bipushout in a pseudo-slice category.

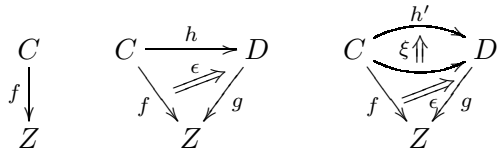
While all the category theory of this section is quite standard and well-understood, the explicit derivation of the notion of GRPO from a general notion of a standard 2-categorical limit is important in order to facilitate a deeper understanding of the concept. For instance, the fact that GRPOs are a certain type of bicolimit automatically implies that GRPOs are defined up to equivalence (Definition 3.1.5).

3.1.1 2-categorical preliminaries

In this section we shall recall some standard notions from the theory of 2-categories. For a more complete overview the reader is referred to [54]. In particular, we shall only develop the underlying theory needed in order to understand the notion of GRPO (see Remark 3.1.6).

The following definition recalls the 2-categorical notion of slice category which shall be appropriate for our needs.

Definition 3.1.1 (Pseudo-slice category). Given a 2-category $\mathbf{C}$ and an



(*i*) (*ii*) (*iii*)

object Z, a *pseudo-slice* category $\mathbf{C}/Z$ is a 2-category with

(i) objects $C \xrightarrow{f} Z$;

(ii) arrows $\langle C, f\rangle \xrightarrow{\langle h, \epsilon\rangle} \langle D, g\rangle$ where $h\colon C \to D$ and $\epsilon\colon f \Rightarrow gh$ is an isomorphism; and

(iii) 2-cells $\xi\colon \langle h, \epsilon\rangle \Rightarrow \langle h', \epsilon'\rangle$ being 2-cells $\xi\colon h \Rightarrow h'$ satisfying the obvious compatibility requirement, namely $g\xi \bullet \epsilon = \epsilon'$.

Composition and identities are defined in the obvious way.

There is a natural notion of homomorphism of 2-categories which is a generalisation of the notion of functor.

Definition 3.1.2. A *2-functor* $F : \mathbf{C} \to \mathbf{D}$ maps objects to objects, arrows to arrows and 2-cells to 2-cells in a way which respects all identities and composition. Let $\operatorname{ob}\mathbf{C}$, $\operatorname{arr}\mathbf{C}$ and $\operatorname{cel}\mathbf{C}$ denote, respectively, the class of objects, arrows and 2-cells.

In more detail, a 2-functor F consists of:

- a map $F : \operatorname{ob}\mathbf{C} \to \operatorname{ob}\mathbf{D}$
- a map $F : \operatorname{arr}\mathbf{C} \to \operatorname{arr}\mathbf{D}$ which preserves domains and codomains (i.e. $F(f : X \to Y) = Ff : FX \to FY$), identities (i.e. $F(\mathrm{id}_X) = \mathrm{id}_{FX}$) and composition (i.e. $F(gf) = FgFf$)[1];
- a map $F : \operatorname{cel}\mathbf{C} \to \operatorname{cel}\mathbf{D}$ which preserves domains and codomains (i.e. $F(\alpha : f \Rightarrow g) = F\alpha : Ff \Rightarrow Fg$), identities (i.e. $F(1_f) = 1_{Ff}$), vertical composition (i.e $F(\beta\alpha) = F\beta F\alpha$)[2], and horizontal composition (i.e. $F(\beta \circ \alpha) = F\beta \circ F\alpha$) ;

The notion of natural transformation between 2-functors which shall be useful for us is that of *pseudo*-natural transformation.

Definition 3.1.3 (Pseudo-natural transformation). A *pseudo-natural transformation* α between 2-functors $F, G : \mathbf{C} \to \mathbf{D}$ consists of an arrow $\alpha_C : FC \to GC$ in $\mathbf{D}$ for every object $C \in \mathbf{C}$, and for every arrow $f : C \to C'$ in $\mathbf{C}$ an invertible 2-cell $\alpha_f : (Gf)\alpha_C \Rightarrow \alpha_{C'}(Ff)$ in $\mathbf{D}$, as illustrated below:

$$\begin{array}{ccc} FC & \xrightarrow{\alpha_C} & GC \\ {\scriptstyle Ff}\downarrow & \swarrow{\scriptstyle \alpha_f} & \downarrow{\scriptstyle Gf} \\ FC' & \xrightarrow[\alpha_{C'}]{} & GC'. \end{array}$$

Additionally, we require the following compatibility conditions with the structure of $\mathbf{C}$:

[1] That is, F is a functor on the category obtained by forgetting the 2-cells.
[2] That is, F defines a functor $\mathbf{C}(X, Y) \to \mathbf{D}(FX, FY)$.

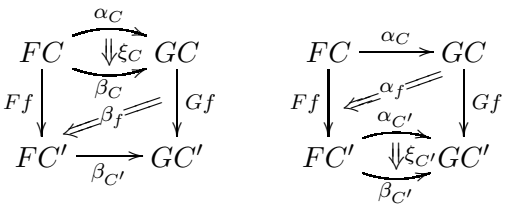
(i) for any object $C \in \mathbf{C}$, $\alpha_{\mathrm{id}_C} = 1_{\alpha_C}$;

(ii) for any arrows $f : C \to C'$, $g : C' \to C''$ of $\mathbf{C}$, α_{gf} is the pasting composite of α_f and α_g (i.e. $\alpha_{gf} = \alpha_g(Ff) \bullet (Gg)\alpha_f$);

(iii) for any 2-cell $\beta : f \Rightarrow g$ in $\mathbf{C}$, $\alpha_{C'}(F\beta) \bullet \alpha_f = \alpha_g \bullet (G\beta)\alpha_C$.

The 2-dimensional structure allows us to go one level further and define mappings between pseudo-natural transformations, these are usually referred to as *modifications*.

Definition 3.1.4. A *modification* ξ between two pseudo-natural transformations α and β between 2-functors F and G is a family of $\xi_C : \alpha_C \Rightarrow \beta_C$ of 2-cells in $\mathbf{D}$ which is suitably compatible with the structures of α and β, that is for any $f : C \to C'$ in $\mathbf{C}$, we require that $\beta_f \bullet (Gf)\xi_C = \xi_{C'}(Ff) \bullet \alpha_f$. We illustrate the 2-cells involved in the two sides of this last equation below.



It is easy to verify that 2-functors from $\mathbf{C}$ to $\mathbf{D}$, pseudo-natural transformations and modifications form a 2-category[3]. We shall denote such a 2-category by $\mathbf{Psd}[\mathbf{C}, \mathbf{D}]$.

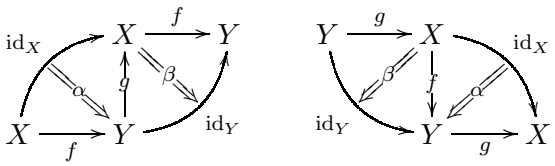
The terminal 2-category $\mathbf{1}$ has one object, one identity arrow and one identity 2-cell. For any 2-category $\mathbf{C}$, we shall denote by $!$ the unique 2-functor to $\mathbf{1}$.

The following definition is of importance as it shall form for us an appropriate notion of when two objects of our 2-dimensional categories may be considered to be the same. It shall turn out that as RPOs are defined *up to isomorphism* so GRPOs are defined *up to equivalence*.

Definition 3.1.5 (Equivalence). Two objects X, Y of a 2-category $\mathbf{C}$ are *equivalent* when there are arrows $f : X \to Y$, $g : Y \to X$ and 2-cells $\alpha : \mathrm{id}_X \Rightarrow gf$, $\beta : fg \Rightarrow \mathrm{id}_Y$. We refer to f and g as equivalences. Given any equivalence f in any 2-category, one can always find 2-cells α and β so that

[3]In fact, 2-categories, 2-functors, pseudo-natural transformations and modifications actually form a *3-category*, but we leave it to the reader to define this concept.

they form the unit and the counit of an *adjunction*. Such an equivalence is usually called an *adjoint equivalence*. The additional equations that α and



β are required to satisfy are sometimes referred to as *triangle* equations; they specify that the 2-cells obtained by pasting the components of the diagrams above are equal to 1_f and 1_g, respectively.

Remark 3.1.6. We have concentrated on *pseudo* (up-to-isomorphism) variants of the definitions of natural transformations and slice categories as these turn out to fit our needs precisely. There are also alternative definitions which give different structures. Roughly, instead of requiring isomorphic 2-cells one can get away with less by taking arbitrary 2-cells – resulting in *lax*-natural transformations and *lax*-slice categories. Alternatively, one can be more strict and require identity 2-cells instead of isomorphisms – resulting in *2*-natural transformations and *2*-slice categories.

3.1.2 Bicolimits

We shall now recall the notion of bicolimit. Bicolimits were first introduced by Street [100]. They are also discussed in Kelly's overview paper [53], amongst other notions of limits an colimits in the setting of 2-categories, We remark that our exposition here is simplified, as the more general notion of *indexed-bicolimits*, which comes from enriched category theory [52], is not needed for our purposes. Such simple colimits are sometimes referred to as *conical* in categorical literature.

Definition 3.1.7 (Bicolimit). Let $G : \mathbf{J} \to \mathbf{C}$ be a 2-functor. A *bicolimit* object of G is an object $\mathrm{Bic}\, G$ of $\mathbf{C}$ which satisfies

$$\mathbf{C}(\mathrm{Bic}\, G,\, A) \simeq \mathbf{Psd}[\mathbf{J}^{\mathrm{op}},\, \mathbf{Cat}](\mathbf{!},\, \mathbf{C}(G-,\, A)) \tag{3.1}$$

where $\simeq$ denotes an equivalence of categories. This equivalence is required to be natural in $A \in \mathbf{C}$.[4] We shall usually refer to the bicolimit of G as the

[4]This means that the functor $H = \lambda A.\, \mathbf{Psd}[\mathbf{J}^{\mathrm{op}}, \mathbf{Cat}](\mathbf{!},\, \mathbf{C}(G-,\, A)) : \mathbf{C} \to \mathbf{Cat}$ admits a *birepresentation*, that is an object $\mathrm{Bic}\, G \in \mathbf{C}$ such that $\lambda A.\mathbf{C}(\mathrm{Bic}\, G,\, A)$ and H are equivalent as objects of $[\mathbf{C},\, \mathbf{Cat}]$, the 2-category of functors $\mathbf{C} \to \mathbf{Cat}$, 2-natural transformations and modifications.

pair $\langle \mathrm{Bic}\, G, \eta \rangle$ where η is the unit of (the image of the object $\mathrm{id}_{\mathrm{Bic}\, G}$ under the equivalence) (3.1).

Remark 3.1.8. Varying (3.1) allows one to capture other notions of 2-categorical limits and colimits. In particular, one may replace the equivalence of categories with an isomorphism of categories and vary the kinds of natural transformations in the right hand side, i.e. instead of pseudo-natural transformations one may consider 2-natural transformations or lax-natural transformations, see [53] for details.

Bicolimits are defined *up to equivalence* (Definition 3.1.5). Certainly, an equivalence $f : C \to D$ induces an equivalence of categories $\mathbf{C}(D,\, A) \simeq \mathbf{C}(C,\, A)$ natural in A. This, together with the fact that equivalences of categories compose, proves that C is a bicolimit of G if and only if D is. It also holds that if C and D are two bicolimits of G, they must be equivalent as objects of $\mathbf{C}$.

We can spell out Definition 3.1.7 in elementary terms. We shall begin by examining the right hand side of (3.1) in detail. Let $G : \mathbf{J} \to \mathbf{C}$ be a 2-functor. To give an arbitrary object of the (ordinary) category $\mathbf{Psd}[\mathbf{J}^{\mathrm{op}}, \mathbf{Cat}](\mathbf{!},\, \mathbf{C}(G-,\, A))$ of pseudo-natural transformations and modifications is to give a pseudo-natural transformation $\tau : \mathbf{!} \Rightarrow \mathbf{C}(G-,\, A) : \mathbf{J} \to \mathbf{Cat}$.

That is, it is to give a collection of (ordinary) functors $\tau_j : \mathbf{1} \to \mathbf{C}(Gj,\, A)$ parametrised over objects $j \in \mathbf{J}$ and (ordinary) natural isomorphisms $\tau_u : \mathbf{C}(u,\, A)\tau_j \Rightarrow \tau_i$, parametrised over morphisms $u : i \to j$ in $\mathbf{J}$. We illustrate the components of such a collection in the diagram below.

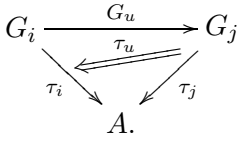
$$\begin{array}{ccc} \mathbf{1} & \xrightarrow{\tau_j} & \mathbf{C}(G_j,\, A) \\ {\scriptstyle \mathrm{id}}\downarrow & \swarrow\!\!\swarrow {\scriptstyle \tau_u} & \downarrow{\scriptstyle \mathbf{C}(Gu,\, A)} \\ \mathbf{1} & \xrightarrow[\tau_i]{} & \mathbf{C}(G_i,\, A) \end{array}$$

To choose a functor $\tau_j : \mathbf{1} \to \mathbf{C}(Gj,\, A)$ is to choose an object of $\mathbf{C}(Gj,\, A)$, that is, a morphism $\tau_j : G_j \to A$ in $\mathbf{C}$. Then to give a natural isomorphism $\tau_u : \mathbf{C}(u,\, A)\tau_j \Rightarrow \tau_i$ is to give an invertible 2-cell $\tau_u : \tau_j(G_u) \Rightarrow \tau_i$.

Moreover, the compatibility conditions for pseudo-natural transformations as stipulated in Definition 3.1.3 amount to requiring that $\tau_{\mathrm{id}_i} = 1_{\tau_i}$, $\tau_{vu} = \tau_u \bullet \tau_v(Gu)$ and, for any 2-cell $\mu : u \Rightarrow u' : i \to j$ of $\mathbf{J}$, $\tau_{u'} \bullet (\tau_j G_\mu) = \tau_u$.

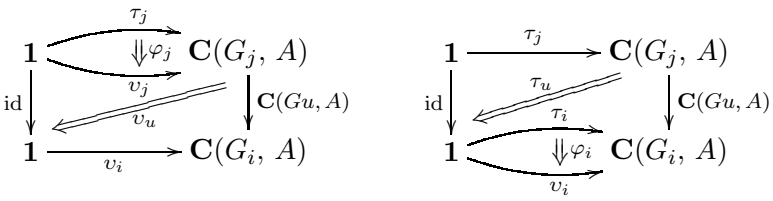
When $\mathbf{J}$ is an ordinary category, we can rephrase the above with the following notion of *pseudo-cocone*.

Definition 3.1.9 (Pseudo-cocone). Suppose that $\mathbf{J}$ is an ordinary category and $G : \mathbf{J} \to \mathbf{C}$ is a 2-functor (which is the same thing as an ordinary functor from $\mathbf{J}$ to the category obtained from $\mathbf{C}$ by forgetting its 2-cells). A *pseudo-cocone* τ from G to $A \in \mathbf{C}$ is a family of arrows $\tau_i : G_i \to A$ for $i \in \mathbf{J}$ and invertible 2-cells $\tau_u : \tau_j G_u \Rightarrow \tau_i$ for $u : i \to j$ in $\mathbf{J}$ as illustrated below:



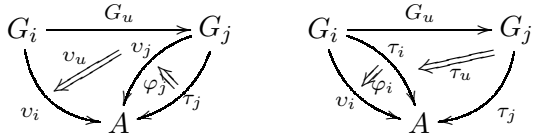
Additionally, $\tau_{\mathrm{id}_i} = 1_{\tau_i}$ and τ_{vu} must be the pasting composite of τ_u and τ_v.

We have verified that objects of $\mathbf{Psd}[\mathbf{J}^{\mathrm{op}}, \mathbf{Cat}](\mathbf{!}, \mathbf{C}(G-, A))$ are pseudo-cocones. The arrows are modifications between pseudo-natural transformations $\tau, \upsilon : \mathbf{!} \to \mathbf{C}(G-, A)$. But as the pseudo-natural transformations consist of functors $\tau_i : \mathbf{1} \to \mathbf{C}(G_i, A)$, so modifications φ are families of natural transformations between such functors. Thus we have a collection of natural transformations $\varphi_i : \tau_i \Rightarrow \upsilon_i : \mathbf{1} \to \mathbf{C}(G_i, A)$ which satisfy $\upsilon_u \bullet (\mathbf{C}(G_u, A)\varphi_j) = \varphi_i \bullet \tau_u$, as illustrated below.



As the τ_i are precisely arrows $G_i \to A$ in $\mathbf{C}$ so $\varphi_j : \tau_i \Rightarrow \tau_j$ are 2-cells in $\mathbf{C}$. In other words, they constitute modifications between pseudo-cocones, which we shall define below.

Definition 3.1.10 (Modifications between pseudo-cocones). Given pseudo-cocones τ and υ from G to $A \in \mathbf{C}$, a modification $\varphi : \tau \Rightarrow \upsilon$ between pseudo cocones is a family of 2-cells $\varphi_i : \tau_i \Rightarrow \upsilon_i$ such that $\upsilon_u \bullet \varphi_j(G_u) = \varphi_i \tau_u$, as illustrated in the diagrams below.



Pseudo cocones from $G : \mathbf{J} \to \mathbf{C}$ to $A \in \mathbf{C}$ and their modifications form a category $\mathbf{PsdCocone}(G, A)$. Using the notions of pseudo-cocone and pseudo-cocone modification, we can give an elementary definition of (conical) bicolimits. Note that we shall specialise to the case when $\mathbf{J}$ is an ordinary category. This is because we shall only be interested in such simple cases; it is nonetheless trivial to extend to the more general situation when $\mathbf{J}$ is a 2-category.

Rephrasing our arguments above, there is an isomorphism of categories and the right hand side of (3.1):

$$\mathbf{PsdCocone}(G, A) \cong \mathbf{Psd}[\mathbf{J}^{\mathrm{op}}, \mathbf{Cat}](!, \mathbf{C}(G-, A)). \tag{3.2}$$

Moreover, for any $f : A \to B$, there is a functor $\mathbf{PsdCocone}(G, f) : \mathbf{PsdCocone}(G, A) \to \mathbf{PsdCocone}(G, B)$, inherited from the right hand side (3.2), which takes $\tau_i : G_i \to A$ to $f\tau_i : G_i \to B$ and $\tau_u : \tau_j G_u \Rightarrow \tau_i$ to $f\tau_u : f\tau_j G_u \Rightarrow f\tau_i$. We shall write $f\tau$ to refer to such a cocone.

Using (3.1) together with (3.2) yields an equivalence of categories

$$\mathbf{C}(\mathrm{Bic}\, G,\ A) \simeq \mathbf{PsdCocone}(G, A)$$

which is natural in A. As with any such equivalence, it is completely determined by where the identity $\mathrm{id}_{\mathrm{Bic}\, G}$ of $\mathbf{C}(\mathrm{Bic}\, G, \mathrm{Bic}\, G)$ is mapped to. Thus we obtain a pseudo cocone η from G to $\mathrm{Bic}\, G$.

We shall use the fact that a functor is an equivalence of categories iff it is fully-faithful and essentially surjective on objects. Note that the "if" part of this characterisation relies on the axiom of choice. Using this characterisation, we obtain the following:

(i) Essential surjectivity: for every pseudo-cocone α from G to A, there exists an arrow $h : \mathrm{Bic}\, G \to A$ and an isomorphic modification $\varphi : h\eta \Rightarrow \alpha$;

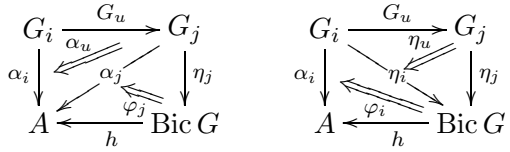
(ii) Fullness: given arrows $h, h' : \mathrm{Bic}\, G \to A$ and a modification $\varphi : h\eta \Rightarrow h'\eta$ between cocones, there exists a 2-cell $\xi : h \to h'$ such that $\varphi = \xi\eta$;

(iii) Faithfulness: different 2-cells $\xi, \xi' : h \Rightarrow h'$ $(\xi \neq \xi')$ give different modifications $\xi\eta \neq \xi'\eta$ between cocones.

The first property allows us to derive the *existence* of a mediating morphism, while the second and the third ensure that such a mediating morphism is *essentially unique*. We are now ready to give a completely elementary account of a conical bicolimit.

Definition 3.1.11 (Bicolimit). Suppose that $\mathbf{J}$ is an ordinary category and that $G : \mathbf{J} \to \mathbf{C}$ is a functor. A bicolimit is a tuple $\langle \operatorname{Bic} G, \eta \rangle$ where $\operatorname{Bic} G \in \mathbf{C}$ and η is a pseudo-cocone from G to $\operatorname{Bic} G$. Two properties are required to ensure equivalence (3.1):

(i) (Existence of mediating morphism). For any pseudo-cocone α from G to A there exists an arrow $h : \operatorname{Bic} G \to A$ and a family of 2-cells $\varphi_i : h\eta_i \Rightarrow \alpha_i$ which make the pseudo-cocones $h\eta$ and α compatible, that is $\alpha_u \bullet \varphi_j G_u = \varphi_i \bullet h\eta_u$, as illustrated below.



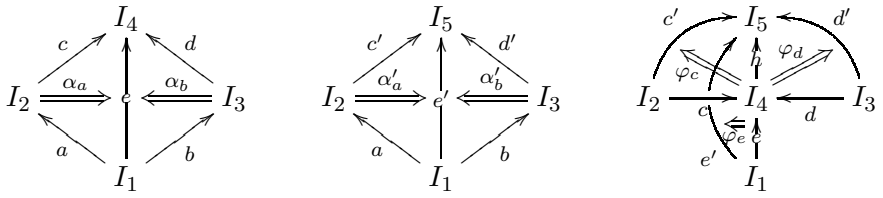
(ii) (Essential uniqueness). Given another arrow $h' : \operatorname{Bic} G \to A$ and a family of 2-cells $\psi_i : h\eta_i \Rightarrow h'\eta_i$ which makes the two pseudo-cocones compatible (i.e.. $\alpha_u \bullet \psi_j G_u = \varphi_i \bullet h'\eta_j$), there exists a unique 2-cell $\xi : h \Rightarrow h'$ such that $\psi_i \bullet \xi\eta_i = \varphi_i$.

3.1.3 Bipushouts and GRPOs

We shall now experiment with a particular choice of $\mathbf{J}$ to examine the 'bi'-version of pushouts, as this will be useful in the definition of relative bipushouts, which specialise to GRPOs in 2-categories where all 2-cells are invertible.

Definition 3.1.12 (Bipushout). Suppose that $\mathbf{J} = \cdot \leftarrow \cdot \rightarrow \cdot$. Then to give a 2-functor $G : \mathbf{J} \to \mathbf{C}$ is to give a span $I_2 \xleftarrow{a} I_1 \xrightarrow{b} I_3$ in $\mathbf{C}$.

Using Definition 3.1.11, a bicolimit of such a cospan, which we shall refer to as *bipushout*, consists of the following data: an object I_4, arrows $c : I_2 \to I_4$, $e : I_1 \to I_4$ and $d : I_3 \to I_4$ and isomorphic 2-cells $\alpha_a : ca \Rightarrow e$ and $\alpha_b : db \Rightarrow e$, as illustrated in diagram (i).



(i) (ii) (iii)

The universal property can be stated as follows:

(i) for any other object I_5, arrows $c' : I_2 \to I_5$, $e' : I_1 \to I_5$ and $d' : I_3 \to I_5$, and invertible 2-cells $\alpha'_a : c'a \Rightarrow e'$ and $\alpha'_b : d'b \Rightarrow e'$, there exists a mediating morphism consisting of an arrow $h : I_4 \to I_5$ and invertible 2-cells $\varphi_c : hc \Rightarrow c'$, $\varphi_e : he \Rightarrow e'$ and $\varphi_d : hd \Rightarrow d'$ so that $\alpha'_a \bullet \varphi_c a = \varphi_e \bullet h\alpha_a$ and $\alpha'_b \bullet \varphi_d b = \varphi_e \bullet h\alpha_b$;

(ii) such a mediating morphism is essentially unique: given an arrow $h' : I_4 \to I_5$ and 2-cells $\eta : hc \Rightarrow h'c$, $\theta : he \Rightarrow h'e$ and $\mu : hd \Rightarrow h'd$ such that $\theta \bullet h\alpha_a = h'\alpha_a \bullet \eta a$ and $\theta \bullet h\alpha_b = h'\alpha_b \bullet \mu b$, there exists a unique 2-cell $\xi : h \Rightarrow h'$ such that $\eta = \xi c$, $\theta = \xi e$ and $\mu = \xi d$.

Remark 3.1.13. The 2-cells η, θ and μ may seem slightly mysterious. Here we give a brief explanation.

Clearly, given another mediating morphism $\langle h', \varphi'_c, \varphi'_e, \varphi'_d \rangle$ which satisfies analogous equations to the ones specified in part (i) of Definition 3.1.12, one can define $\eta = \varphi'^{-1}_c \bullet \varphi_c$, $\theta = \varphi'^{-1}_e \bullet \varphi_e$ and $\mu = \varphi'^{-1}_d \bullet \varphi_d$ which then satisfy the required equations as specified in part (ii) of Definition 3.1.12. The existence of a unique $\xi : h \Rightarrow h'$ then implies that $\varphi'_c \bullet \xi c = \varphi_c$, $\varphi'_e \bullet \xi e = \varphi_e$ and $\varphi'_d \bullet \xi d = \varphi_d$. Moreover, ξ is in this case an isomorphism.

Conversely, starting with 2-cells η, θ and μ which are *isomorphisms*, one may recover the isomorphisms $\varphi'_c = \varphi_c \bullet \eta^{-1}$, $\varphi'_e = \varphi_e \bullet \theta^{-1}$ and $\varphi'_d = \varphi_d \bullet \mu^{-1}$. However, η, θ and μ do not have to be, in general, isomorphism.

Definition 3.1.12 actually contains redundant information as demonstrated by the following lemma.

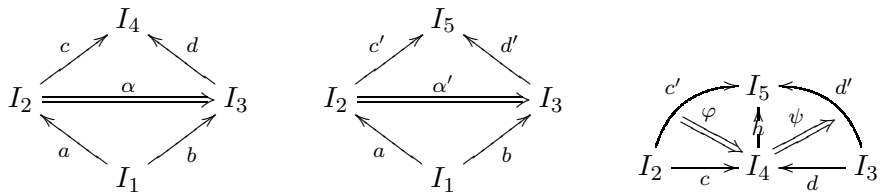
Lemma 3.1.14. Suppose that $\langle I_4, c, d, e, \alpha_a, \alpha_b \rangle$ is a bipushout of $I_2 \xleftarrow{a} I_1 \xrightarrow{b} I_3$. Then so is $\langle I_4, c, d, e', \alpha'_a, \alpha'_b \rangle$ for any $e' : I_1 \to I_4$, α'_a and α'_b such that $\alpha'^{-1}_b \bullet \alpha'_a = \alpha^{-1}_b \bullet \alpha_a$.

Proof. It suffices to note that $\langle \mathrm{id} : I_4 \to I_4, 1_c, \alpha'_b \bullet \alpha^{-1}_b, 1_d \rangle$ is a mediating morphism and id is clearly an equivalence. □

Throughout this thesis we are interested in 2-categories where all 2-cells are isomorphisms, of G-categories (Definition 2.2.3). The insights provided by Remark 3.1.13 and Lemma 3.1.14 allow us to simplify the definition of bipushout in G-categories as follows.

Definition 3.1.15 (Bipushout). A *bipushout* of arrows $I_2 \xleftarrow{a} I_1 \xrightarrow{b} I_3$ in a G-category $\mathbf{C}$ is a quadruple $\langle I_4, c, d, \alpha \rangle$ where $c\colon I_2 \to I_4$, $d\colon I_3 \to I_4$

and $\alpha\colon ca \Rightarrow db$ is an isomorphism, as illustrated in diagram (i). Moreover, for any other such quadruple $\langle I_5, c', d', \alpha' \rangle$, as illustrated diagram (ii), we have the following:



(i) (ii) (iii)

(i) there exists an arrow $h : I_4 \to I_5$ and isomorphisms $\varphi : c' \Rightarrow hc$, $\psi : hd \Rightarrow d'$ satisfying the obvious compatibility condition, namely

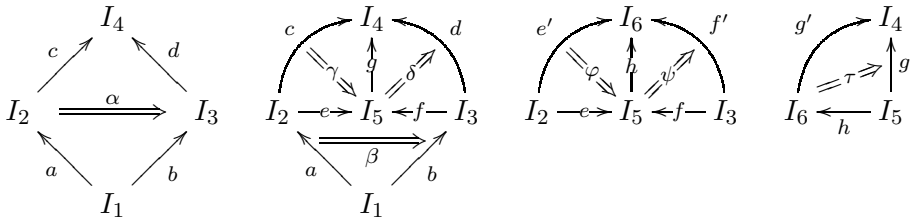
$$\psi b \bullet h\alpha \bullet \varphi a = \alpha';$$

(ii) for any other arrow $h'\colon I_4 \to I_5$ and isomorphisms $\varphi' : c' \Rightarrow h'c$, $\psi' : h'd \Rightarrow d'$ which satisfy $\psi' b \bullet h'\alpha \bullet \varphi' a = \alpha'$ there exists a unique isomorphism $\xi\colon h \Rightarrow h'$ such that $\xi c \bullet \varphi = \varphi'$ and $\psi \bullet \xi^{-1} d = \psi'$.

Analogously to Leifer and Milner's RPO being a pushout in a slice-category, we shall define a relative bipushout to be a bipushout in a pseudo-slice category. Specialising these to G-categories results in the GRPOs of Definition 2.2.9.

Definition 3.1.16 (Relative bipushout). Let $\alpha : ca \Rightarrow db : I_1 \to I_4$ be an isomorphic 2-cell in a 2-category $\mathbf{C}$, as illustrated in diagram (i). A *relative bipushout* for α is a bipushout of the pair or arrows $\langle a, 1_{ca} \rangle : ca \to c$ and $\langle b, \alpha \rangle : ca \to d$ of $\mathbf{C}/I_4$. The reader will notice that to close such a span is to give an object $g : I_5 \to I_4$, arrows $\langle e, \gamma \rangle : c \to g$, $\left\langle f, \delta^{-1} \right\rangle : d \to g$, and an isomorphic 2-cell $\beta : ea \Rightarrow fb$. Moreover, since β is actually a 2-cell of $\mathbf{C}/I_4$, it is required to satisfy a compatibility condition: $g\beta \bullet \gamma a = \delta^{-1} b \bullet \alpha$, or equivalently: $\delta b \bullet g\beta \bullet \gamma a = \alpha$. In other words, the 2-cells γ, β and δ, illustrated in diagram (ii), are required to paste together to yield α. We

shall often refer to $\langle I_5, e, f, g, \beta, \gamma, \delta\rangle$ as a candidate for α.



(i) (ii) (iii) (iv)

Such data defines a bipushout when, given another way of closing the span, i.e. a candidate $\langle I_6, e', f', g', \beta', \gamma', \delta'\rangle$ for α, there exists a mediating morphism: an arrow $h : I_5 \to I_6$ and 2-cells $\varphi : e' \Rightarrow he$ and $\psi : hf \Rightarrow f'$ (diagram (iii)); moreover h is required to be a morphism in $\mathbf{C}/I_4$, meaning that there is an additional 2-cell $\tau^{-1} : g \Rightarrow g'h$ (diagram (iv)).

The fact that φ is a 2-cell in $\mathbf{C}/I_4$ means that $g'\varphi \bullet \gamma' = \tau^{-1}e \bullet \gamma$ which is the same as $\tau e \bullet g'\varphi \bullet \gamma' = \gamma$, similarly, the fact that ψ is a 2-cell means that $g'\psi \bullet \tau^{-1}f \bullet \delta^{-1} = \delta'^{-1}$ which is the same as saying that $\delta' \bullet g'\psi \bullet \tau^{-1}f = \delta$. From the definition of bipushout, we know that $\psi b \bullet h\beta \bullet \varphi a = \beta'$. We summarise the conditions required of the 2-cells below:

(i) $\tau e \bullet g'\varphi \bullet \gamma' = \gamma$;

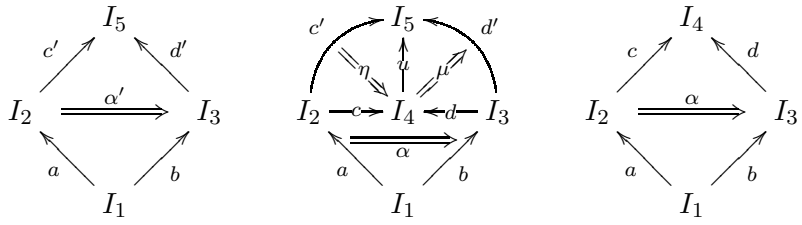
(ii) $\delta' \bullet g'\psi \bullet \tau^{-1}f = \delta$;

(iii) $\psi b \bullet h\beta \bullet \varphi a = \beta'$.

Such a mediating morphism must be essentially unique, namely, given an arrow $h' : I_5 \to I_6$ together with an isomorphism $\tau'^{-1} : g \Rightarrow g'h'$ and arbitrary 2-cells $\eta : he \Rightarrow h'e$ and $\mu : hf \Rightarrow h'f$ which satisfy $h'\beta \bullet \eta a = \mu b \bullet h\beta$ as well as $g'\eta \bullet \tau^{-1}e = \tau'^{-1}e$ and $g'\mu \bullet \tau^{-1}f = \tau'^{-1}f$, then there exists a unique 2-cell $\xi : h \Rightarrow h'$ such that $g'\xi \bullet \tau^{-1} = \tau'^{-1}$, $\eta = \xi e$ and $\mu = \xi f$.

When working in a G-category (see Remark 3.1.13), we can write the essential uniqueness condition as follows: for any other mediating morphism $\langle h', \varphi', \psi', \tau'\rangle$ there must exist a unique invertible 2-cell $\xi\colon h \Rightarrow h'$ which makes the two mediating morphisms compatible, i.e.:

1. $\xi e \bullet \varphi = \varphi'$;
2. $\psi \bullet \xi^{-1}f = \psi'$;
3. $\tau' \bullet g'\xi = \tau$.



(i) (ii) (iii)

3.2 Properties of GIPOs

In this section we shall prove the crucial properties of GRPOs and GIPOs stated in §2.2.3 and used, for instance, in the proof of the congruence theorems of section 2.3. The proofs are rather lengthy, but not complicated. Indeed, the interested reader should first prove the one-dimensional versions of the results (replacing GRPOs with RPOs and GIPOs with IPOs), as first done by Leifer and Milner [66]. The 2-dimensional versions follow basically the same proof strategies with the extra complications coming because diagrams commute only up-to invertible 2-cells and ordinary uniqueness of arrows is replaced with essential uniqueness - arrows are unique up to a unique invertible 2-cell.

The proof of Lemma 2.2.19 is divided into two parts. First, in §3.2.1 we shall prove that decomposing an arbitrary GRPO and taking the resulting closure of the original span yields a GIPO. Conversely, in §3.2.2 we shall show that starting with a GIPO which can be extended into a candidate for a square which has a GRPO then the resulting candidate is actually a GRPO. In §3.2.3 we shall prove that GIPOs compose and decompose in a manner similar to pushouts (see Lemma 1.3.2).

3.2.1 From GRPOs to GIPOs

Recall the first part of Lemma 2.2.19. It states that in an arbitrary G-category, if (3.3)

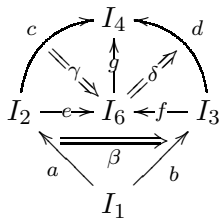
$$\left\langle I_4,\, c,\, d,\, u,\, \alpha : ca \Rightarrow db,\, \eta : c' \Rightarrow uc,\, \mu : ud \Rightarrow d' \right\rangle \tag{3.3}$$

is a GRPO for diagram (i), as illustrated in diagram (ii), then diagram (iii) is a GIPO;

Proof. As (3.3) is GRPO for diagram (i), it is a candidate and thus equation (3.4) holds.

$$\mu b \bullet u\alpha \bullet \eta a = \alpha' \tag{3.4}$$

Suppose that $\langle I_6,\, e,\, f,\, g,\, \beta : ea \Rightarrow fb,\, \gamma : c \Rightarrow ge,\, \delta : gf \Rightarrow d\rangle$ is a candidate for diagram (iii) as illustrated in diagram (iv), that is equation (3.5) holds.



(iv)

$$\delta b \bullet g\beta \bullet \gamma a = \alpha \tag{3.5}$$

Then it is easy to verify that

$$\big\langle I_6,\, e,\, f,\, ug,\, \beta,\, u\gamma \bullet \eta : c' \Rightarrow (ug)e,\, \mu \bullet u\delta : (ug)f \Rightarrow d' \big\rangle$$

is a candidate for (i): substituting equation (3.5) into equation (3.4) yields

$$\begin{aligned} \alpha' &= \mu b \bullet u(\delta b \bullet g\beta \bullet \gamma a) \bullet \eta a \\ &= \mu b \bullet u\delta b \bullet ug\beta \bullet u\gamma a \bullet \eta a \\ &= (\mu \bullet u\delta) b \bullet ug\beta \bullet (u\gamma \bullet \eta) a. \end{aligned}$$

Thus there exists $h\colon I_4 \to I_6$ and isomorphisms

$$\varphi\colon e \Rightarrow hc,\ \psi\colon hd \Rightarrow f \text{ and } \tau\colon ugh \Rightarrow u$$

satisfying

$$\tau c \bullet ug\varphi \bullet u\gamma \bullet \eta = \eta, \tag{3.6}$$

$$\mu \bullet u\delta \bullet ug\psi \bullet \tau^{-1} d = \mu, \text{ and} \tag{3.7}$$

$$\psi b \bullet h\alpha \bullet \varphi a = \beta. \tag{3.8}$$

It follows that

$$\langle gh,\, g\varphi \bullet \gamma,\, \delta \bullet g\psi,\, \tau\rangle$$

is a mediating morphism between $\langle I_4,\, c,\, d,\, u,\, \alpha,\, \eta,\, \mu\rangle$ and itself as a GRPO of diagram (i). Indeed, equations (3.6) and (3.7) can be rearranged slightly

into $\tau c \bullet u(g\varphi \bullet \gamma) \bullet \eta = \eta$ and $\mu \bullet u(\delta \bullet g\psi) \bullet \tau^{-1} d = \mu$. It remains to show that

$$(\delta \bullet g\psi)b \bullet gh\alpha \bullet (g\varphi \bullet \gamma)a = \alpha,$$

and this is a simple diagram chase:

$$\begin{aligned}(\delta \bullet g\psi)b \bullet gh\alpha \bullet (g\varphi \bullet \gamma)a &= \delta b \bullet g\psi b \bullet gh\alpha \bullet g\varphi a \bullet \gamma a \\ &= \delta b \bullet g(\psi b \bullet h\alpha \bullet \varphi a) \bullet \gamma a \\ &=_{(3.8)} \delta b \bullet g\beta \bullet \gamma a \\ &=_{(3.5)} a\end{aligned}$$

On the other hand, $\langle id_{I_4}, 1_c, 1_d, 1_u \rangle$ is clearly also a mediating morphism. Using essential uniqueness, there exists a unique 2-cell $\xi \colon gh \Rightarrow \mathrm{id}_{I_4}$ such that

$$\xi c \bullet g\varphi \bullet \gamma = 1_c, \tag{3.9}$$

$$\delta \bullet g\psi \bullet \xi^{-1} d = 1_d \quad \text{and} \tag{3.10}$$

$$u\xi = \tau. \tag{3.11}$$

Equations (3.9), (3.10) and (3.8) imply that

$$\langle h : I_4 \to I_6,\ \varphi : e \Rightarrow hc,\ \psi : hd \Rightarrow f,\ \xi : gh \Rightarrow \mathrm{id}_{I_4} \rangle$$

is a mediating morphism from $\langle I_4, c, d, \mathrm{id}, \alpha, 1_c, 1_d \rangle$ to $\langle I_6, e, f, g, \beta, \delta, \gamma \rangle$ as candidates for diagram (*iii*).

It remains to show that the mediating morphism is essentially unique. Let

$$\left\langle h' : I_4 \to I_6,\ \varphi' : e \Rightarrow h'c,\ \psi' : h'd \Rightarrow f,\ \xi' : gh' \Rightarrow \mathrm{id}_{I_4} \right\rangle$$

be another such mediating morphism, thus

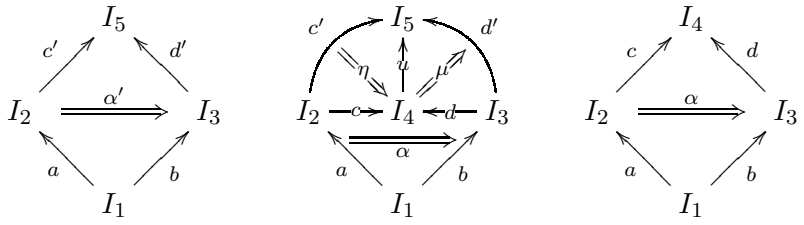
$$\xi' c \bullet g\varphi' \bullet \gamma = 1_c, \tag{3.12}$$

$$\delta \bullet g\psi' \bullet \xi'^{-1} d = 1_d, \text{ and} \tag{3.13}$$

$$\psi' b \bullet h\alpha \bullet \varphi' a = \beta. \tag{3.14}$$

It is easy to verify that $\langle h', \varphi', \psi', u\xi' \rangle$ constitutes another mediating morphism from $\langle I_4, c, d, u, \alpha, \eta, \mu \rangle$ to $\langle I_5, e, f, ug, \beta, u \circ \gamma \bullet \eta, \mu \bullet u \circ \delta \rangle$: indeed

$$\begin{aligned}u\xi' c \bullet ug\varphi' \bullet u\gamma \bullet \eta &= u(\xi' c \bullet g\varphi' \bullet \gamma) \bullet \eta \\ &=_{(3.12)} u(1_c) \bullet \eta = \eta\end{aligned}$$



(i) (ii) (iii)

and similarly, using equation (3.13) we derive $\mu \bullet u\delta \bullet ug\psi' \bullet u\xi'^{-1}d = \mu$, and equation (3.14) ends the verification.

Thus there exists a unique $\lambda\colon h \Rightarrow h'$ which satisfies

$$\lambda c \bullet \varphi = \varphi', \tag{3.15}$$

$$\psi \bullet \lambda^{-1} d = \psi', \quad \text{and} \tag{3.16}$$

$$u\xi' \bullet ug\lambda = \tau. \tag{3.17}$$

It remains to check that $\xi' \bullet g\lambda = \xi$. Recall that $\xi : gh \Rightarrow \mathrm{id}_{I_4}$ was the unique isomorphism which satisfies equations (3.9), (3.10) and (3.11). Equation (3.17) demonstrates that $\xi' \bullet g\lambda$, substituted in place of ξ satisfies the last of these. Also

$$\begin{aligned}(\xi' \bullet g\lambda)c \bullet g\varphi \bullet \gamma &= \xi' c \bullet g\lambda c \bullet g\varphi \bullet \gamma \\ &= \xi' c \bullet g(\lambda c \bullet \varphi) \bullet \gamma \\ &=_{(3.15)} \xi' c \bullet g\varphi' \bullet \gamma \\ &=_{(3.12)} 1_c\end{aligned}$$

and similarly, using equations (3.16) and (3.13) allows us to derive

$$\delta \bullet g\psi \bullet (\xi' \bullet g\lambda)^{-1} d = 1_d.$$

Thus $\xi \bullet g\lambda = \xi$, as required. □

3.2.2 From GIPOs to GRPOs

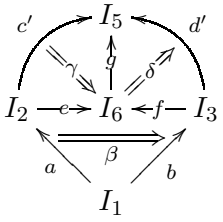
Recall the second part of Lemma 2.2.19. In an arbitrary G-category, if diagram (iii) is a GIPO, diagram (i) has a GRPO, and (3.18) is a candidate for it as shown in diagram (ii), then (3.18) is a GRPO for diagram (i).

$$\big\langle I_4,\, c,\, d,\, u,\, \alpha : ca \Rightarrow db,\, \eta : c' \Rightarrow uc,\, \mu : ud \Rightarrow d' \big\rangle \tag{3.18}$$

Proof. Using the fact that a GRPO for diagram (i) is assumed to exist, let

$$\langle I_6,\, e,\, f,\, g,\, \beta : ea \Rightarrow gb,\, \gamma : c' \Rightarrow fe,\, \delta : fg \Rightarrow d' \rangle$$

illustrated in diagram (v) be a GRPO for diagram (i).



(v)

Because of the candidate illustrated in diagram (ii) we use the defining property of GRPOs in order to derive the existence of a mediating morphism

$$\langle v : I_6 \to I_4,\, \varphi\colon c \Rightarrow ve,\, \psi\colon vf \Rightarrow d,\, \tau\colon uv \Rightarrow g \rangle$$

The equations satisfied are:

$$\tau e \bullet u\varphi \bullet \eta = \gamma, \tag{3.19}$$

$$\mu \bullet u\psi \bullet \tau^{-1} f = \delta, \text{ and} \tag{3.20}$$

$$\psi b \bullet v\beta \bullet \varphi a = \alpha. \tag{3.21}$$

Equation (3.21) asserts that $\langle I_6,\, e,\, f,\, v,\, \beta,\, \varphi,\, \psi \rangle$ is a candidate for diagram (iii) and using the fact that diagram (iii) is a GIPO, there exists a mediating morphism

$$\langle w\colon I_4 \to I_6,\, \varphi'\colon e \Rightarrow wc,\, \psi'\colon wd \Rightarrow f,\, \tau'\colon vw \Rightarrow \mathrm{id}_{I_4} \rangle$$

the isomorphisms of which satisfy

$$\tau' c \bullet v\varphi' \bullet \varphi = 1_c, \tag{3.22}$$

$$\psi \bullet v\psi' \bullet \tau'^{-1} d = 1_d, \text{ and} \tag{3.23}$$

$$\psi' b \bullet w\alpha \bullet \varphi' a = \beta. \tag{3.24}$$

We claim that

$$\langle wv : I_6 \to I_6,\, w\varphi \bullet \varphi' : e \Rightarrow wve,\, \psi' \bullet w\psi : wvf \Rightarrow f,\, \tau \bullet u\tau' v \bullet \tau^{-1} wv : gwv \Rightarrow g \rangle$$

is a mediating morphism from $\langle I_6,\, e,\, f,\, g,\, \beta,\, \gamma,\, \delta\rangle$ to itself as a GRPO.

The equations that need to be checked are:

$$(\tau \bullet u\tau' v \bullet \tau^{-1} wv)e \bullet g(w\varphi \bullet \varphi') \bullet \gamma = \gamma \tag{3.25}$$

$$\delta \bullet g(\psi' \bullet w\psi) \bullet (\tau \bullet u\tau' v \bullet \tau^{-1} wv)^{-1} f = \delta \tag{3.26}$$

$$(\psi' \bullet w\psi)b \bullet wv\beta \bullet (w\varphi \bullet \varphi')a = \beta \tag{3.27}$$

Equation (3.27) is easy to check, indeed:

$$\begin{aligned}(\psi' \bullet w\psi)b \bullet wv\beta \bullet (w\varphi \bullet \varphi')a &= \psi' b \bullet w\psi b \bullet wv\beta \bullet w\varphi a \bullet \varphi' a \\ &= \psi' b \bullet w(\psi b \bullet v\beta \bullet \varphi a) \bullet \varphi' a \\ &=_{(3.21)} \psi' b \bullet w\alpha \bullet \varphi' a \\ &=_{(3.24)} \beta\end{aligned}$$

We shall now show that equation (3.25) holds. First notice that we can transform equation (3.19) into

$$u\varphi \bullet \eta = \tau^{-1} e \bullet \gamma. \tag{3.28}$$

Now

$$\begin{aligned}(\tau \bullet u\tau' v \bullet \tau^{-1} wv)e \bullet g(w\varphi \bullet \varphi') \bullet \gamma &= \tau e \bullet u\tau' ve \bullet \tau' wve \bullet gw\varphi \bullet g\varphi' \bullet \gamma \\ &=_{(\text{pasting})} \tau e \bullet u\varphi \bullet u\tau' c \bullet uv\varphi' \bullet \tau^{-1} e \bullet \gamma \\ &=_{(3.28)} \tau e \bullet u\varphi \bullet u\tau' c \bullet uv\varphi' \bullet u\varphi \bullet \eta \\ &= \tau e \bullet u\varphi \bullet u(\tau' c \bullet v\varphi' \bullet \varphi) \bullet \eta \\ &=_{(3.22)} \tau e \bullet u\varphi \bullet u(1_c) \bullet \eta \\ &= \tau e \bullet u\varphi \bullet \eta \\ &=_{(3.19)} \gamma\end{aligned}$$

The method of showing that equation (3.26) holds is similar.

Since $\langle \mathrm{id}_{I_6},\, 1_e,\, 1_f,\, 1_g\rangle$ is clearly also a mediating morphism, there exists a unique $\xi\colon wv \Rightarrow \mathrm{id}_{I_6}$ which makes the two mediating morphisms compatible, that is, equations

$$\xi e \bullet w\varphi \bullet \varphi' = 1_e, \tag{3.29}$$

$$\psi' \bullet w\psi \bullet \xi^{-1} f = 1_f \text{ and} \tag{3.30}$$

$$g\xi = \tau \bullet u\tau' v \bullet \tau^{-1} wv \tag{3.31}$$

hold.

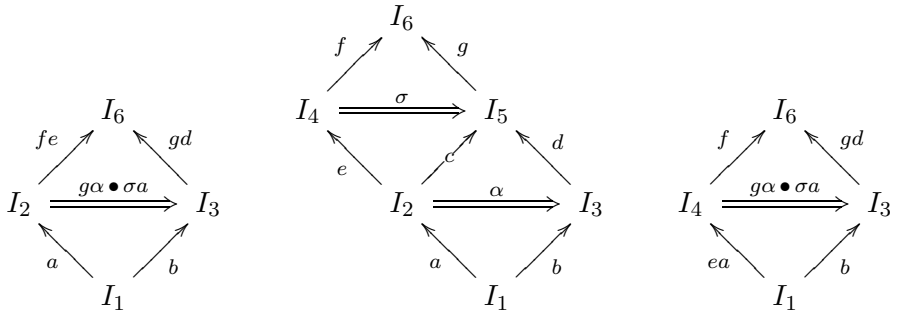
We have derived the existence of isomorphisms

$$\tau'^{-1} : \mathrm{id}_{I_4} \Rightarrow vw \quad \text{and} \quad \xi : wv \Rightarrow \mathrm{id}_{I_6}\,.$$

Since GRPOs are defined up to an equivalence, this completes the proof. □

3.2.3 Composition and Decomposition

Here we shall be concerned with Lemma 2.2.20. We shall first restate it below and then proceed to give a detailed proof. The lemma holds in any G-category.

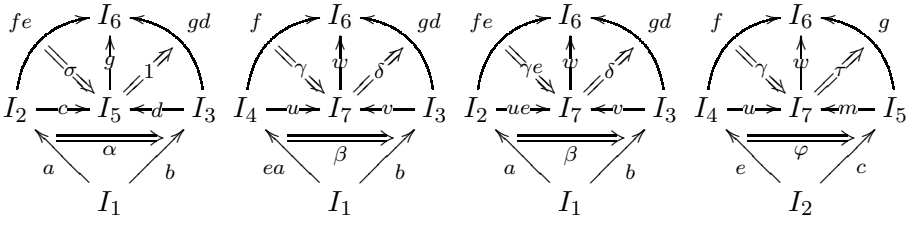


(i) (ii) (iii)

Suppose that diagram (i) has a GRPO. Then:

1. if both squares in diagram (ii) are GIPOs then the exterior (diagram (iii)) is also a GIPO;
2. if the lower square and the exterior (diagram (iii)) of diagram (ii) are GIPOs then so is the upper square.

Proof. **(1).** Using the conclusion of the second part of Lemma 2.2.19, the components of $\langle I_5, c, d, g, \alpha, 1_{gd}, \sigma\rangle$ form a GRPO for diagram (i), as illustrated in the diagram (iv) below.



(iv) (v) (vi) (vii)

Suppose that, as illustrated in diagram (v),

$$\langle I_7,\ u : I_4 \to I_7,\ v : I_3 \to I_7,\ w : I_7 \to I_6,$$
$$\beta : uea \Rightarrow vb,\ \gamma : f \Rightarrow wv,\ \delta : wv \Rightarrow gd\rangle$$

is a candidate for diagram (*iii*), that is $\delta b \bullet w\beta \bullet \gamma ea = g\alpha \bullet \sigma a$.

It follows immediately that $\langle I_7,\, ue,\, v,\, w,\, \beta,\, \gamma e,\, \delta\rangle$ is a candidate for diagram (*i*), as illustrated in diagram (*vi*), and thus there exists a mediating morphism

$$\langle m\colon I_5 \to I_7,\, \varphi : ue \Rightarrow mc,\, \psi\colon md \Rightarrow v,\, \tau\colon wm \Rightarrow g\rangle$$

satisfying the usual compatibility requirements, namely:

$$\tau c \bullet w\varphi \bullet \gamma e = \sigma, \tag{3.32}$$

$$\delta \bullet w\psi \bullet \tau^{-1} d = 1_{gd} \quad \text{and} \tag{3.33}$$

$$\psi b \bullet m\alpha \bullet \varphi a = \beta. \tag{3.34}$$

In particular, equation (3.32) implies that $\langle I_7,\, u,\, m,\, w,\, \varphi,\, \gamma,\, \tau\rangle$ is a candidate for upper square of diagram (*ii*). By assumption, this in a GIPO and thus there exists an mediating morphism

$$\big\langle n\colon I_6 \to I_7,\, \varphi'\colon u \Rightarrow nf,\, \psi'\colon ng \Rightarrow m,\, \tau'\colon wn \Rightarrow \mathrm{id}_{I_6}\big\rangle$$

the components of which satisfy

$$\tau' f \bullet w\varphi' \bullet \gamma = 1_f, \tag{3.35}$$

$$\tau \bullet w\psi' \bullet \tau'^{-1} g = 1_g \quad \text{and} \tag{3.36}$$

$$\psi' c \bullet n\sigma \bullet \varphi' e = \varphi. \tag{3.37}$$

Now it follows that

$$\big\langle n : I_6 \to I_7,\, \varphi' : u \Rightarrow nf,\, \psi \bullet \psi' d : ngd \Rightarrow v,\, \tau' : wn \Rightarrow \mathrm{id}_{I_6}\big\rangle$$

constitutes a mediating morphism from $\langle I_6,\, f,\, gd,\, \mathrm{id}_{I_4},\, g\alpha \bullet \sigma a,\, 1_f,\, 1_{gd}\rangle$ to $\langle I_7,\, u,\, v,\, w,\, \beta,\, \gamma,\, \delta\rangle$. Indeed, the first equation that needs to be checked is equation (3.35). The second equation can be derived as follows:

$$\begin{aligned}
\delta \bullet w\psi \bullet w\psi' d \bullet \tau'^{-1} gd &=_{(3.33)} \tau d \bullet w\psi' d \bullet \tau'^{-1} gd \\
&= (\tau \bullet w\psi' \bullet \tau'^{-1} g) d \\
&=_{(3.36)} 1_{gd}
\end{aligned}$$

We have derived the existence of a mediating morphism. We shall now show that it is essentially unique. Indeed, suppose that

$$\big\langle n' : I_6 \to I_7,\, \varphi'' : u \Rightarrow n'f,\, \psi'' : n'gd \Rightarrow v,\, \tau'' : wn' \Rightarrow \mathrm{id}_{I_6}\big\rangle$$

is another such mediating morphism. Then

$$\tau'' f \bullet w\varphi'' \bullet \gamma = 1_f, \tag{3.38}$$

$$\delta \bullet w\psi'' \bullet \tau''^{-1} gd = 1_{gd} \quad \text{and} \tag{3.39}$$

$$\psi'' b \bullet n'g\alpha \bullet n'\sigma a \bullet \varphi'' ea = \beta. \tag{3.40}$$

Then it follows that

$$\langle n'g : I_5 \to I_7,\ n'\sigma \bullet \varphi'' e : ue \Rightarrow n'gc,\ \psi'' : n'gd \Rightarrow v,\ \tau'' g : wn'g \Rightarrow g \rangle$$

is a mediating morphism from $\langle I_5,\ c,\ d,\ g,\ \alpha,\ \sigma,\ 1_{gc} \rangle$ to $\langle I_7,\ ue,\ v,\ w,\ \beta,\ \gamma e,\ \delta \rangle$. Indeed, the only equation that needs to be checked is

$$\begin{aligned} \tau'' gc \bullet wn'\sigma \bullet w\varphi'' e \bullet \gamma e &= \sigma \bullet \tau'' f e \bullet w\varphi'' e \bullet \gamma e \\ &= \sigma \bullet (\tau'' f \bullet w\varphi'' \bullet \gamma) e \\ &=_{(3.38)} \sigma. \end{aligned}$$

We have already seen the two other conditions as equations (3.39) and (3.40).

Since we know that $\langle m,\ \varphi,\ \psi,\ \tau \rangle$ is also such a mediating morphism, there exists a unique $\xi \colon m \Rightarrow n'g$ so that

$$\xi c \bullet \varphi = n'\sigma \bullet \varphi'' e, \tag{3.41}$$

$$\psi \bullet \xi^{-1} d = \psi'' \quad \text{and} \tag{3.42}$$

$$\tau'' g \bullet w\xi = \tau. \tag{3.43}$$

We are now ready to conclude that

$$\langle n' : I_6 \to I_7,\ \varphi'' : u \Rightarrow n'f,\ \xi^{-1} : n'g \Rightarrow m,\ \tau'' : wn' \Rightarrow \mathrm{id}_{I_6} \rangle$$

is a mediating morphism between the two candidates for the upper square of diagram (i):

$$\langle I_6,\ f,\ g,\ \mathrm{id}_{I_6},\ \sigma,\ 1_f,\ 1_g \rangle \ \text{ and } \ \langle I_7,\ u,\ m,\ w,\ \varphi,\ \gamma,\ \tau \rangle .$$

Indeed, two candidates are made compatible because:

1. equation (3.38);
2. $\tau \bullet w\xi^{-1} \bullet \tau''^{-1} = 1_g$ as a result of rearranging equation (3.43);
3. $\xi^{-1} c \bullet n'\sigma \bullet \varphi'' e = \varphi$ as a result of rearranging equation (3.41).

Thus there exists a unique $\xi' \colon n \Rightarrow n'$ which makes the mediating morphism compatible with $\langle n,\, \varphi',\, \psi',\, \tau' \rangle$, which means that

$$\xi' f \bullet \varphi' = \varphi'', \tag{3.44}$$

$$\psi' \bullet \xi'^{-1} g = \xi^{-1} \quad \text{and} \tag{3.45}$$

$$\tau'' \bullet w\xi' = \tau'. \tag{3.46}$$

It is easy to check that ξ' makes $\langle n,\, \varphi',\, \psi \bullet \psi' d,\, \tau' \rangle$ compatible with $\langle n',\, \varphi'',\, \psi'',\, \tau'' \rangle$ also. Indeed, the equations that we need to check are just equation (3.44),

$$\begin{aligned} \psi \bullet \psi' d \bullet \xi'^{-1} g d &= \psi \bullet (\psi' \bullet \xi' g) d \\ &=_{(3.45)} \psi \bullet \xi^{-1} d \\ &=_{(3.42)} \psi'' \end{aligned}$$

and equation (3.46).

If there is another such $\xi'' : n \Rightarrow n'$ which satisfies equations

$$\xi'' f \bullet \varphi = \varphi'', \tag{3.47}$$

$$\psi \bullet \psi' d \bullet \xi''^{-1} g d = \psi'' \quad \text{and} \tag{3.48}$$

$$\tau'' \bullet w\xi'' = \tau', \tag{3.49}$$

then to check that ξ'' is a modification between the mediating morphisms $\langle n,\, \varphi',\, \psi',\, \tau' \rangle$ and $\langle n', \varphi'', \xi^{-1}, \tau'' \rangle$ we need only check that $\psi' \bullet \xi''^{-1} g = \xi^{-1}$. This follows by uniqueness of ξ, it suffices to check that $(\psi' \bullet \xi''^{-1} g)^{-1} = \xi'' g \bullet \psi'^{-1}$ is a modification between mediating morphisms $\langle m,\, \varphi,\, \psi,\, \tau \rangle$ and $\langle n'g,\, n'\sigma \bullet \varphi'' e,\, \psi'',\, \tau'' g \rangle$. Indeed,

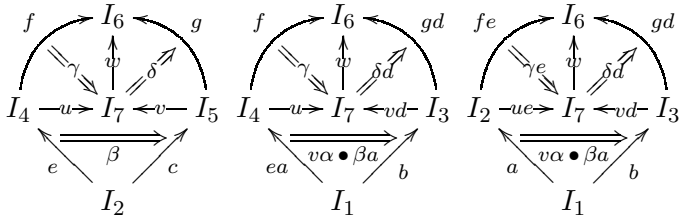
$$\begin{aligned} (\xi'' g \bullet \psi'^{-1}) c \bullet \varphi &= \xi'' g c \bullet \psi'^{-1} c \bullet \varphi \\ &=_{(3.37)} \xi'' g c \bullet n\sigma \bullet \varphi' e \\ &= n'\sigma \bullet \xi'' f e \bullet \varphi' e \\ &= n'\sigma \bullet (\xi'' f \bullet \varphi') e \\ &=_{(3.47)} n'\sigma \bullet \varphi'' e, \end{aligned}$$

$$\begin{aligned} \psi \bullet (\psi' \bullet \xi''^{-1} g) d &= \psi \bullet \psi' d \bullet \xi''^{-1} g d \\ &=_{(3.48)} \psi'' \quad \text{and} \end{aligned}$$

$$\begin{aligned}\tau'' g \bullet w(\xi'' g \bullet \psi'^{-1}) &= \tau'' g \bullet w\xi'' g \bullet w\psi'^{-1} \\ &= (\tau'' \bullet w\xi'')g \bullet w\psi'^{-1} \\ &=_{(3.49)} \tau' g \bullet w\psi'^{-1} \\ &=_{(3.36)} \tau\end{aligned}$$

as required.

(2). Suppose that $\langle I_5,\, u,\, v,\, w,\, \beta,\, \delta,\, \gamma\rangle$ is a candidate for the upper square



$(viii)$ (ix) (x)

of diagram (ii), as illustrated in diagram $(viii)$. Then

$$\langle I_7,\, u,\, vd,\, w,\, v\alpha \bullet \beta a,\, \gamma,\, \delta d\rangle$$

is a candidate for diagram (iii), as illustrated in diagram (ix). In particular, this means that

$$\delta c \bullet w\beta \bullet \gamma e = \sigma. \tag{3.50}$$

This, by assumption, is a GIPO so there exists a mediating morphism

$$\langle m\colon I_6 \to I_7,\, \varphi\colon u \Rightarrow mf,\, \psi\colon mgd \Rightarrow vd,\, \tau\colon wm \Rightarrow \mathrm{id}_{I_6}\rangle$$

satisfying

$$\tau f \bullet w\varphi \bullet \gamma = 1_f, \tag{3.51}$$

$$\delta d \bullet w\psi \bullet \tau^{-1} gd = 1_{gd} \quad \text{and} \tag{3.52}$$

$$\psi b \bullet mg\alpha \bullet m\sigma a \bullet \varphi ea = v\alpha \bullet \beta a \tag{3.53}$$

Notice that also

$$\langle I_7,\, ue,\, vd,\, w,\, v\alpha \bullet \beta a,\, \gamma e,\, \delta d\rangle$$

is a candidate for diagram (i). Recall that by using the conclusion of the second part Lemma 2.2.19 we have that candidate

$$\langle I_5,\, c,\, d,\, g,\, \alpha,\, \sigma,\, 1_{gd}\rangle$$

is a GRPO for diagram (i), as illustrated in diagram (iv).

Now $\langle v : I_5 \to I_7,\ \beta,\ 1_{vd},\ \delta\rangle$ is a mediating morphism, indeed equation (3.50) confirms that $\delta c \bullet w\beta \bullet \gamma e = \sigma$, clearly $\delta d \bullet w1_{vd} \bullet \delta^{-1} d = 1_{gd}$ and $1_{vd} b \bullet b\alpha \bullet \beta a = v\alpha \bullet \beta a$.

But also $\langle mg,\ m\sigma \bullet \varphi e,\ \psi,\ \tau g\rangle$ is such a mediating morphism, indeed:

$$\begin{aligned}\tau g c \bullet wm\sigma \bullet w\varphi w \bullet \gamma e &= \sigma \bullet \tau f e \bullet w\varphi e \bullet \gamma e \\ &= \sigma \bullet (\tau f \bullet w \bullet \gamma) e \\ &=_{(3.51)} \sigma,\end{aligned}$$

while the second and third equations are just equations (3.52) and (3.53).

Thus there exists a unique two-cell $\xi \colon v \Rightarrow mg$ making the two mediating morphisms compatible, we have that:

$$\xi c \bullet \beta = m\sigma \bullet \varphi e, \tag{3.54}$$

$$\xi^{-1} d = \psi \quad \text{and} \tag{3.55}$$

$$\tau g \bullet w\xi = \delta. \tag{3.56}$$

These equations imply that

$$\left\langle m : I_6 \to I_7,\ \varphi : u \Rightarrow mf,\ \xi^{-1} : mg \Rightarrow v,\ \tau : wm \Rightarrow \mathrm{id}_{I_6}\right\rangle$$

is a mediating morphism from $\langle I_6,\ f,\ g,\ \mathrm{id}_{I_6},\ \sigma,\ 1_f,\ 1_g\rangle$ to $\langle I_7,\ u,\ v,\ w,\ \beta,\ \gamma,\ \delta\rangle$ in the upper square of diagram (ii). Indeed, $\tau f \bullet w\varphi \bullet \gamma = 1_f$ by equation (3.51),

$$\begin{aligned}\delta \bullet w\xi^{-1} \bullet \tau^{-1} g &=_{(3.56)} \tau g \bullet w\xi \bullet w\xi^{-1} \bullet \tau^{-1} g \\ &= 1_g\end{aligned}$$

and $\xi^{-1} c \bullet m\sigma \bullet \varphi e = \beta$ as a result of rearranging equation (3.54).

We need only to show that the mediating morphism is essentially unique. Suppose that

$$\left\langle m' : I_6 \to I_7,\ \varphi' : u \Rightarrow m'f,\ \psi' : m'g \Rightarrow v,\ \tau' : wm' \Rightarrow \mathrm{id}_{I_6}\right\rangle$$

is another such morphism, then:

$$\tau' f \bullet w\varphi' \bullet \gamma = 1_f, \tag{3.57}$$

$$\delta \bullet w\psi' \bullet \tau'^{-1} g = 1_g \quad \text{and} \tag{3.58}$$

$$\psi' c \bullet m'\sigma \bullet \varphi' e = \beta. \tag{3.59}$$

Now

$$\langle m',\, \varphi',\, \psi' d,\, \tau' \rangle$$

is a mediating morphism, guaranteed by equation (3.57), $\delta d \bullet w\psi' d \bullet \tau'^{-1} g d = (\delta \bullet w\psi' \bullet \tau'^{-1} g)d = 1_{gd}$ using equation (3.58) and

$$\begin{aligned} \psi' db \bullet m' g\alpha \bullet m'\sigma a \bullet \varphi' ea &= v\alpha \bullet \psi' ca \bullet m'\sigma a \bullet \varphi' ea \\ &= v\alpha \bullet (\psi' c \bullet m'\sigma \bullet \varphi' e)a \\ &=_{(3.59)} v\alpha \bullet \beta a. \end{aligned}$$

Hence there is a unique $\xi' \colon m \Rightarrow m'$ which makes this mediating morphism compatible with $\langle m,\, \varphi,\, \psi,\, \tau \rangle$ meaning that we have

$$\xi' f \bullet \varphi = \varphi', \tag{3.60}$$

$$\psi \bullet \xi'^{-1} g d = \psi' d \quad \text{and} \tag{3.61}$$

$$\tau' \bullet w\xi' = \tau. \tag{3.62}$$

To show that ξ' is also a modification between mediating morphisms $\langle m,\, \varphi,\, \xi^{-1},\, \tau \rangle$ and $\langle m',\, \varphi',\, \psi',\, \tau' \rangle$ it remains to show that $\xi^{-1} \bullet \xi'^{-1} g = \psi'$. We shall show, equivalently, that $\xi'^{-1} g \bullet \psi'^{-1} = \xi$, using the fact that ξ was defined as the unique 2-cell satisfying equations (3.54), (3.55) and (3.56). Indeed, we have

$$\begin{aligned} (\xi'^{-1} g \bullet \psi'^{-1})c \bullet \beta &= \xi'^{-1} gc \bullet \psi'^{-1} c \bullet \beta \\ &=_{(3.59)} \xi'^{-1} gc \bullet \psi'^{-1} c \bullet \psi' c \bullet m'\sigma \bullet \varphi' e \\ &= \xi'^{-1} gc \bullet m'\sigma \bullet \varphi' e \\ &= m\sigma \bullet \xi'^{-1} f e \bullet \varphi' e \\ &= m\sigma \bullet (\xi'^{-1} f \bullet \varphi')e \\ &=_{(3.60)} m\sigma \bullet \varphi e \end{aligned}$$

$$\begin{aligned} (\psi' \bullet \xi' g)d &= \psi' d \bullet \xi' g d \\ &=_{(3.61)} \psi \end{aligned}$$

$$\begin{aligned} \tau g \bullet w(\xi'^{-1} g \bullet \psi^{-1}) &= \tau g \bullet w\xi'^{-1} g \bullet w\psi^{-1} \\ &= (\tau \bullet w\xi'^{-1})g \bullet w\psi^{-1} \\ &=_{(3.62)} \tau' g \bullet w\psi^{-1} \\ &=_{(3.58)} \delta \end{aligned}$$

It remains only to show that ξ' is the unique such modification. Indeed, suppose that there is another $\xi'' : m \Rightarrow m'$ which satisfies

$$\xi'' f \bullet \varphi = \varphi', \tag{3.63}$$

$$\xi^{-1} \bullet \xi''^{-1} g = \psi' \quad \text{and} \tag{3.64}$$

$$\tau' \bullet w\xi'' = \tau. \tag{3.65}$$

We shall use the fact that ξ' was originally defined as the unique modification between $\langle m,\, \varphi,\, psi,\, \tau \rangle$ and $\langle m', \varphi', \psi' d,\, \tau' \rangle$, it remains only to check that

$$\begin{aligned} \psi \bullet \xi''^{-1} g d &=_{(3.55)} \xi^{-1} d \bullet \xi''^{-1} g d \\ &= (\xi^{-1} \bullet \xi''^{-1} g) d \\ &=_{(3.64)} \psi' d \end{aligned}$$

which confirms that $\xi'' = \xi'$. □

Chapter 4

Extensive categories and bunch contexts

In this chapter we shall continue our study of GRPOs, giving fully worked out constructions of GRPOs for the two running examples of Chapter 2: the G-category $\mathbf{M}_\Sigma$, the arrows of which model the terms of the simple process calculus (see Examples 2.1.7 and 2.2.4) and the G-category **Bun** of concrete bunch contexts (Definition 2.2.5). The constructions and the proofs of universality are done categorically. Thus, for instance, we do not need to talk about the elements of the carriers of the bunch contexts.

In fact, we assume only that the the carriers are objects of an *extensive* [12] category; for the construction of GRPOs in **Bun** we also need this category to have pushouts. Extensive categories have an associated slogan: "coproducts exist and are well-behaved". Of course, the paradigm for good behaviour is the category **Set** of sets and functions. The good behaviour of coproducts is useful for us because, for example, the carrier set of the composite of two bunch contexts is the coproduct of their carrier sets. Extensive categories are also a precursor to adhesive categories of Chapter 5. Indeed, the main axiom in the axiomatic presentation of the definition of extensive categories, is very similar to the main axiom of adhesive categories.

The second main contribution of this chapter is a translation of the theory of *S-precategories* developed by Milner [73], and more recently by Jensen and Milner [46], into the theory of G-reactive systems and GRPOs. We shall also briefly discuss the theory of *functorial reactive systems* of Leifer [64]. These theories have been developed in order to deal with the problems caused by non-trivial structural congruences and isomorphism (see §2.2.1). We shall show, via the translation, that the theory of G-reactive systems and GRPOs provides a unifying, elegant and general framework which subsumes the previous theories.

The structure of this chapter is as follows: in section 4.1 we shall recall

the definition of extensive categories and prove several lemmas which shall be needed later in this chapter, as well as in Chapter 5. In section 4.2 we shall present the construction of GRPOs in the G-categories $\mathbf{M}_\Sigma$ and **Bun**. Finally, in section 4.3 we show how previously introduced theory developed in order to treat examples such as bunch contexts can actually be seen as a special case of the theory of G-reactive systems and GRPOs.

4.1 Extensive categories

In this section we shall introduce the theory of extensive categories. First, in §4.1.1, we briefly recall the theory of distributive categories, which led to the development of extensive categories. We shall recall the definition of extensive categories in §4.1.2. Finally, in §4.1.3, we shall derive several properties of extensive categories which give an intuition of their structure and which shall be needed in section 4.2 as well as in Chapter 5.

4.1.1 Distributive categories

Extensive categories arose in the process of investigations into the structure of *distributive* categories, which are categories with products and coproducts where the canonical arrow

$$A \times B + A \times C \to A \times (B + C) \tag{4.1}$$

defined by $\mathrm{id}_A \times i_{B \to B+C} : A \times B \to A \times (B + C)$ on the first injection and $\mathrm{id}_A \times i_{C \to B+C} : A \times C \to A \times (B + C)$ on the second, is invertible.

Distributive categories have been used in computer science, see the book by Walters [107] for an introduction. In particular, they have been used to model circuits [37] and data-types [106].

Several different definitions of the concept of "distributive category" have been considered in literature. One, advocated by Walters [107], is that a category is distributive when it has finite products and coproducts and (4.1) is invertible.

Alternatively, Schanuel [96] and Lawvere [63] proposed a category with all finite limits (not just products) and finite coproducts and additionally, the canonical functor

$$+ : \mathbf{C}/A \times \mathbf{C}/B \to \mathbf{C}/(A + B) \tag{4.2}$$

defined on objects by taking objects $\langle r : X \to A,\ s : Y \to B \rangle$ to $r + s : X + Y \to A+B$ and sending arrows $\langle f : X \to X',\ g : Y \to Y' \rangle$ to $f+g : X+Y \to$

$X' + Y'$, is required to be an equivalence of categories. In such categories one can verify that (4.1) is indeed invertible.

Such categories have been interesting to category theorists, not least because they can be seen to generalise the theory or *rings*, which are algebraic structures with addition, multiplication and where multiplication distributes over addition. More precisely, they generalise *rigs*, which are "rings without additive inverses"; additive inverses, in the naive sense, are impossible for categories: if $A + B \cong 0$ then it follows that $A \cong 0$ and $B \cong 0$ (see [96]).

Carboni, Lack and Walters [12] argued that requiring (4.2) is actually a property of coproducts which they dubbed *extensivity*. Indeed, it makes sense without requiring the category to have products, and it implies that coproducts have useful additional structure, not necessarily present in ordinary categories. However, as mentioned before, in the presence of products the fact that functor (4.2) is an equivalence does guarantee that (4.1) is invertible.

4.1.2 Extensive categories

This subsection is meant as a very brief introduction to the theory of extensive categories – in particular, we shall only concentrate on the properties of extensive categories which shall be useful later in the thesis. For a more complete introduction to extensive categories, the reader is directed to [12] and also Gates' PhD dissertation [37].

We shall first present the axiomatic definition of extensive categories [12], this is equivalent to requiring the canonical functor (4.2) to be an equivalence in a category with finite products (see Proposition 4.1.2).

Definition 4.1.1 (Extensive category). A category $\mathbf{C}$ is said to be *extensive* when

(i) it has finite coproducts

(ii) it has pullbacks along coproduct injections

(iii) given a diagram where the bottom row is a coproduct diagram

$$\begin{array}{ccccc} X & \xrightarrow{m} & Z & \xleftarrow{n} & Y \\ {\scriptstyle r}\downarrow & & \downarrow{\scriptstyle h} & & \downarrow{\scriptstyle s} \\ A & \longrightarrow & A+B & \longleftarrow & B \end{array}$$

the two squares are pullbacks if and only if the top row is a coproduct.

The third axiom is the most interesting. Notice that it implies that coproduct diagrams are stable under pullback – but this just the *only if* part of the axiom. The second part of the axiom states that if we have a commutative diagram in which the top and bottom rows are coproducts then the two squares are pullbacks.

Indeed, it is the third axiom of Definition 4.1.1 that captures what we mean when we say that the coproduct $A + B$ is "well-behaved": it includes the fact that coproducts are stable under pullback, and it implies that coproducts are disjoint[1] and that initial objects are strict.[2] It also implies a cancellativity property of coproducts: given an isomorphism $A + B \cong A + C$ compatible with the injections, one can construct an isomorphism $B \cong C$.[3] For an object Z of an extensive category, the lattice $\mathrm{Sub}(Z)$ of coproduct summands of Z is a Boolean algebra [12, 37].

The proof of the following proposition, which relates the axiomatic and the "equivalence of categories" definitions of extensive categories, can be found in [12].

Proposition 4.1.2. Given a category $\mathbf{C}$, the following conditions are equivalent if they hold for all objects A and B:

(i) the functor $\mathbf{C}/A \times \mathbf{C}/B \to \mathbf{C}/(A + B)$, which forms the coproducts of morphisms in $\mathbf{C}$, is an equivalence of categories;

(ii) in a commutative diagram

$$\begin{array}{ccccc} X & \xrightarrow{m} & Z & \xleftarrow{n} & Y \\ {\scriptstyle r}\downarrow & & \downarrow{\scriptstyle h} & & \downarrow{\scriptstyle s} \\ A & \longrightarrow & A + B & \longleftarrow & B \end{array}$$

the top row is a coproduct diagram if and only if the squares are pullbacks;

Examples of extensive categories include **Set**, and more generally any topos. The category of topological spaces and continuous functions **Top** is extensive. Any category with freely generated coproducts is extensive [12].

The category of sets and partial functions **Par**, isomorphic to the coslice category $1/\mathbf{Set}$, fails to be adhesive. This follows easily, since every set has the fully undefined map to the initial object (the singleton), in other words, the initial object is not strict.

[1] The pullback of the coproduct injections is initial, see Lemma 4.1.3, part (i).
[2] Any arrow to an initial object must be an isomorphism, see Lemma 4.1.3, part (ii).
[3] See Lemma 4.1.3, part (iv).

4.1.3 Properties of extensive categories

In order to provide the reader with some intuition for the good behaviour of coproducts in extensive categories, we recall below some of their properties. The individual parts of Lemma 4.1.3 have appeared previously in [12]. We include the proofs here because we shall prove similar properties, about pushouts along monos instead of coproducts, for adhesive categories in Chapter 5. Several of these properties shall also be used in section 4.2 during the construction of GRPOs in the G-category of bunch contexts.

Lemma 4.1.3. Let **C** be an extensive category. Then,

(i) sums are disjoint; that is, the pullback of the two injections of a binary coproduct is the initial object;

(ii) coproduct injections are mono;

(iii) summand "complements" are unique up to isomorphism: if we have coproduct diagrams $A \xrightarrow{i_1} C \xleftarrow{i_2} B$ and $A' \xrightarrow{i'_1} C \xleftarrow{i_2} B$ then there exists a unique isomorphism $\varphi : A \to A'$ such that $i'_1\varphi = i_1$;

(iv) if $\varphi : A+C \to B+C$ is an isomorphism such that $\varphi i_{C\to A+C} = i_{C\to B+C}$ then there exists a unique isomorphism $\psi : A \to B$ so that $\varphi = \psi + C$;

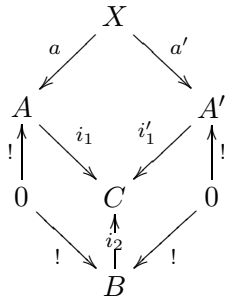
(v) initial objects are strict, that is, any arrow $\varphi : C \to 0$ is necessarily an isomorphism.

Proof. We shall begin by proving (i) and (ii).

$$\begin{array}{ccccc} 0 & \xrightarrow{!} & B & \xleftarrow{\mathrm{id}} & B \\ {\scriptstyle !}\downarrow & & {\scriptstyle i_2}\downarrow & & \downarrow{\scriptstyle \mathrm{id}} \\ A & \xrightarrow[i_1]{} & A+B & \xleftarrow[i_2]{} & B \end{array}$$

and the two squares are clearly commutative. Using the definition of extensivity, the two squares are, therefore, pullbacks. The left square being a pullback means that coproducts are disjoint. The fact that the right hand side is a pullback implies that i_2 is mono. By a similar argument, i_1 is also mono.

We shall now proceed with (iii). Consider the following diagram,



using part (i), we deduce that the two lower regions are pullbacks. Let the upper region be a pullback. Using extensivity, $0 \xrightarrow{!} A \xleftarrow{a} X$ and $0 \xrightarrow{!} A' \xleftarrow{a'} X$ are coproduct diagrams, and therefore, it follows that a and a' are isomorphisms. Let $\varphi = a'a^{-1}$, which satisfies $i'_1\varphi = i_1$, as required. Given another such φ', we have $i'_1\varphi = i_1 = i'_1\varphi$. We can now use part (ii) to deduce that i'_1 is mono, and therefore, that $\varphi = \varphi'$.

It remains to prove (iv). Consider the diagram below, where

$$\begin{array}{ccccc} X & \xrightarrow{f} & A+C & \xleftarrow[i_2]{} & C \\ {\scriptstyle \psi'}\downarrow & & \downarrow{\scriptstyle \varphi} & & \downarrow{\scriptstyle \mathrm{id}} \\ B & \xrightarrow[i_1]{} & B+C & \xleftarrow[i_2]{} & C \end{array}$$

the right square can be verified to be pullback, using the fact that φ is mono. Let the left-square be a pullback. Note that ψ' is an isomorphism, since it is a pullback of an isomorphism. Using extensivity, the resulting top row is a coproduct diagram, and using part (iii), we can deduce that there exists an isomorphism $\varphi : A \to X$ such that $f\varphi = i_1 : A \to A + C$. Letting $\psi = \psi'\varphi$, we obtain $\varphi = \psi + C$. The fact that $i_1 : B \to B + C$ is mono implies uniqueness.

Finally, to prove part (v), suppose that there exists an arrow $\varphi : C \to 0$. In the following commutative diagram, the two squares are clearly pullbacks.

$$\begin{array}{ccccc} C & \xrightarrow{\mathrm{id}} & C & \xleftarrow{\mathrm{id}} & C \\ {\scriptstyle \varphi}\downarrow & & \downarrow{\scriptstyle \varphi} & & \downarrow{\scriptstyle \varphi} \\ 0 & \xrightarrow[\mathrm{id}]{} & 0 & \xleftarrow[\mathrm{id}]{} & 0 \end{array}$$

Then C is a coproduct of two copies of itself with identity injections. Thus C admits at most one arrow to any other object. Since there is an arrow to

the initial object, it admits exactly one arrow to any other object. Thus C is initial and therefore φ is invertible. □

The following lemma is actually a simple corollary of the fact that coproduct injections are mono in extensive categories (Lemma 4.1.3, part (ii)). However, because it shall prove to be useful in the next section, we include it here.

Lemma 4.1.4. In an extensive category, if for any pair of arrows $\varphi, \psi : A \to B$ and any object C we have $\varphi + C = \psi + C : A + C \to B + C$ then $\varphi = \psi$.

Proof. We have $i_{B \to B+C}\varphi = (\varphi + C) i_{A \to A+C} = (\psi + C) i_{A \to A+C} = i_{B \to B+C}\psi$. The conclusion follows because $i_{B \to B+C}$ is mono (Lemma 4.1.3, part (ii)). □

The following is a straightforward lemma about pullbacks in extensive categories. It says, roughly, that the class of pullbacks is closed under coproduct and it shall be useful for us in Chapter 5.

Lemma 4.1.5. Suppose that $\mathbf{C}$ is an extensive category and that diagrams (i) and (ii) below are pullbacks in $\mathbf{C}$.

$$\begin{array}{ccc} A_1 & \xrightarrow{f_1} & B_1 \\ {\scriptstyle g_1}\downarrow & & \downarrow{\scriptstyle u_1} \\ C_1 & \xrightarrow[v_1]{} & D_1 \end{array} \qquad \begin{array}{ccc} A_2 & \xrightarrow{f_2} & B' \\ {\scriptstyle g_2}\downarrow & & \downarrow{\scriptstyle u_2} \\ C_2 & \xrightarrow[v_2]{} & D_2 \end{array} \qquad \begin{array}{ccc} A_1 + A_2 & \xrightarrow{f_1+f_2} & B_1 + B_2 \\ {\scriptstyle g_1+g_2}\downarrow & & \downarrow{\scriptstyle u_1+u_2} \\ C_1 + C_2 & \xrightarrow[v_1+v_2]{} & D_1 + D_2 \end{array}$$

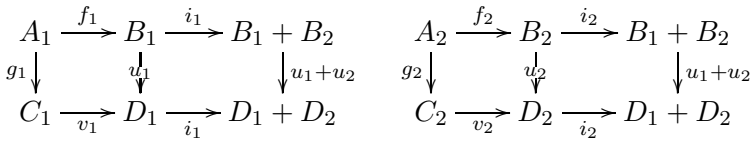
Then diagram (iii) is a pullback in $\mathbf{C}$.

Proof. Take an arbitrary object X with maps $\alpha : X \to B_1 + B_2$ and $\beta : X \to C_1 + C_2$ satisfying $(u_1 + u_2)\alpha = (v_1 + v_2)\beta$. Using extensivity, we can express X as a coproduct of X_1 and X_2 by taking pullbacks

$$\begin{array}{ccccc} X_1 & \xrightarrow{i_1} & X & \xleftarrow{i_2} & X_2 \\ {\scriptstyle \beta_1}\downarrow & & {\scriptstyle \beta}\downarrow & & \downarrow{\scriptstyle \beta_2} \\ C_1 & \xrightarrow[i_1]{} & C_1 + C_2 & \xleftarrow[i_2]{} & C_2 \end{array}$$

as above. Now, observe that the exteriors of the two diagrams below are pullbacks, given that they are constructed by pasting two pullbacks together;

the right-hand side squares are pullbacks because of extensivity.



Now $(u_1 + u_2)\alpha i_{X_1 \to X} = (v_1 + v_2)\beta i_{X_1 \to X} = (v_1 + v_2) i_{C_1 \to C_1 + C_2} \beta_1 = i_{D_1 \to D_1 + D_2} v_1 \beta_1$ and we obtain a unique map $h_1 : X_1 \to A_1$ satisfying $g_1 h_1 = \beta_1$ and $i_1 f_1 h_1 = \alpha i_{X_1 \to X}$. Similarly, we obtain a unique $h_2 : X_2 \to A_2$ satisfying $g_2 h_2 = \beta_2$ and $i_2 f_2 h_2 = \alpha i_{X_2 \to X}$. Together they induce $h_1 + h_2 : X \to A_1 + A_2$ which is easily checked to satisfy $(g_1 + g_2)(h_1 + h_2) = \beta$ and $(f_1 + f_2)(h_1 + h_2) = \alpha$. Uniqueness of $h_1 + h_2$ follows easily from the uniqueness of the two components. □

4.2 Constructing GRPOs

In this section we shall construct GRPOs in two simple G-categories which correspond to the two running examples of Chapter 2.

The first of these is the G-category $\mathbf{M}_\Sigma$ of Example 2.2.4, the arrows of which model the terms of the simple process calculus introduced in Example 2.1.7. The construction shall actually be performed in a simpler G-category, which we denote **Prl**, to which there is a 2-functor from $\mathbf{M}_\Sigma$ which creates GRPOs. The second is the G-category of bunch contexts. We started in Definition 2.1.10 by identifying the algebra of bunch contexts. Bunch contexts can be composed, although the composition may not be associative on the nose (see Remark 2.1.11). We argued that isomorphic bunches should not be distinguished – this resulted in the definition of abstract bunch contexts (Definition 2.1.12) and an ordinary category $\mathbf{Bun}_0$ (Definition 2.1.13) with arrows the abstract bunch contexts.

Unfortunately, quotienting the arrows by isomorphism introduces some complications. Indeed, as we have demonstrated in Example 2.2.2, $\mathbf{Bun}_0$ does not have RPOs. The solution suggested in Chapter 2 is to keep the 2-dimensional structure of bunches - that is work with the G-category **Bun** (Definition 2.2.5) and construct GRPOs (Definition 2.2.9). We get the best of both worlds in that we can use this extra information to derive labels – as we shall show in this section, GRPOs do exist in **Bun** – but we can forget about the 2-dimensional information, when reasoning about dynamics, by using the abstract labelled transition system of Definition 2.2.16.

The example of bunches is important for us for several reasons. Firstly, it gives a simple test case for the theory of G-reactive systems and GRPOs. Secondly, it provides a nice example of the relationship between the theory of G-reactive systems and the various solutions proposed by Leifer and Milner; we shall exhibit this relationship in section 4.3. Thirdly, it is a nice "warm-up" for us before embarking on the more involved construction of GRPOs in cospan bicategories – the subject of Chapter 6.

Starting in §4.2.1 with the definition of the two G-categories we shall be dealing with in this section, **Prl** and **Bun**, we shall proceed with presenting a construction of GRPOs in **Prl** in §4.2.2 and finally, a construction of GRPOs in **Bun** in §4.2.3.

4.2.1 Prl and Bun

We have introduced concrete bunch context in Definition 2.1.10. Recall that such a bunch context of type $m_1 \to m_2$ consists of an ordered set of m_2 trees of depth one containing exactly m_1 holes. Leaves are labelled from an alphabet $\mathcal{K}$. These data represent m_2 bunches of unspecified controls (the leaves), together with m_1 places (the holes) where further bunch contexts can be plugged to.

Composition of two (composable) bunch contexts is performed by taking the disjoint union of the carrier sets of the two bunches. This leads to a problem – the operation of disjoint union is not associative on the nose but up to a canonical isomorphism.

For example, a standard ad-hoc way of defining disjoint union for sets is to let $A + B = \{ \langle a, 0\rangle \mid a \in A \} \cup \{ \langle b, 1\rangle \mid b \in B \}$. Then $(A + B) + C$ has elements of the form $\langle\langle a, 0\rangle, 0\rangle$, $\langle\langle b, 1\rangle, 0\rangle$ and $\langle c, 1\rangle$ while $A + (B + C)$ has elements of the form $\langle a, 0\rangle$, $\langle\langle b, 0\rangle, 1\rangle$ and $\langle\langle c, 1\rangle, 1\rangle$. Thus the sets $(A + B) + C$ and $A + (B + C)$, while being clearly bijective in a canonical way, are not equal.

However, this problem is easily solved by replacing finite sets with finite ordinals, or indeed, any category with a natural definition of an associative binary coproduct and a forgetful functor to the category of finite sets $\mathbf{Set}_f$.

Replacing the carrier set X with a finite ordinal x allows us to avoid the unnecessary burden of working in a bicategory, which would arise because sum on sets is only associative up to isomorphism. Observe that this simplification is harmless since the set-theoretical identity of the elements of the carrier is irrelevant. We remark, however, that GRPOs are naturally a bicategorical notion, being a local bicolimit (see Chapter 3). Thus they pose no particular challenge in the setting of bicategories. Indeed, in Chapter 6

we shall construct GRPOs in a particular bicategory of cospans.

Before restating the definition of the G-category of bunch contexts, we shall first recall the definition of the category of ordinals.

Definition 4.2.1 (Ord). Let **Ord** denote the category of finite ordinals. The objects of this category are the natural numbers $0, 1, 2, \ldots$. The morphisms from m to n are the all the functions from the ordered m-element set $[m] = \{0,\, 2, \ldots m-1\}$ to $[n] = \{0,\, 2, \ldots n-1\}$. Composition is the usual compositions of functions. The category is skeletal, in that we have $n \cong n'$ if and only if $n = n'$. We assume that **Ord** has chosen coproducts, namely ordinal addition $\oplus$. One possible way to define this is to let, on objects, $m \oplus n = m + n$, while on arrows, given $f : m \to m'$ and $g : n \to n'$, let $f \oplus g : m + n \to m' + n'$ be the function defined $(f \oplus g)(x) = f(x)$ for $0 \leq x < m$ and $(f \oplus g)(x) = g(x - m) + m'$ otherwise. Intuitively, $f + g$ is constructed by putting f and g side by side.

For any finite set x, let $\mathrm{ord}(x)$ be the finite ordinal of the same cardinality and $t_x \colon x \to \mathrm{ord}(x)$ be a chosen isomorphism. There is an equivalence of categories $F \colon \mathbf{Set}_f \to \mathbf{Ord}$. On objects it sends x to $\mathrm{ord}(x)$; on morphisms, it maps $f \colon x \to y$ to $t_y f t_x^{-1} \colon \mathrm{ord}(x) \to \mathrm{ord}(y)$.

Having illustrated a way of avoiding the problem of non-associativity, we are ready to give a full version of Definition 2.2.5 and introduce the G-category **Bun**. We shall first define a G-category which is simpler than **Bun**, and which we have seen already in the guise of $\mathbf{M}_\Sigma$ of Example 2.2.4.

Definition 4.2.2 (Prl). Let **Prl** denote the G-category with

- a single object $\bullet$;
- arrows the finite ordinals m, n, …, with composition defined $m \circ n = n \oplus m$;
- if $m \neq n$ then there are no 2-cells between m and n, otherwise all bijections $m \to m$.

Clearly 0, the empty ordinal, is the identity arrow of $\bullet$. Vertical composition is function composition. Horizontal composition is defined as follows: given $f : m \to m'$ and $g : n \to n'$, $gf : m \oplus n \to m' \oplus n'$ is defined by putting the functions "side by side"; see Example 2.2.4 for a more detailed presentation.

Definition 4.2.3 (Bun). The G-category of bunch contexts **Bun** has:

- objects the finite ordinals, denoted $m_1, m_2, \ldots$

- arrows $c = \langle x, \mathrm{ch}, \mathrm{rt}\rangle : m_1 \to m_2$ consist of a finite ordinal x, a surjective function $\mathrm{rt}\colon m_1 \oplus x \to m_2$ and a labelling function $\mathrm{ch}\colon x \to \mathcal{K}$.
- 2-cells α are isomorphisms between bunches' carriers which preserve the structure, that is respect ch and rt.

Composition of arrows is defined as follows: given bunch contexts $c_0 : m_1 \to m_2$ and $c_1 : m_2 \to m_3$, let $c_1c_0 : m_1 \to m_3 = \langle X, \mathrm{rt}, \mathrm{ch}\rangle$ where $X = X_0 \oplus X_1$, $\mathrm{rt} = \mathrm{rt}_1(\mathrm{rt}_0 \oplus \mathrm{id}_{X_1})$ and $\mathrm{ch} = [\mathrm{ch}_0, \mathrm{ch}_1]$ (see also Definition 2.1.10). Vertical composition of 2-cells is function composition, horizontal composition of $\alpha : c_0 \Rightarrow c_0' : m_1 \to m_2$ and $\beta : c_1 \Rightarrow c_1' : m_2 \to m_3$ is $\alpha \oplus \beta : c_1c_0 \Rightarrow c_1'c_0' : m_1 \to m_3$. Notice that since $\oplus$ is associative, composition in **Bun** is associative. Therefore **Bun** is a G-category.

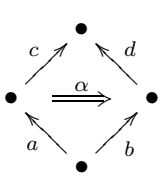
4.2.2 GRPOs in $\mathbf{M}_\Sigma$

When constructing GRPOs, we shall use general categorical constructions defined using universal properties. This not only simplifies the proofs, freeing one from unnecessary set-theoretical detail but also makes them more robust in that the proofs lift relatively easily to other models.

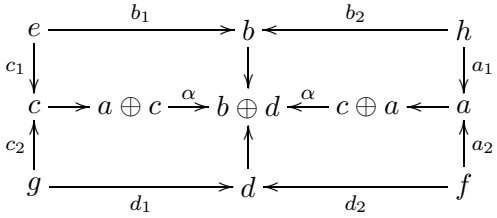
Theorem 4.2.4. Prl *has GRPOs.*

The proof is divided into two sections. Firstly we shall construct a candidate and secondly we shall show that this candidate satisfies the required universal property (see Definition 2.2.9). The only structure of the ordinals which we shall use is that **Ord** is an extensive category.

Construction of candidate. Consider diagram (i) below.



(i)



(ii)

Now consider diagram (ii) where the unlabelled morphisms are the canonical coproduct injections and e, f, g and h as well as a_i, b_i, c_i and d_i are chosen so as to make the four rectangles pullbacks. Using the fact that we are in an extensive category, we can deduce that each of the sides of the exterior

of diagram (*ii*) is a coproduct diagram. The intuition is that e is the set of elements common to c and b when related by α; with similar intuitions for f, g and h.

Using the fact that the sides of diagram (*ii*) are coproduct injections, we are able to deduce the existence of isomorphisms $\gamma = [c_1, c_2]^{-1} : c \to e \oplus g$, $\delta = [d_2, d_1] : f \oplus g \to d$ and $\beta : a \oplus e \to b \oplus f$ defined as the composite

$$a \oplus e \xrightarrow{[a_2,a_1]^{-1} \oplus e} f \oplus h \oplus e \xrightarrow{f \oplus [b_2,b_1]} f \oplus b \xrightarrow{tw} b \oplus f,$$

where tw is the canonical "twist" isomorphism.

We shall show that $\langle e, f, g, \beta, \gamma, \delta \rangle$ is a candidate for diagram (*i*). In order to do this, one must check that

$$(b \oplus \delta)(\beta \oplus g)(a \oplus \gamma) = \alpha \tag{4.3}$$

It is straightforward but tedious to check that equation (4.3) holds, we start by rearranging the equation

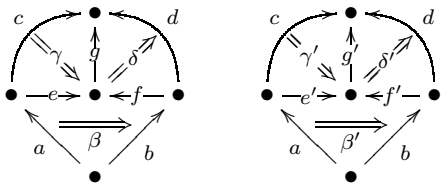
$$(b \oplus [d_2, d_1])(tw \oplus g)(f \oplus [b_2, b_1] \oplus g)([a_2, a_1]^{-1} \oplus e \oplus g)(a \oplus [c_1, c_2]^{-1}) = \alpha$$

into

$$(b \oplus [d_2, d_1])(tw \oplus g)(f \oplus [b_2, b_1] \oplus g) = \alpha(a \oplus [c_1, c_2])([a_2, a_1] \oplus e \oplus g)$$

and checking each of the four coproduct injections to $x_f \oplus x_h \oplus x_e \oplus x_g$. The equality on the first injection follows because $\alpha i_{x_a \to x_c \oplus x_a} a_2 = i_{x_d \to x_b \oplus x_d} d_2$, as shown in the lower right rectangle of diagram (*ii*). Similarly, the equality on the second, third and fourth injections follows, respectively, from the commutativity of the upper right, upper left and lower left rectangles of diagram (*ii*).

We have shown the existence of the candidate illustrated in diagram (*iii*).



(*iii*) (*iv*)

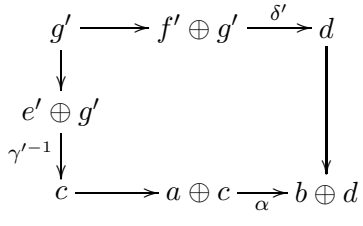
Verification of universal property. Suppose that

$$\left\langle e',\, f',\, g',\, \beta' : a \oplus e' \to b \oplus f',\, \gamma' : c \to e' \oplus g',\, \delta' : f' \oplus g' \to d \right\rangle,$$

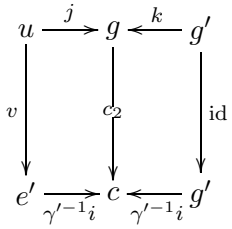
forms another candidate, as illustrated in diagram (iv). Then

$$(b \oplus \delta')(\beta' \oplus g')(a \oplus \gamma') = \alpha, \tag{4.4}$$

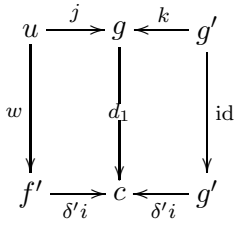
which rearranges to $(b \oplus \delta')(\beta' \oplus g') = \alpha(a \oplus \gamma'^{-1}) : a \oplus e' \oplus g' \to b \oplus d$. Precomposing this last equation with the injection $g \to a \oplus e' \oplus g'$ implies the commutativity of diagram (v).



(v)



(vi)

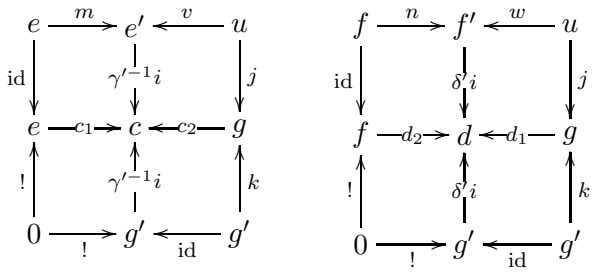


(vii)

Using the fact that the lower left rectangle of diagram (ii) is a pullback, we obtain an arrow $k : g' \to g$ which satisfies $c_2 k = \gamma'^{-1} i_{g' \to e' \oplus g'}$ and $d_1 k = \delta' i_{g' \to f' \oplus g'}$. The first of these equations implies that the right rectangle of diagram (vi) is commutative. Using the fact that injections are mono in extensive categories (see part (ii) of Lemma 4.1.3), $c_2 : g \to c$ is mono. This in turn, using Lemma 1.3.3, implies that the right rectangle of diagram (vi) is a pullback. We obtain an object u and arrows $j : u \to g$ and $v : u \to e'$ by taking the pullback of c_2 and $\gamma'^{-1} i_{e' \to e' \oplus g'}$.

Then the top row of diagram (vi) is a coproduct diagram, using extensivity. Using the second of k's defining equations and Lemma 1.3.3 we derive that the right rectangle of diagram (vii) is a pullback. Using extensivity, any pullback diagram of d_1 and $\delta' i_{f' \to f' \oplus g'}$ makes the top row of diagram (vii) a coproduct diagram. Since we already know that $u \xrightarrow{j} g \xleftarrow{k} g'$ is a coproduct diagram, we can use part (iii) of Lemma 4.1.3 which allows us to conclude that there exists an arrow $w : u \to f'$ so that the left rectangle of diagram (vii) is a pullback.

Let $\tau : u \oplus g' \to g$ be the isomorphism defined $\tau = [j, k]$. Now consider



(viii) *(ix)*

diagram *(viii)*. We know (see diagram *(vi)*) that the upper right and lower right squares are pullbacks. Because c_1 and c_2 form a coproduct diagram, extensivity implies that lower left square is a pullback. Using the fact that $\gamma'^{-1}i_{e' \to e' \oplus g'}$ and $\gamma'^{-1}i_{g' \to e' \oplus g'}$ form a coproduct diagram, and part (iii) of Lemma 4.1.3 implies that also the upper left square is a pullback, for some $m : e \to e'$. Moreover, the top row is a coproduct diagram. Repeating all this for diagram *(ix)* allows us to obtain $n : f \to f'$ which makes the upper left square a pullback and the top row a coproduct diagram.

Define $\varphi = [m, v]^{-1} : e' \to e \oplus u$ and $\psi = [n, w] : f \oplus u \to f'$. To check

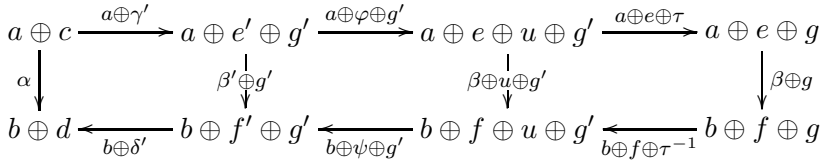
$$(e \oplus \tau)(\varphi \oplus g')\gamma' = \gamma, \tag{4.5}$$

we rearrange the equation to $[c_1, c_2]^{-1}\gamma'^{-1}([m, v] \oplus g') = e \oplus [j, k] : e \oplus u \oplus g' \to e \oplus g$ and check each of the three injections. Equality on the first injection follows from the commutativity of the upper left square of diagram *(viii)*, on the second injection from the upper right square and on the third injection from the lower right. One shows that the corresponding equation

$$\delta'(\psi \oplus g')(f \oplus \tau^{-1}) = \delta, \tag{4.6}$$

relating δ and δ', holds in a similar way, using the commutativity of diagram *(ix)*.

Finally, consider diagram (x) below, the exterior of which is commutative owing to equation (4.3).



(x)

Equation (4.4) implies that the leftmost region is commutative and the rightmost region is easily seen to be commutative. Because all of the components of the diagram are isomorphisms, we may conclude that the middle region of the diagram is commutative, that is $(b\oplus\psi\oplus g')(\beta\oplus u\oplus g')(a\oplus\varphi\oplus g') = \beta'\oplus g'$. Using Lemma 4.1.4, we obtain

$$(b \oplus \psi)(\beta \oplus u)(a \oplus \varphi) = \beta' \tag{4.7}$$

which proves that $\langle u, \varphi, \psi, \tau\rangle$ is a mediating morphism.

It remains to show essential uniqueness. Indeed, suppose that the components of $\langle u', \varphi', \psi', \tau'\rangle$ constitute another mediating morphism. Then

$$(e \oplus \tau')(\varphi' \oplus g')\gamma' = \gamma, \tag{4.8}$$

$$\delta'(\psi' \oplus g')(f \oplus \tau'^{-1}) = \delta \tag{4.9}$$

and $(b \oplus \psi')(\beta \oplus u')(a \oplus \varphi) = \beta'$. Recall from Definition 2.2.9 that we need to show that there exists a unique $\xi : u \to u'$ such that

$$(e \oplus \xi)\varphi = \varphi', \tag{4.10}$$

$$\psi(f \oplus \xi^{-1}) = \psi' \text{ and} \tag{4.11}$$

$$\tau'(\xi \oplus g') = \tau. \tag{4.12}$$

We start by noticing that equations (4.5) and (4.8) together imply that

$$(e \oplus \tau)(\varphi \oplus g') = (e \oplus \tau')(\varphi' \oplus g'). \tag{4.13}$$

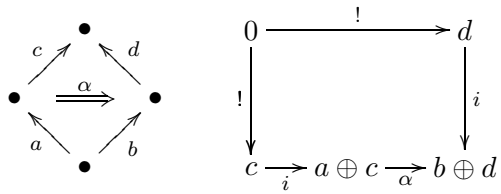
Precomposing this equation with $i_{g'\to e'\oplus g'}$ gives $\tau' i_{g'\to u'\oplus g'} = \tau i_{g'\to u\oplus g'} = k$. In particular, we have two coproduct diagrams:

$$u \xrightarrow{\ j\ } g \xleftarrow{\ k\ } g' \qquad \text{and} \qquad u' \xrightarrow{\ \tau' i_{u'\to u'\oplus g'}\ } g \xleftarrow{\ k\ } g'.$$

Using part (iii) of Lemma 4.1.3, we obtain a unique isomorphism $\xi : u \to u'$ such that $\tau' i_{u'\to u'\oplus g'}\xi = j$, and therefore equation (4.12) holds. Substituting equation (4.12) into equation (4.13) yields $(e \oplus \tau')(e \oplus \xi \oplus g')(\varphi \oplus g') = (e \oplus \tau')(\varphi' \oplus g')$ which one can manipulate to get $((e \oplus \xi)\varphi) \oplus g' = \varphi' \oplus g'$. A straightforward application of Lemma 4.1.4 yields equation (4.10) One derives equation (4.11) in a similar way, using equations (4.6) and (4.9) in the process. □

There is a simple characterisation of GIPOs in **Prl**.

Lemma 4.2.5. [GIPOs in **Prl**] Diagram (i), below, is a GIPO if and only



(i) (ii)

if diagram (ii) is a pullback.

Proof. If diagram (ii) is a pullback then the GRPO construction of Theorem 4.2.4 yields the identity candidate. On the other hand, starting with a diagram (i) which is a GIPO, notice that any candidate must be of the form $\langle e, f, 0, \beta, \gamma, \delta\rangle$ since 0 is the only arrow which factorises 0 in **Prl**. Using the fact that we construct the components of the candidate in the proof of Theorem 4.2.4 by taking pullbacks, this means that the pullback of $\alpha i_{c\to a\oplus c}$ and $i_{d\to b\oplus d}$ must be 0. □

Corollary 4.2.6. $\mathbf{M}_\Sigma$ has GRPOs.

Proof. There is an obvious 2-functor which takes $\bullet$ to $\bullet$, $a_0 \mid \ldots \mid a_{n-1}$ to the ordinal n and acts as an identity on the 2-cells. Thus the functor can be said to forget names. It is easy to see that this functor creates GRPOs, in the sense that one can construct the required candidate in **Prl** and reintroduce names in a unique way to obtain a candidate in $\mathbf{M}_\Sigma$ which satisfies the required universal property. □

4.2.3 GRPOs in Bun

In the proof of Theorem 4.2.7 below, we use only the fact that *Ord* is an extensive category with pushouts.

Theorem 4.2.7. Bun *has GRPOs.*

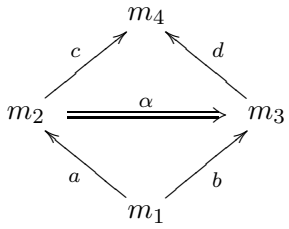
The proof is only slightly more involved than the proof of Theorem 4.2.4. Indeed, the candidate is constructed in the same way, the extra work involved is in showing that all the arrows constructed in that proof can be given the structure of a bunch contexts – i.e. the root function rt and the character function ch.

Another way of saying the above is that the 2-functor **Bun** $\to$ **Prl**, which sends all objects to $\bullet$, sends a bunch context $c : m_0 \to m_1 =$

$\langle x_c, \mathrm{ch}_c : x_c \to \mathcal{K}, \mathrm{rt}_c : m_0 \oplus x_c \to m_1 \rangle$ is sent to x_c while bunch homomorphisms are sent to the their underlying functions, creates GRPOs.

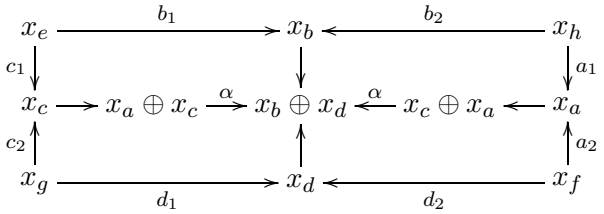
The proof is divided into two parts. In the first part we construct the required candidate and in the second part we verify that the universal property holds.

Construction of candidate. Suppose that we have an isomorphic 2-cell $\alpha\colon ca \Rightarrow db$ as illustrated below.



The intuition here is that, for an 'agent' a and a left hand side b of some reaction rule, we are given bunch contexts c and d so that ca is db, up to α (in symbols, $ca \cong_\alpha db$). We shall find the smallest upper bound of a and b which 'respects' α.

Using $\alpha : x_a \oplus x_c \to x_b \oplus x_d$ we take four pullbacks obtaining the following diagram. (as in diagram (ii) in the proof of Theorem 4.2.4)



(i)

We shall show that x_e and x_f form the nodes of the minimal candidate.

Using the morphisms from the diagram above as building blocks, we can construct bijections $\gamma\colon x_c \to x_e \oplus x_g$, $\delta\colon x_f \oplus x_g \to x_d$ and $\beta\colon x_a \oplus x_e \to x_b \oplus x_f$ such that

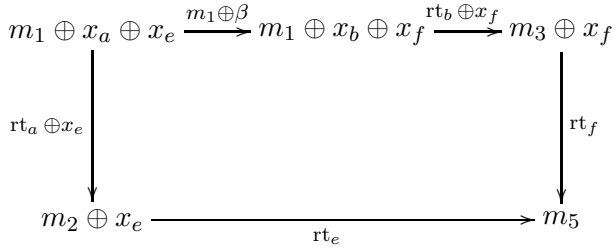
$$x_b \oplus \delta.\beta \oplus x_g.x_a \oplus \gamma = \alpha, \tag{4.14}$$

more precisely, $\gamma = [c_1, c_2]^{-1}$, $\delta = [d_2, d_1]$ and β is the following composition,

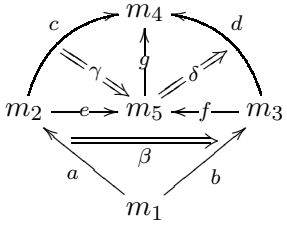
$$x_a \oplus x_e \xrightarrow{[a_2,a_1]^{-1} \oplus x_e} x_f \oplus x_h \oplus x_e \xrightarrow{x_f \oplus [b_2,b_1]} x_f \oplus x_b \xrightarrow{tw} x_b \oplus x_f$$

where $tw : x_f \oplus x_b \to x_b \oplus x_f$ is the 'twist' isomorphism. One checks that the equation (4.14) holds in the same way as in the proof of Theorem 4.2.4.

Let m_5 and morphisms $\mathrm{rt}_e : m_2 \oplus x_e \to m_5$ $\mathrm{rt}_f : m_2 \oplus x_f \to m_5$ be such so that diagram (ii), below, a pushout diagram.



(ii)

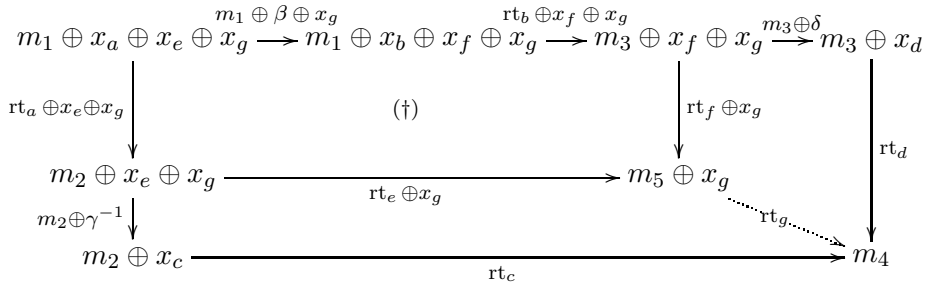


(iii)

We can then define $\mathrm{ch}_e = \mathrm{ch}_c\, c_1$, $\mathrm{ch}_f = \mathrm{ch}_d\, d_2$ and $\mathrm{ch}_g = \mathrm{ch}_c\, c_2$. Notice that the commutativity of diagram (ii) implies that β is a bunch homomorphism.

In order to form bunch contexts e, g and f which make diagram (iii), above, a candidate, it remains to define rt_g and prove that γ and δ are bunch homomorphisms.

Consider diagram (iv), below.



(iv)

The exterior of diagram (iv) is commutative since α is a bunch homomorphism, this can be verified by precomposing with $m_1 \oplus x_a \oplus \gamma : m_1 \oplus x_a \oplus x_c \to m_1 \oplus x_a \oplus x_e \oplus x_g$ and using (4.14). Now, since digram (ii) is a pushout, an application of Lemma 1.3.4 yields that (†) is a pushout. We obtain a morphism $\mathrm{rt}_g : m_5 \oplus x_g \to m_4$ which makes the remaining regions of diagram (iv) commute. The commutativity of these two regions amounts means that γ and δ are bunch homomorphisms. We can deduce that rt_g is epi since $\mathrm{rt}_g \,.\, \mathrm{rt}_f \oplus x_g = \mathrm{rt}_d \,.\, m_3 \oplus \delta$, rt_d is epi and $m_3 \oplus \delta$ is an isomorphisms.

Thus, diagram (ii) is indeed a candidate for the 2-cell $\alpha\colon ca \Rightarrow db$.

Verification of universal property. Suppose that $\langle m_6,\, r,\, s,\, t,\, \beta',\, \gamma',\, \delta' \rangle$ is another candidate for α, i.e. $\delta' b \bullet t\beta' \bullet \gamma' a = \alpha$.

Using the same process as in the proof of Theorem 4.2.4 we obtain x_u and coproduct diagrams

$$x_u \xrightarrow{j} x_g \xleftarrow{k} x_{g'},\ x_f \xrightarrow{n} x_{f'} \xleftarrow{w} x_u \text{ and } x_e \xrightarrow{m} x_{e'} \xleftarrow{v} x_u.$$

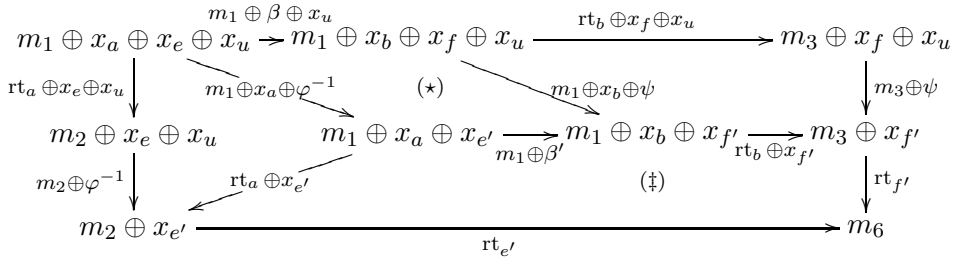
From these coproduct diagrams one defines isomorphisms $\tau : x_u \oplus x_{g'} \to x_g$, $\varphi : x_{e'} \to x_e \oplus x_u$ and $\psi : x_f \oplus x_u \to x_{f'}$ which satisfy the required equations:

$$(x_e \oplus \tau)(\varphi \oplus x_{g'})\gamma' = \gamma, \tag{4.15}$$

$$\delta'(\psi \oplus x_{g'})(x_f \oplus \tau^{-1}) = \delta \text{ and} \tag{4.16}$$

$$(x_b \oplus \psi)(\beta \oplus x_u)(x_a \oplus \varphi) = \beta'. \tag{4.17}$$

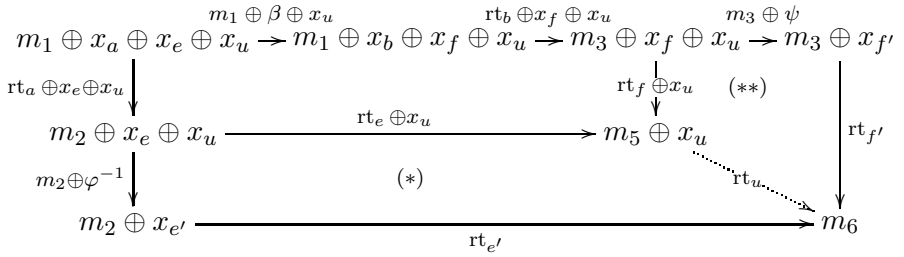
In order to show that $\langle u : m_5 \to m_6, \varphi, \psi, \tau \rangle$ is a mediating morphism it remains to show that x_u can be equipped with appropriate structure which makes it into a bunch context, and that φ, ψ and τ are bunch homomorphisms.



(x)

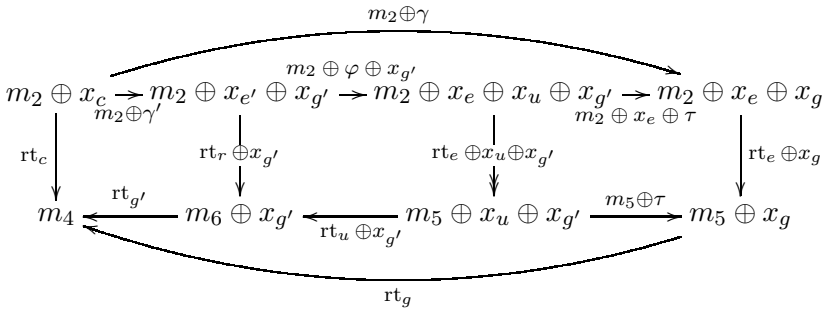
Consider diagram (x), above. The commutativity of region $(\star)$ follows from equation (4.17). Region $(\ddagger)$ is commutative because β' is a bunch homomorphism. Thus, the exterior of the diagram is commutative. The commutativity of diagram (x) implies that the exterior of diagram (xi) is commutative. Applying the conclusion of Lemma 1.3.4 to diagram (ii) implies that the inner region is a pushout diagram, and therefore, that there exists a unique morphism $\mathrm{rt}_g : m_5 \oplus x_g \to m_4$ which renders regions (*) and (**) commu-

tative.



(xi)

Thus $u\colon m_5 \to m_6$ is a bunch context. Regions (*) and (**) of digram (xi) imply that $\varphi\colon e' \Rightarrow ue$ and $\psi\colon uf \Rightarrow f'$, respectively, are homomorphisms. To see that $\tau\colon g'u \Rightarrow g$ is a homomorphism, consider diagram (xii), below.



(xii)

The three rectangles are commutative since, the leftmost since γ' is a homomorphism, the middle since φ is a homomorphism and the rightmost for trivial reasons. Using equation (4.15), the top row is equal to $m_2 \oplus \gamma$. Using the fact that γ is a homomorphism, we conclude that the exterior is commutative. Finally, the fact that the marked arrow is epi, we conclude that $\mathrm{rt}_g(m_5 \oplus \tau) = \mathrm{rt}_{g'}(\mathrm{rt}_u \oplus x_{g'})$. Thus $\langle u, \varphi, \psi, \tau\rangle$ is a mediating morphism.

Now consider any other mediating morphism $\langle u', \tau', \varphi', \psi'\rangle$. We have that

$$(x_e \oplus \tau')(\varphi' \oplus x_{g'})\gamma' = \gamma, \tag{4.18}$$

$$\delta'(\psi' \oplus x_{g'})(x_f \oplus \tau'^{-1}) = \delta \text{ and} \tag{4.19}$$

$(x_b \oplus \psi')(\beta \oplus x_{u'})(x_u \oplus \varphi') = \beta'$. Using the process described in the proof of Theorem 4.2.4, we obtain a unique $\xi : x_u \to x_{u'}$ which satisfies $\tau'(\xi \oplus x_{g'}) = \tau$, $\varphi' = (x_e \oplus \xi)\varphi$ and $\psi(x_f \oplus \xi^{-1}) = \psi'$. It remains only to check that ξ is

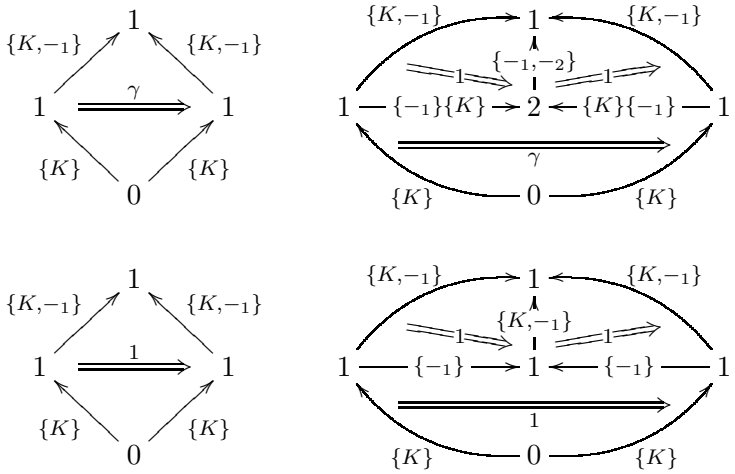
a homomorphism, that is $\mathrm{rt}_u = \mathrm{rt}_{u'}(m_5 \oplus \xi)$. We do this by showing that $\mathrm{rt}_{u'}(m_5 \oplus \xi)$ also serves as the indicated morphism in diagram (xi); equality then follows from the universal property of pushouts. Indeed,

$$\begin{aligned}
\mathrm{rt}_{u'}(m_5 \oplus \xi)(\mathrm{rt}_f \oplus x_u) &= \mathrm{rt}_{u'}(\mathrm{rt}_f \oplus x_{u'})(m_3 \oplus x_f \oplus \xi) \\
&= \mathrm{rt}_{f'}(m_3 \oplus \psi')(m_3 \oplus x_f \oplus \xi) \qquad (\psi' \text{ homo.}) \\
&= \mathrm{rt}_{f'}(m_3 \oplus \psi) \qquad (4.19)
\end{aligned}$$

Similarly, using the fact that φ' is a homomorphisms and equation (4.18) yields that $\mathrm{rt}_{u'}(m_5 \oplus \xi)(\mathrm{rt}_e \oplus x_u) = \mathrm{rt}_{e'}(m_2 \oplus \varphi^{-1})$. □

Recall that in Example 2.2.2 we showed that $\mathbf{Bun}_0$, the category of abstract bunch contexts obtained by quotienting the arrows of **Bun** by isomorphism, does not have RPOs. We demonstrated this by considering to candidates which had no common smaller candidate. It turns out that both of those candidates are actually minimal, in the sense that they are GRPOs.

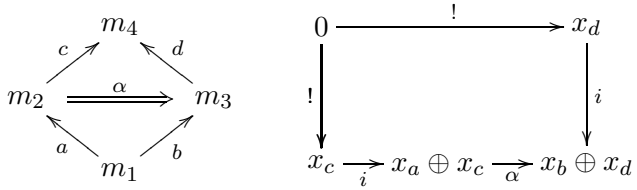
Example 4.2.8. Let $\gamma\colon 2 \to 2$ be the function taking $1 \mapsto 2$ and $2 \mapsto 1$. We give below on the right the GRPOs for the squares on the left.



The ambiguity in $\mathbf{Bun}_0$ about 'how' the diagrams commute – which ultimately leads to $\mathbf{Bun}_0$ failing to have RPOs – is resolved here by the explicit presence of 2-cells 1 or γ.

Finally, we remark that **Bun** clearly inherits the simple characterisation of GIPOs from **Prl** (See Lemma 4.2.5).

Proposition 4.2.9. [GIPOs in **Bun**] Diagram (i), below, is a GIPO if and



(i) (ii)

only if diagram (ii) is a pullback.

4.3 2-categories vs S-precategories

In this section we shall compare the theory of G-reactive systems and GRPOs, developed in Chapter 2, and the theory introduced by Leifer [64] and refined by Jensen and Milner [46] which also deals with the problems of RPOs not existing when the 2-dimensional structure of categories is forgotten.

Leifer, Jensen and Milner's solution is to decorate the categorical structure – either objects, as in Leifer's *functorial reactive systems*, or arrows, as in Jensen and Milner's *S-precategories* – with sets. The idea is that, starting with some structure which has an underlying carrier set, the extra set theoretical information allows one to specify precisely the identity of the various components; avoiding problems such as Example 2.2.2. Because of the extra book-keeping introduced, there are complications. For example, in the theory of S-precategories, ordinary composition of arrows is not always defined – it is defined only if the sets associated with the arrows to be composed are disjoint. Interestingly, Leifer and Milner were already aware of these problems when they introduced the theory of reactive systems [66] – they present the category of bunch contexts in the style of Leifer's *track* category (see §4.3.3).

The main contribution of this section is the proof that this arguably ad-hoc theory can be actually seen as an instance of the general theory introduced in Chapter 2. This is meant in a strong sense - given a functorial reactive system or an S-precategory, one can perform the translation specified in this section and obtain a G-category. This translation is, of course, well-behaved meaning that the labelled transition systems proposed by Leifer and Milner are identical to the abstract labelled transition system (Definition 2.2.16) obtained using GRPOs.

We shall start in §4.3.1 with the definition of S-precategory and related technology. In §4.3.2 we prove the main result of this section, that the theory of S-precategories is a special case of the theory of G-categories. In §4.3.3 we briefly recall Leifer's functorial reactive systems, which is also easily seen to fit within the framework of Chapter 2.

4.3.1 S-precategories

Other categories which, besides $\mathbf{Bun}_0$, lack RPOs include the categories of closed *shallow action contexts* [64, 65] and *bigraph contexts* [73, 46]. The solution adopted by Milner [46] is to introduce a notion of an *S-precategory*, where the algebraic structures at hand are decorated by finite 'support sets.' The result is no longer a category – since composition of arrows is defined only if their supports are disjoint.

In this section we present a translation from arbitrary S-precategories to G-categories. Our main result shows that the lts derived using precategories and functorial reactive systems is identical to the (abstract) lts of Definition 2.2.16 derived using GRPOs. We begin with a brief recapitulation of the definitions from [65], to which the reader is referred for further details.

In the following we shall use the symbol $\uplus$ to mean set-theoretical union when we additionally know a priori that the sets under consideration are disjoint.

Definition 4.3.1 (Precategory). A *precategory* $\acute{}\mathbf{A}$ consists of the same data as a category. The composition operator $\circ$ is, however, a partial function which satisfies:

1. for any arrow $f : A \to B$, $\mathrm{id}_B \circ f$ and $f \circ \mathrm{id}_A$ are defined and $\mathrm{id}_B \circ f = f = f \circ \mathrm{id}_A$;
2. for any $f : A \to B$, $g : B \to C$, $h : C \to D$, $(h \circ g) \circ f$ is defined iff $h \circ (g \circ f)$ is defined and then $(h \circ g) \circ f = h \circ (g \circ f)$.

Definition 4.3.2 (S-precategory). Let $\mathbf{Set}_f$ be the category of finite sets. A supported precategory, or *S-precategory* is a pair $\langle \acute{}\mathbf{A}, |-| \rangle$, where $\acute{}\mathbf{A}$ is a precategory and $|-|$ is a map from the arrows of $\acute{}\mathbf{A}$ to $\mathbf{Set}_f$, the so-called support function, satisfying:

1. $g \circ f$ is defined iff $|g| \cap |f| = \varnothing$, and if $g \circ f$ is defined then $|g \circ f| = |g| \uplus |f|$;
2. $|\,\mathrm{id}_A\,| = \varnothing$.

For any $f : A \to B$ and any injective function α in $\mathbf{Set}_f$ the domain of which contains $|f|$ there exists an arrow $\alpha{\cdot}f : A \to B$ called the *support translation* of f by α. The following axioms are to be satisfied.

1. $\alpha{\cdot}\,\mathrm{id}_A = \mathrm{id}_A$;
2. $\mathrm{id}_{|f|}{\cdot}f = f$;
3. $\alpha_0|f| = \alpha_1|f|$ implies $\alpha_0{\cdot}f = \alpha_1{\cdot}f$;
4. $\alpha{\cdot}(g \circ f) = \alpha{\cdot}g \circ \alpha{\cdot}f$;
5. $(\alpha_1 \circ \alpha_0){\cdot}f = \alpha_1{\cdot}(\alpha_0{\cdot}f)$;
6. $|\alpha{\cdot}f| = \alpha|f|$.

We illustrate these definitions giving a definition of the S-precategory of bunch contexts. The reader can compare this to Definition 4.2.3 of the G-category of bunch contexts.

Example 4.3.3. The precategory of bunch contexts **ABun** has objects as in **Bun** (Definition 4.2.3), namely finite ordinals. Also as in **Bun**, arrows are concrete bunch contexts. The support of $c = \langle X, \mathrm{ch}, \mathrm{rt}\rangle$ is X. Composition $c_1c_0 = \langle X, \mathrm{ch}, \mathrm{rt}\rangle : m_1 \to m_3$ of $c_0 = \langle X_0, \mathrm{ch}_0, \mathrm{rt}_0\rangle : m_1 \to m_2$ and $c_1 = \langle X_1, \mathrm{ch}_1, \mathrm{rt}_1\rangle : m_2 \to m_3$ is defined if and only if $X_0 \cap X_1 = \varnothing$ and, if so, we have $X = X_0 \uplus X_1$. Functions ch and rt are defined as in **Bun**, with X taking the place of $X_0 \oplus X_1$. The identity arrows are the same as in **Bun**. Given an injective function $\alpha\colon X \to Y$, the support translation $\alpha{\cdot}c$ is $\big\langle \alpha X, \mathrm{ch}\ \alpha^{-1}, \mathrm{rt}\,(\mathrm{id}_{m_1} + \alpha^{-1})\big\rangle$. It is easy to verify that this satisfies the axioms of precategories.

We shall now recall the construction of the support quotient category from a S-precategory ´**A**. The support quotient category "forgets" the support information, in much the same way the functor $(-)_0 : G\text{-}\mathbf{Cat} \to \mathbf{Cat}$, defined in Chapter 2, forgets the 2-dimensional structure of a G-category.

Definition 4.3.4 (Support quotient). The *support quotient* of ´**A** is a category **A** with

- objects: as in ´**A**;
- arrows: equivalence classes $[f]$ of arrows of ´**A**, where f and g are equated if there exist a bijective α such that $\alpha{\cdot}f = g$.

Example 4.3.5. The support quotient of **ABun** is $\mathbf{Bun}_0$ of Definition 2.1.13.

There is a straightforward way of defining a reactive system over an S-precategory, akin to the definition of G-reactive system for a G-category (Definition 2.2.6). For the reader familiar with Jensen and Milner's development [46], we do not treat the *wide* aspects of the theory – we shall concentrate on narrow contexts and stay in the realm of ordinary reactive systems. However, it appears that this development is independent of the translation from S-categories and G-categories presented in §4.3.2.

Definition 4.3.6 (Reactive system). A *reactive system* $\mathbb{A}$ over an S-precategory $\acute{\mathbf{A}}$ consists of

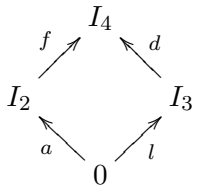
1. a collection $\mathbf{D}$ of arrows of $\acute{\mathbf{A}}$, the reactive contexts; it is required to be closed under support translation and to be composition-reflecting,
2. a distinguished object $0 \in \acute{\mathbf{A}}$,
3. a set of pairs $\mathcal{R} \subseteq \bigcup_{A \in \acute{\mathbf{A}}} \acute{\mathbf{A}}(0, A) \times \acute{\mathbf{A}}(0, A)$ called the reaction rules. These are required to be pointwise closed under support translation, that is, given $\langle l, r \rangle \in \mathcal{R}$ and support translations α, α' whose domains contain respectively $|l|$ and $|r|$, we require that $\langle \alpha \bullet l, \alpha' \bullet r \rangle \in \mathcal{R}$.

Notice that a reactive system $\mathbb{A}$ over a precategory $\acute{\mathbf{A}}$ translates in the obvious way to an ordinary reactive system over the support quotient category $\mathbf{A}$.

The notions of RPO and IPO (see §2.1.3) can be easily extended to precategories, see Jensen and Milner [46]. Thus one can define an lts using IPOs.

Definition 4.3.7 (LTS). Let $\mathbb{A}$ be a reactive system over an S-precategory $\acute{\mathbf{A}}$. The lts STS($\mathbb{A}$) has

- States: arrows $[a] \colon 0 \to I_2$ in $\mathbf{A}$;
- Transitions: $[a] \xrightarrow{[f]} [dr]$ if and only if $\langle l, r \rangle \in \mathcal{R}$, $d \in \mathbf{D}$, and



is an IPO in $\acute{\mathbf{A}}$.

It is shown in [46] that if the underlying S-precategory $\acute{\mathbf{A}}$ of a reactive system $\mathbb{A}$ has enough RPOs then bisimilarity on STS($\mathbb{A}$) is a congruence. This result is akin to Corollary 2.3.7, in which we proved that bisimilarity on the abstract lts (Definition 2.2.16) is a congruence.

4.3.2 S-precategories are G-categories

All the theory presented so far can be elegantly assimilated into the theory of GRPOs. In [65], Leifer predicted that instead of precategories, one could

consider a bicategorical notion of RPO in a bicategory of supports. This is indeed the case, with GRPOs being the bicategorical notion of RPO. However, working with ordinals for support sets we can avoid bicategories and, as in the case of **Bun**, stay within the realm of 2-categories. It is worth noticing, however, that a bicategory of supports as above and the G-category we introduce below would be (bi)equivalent (in the precise sense of [100]).

Definition 4.3.8 (G-Category of Supports). Given an S-precategory $\acute{\mathbf{A}}$, the G-category of supports $\mathbf{B}$ has

- objects: as in $\acute{\mathbf{A}}$;
- arrows: $f\colon A \to B$, where $f\colon A \to B$ is an arrow of $\acute{\mathbf{A}}$ and $|f|$ (the support) is an ordinal;
- 2-cells: $\alpha\colon f \Rightarrow g$ when α is a 'structure preserving' support bijection, i.e. $\alpha_{\bullet} f = g$ in $\acute{\mathbf{A}}$.

Composition is defined as follows. Given $f : A \to B$ and $g : B \to C$,

$$g \circ_{\mathbf{B}} f = i_{|g| \to |f| \oplus |g|}{}_{\bullet} g \circ_{\acute{\mathbf{A}}} i_{|f| \to |f| \oplus |g|}{}_{\bullet} f$$

where $|f| \to |f| \oplus |g| \leftarrow i_2|g|$ is the chosen coproduct diagram in **Ord**. Recall that, as stated in Definition 4.2.1, we assume that for every finite set x we have a chosen bijection $t : x \to \mathrm{ord}(x)$, where $\mathrm{ord}(x)$ is the unique finite ordinal with x's cardinality. Given an arrow f in $\acute{\mathbf{A}}$, we use $\widetilde{f} = t_{|f|}{}_{\bullet} f$ in $\mathbf{B}$, the 'canonical representative' of f in $\mathbf{B}$. To simplify the notation in the following we write t_f for $t_{|f|}$. Observe that, with these conventions, $t_f\colon |f| \to |\widetilde{f}|$.

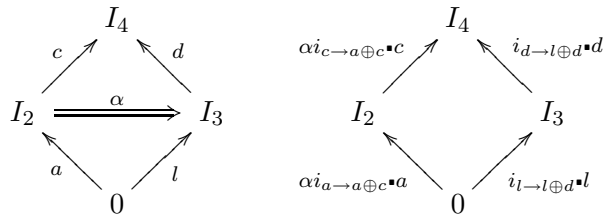
Notice that the translation can be easily extended to reactive systems. That is, starting with a reactive system $\mathbb{A}$ over an S-precategory $\acute{\mathbf{A}}$, one uses the translation of Definition 4.3.8 to obtain a G-reactive system $\mathbb{B}$ over the G-category of supports $\mathbf{B}$.

The following theorem guarantees that the lts generated is the same as the one generated with the theory of functorial reactive systems. Unfortunately, the proof of this theorem, while trivial, is quite awkward (and therefore, technical) because of the clash of philosophies: the idea that we should compare algebraic structures through the set theoretical identity of their underlying carriers – championed by the idea of S-precategories versus the idea that they should be compared via a natural notion of homomorphism, suggested by the G-categorical point of view.

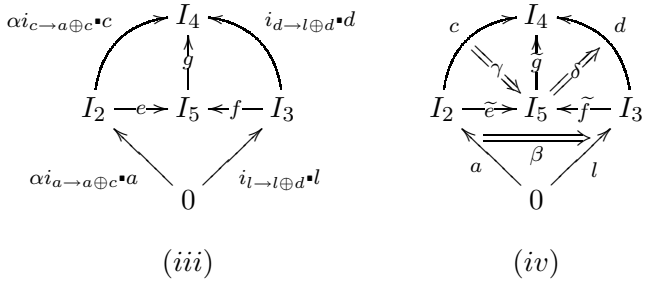
Theorem 4.3.9. *Let* $\mathbb{A}$ *be a reactive system over a supported precategory* $\acute{}\mathbf{A}$*, and let* $\mathbb{B}$ *and* $\mathbf{B}$ *be respectively the G-reactive system and G-category obtained as above. Then,* $\mathrm{STS}(\mathbb{A}) = \mathrm{ATS}(\mathbb{B})$.

Proof. It is enough to present a translation between GIPOs in $\mathbf{B}$ and IPOs in $\acute{}\mathbf{A}$ which preserves the resulting label in the derived lts. Although the details appear technical, there is nothing deep going on – it is just a series of translations between support translations and 2-cells.

Suppose that diagram (i), below, is a GIPO.



Then we claim that digram (ii) above is an IPO in $\acute{}\mathbf{A}$. Note that (ii) is commutative since α is by definition a structure-preserving support bijection and, therefore, $(\alpha i_{c\to a\oplus c}{\scriptstyle\bullet} c) \circ (\alpha i_{a\to a\oplus c}{\scriptstyle\bullet} a) = \alpha{\scriptstyle\bullet}(i_{c\to a\oplus c}{\scriptstyle\bullet} c \circ i_{a\to a\oplus c}{\scriptstyle\bullet} a) = (i_{d\to l\oplus d}{\scriptstyle\bullet} d) \circ (i_{l\to l\oplus d}{\scriptstyle\bullet} l)$.



Suppose that $\langle I_5, e, f, g\rangle$ is a candidate for diagram (ii), as illustrated in diagram (iii). We shall show how to find β, γ and δ such that the components of $\left\langle I_5, \widetilde{e}, \widetilde{f}, \widetilde{g}, \beta, \gamma, \delta \right\rangle$ form a candidate for diagram (i), as illustrated in diagram (iv). This amounts to showing that β, γ, and δ are structure-preserving bijections and that they paste together to give α.

In the following, in order to simplify notation, we shall leave out the support function, that is, we shall write a instead of $|a|$. Let β represent the

following composite

$$a \oplus \widetilde{e} \xrightarrow{[\alpha i_{a\to a\oplus c},\, t_e^{-1}]} (\alpha i_{a\to a\oplus c})\text{·}a \uplus e = e \circ (\alpha i_{a\to a\oplus c}\text{·}a)$$
$$= f \circ (i_{l\to l\oplus d}\text{·}l) = (i_{l\to l\oplus d})\text{·}l \uplus f \xrightarrow{[i_{l\to l\oplus \widetilde{f}}\, i_{l\to l\oplus d}^{-1},\, i_{\widetilde{f}\to l\oplus \widetilde{f}}\, t_f]} l \oplus \widetilde{f},$$

and similarly let γ and δ be respectively

$$c \xrightarrow{\alpha i_{c\to a\oplus c}} \alpha i_{c\to a\oplus c}\text{·}c = g \circ e = e \uplus g \xrightarrow{[i_{\widetilde{e}\to \widetilde{e}\oplus \widetilde{g}}\, t_e,\, i_{\widetilde{g}\to \widetilde{e}\oplus \widetilde{g}}\, t_g]} \widetilde{e} \oplus \widetilde{g}$$

and

$$\widetilde{f} \oplus \widetilde{g} \xrightarrow{[t_f^{-1},\, t_g^{-1}]} f \uplus g = g \circ f = i_2 \circ d \xrightarrow{i_2^{-1}} d$$

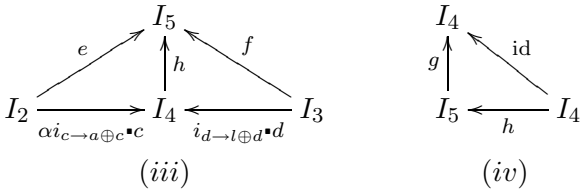
It is easy to check that the pasting composite of γ, β and δ yields α:

$$(l \oplus \delta)(\beta \oplus \widetilde{g})(a \oplus \gamma) = \alpha : a \oplus c \Rightarrow l \oplus d.$$

We shall now show that γ is a structure-preserving bijection, which means that it is a 2-cell in $\mathbf{B}$. The argument for the other morphisms is similarly trivial. Since $\alpha i_{c\to a\oplus c}\text{·}c = g \circ e$ we have $[i_{\widetilde{e}\to \widetilde{e}\oplus \widetilde{g}}\, t_e,\, i_{\widetilde{g}\to \widetilde{e}\oplus \widetilde{g}}\, t_g]\alpha i_{c\to a\oplus c}\text{·}c = [i_{\widetilde{e}\to \widetilde{e}\oplus \widetilde{g}}\, t_e,\, i_{\widetilde{g}\to \widetilde{e}\oplus \widetilde{g}}\, t_g]\text{·}g \circ e$ and so $\gamma\text{·}c = \widetilde{g} \circ \widetilde{e}$.

Thus $\left\langle I_5,\, \widetilde{e},\, \widetilde{f},\, \widetilde{g},\, \beta,\, \gamma,\, \delta \right\rangle$ is a candidate for diagram (i). Thus there exists $h\colon I_4 \to I_5$ and 2-cells (structure-preserving support bijections) $\varphi\colon \widetilde{e} \Rightarrow hc$, $\psi\colon hd \Rightarrow \widetilde{f}$ and $\tau\colon \widetilde{g}h \Rightarrow \mathrm{id}_{I_4}$.

From the existence of τ and the definition of supported category, we can deduce that $|\widetilde{g}| = |g| = \varnothing$ and $|h| = \varnothing$. Note that $\tau = \mathrm{id}$, since there is only one endofunction on $\varnothing$. We can therefore conclude that $\widetilde{g} = g$.

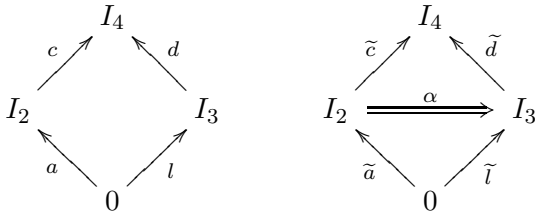


(iii) (iv)

We also get immediately that diagram (iv) above commutes. We shall show that the left triangle of diagram (iii) commutes, the proof for the right one is similar. From the definition of GRPO, we have that $\mathrm{id}_c = \tau c \bullet \widetilde{g}\varphi \bullet \gamma = g\varphi \bullet \gamma$ which then implies that $\varphi = \gamma^{-1}$. Using the definition of γ, $\alpha i_{c\to a\oplus c} \bullet \varphi \bullet t_e = \mathrm{id}$ which amounts to saying that the triangle is commutative.

Uniqueness in ´**A** easily follows from essential uniqueness in **B** (which is in this case the same as uniqueness, since there is only one endofunction on the $\varnothing$).

Going the other way, suppose that diagram (v) below



is a RPO. Then we claim that diagram (vi) is a GRPO where α is

$$\widetilde{a} \oplus \widetilde{c} \xrightarrow{[t_a^{-1}, t_c^{-1}]} a \uplus c = l \uplus d \xrightarrow{[i_{\widetilde{l} \to \widetilde{l} \oplus \widetilde{d}} t_l, i_{\widetilde{d} \to \widetilde{l} \oplus \widetilde{d}} t_d]} \widetilde{l} \oplus \widetilde{d}$$

It is trivial to show that that α is structure-preserving, i.e. $\alpha \cdot (\widetilde{c} \circ \widetilde{a}) = \widetilde{d} \circ \widetilde{l}$. Now consider a candidate $\langle I_5, e, f, g, \beta, \gamma, \delta \rangle$ for diagram (vi), above. Since the pasting composite of γ, β and δ yields α, we have that $t_c^{-1}\gamma^{-1} i_{g \to e \oplus g} \cdot g = t_d^{-1} \delta i_{g \to f \oplus g} \cdot g = g'$. Let $e' = t_c^{-1}\gamma^{-1} i_{e \to e \oplus g} \cdot e$ and $f' = t_d^{-1} \delta i_{f \to f \oplus g} \cdot f$. It is easy but tedious to check that $\langle I_5, e', f', g' \rangle$ is a candidate for diagram (i). By assumption, there exists an arrow $h \colon I_4 \to I_5$ which satisfies $hc = e'$, $hd = f'$ and $g'h = f'$. This can be translated in the by-now standard way into a mediating morphism $\langle h, \varphi, \psi, \tau \rangle$ where τ is again the unique endofunction on the $\varnothing$. Essential uniqueness again follows. □

Example 4.3.10. The G-category of supports of the precategory **ABun** is **Bun**. Note that a 'structure preserving' support bijection is, in this case, exactly a bunch homomorphism. Indeed, $\alpha \colon \langle X, \mathrm{ch}, \mathrm{rt} \rangle \Rightarrow \langle X', \mathrm{ch}', \mathrm{rt}' \rangle$ if $X' = \alpha X$, $\mathrm{ch}' = \mathrm{ch}\ \alpha^{-1}$ and $\mathrm{rt}' = \mathrm{rt}(\mathrm{id} \oplus \alpha^{-1})$ which is the same as saying $\mathrm{ch} = \mathrm{ch}'\ \alpha$ and $\mathrm{rt} = \mathrm{rt}'(\mathrm{id} \oplus \alpha)$.

In other words, our general construction translating from S-precategories to G-categories applied to the particular case of bunch contexts extracts **Bun** out of **ABun**. It is worth remarking how in Definition 4.3.8 precategories' support-translation isomorphisms are subsumed in G-categories as 2-cells. It is also worth reflecting on the fact that the natural notion of bunch homomorphism comes "for free" out of the ad-hoc definition of the S-precategory **ABun** when translated into a G-category.

One consequence of Theorem 4.3.9 is that Jensen and Milner's congruence theorem [46] follows from of the fact that bisimilarity is a congruence on the abstract lts generated by GIPOs (Corollary 2.3.7). Indeed, it appears that the congruence results presented in this thesis are more general then the congruence results for S-precategories – this is because the translation of Theorem 4.3.9 is a one way translation (from S-precategories to G-categories) and it is unclear how one would go about providing a general translation going the other way. Indeed, an arbitrary G-category does not need to have 2-cells which behave like bijections.

4.3.3 Functorial reactive systems

In this section we shall briefly recall the theory of *functorial reactive systems* due to Leifer [64] (see also [65]) which is an alternative to reactive systems over S-precategories. Similarly to the translation exhibited in Theorem 4.3.9, there is a general translation from any functorial reactive systems to a G-reactive system so that the labelled transition systems generated are equal.

Functorial reactive systems. While staying with the idea of using sets to keep track of the concrete identity of algebraic objects, Leifer considers an ordinary category (composition is always defined), called the track category which also has the concrete set theoretical identities on the arrows, but also the objects have extra set theoretical information built in. This extra data in the objects essentially ensures that the support sets in any composition are disjoint. There is an ordinary functor relating the track category and the support quotient category, defined in a similar way to the support quotient of an S-precategory (Definition 4.3.4). This functor is required to satisfy the properties which Leifer identifies and uses in order to prove his central congruence theorem. The congruence theorem essentially states that if enough RPOs exist in the track category then one may generate an lts with states and transitions from the support quotient category, and bisimilarity (as well as other equivalences/preorders) is a congruence. In [65] Leifer showed that starting with an S-precategory one can generate a track category, so that the functor from the track to the support quotient categories automatically satisfies the required properties.

The track has the support information built into the objects. On the contrary, the support quotient consists of isomorphism classes of arrows with respect to support translation.

Definition 4.3.11 (Track category). The *track* category $\widehat{\mathbf{A}}$ of an S-precategory $\acute{\mathbf{A}}$ has:

- objects: pairs $\langle A,\, M\rangle$ where $A \in \acute{}\mathbf{A}$ and $M \in \mathbf{Set}_f$;
- arrows: $\langle A, M\rangle \xrightarrow{f} \langle B,\, N\rangle$ where $f\colon A \to B$ is in $\acute{}\mathbf{A}$, $M \subseteq N$ and $|f| = N \backslash M$.

Composition of arrows is as in $\acute{}\mathbf{A}$. Observe that the definition of $|f|$ ensures that composition of arrows is always defined. We leave it to the reader to check that the data defines a category (cf. [65]).

There is an obvious functor $F\colon \widehat{\mathbf{A}} \to \mathbf{A}$, the *support-quotienting functor*. It is proved in [65] that this functor satisfies the conditions required by the theory of functorial reactive systems [64, 65]. One can define a reactive system on the track category in a similar way to the reactive system over an S-precategory (Definition 4.3.6). The data of such a reactive system maps down to the support-quotient category giving an ordinary reactive system.

Roughly, one can generate a labelled transition system for a reactive system in $\mathbf{A}$ by using the labels of redex squares which are IPOs "upstairs" in the track category. Then if the track category $\widehat{\mathbf{A}}$ has enough RPOs, bisimulation on the resulting labelled transition systems is a congruence.

Every functorial reactive system can be translated, in a manner almost identical to the translation given in the proof of Theorem 4.3.9, to a G-reactive system. Moreover, the labelled transition systems generated are the same. Because the proofs are almost identical, we shall not give the details here and refer the interested reader to [92].

Chapter 5

Adhesive and quasiadhesive categories

In this chapter we shall introduce and develop the theory of adhesive categories. They are categories in which pushouts along monomorphisms are "well-behaved", where the paradigm for the good-behaviour is given by the category of sets. The idea is analogous to that of extensive categories [12], briefly recalled in Chapter 4, which have well-behaved coproducts in a similar sense.

Various notions of graphical structures used in computer science form adhesive categories. This includes ordinary directed graphs, typed graphs [4] and hypergraphs [23], amongst others. The structure of adhesive category allows us to derive useful properties. For instance, the union of two subobjects is calculated as the pushout over their intersection.

While the structure of adhesive categories shall be used in Chapter 6 in order to construct GRPOs in a suitable bicategory of cospans over an arbitrary adhesive category, it turns out that adhesive categories have a more direct application. As we shall demonstrate, adhesive categories provide an elegant setting in which one can develop the well-known theory of double-pushout (dpo) graph rewriting [27].

The advantages provided by suitable categorical settings for theoretical computer science is that proofs of various useful properties are provided uniformly across a wide range of models. The usual approach is to find a natural class of categories with the right structure to support the range of constructions particular to the application area. The structure should, ideally, be robust with respect to common operations on categories, such as product or slice category which allow one to construct new models from old. Such robustness means that each of the properties that the structure implies is also robust under such operations. A well-known example of a natural class of categories which capture the relevant structure is the class of

cartesian-closed categories, which provides models for simply typed lambda calculi [60].

The chapter is divided into three sections. In section 5.1 we shall define adhesive categories, give examples and derive several of their basic properties. Section 5.2 recalls the notion of dpo graph rewriting [27] and high level replacement (hlr) categories [23], which are a class of categories designed to provide a general class of models for dpo rewriting. We show that many of the axioms of hlr-categories hold in adhesive categories and develop some of the rewriting theory for dpo systems over adhesive categories. Finally, in section 5.3 we develop the theory of quasiadhesive categories which are a weaker version of adhesive categories, meaning that every adhesive category is quasiadhesive, but not vice-versa. This weakening of the theory is crucial in order to cover interesting examples from the theory of algebraic specifications [90].

5.1 Adhesive categories

In this section we shall introduce the theory of adhesive categories. First, in §5.1.1, we shall introduce the notion of Van Kampen square, which is central to the definitions of both adhesive and quasiadhesive categories. We shall state the definition of adhesive categories, give several examples and prove that the notion of adhesivity is robust under several categorical operations in §5.1.2. Finally, in §5.1.3 we shall state and prove several of the basic properties which hold in any adhesive category.

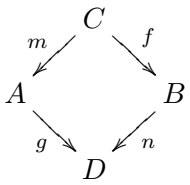
5.1.1 Van Kampen squares

The definition of adhesive category is stated in terms of something called a *van Kampen square*, which can be thought of as a "well-behaved pushout", in a similar way to which coproducts can be thought of as "well-behaved" in an extensive category; essentially this means that they behave as they do in the category of sets.

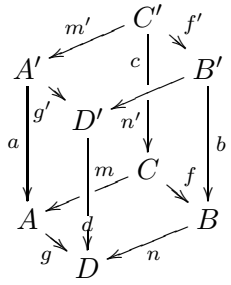
The name van Kampen derives from the relationship between these squares and the van Kampen theorem in topology, in its "coverings version", as presented for example in [11]. This relationship is described in detail in [58].

Definition 5.1.1 (van Kampen square). A *van Kampen (VK) square* (i) is a pushout which satisfies the following condition: given a commutative cube (ii) of which diagram (i) forms the bottom face and the back faces

are pullbacks, the front faces are pullbacks if and only if the top face is a pushout. Another way of stating the "only if" condition is that such a pushout is required to be stable under pullback.

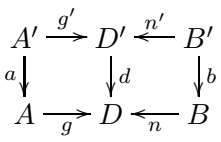


(i)

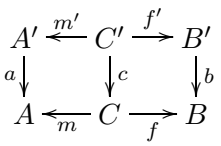


(ii)

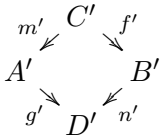
Another, equivalent, way of defining a VK square in a category with pullbacks is as follows. A VK square (i) is a pushout which satisfies the property that given a commutative diagram (iii), the two squares are pullbacks if and only if there exists an object C' and morphisms



(iii)



(iv)



(v)

so that the squares in diagram (iv) are pullbacks and diagram (v) is a pushout.

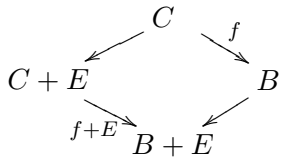
By a *pushout along a monomorphism* we mean a pushout, as in diagram (i) above, in which m is a monomorphism. Similarly, if m is a coproduct injection, we have a pushout along a coproduct injection.

A crucial class of examples of VK squares is provided by:

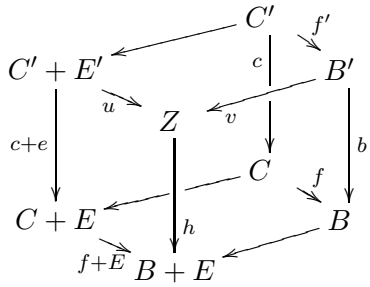
Theorem 5.1.2. *In an extensive category, pushouts along coproduct injections are VK squares.*

Proof. If $m : C \to A$ is a coproduct injection, say $C \to C + E$, then the

diagrams (i) and (ii) have the form



(i)



(ii)

where the unnamed arrows are coproduct injections.

If the top face is a pushout then we may take $Z = B' + E'$, it then follows that $h = b + e$. The front right face of the cube is then a pullback, using extensivity. The front left face is a pullback using the conclusion of Lemma 4.1.5; it is constructed by "adding together" two pullbacks, namely the back right face of the cube and the pullback of the identity on E and $e : E' \to E$.

Conversely, suppose that the front faces are pullbacks. Then, as the bottom row of the following diagram

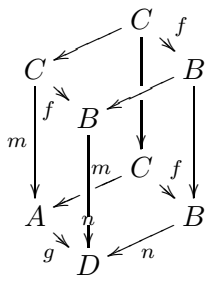
$$\begin{array}{ccccccc} E' & \rightarrowtail & C' + E' & \xrightarrow{u} & Z & \xleftarrow{v} & B' \\ {\scriptstyle e}\downarrow & & {\scriptstyle c+e}\downarrow & & \downarrow{\scriptstyle h} & & \downarrow{\scriptstyle b} \\ E & \rightarrowtail & C + E & \xrightarrow[f+E]{} & B + E & \leftarrowtail & B \end{array}$$

is a coproduct diagram and all the squares are pullbacks, we may deduce that the top row is a coproduct diagram, that is $Z = B' + E'$. Thus the top face of the cube is a pushout. □

We have the following important properties of VK squares:

Lemma 5.1.3. In a VK square as in diagram (i), if m is a monomorphism then n is a monomorphism and the square is also a pullback.

Proof. Suppose that the bottom face of the cube



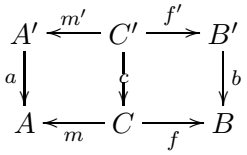
is VK. Then the top and bottom squares are pushouts, while the back squares are pullbacks if m is a monomorphism. Thus the front faces will be pullbacks: the front right face being a pullback means that n is a monomorphism, and the front left face being a pullback means that the original square is a pullback. □

There is an alternative way of defining a VK square which involves requiring a certain functor to be an equivalence of categories. This is akin to the alternative way of presenting the main axiom of extensive categories (see Proposition 4.1.2).

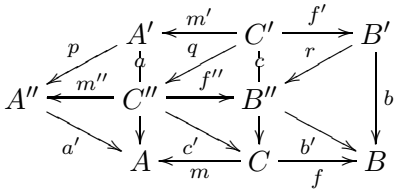
In the following we shall assume that $\mathbf{C}$ is a category with all pullbacks.

Definition 5.1.4. Given a span $A \xleftarrow{m} C \xrightarrow{f} B$, let $\mathbf{C}/A \times_{\mathbf{C}/C} \mathbf{C}/B$ denote the category with:

- objects: commutative diagrams, as illustrated below, where both squares are pullbacks;

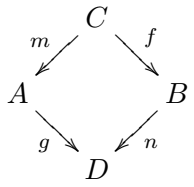


- arrows: given two such diagrams, as illustrated below, an arrow is a triple $p : A' \to A''$, $q : C' \to C''$ and $r : B' \to B''$ so that $a'p = a$, $c'q = c$ and $b'r = b$.



Composition and identities are obvious.

For a morphism $u : U \to V$ we shall write $u^* : \mathbf{C}/V \to \mathbf{C}/U$ for the functor given by pulling back along u. Now, suppose that we have a pushout diagram as below.



Then the functors n^* and g^* induce a functor

$$\mathsf{Pb} : \mathbf{C}/D \to \mathbf{C}/A \times_{\mathbf{C}/C} \mathbf{C}/B$$

which takes an arrow $d : D' \to D$ and gives the object of $\mathbf{C}/A \times_{\mathbf{C}/C} \mathbf{C}/B$ obtained by taking the rear faces of the cube illustrated in diagram (ii). One constructs the cube by taking pullbacks, first in order to construct the front faces and then the back faces. It is easy to verify that this definition defines the functor

On the other hand, if $\mathbf{C}$ has pushouts (or pushouts along monos, if we assume m to be mono) we can define a functor

$$\mathsf{Po} : \mathbf{C}/A \times_{\mathbf{C}/C} \mathbf{C}/B \to \mathbf{C}/D$$

as follows: starting with the back faces of diagram (ii) we construct the pushout of m' and f' and obtain a unique arrow $d : D' \to D$ given by the universal property of pushouts.

Proposition 5.1.5. Pb is right adjoint to Po.

The composite PoPb is given by pulling back $d : D' \to D$ and then forming a pushout; thus the counit of the adjunction is invertible if and only if, in the cube, if the vertical faces are pullbacks then the top face is a pushout; in other words, if the pushout (i) is stable under pullback. On the other hand, the unit of the adjunction is invertible if and only if, whenever the back faces are pullbacks, and the top (and bottom) faces are pushouts, then the front faces are also pullbacks. We may summarize all this as follows:

Proposition 5.1.6. For the pushout diagram (i), the following conditions are equivalent:

(i) Pb is an equivalence;

(ii) Po is an equivalence;

(iii) diagram *(i)* is a VK-square;

(iv) the pushout is stable under pullback, and the functor Pb is essentially surjective on objects.

Condition (i) of the proposition makes sense without assuming all pushouts, and should be taken as the definition of van Kampen square when not all pushouts are assumed to exist. Furthermore, if m in diagram (i) is a monomorphism, then Po will exist provided that pushouts along monomorphisms do so, and the proposition will hold in that generality.

5.1.2 Adhesive categories

We shall now proceed to define the notion of adhesive category, and provide various examples and counterexamples.

Definition 5.1.7 (Adhesive category). A category $\mathbf{C}$ is said to be *adhesive* if

(i) $\mathbf{C}$ has pushouts along monomorphisms;

(ii) $\mathbf{C}$ has pullbacks;

(iii) pushouts along monomorphisms are VK-squares.

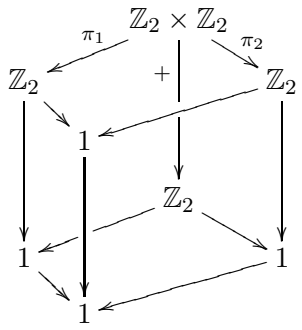
Just as the third axiom of extensive categories (Definition 4.1.1) ensures that coproducts are "well-behaved", it is the third axiom of adhesive categories which ensures that pushouts along monomorphisms are "well-behaved". This includes the fact that such pushouts are stable under pullback.

Since every monomorphism in **Set** is a coproduct injection, and **Set** is extensive, we immediately have:

Example 5.1.8. Set is adhesive.

Observe that the restriction to pushouts along monomorphisms is necessary: there are pushouts in **Set** which are not VK squares. Consider the 2

element abelian group $\mathbb{Z}_2$ (the following argument works for any non-trivial group). In the diagram



both the bottom and the top faces are easily verified to be pushouts and the rear faces are both pullbacks. However, the front two faces are not pullbacks.

Even with the restriction to pushouts along a monomorphism, many well-known categories fail to be adhesive.

Counterexample 5.1.9. The categories **Pos**, **Top**, **Gpd** and **Cat** are not adhesive.

Proof. For the case of the category of posets, write $[n]$ for the ordered set $\{0 \leq 1 \leq \ldots n-1\}$. The pushout square

$$\begin{array}{ccc} [1] & \xrightarrow{0} & [2] \\ {\scriptstyle 1}\downarrow & & \downarrow \\ [2] & \to & [3] \end{array}$$

is not van Kampen, since it is not stable under pullback along the map $[2] \to [3]$ sending 0 to 0 and sending 1 to 2. Thus **Pos** is not adhesive. The same pushout square, regarded as a pushout of categories, shows that **Cat** is not adhesive. For the case of **Gpd**, one simply replaces the poset $[n]$ by the groupoid with n objects and a unique isomorphism between each pair of objects.

Finally consider the category **Top** of topological spaces. A finite poset induces a finite topological space on the same underlying set: the topology is determined by specifying that y is in the closure of x if and only if $x \leq y$. Applying this process to the previous example yields an example showing that **Top** is not adhesive. □

Since the definition of adhesive category only uses pullbacks, pushouts, and relationships between these, we have the following constructions involving adhesive categories:

Proposition 5.1.10.

(i) If $\mathbf{C}$ and $\mathbf{D}$ are adhesive categories then so is $\mathbf{C} \times \mathbf{D}$;

(ii) If $\mathbf{C}$ is adhesive then so are $\mathbf{C}/C$ and $C/\mathbf{C}$ for any object C of $\mathbf{C}$;

(iii) If $\mathbf{C}$ is adhesive then so is any functor category $[\mathbf{X}, \mathbf{C}]$.

Since **Set** is adhesive, part (iii) of the proposition implies that any presheaf topos $[\mathbf{X}, \mathbf{Set}]$ is adhesive. In particular, the category **Graph** of directed graphs is adhesive. Indeed, if $\mathbf{C}$ is adhesive, then so is the category $Graph(\mathbf{C}) = [\cdot \rightrightarrows \cdot, \mathbf{C}]$ of internal graphs in $\mathbf{C}$.

Part (ii) implies that categories of typed graphs [4], coloured (or labelled) graphs [15], ranked graphs [36] and hypergraphs [23], considered in the literature on graph grammars, are adhesive.

As a consequence, all proof techniques and constructions in adhesive categories can be readily applied to any of the aforementioned categories of graphs. In fact, more generally, we have:

Proposition 5.1.11. *Any elementary topos is adhesive.*

This is somewhat harder to prove than the result for presheaf toposes; the proof can be found in [58].

Part (ii) of Proposition 5.1.10 also allows us to construct examples of adhesive categories which are not toposes.

Example 5.1.12. The category $\mathbf{Set}_* = 1/\mathbf{Set}$ of pointed sets (or equivalently, sets and partial functions) is adhesive, but is not extensive, and therefore, is not a topos.

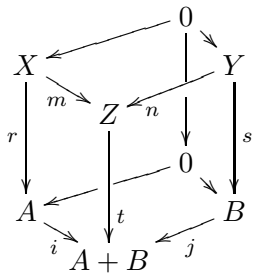
Proof. In the category of pointed sets, the initial object is the one-point set 1. Since every non-initial object has a map into 1, the initial object is not strict, and so the category is not extensive [12, Proposition 2.8]. □

5.1.3 Basic properties of adhesive categories

Here we provide several simple lemmas which hold in any adhesive category. Lemma 5.1.13 demonstrates why adhesive categories can be considered as a generalisation of extensive categories. Lemmas 5.1.15, 5.1.16, 5.1.18 and 5.1.19 shed some light on pushouts along monos in adhesive categories.

Lemma 5.1.13. An adhesive category is extensive if and only if it has a strict initial object.

Proof. In an extensive category the initial object is strict [12, Proposition 2.8]. On the other hand, in an adhesive category with strict initial object, any arrow with domain 0 is mono. Consider the cube



in which the bottom square is a pushout along a monomorphism, while the back squares are pullbacks since the initial object is strict. By adhesiveness, front squares are pullbacks if and only if the top squares is a pushout; but this says that the front squares are pullbacks if and only if the top row of these squares is a coproduct (Z=X+Y). □

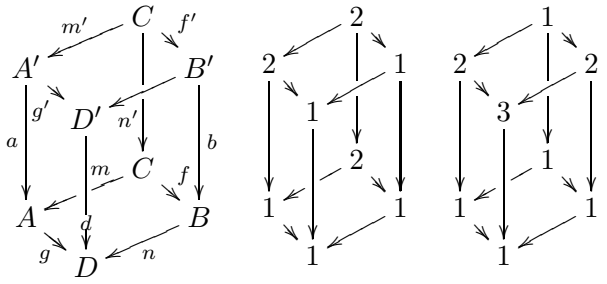
Remark 5.1.14. Notice that coproducts in coslice categories are pushouts. It is then perhaps useful to point out that a category which is extensive in every coslice is not the same thing as an adhesive category.[1]

Indeed, categories which are extensive in every slice are equivalent to the terminal category **1**. This is because extensive categories have strict initial objects, meaning that if $X/\mathbf{C}$ is extensive and if there exist arrows $f : X \to Y$ and $g : Y \to X$ such that $gf = \mathrm{id}_X$ then f must be an isomorphism. Applying this to the codiagonal $X + X \to X$ yields that $X \cong X + X$ for every $X \in \mathbf{C}$. This is another way of saying that $\mathbf{C}$ is a preorder. But every extensive preorder is equivalent to **1**: indeed, since coproducts are disjoint we have $X = X \cap X \cong \bot$.

Axiomatically, a category which is extensive in every coslice satisfies the

[1] This question was posed to the author by Paul-André Melliès.

following axiom: for any cube, as in diagram (i)



(i) (ii) (iii)

where the bottom face is a pushout, the top face is a pushout if and only if the front faces are pullbacks. It is easy to see that **Set** fails to satisfy this axiom, apart from the fact that **Set** is not equivalent to **1**, we may consider the cube illustrated in diagram (ii), where the top and the bottom faces are pushouts, yet the front left face is not a pullback. This problem persists even if we restrict the sort of pushouts we allow by requiring m and m' in diagram (i) to be mono: indeed, consider the cube in diagram (iii), the top and bottom faces are pushouts but neither of the front faces are pullbacks. Finally, one may strengthen the axiom of extensive categories, requiring extensivity only when the "vertical" morphisms of the diagram in the definition of extensive categories (Definition 4.1.1) are mono. It then follows by Lemma 1.3.3 that the two rear squares of diagram (i) are pullbacks and the axiom follows as a special case of adhesivity. However, such categories are then less general then adhesive categories.

The conclusions of the following two lemmas are used extensively in literature on algebraic graph rewriting. Indeed, they are usually assumed as axioms (see [21] and section 5.2.2) in attempts at generalising graph rewriting. They hold in any adhesive category by Lemma 5.1.3:

Lemma 5.1.15. Monomorphisms are stable under pushout.

Lemma 5.1.16. Pushouts along monomorphisms are also pullbacks.

The notion of pushout complement [27] is vital in algebraic approaches to graph rewriting.

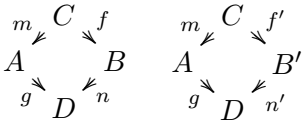
Definition 5.1.17. Let $m : C \to A$ and $g : A \to B$ be arrows in an arbitrary category (m is not assumed to be mono). A *pushout complement* of the pair (m, g) consists of arrows $f : C \to B$ and $n : B \to D$ for which the resulting

square commutes and is a pushout. We shall sometimes refer to pushout complements of *monos*, this refers to pushout complements of pairs (m, g) where m is mono.

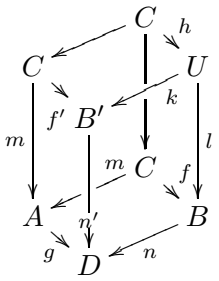
The conclusion of the following lemma is a crucial ingredient in many applications of graph rewriting. It has also been assumed as an axiom [23] in order to prove the concurrency theorem (cf. Theorem 5.2.14). It is important mainly because it assures that once an occurrence of a left hand side of a rewrite rule is found within a structure, then the application of the rewrite rule results in a structure which is unique up to isomorphism (see section 5.2). In other words, rewrite rule application is functional up to isomorphism.

Lemma 5.1.18. Pushout complements of monos (if they exist) are unique up to isomorphism.

Proof. Suppose that the following diagrams



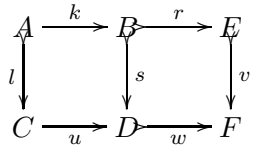
are pushouts and that m is mono. Consider the cube



in which the front right face is a pullback, $h : C \to U$ is the map induced by f and f', and the unnamed arrows are identities. Then the front faces and the back left face are pullbacks, hence the back right face is also a pullback; and the bottom face is a pushout, hence the top face is a pushout. But this implies that k is invertible, since it is the pushout of 1_C. By symmetry, so too is l. The induced isomorphism $j = kl^{-1} : B \to B'$ satisfies $n'j = n$ and $jf = f'$. □

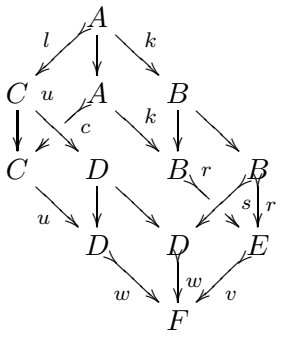
The following lemma will be used in §5.2.2 to show that adhesive categories are high-level replacement categories:

Lemma 5.1.19. Consider a diagram



in which the marked morphisms are mono, the exterior is a pushout and the right square is a pullback. Then the left square is a pushout, and so all squares are both pullbacks and pushouts.

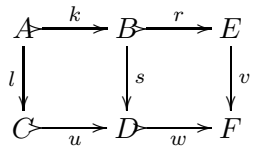
Proof. This amounts to stability of the exterior pushout under pullback along $w : D \to F$. We illustrate this in the diagram below, where we leave the identity morphisms unlabelled.



□

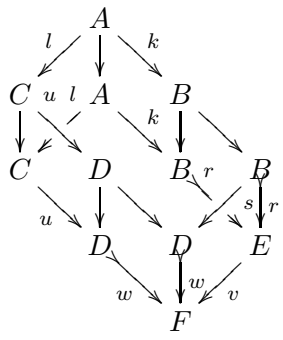
The following lemma is of a similar nature and it shall be useful in the construction of GRPOs of Chapter 6.

Lemma 5.1.20. Consider a diagram



in which the marked morphisms are mono, the exterior is a pushout, the right square is a pullback, and morphisms k, r, u and w are mono. Then the left square is a pushout.

Proof. The exterior pushout is stable under pullback along the morphism w, as illustrated below.



□

The following lemma shows that all monomorphisms in adhesive categories are regular. For a monomorphism to be regular, it has to be an equaliser

$$A \xrightarrow{m} B \underset{g}{\overset{f}{\rightrightarrows}} C$$

of two morphisms. In fact, it follows easily from the universal property of equalisers that any equaliser is mono.

Lemma 5.1.21. Monomorphisms are regular.

Proof. Given $m : A \to B$, construct the pushout of m with itself:

$$\begin{array}{ccc} A & \xrightarrow{m} & B \\ {\scriptstyle m}\downarrow & & \downarrow{\scriptstyle n_1} \\ B & \xrightarrow[n_2]{} & C. \end{array}$$

Since m is mono, the diagram above is also a pullback (Lemma 5.1.16). It is now easy to show that m is the equaliser of n_1 and n_2. Indeed, suppose we're given a morphism $p : X \to B$ such that $n_1 p = n_2 p$. Using the pullback property, there exists a unique morphism $h : X \to A$ such that $mh = p$. □

The following lemma can be seen as an easy corollary of the former. Indeed, it is easy to show that in any category, if a morphism is both epi and regular mono then it is an isomorphism. Nonetheless, we include it as it has a very simple direct proof.

Lemma 5.1.22. Adhesive categories are balanced. That is, a morphism which is both a monomorphism and an epimorphism is an isomorphism.

Proof. Suppose that $f : A \to B$ is mono and epi. Because it is epi,

$$\begin{array}{ccc} A & \xrightarrow{f} & B \\ {\scriptstyle f}\downarrow & & \downarrow \\ B & \to & B \end{array}$$

the diagram above is a pushout. Since f is mono, using Lemma 5.1.16 we can conclude that it is also a pullback. This implies that there exists an arrow $g : B \to A$ such that $fg = \mathrm{id}_B$. Then $fgf = f$ and since f is mono, $gf = \mathrm{id}_A$. □

We can put a preorder on monomorphisms into an object Z of an arbitrary category by defining a monomorphism $a : A \to Z$ to be less than or equal to a monomorphism $b : B \to Z$ precisely when there exists an arrow $c : A \to B$ such that $bc = a$. A subobject (of Z) is an equivalence class with respect to the equivalence generated by this preorder. For example, subobjects in **Set** are subsets while subobjects in **Graph** are subgraphs.

Here we shall demonstrate that, in adhesive categories, unions of two subobjects can be constructed by pushout over their intersection. This provides further evidence of how pushouts behave in adhesive categories as well as making more precise the intuition that the pushout operation "glues together" two structures along a common substructure. As a corollary, it follows that in an adhesive category the lattices of subobjects are distributive.

Let $\mathbf{C}$ be an adhesive category, and Z a fixed object of $\mathbf{C}$. We write $\mathrm{Sub}(Z)$ for the category of subobjects of Z in $\mathbf{C}$; it has products (=intersections), given by pullback in $\mathbf{C}$. It has a top object, given by Z itself. If $\mathbf{C}$ has a strict initial object 0, then the unique map $0 \to Z$ is a monomorphism, and is the bottom object of $\mathrm{Sub}(Z)$.

Theorem 5.1.23. *For an object Z of an adhesive category $\mathbf{C}$, the category Sub(Z) of subobjects of Z has binary coproducts: the coproduct of two subobjects is the pushout in $\mathbf{C}$ of their intersection.*

Proof. We shall show how to form binary coproducts (=unions) in $\mathrm{Sub}(Z)$. Let $a : A \to Z$ and $b : B \to Z$ be subobjects of Z, and form the intersection $A \cap B \to Z$, with projections $p : A \cap B \to A$ and $q : A \cap B \to B$; and now

the pushout

$$\begin{array}{ccc} A \cap B & \xrightarrow{q} & B \\ {\scriptstyle p}\downarrow & & \downarrow{\scriptstyle v} \\ A & \xrightarrow[u]{} & C \end{array}$$

in $\mathbf{C}$. Let $c : C \to Z$ be the unique map satisfying $cu = a$ and $cv = b$. We shall show that c is a monomorphism, and so that C is the coproduct $A \cup B$ in $\mathrm{Sub}(Z)$ of A and B. Suppose then that $f, g : K \to C$ satisfy $cf = cg$. Form the following pullbacks

$$\begin{array}{ccccc} L_1 & \xrightarrow{l_1} & K & \xleftarrow{l_2} & L_2 \\ {\scriptstyle f_1}\downarrow & & \downarrow{\scriptstyle f} & & \downarrow{\scriptstyle f_2} \\ A & \xrightarrow[u]{} & C & \xleftarrow[v]{} & B \end{array} \qquad \begin{array}{ccccc} M_1 & \xrightarrow{m_1} & K & \xleftarrow{m_2} & M_2 \\ {\scriptstyle g_1}\downarrow & & \downarrow{\scriptstyle g} & & \downarrow{\scriptstyle g_2} \\ A & \xrightarrow[u]{} & C & \xleftarrow[v]{} & B \end{array} \qquad \begin{array}{ccccc} N_{11} & \xrightarrow{m_{11}} & M_1 & \xleftarrow{m_{12}} & N_{12} \\ {\scriptstyle l_{11}}\downarrow & & \downarrow{\scriptstyle m_1} & & \downarrow{\scriptstyle l_{12}} \\ L_1 & \xrightarrow{l_1} & K & \xleftarrow{l_2} & L_2 \\ {\scriptstyle l_{21}}\uparrow & & \uparrow{\scriptstyle m_2} & & \uparrow{\scriptstyle l_{22}} \\ N_{21} & \xrightarrow[m_{21}]{} & M_2 & \xleftarrow[m_{22}]{} & N_{22} \end{array}$$

and note that each of the following pairs are the coprojections of a pushout, hence each pair is jointly epimorphic: (l_1, l_2), (m_1, m_2), (m_{11}, m_{12}), and (m_{21}, m_{22}). We are to show that $f = g$; to do this, it will suffice to show that $fm_1 = gm_1$ and $fm_2 = gm_2$; we shall prove only the former, leaving the latter to the reader. To show that $fm_1 = gm_1$ it will in turn suffice to show that $fm_1m_{11} = gm_1m_{11}$ and $fm_1m_{12} = gm_1m_{12}$.

First note that $af_1l_{11} = cuf_1l_{11} = cfl_1l_{11} = cgl_1l_{11} = cgm_1m_{11} = cug_1m_{11} = ag_1m_{11}$, so that $f_1l_{11} = g_1m_{11}$ since a is monic; thus $fm_1m_{11} = fl_1l_{11} = uf_1l_{11} = ug_1m_{11} = gm_1m_{11}$ as required.

On the other hand, $bf_2l_{12} = cvf_2l_{12} = cfl_2l_{12} = cgl_2l_{12} = cgm_1m_{12} = cug_1m_{12} = ag_1m_{12}$, so by the universal property of the pullback $A \cap B$, there is a unique map $h : N_{12} \to A \cap B$ satisfying $ph = g_1m_{12}$ and $qh = f_2l_{12}$. Now $fm_1m_{12} = fl_2l_{12} = vf_2l_{12} = vqh = uph = ug_1m_{12} = gm_1m_{12}$, and so $fm_1 = gm_1$ as claimed. As promised, we leave the proof that $fm_2 = gm_2$ to the reader, and deduce that $f = g$, so that c is monic. □

Since pushouts are stable it follows that intersections distribute over unions:

Corollary 5.1.24. The lattice $\mathrm{Sub}(Z)$ is distributive.

5.2 Adhesive grammars

In this section we shall demonstrate that adhesive categories have structure which allows a development of a rich *general* theory of double-pushout (dpo)

rewriting [27]. Dpo *graph* rewriting has been widely studied and the field can be considered relatively mature [87, 20, 26].

Dpo rewriting is formulated in categorical terms and is therefore portable to structures other than directed graphs. There have been several attempts [23, 21] to isolate classes of categories in which one can perform dpo rewriting and in which one can develop the rewriting theory to a satisfactory level. In particular, several axioms were put forward in [23] in order to prove a local Church-Rosser theorem (Theorem 5.2.10) for such general rewrite systems. Additional axioms were needed to prove a general version of the so-called concurrency theorem [57] (Theorem 5.2.14). Categories satisfying such axioms are referred to in the literature as hlr-categories. Hlr-categories are usually parametrised by a class of morphisms M. In the majority of the relevant examples covered by hlr theory, M is the class of monomorphisms. We shall show that many of the axioms of hlr-categories, when M is the class of monomorphisms, hold in adhesive categories.

First, in §5.2.1, we shall recall the basic definitions of dpo-rewriting. In §5.2.2 we shall examine the relationship between adhesive categories and hlr-categories. This shall include *proving*, in the setting of adhesive categories, several of the axioms put forward in the literature on hlr-categories. Finally, in §5.2.3 we shall prove two of the classic results from the theory of hlr-systems, the local Church-Rosser and the concurrency theorems.

5.2.1 Double-pushout graph rewriting

Here we shall recall the basic notions of double-pushout rewriting [27, 87] and show that it can be defined within an arbitrary adhesive category.

Henceforth we shall assume that $\mathbf{C}$ is an adhesive category.

Definition 5.2.1 (Production). A production p is a span

$$L \xleftarrow{l} K \xrightarrow{r} R \tag{5.1}$$

in $\mathbf{C}$. We shall say that p is *left-linear* when l is mono, and *linear* when both l and r are mono. We shall let $\mathcal{P}$ denote an arbitrary set of productions and let p range over $\mathcal{P}$.

In order to develop an intuition of why a production is defined as a span, we shall restrict our attention to linear production rules. One may then consider K as a substructure of both L and R. We think of L and R as respectively the left-hand side and the right-hand side of the rewrite rule p. In order to perform the rewrite, we need to match L as a substructure of

a redex C. The structure K, thought of as a substructure of L, is exactly the part of L which is to remain invariant as we apply the rule to C.

Thus, an application of a rewrite rule consists of three steps. First we must match L as a substructure of the redex C; secondly, we delete all of parts of the redex matched by L which are not included in K. Thirdly, we add all of R which is not contained in K, thereby producing a new structure D. The deletion and addition of structure is handled, respectively, by finding a pushout complement and constructing a pushout.

Definition 5.2.2 (Gluing Conditions). Given a production p as in (5.1), a *match* in C is a morphism $f : L \to C$. A match f satisfies the *gluing conditions* with respect to p precisely when there exists an object E and morphisms $g : K \to E$ and $v : E \to C$ such that

$$\begin{array}{ccc} L & \xleftarrow{l} & K \\ {\scriptstyle f}\downarrow & & \downarrow{\scriptstyle g} \\ C & \xleftarrow[v]{} & E \end{array}$$

is a pushout diagram. (In other words, there exists a pushout complement of (l, f) in the sense of Definition 5.1.17.)

Definition 5.2.3 (Derivation). Given an object $C \in \mathbf{C}$ and a set of productions $\mathcal{P}$, we write $C \Longrightarrow_{p,f} D$ for a production $p \in \mathcal{P}$ and a morphism $f : L \to C$ if (a) f satisfies the gluing conditions with respect to l, and (b) there is a diagram

$$\begin{array}{ccccc} L & \xleftarrow{l} & K & \xrightarrow{r} & R \\ {\scriptstyle f}\downarrow & & {\scriptstyle g}\downarrow & & \downarrow{\scriptstyle h} \\ C & \xleftarrow[v]{} & E & \xrightarrow[w]{} & D \end{array}$$

in which both squares are pushouts.

The object E in the above diagram can be thought of as a temporary state in the middle of the rewrite process. Returning briefly to our informal description, it is the structure obtained from C by deleting all the parts of L not contained in K. Recall from Lemma 5.1.18 that if l is mono (that is, if p is left-linear) then E is unique up to isomorphism. Indeed, if p is a left-linear production, $C \Longrightarrow_{p,f} D$ and $C \Longrightarrow_{p,f} D'$ then we must have $D \cong D'$. This is a consequence of Lemma 5.1.18 and the fact that pushouts are unique up to isomorphism.

Definition 5.2.4 (Adhesive Grammar). An adhesive grammar $\mathbf{G}$ is a pair $\langle \mathbf{C}, \mathbf{P} \rangle$ where $\mathbf{C}$ is an adhesive category and $\mathbf{P}$ is a set of linear productions.

Assuming that all the productions are linear allows us to derive a rich rewriting theory on adhesive categories. Henceforward we assume that we are working over an adhesive grammar **G**.

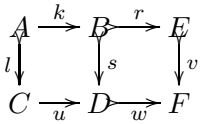
5.2.2 Adhesive vs. high-level replacement categories

High-level replacement categories [21,22,23] or hlr-categories encompass several attempts to isolate general categorical axioms which lead to categories in which one can define double-pushout graph rewriting and prove useful theorems such as the local Church-Rosser theorem and the concurrency theorem.

Hlr-categories usually have axioms which are parametrised over an arbitrary class of morphisms $\mathcal{M}$. Here we give a simplified version of the definition which appears in [21]. The simplification is that we take $\mathcal{M}$ to be the class of monomorphisms: we justify this by noting that this is the case in the majority of examples.

Definition 5.2.5 (Hlr-categories). A category **S** is an hlr-category if it satisfies the following axioms:

1. pairs $C \leftarrow A \rightarrow B$ with at least one of the arrows mono have a pushout;
2. pairs $B \rightarrow D \leftarrow C$ with both morphisms mono have pullbacks;
3. monos are preserved by pushout;
4. finite coproducts exist;
5. pushouts along monos are pullbacks;
6. pushout-pullback decomposition holds: that is, given a diagram



   if the marked morphisms are mono, the whole rectangle is a pushout and the right square is a pullback, then the left square is a pushout.

Lemma 5.2.6. Any adhesive category with an initial object is an hlr-category.

Proof. This follows immediately from Lemmas 5.1.15, 5.1.16, and 5.1.19. □

The axioms listed above are enough to prove the local Church-Rosser theorem (cf. Theorem 5.2.10), but *not* the concurrency theorem (cf. Theorem 5.2.14). To prove the latter, extra axioms had to be introduced in [23], such as the conclusion of the following lemma. Interestingly, it is almost the dual of the main axiom of adhesive categories.

Lemma 5.2.7 (Cube-pushout-pullback-lemma [23]). Given a cube in which all arrows in the top and bottom faces are mono, if the top face is a pullback and the front faces are pushouts, then the bottom face is a pullback if and only if the back faces are pushouts.

Proof. Since the front faces are pushouts along monomorphisms, they are also pullbacks.

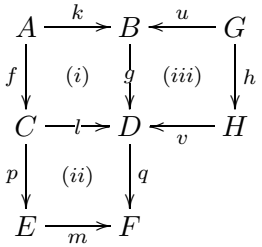
If the bottom face is a pullback, then the back faces are pushouts by stability of the pushouts on the front faces. Suppose conversely that the back faces are pushouts; since they are pushouts along monomorphisms, they are also pullbacks. One now simply "rotates the cube": since the front right and back left faces are pushouts, and the top and back right faces are pullbacks, it follows by adhesiveness that the bottom square is a pullback. □

An hlr-category which has the conclusion of Lemma 5.2.7 as an additional axiom is sometimes referred to as an hlr2-category [23]. It is immediate, therefore, that any adhesive category with an initial object is an hlr2-category.

The strongest axiom system for general rewriting is enjoyed by the so-called hlr2*-categories [23]. These are hlr2-categories which, additionally, have the conclusion of Lemma 5.1.18 as an axiom, that is, pushout complements of monos are, if they exist, unique up to isomorphism. Finally, they satisfy an axiom known as the twisted-triple-pushout condition. We believe that this axiom does not hold in an arbitrary adhesive category, although it does hold, for instance, in any topos. Indeed, it is possible to extend the definition of adhesive categories in a natural way so that the twisted-triple-pushout-condition holds.

Lemma 5.2.8. If $\mathbf{C}$ is adhesive and, additionally, all pushouts are stable under pullback then $\mathbf{C}$ satisfies the twisted-triple-pushout condition: that

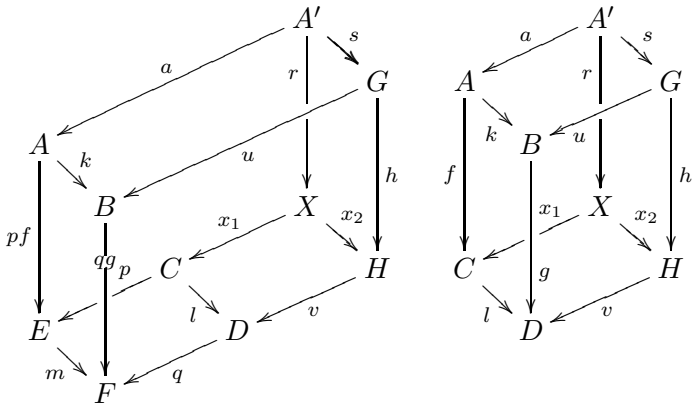
is, given a diagram



with k, l and m mono, diagram (i) is a pushout when (i)+(ii) a pushout, (ii) a pullback, (iii) a pushout and u, h forming a pullback of qg and qv.

Proof. First note that, using Lemma 5.1.16, *(i)+(ii)* is a pullback. Using the pullback version of Lemma 1.3.2 it follows that diagram (i) is a pullback. It is easy to verify that diagram (iii) is a pullback.

Consider diagram (iv) below. By assumption, the front right face is a pullback, while the front left face is (i)+(ii), a pushout.



(iv) (v)

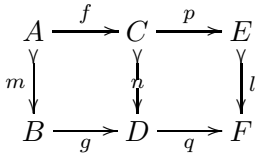
Furthermore, the leftmost half of the bottom face is (ii) which is, by assumption, a pullback. We obtain X and morphisms $x_1 : X \to C$ and $x_2 : X \to H$ by taking the pullback of $l : C \to D$ and $v : H \to D$. We can now complete the diagram with an object A', and morphisms $a : A' \to A$, $r : A' \to X$ and $s : A' \to G$ so that the back left face and the top face are pullbacks. Using adhesivity, the back right face is a pushout.

We obtain diagram (v) by "cutting off" the bottom left corner of diagram (iv). The front left face is (i), the front right face is (iii), which is, by assumption, both a pushout and a pullback. The bottom and back right

faces are pullbacks, since they were constructed as such in diagram (iv). To see that the back left face is commutative, note that $lfa = gka = gus = vhs = vx_2r = lx_1r$ and use the fact that l is mono. It is also a pullback, using the pullback version of Lemma 1.3.2. Using the fact that arbitrary pushouts are stable under pullback, we can conclude that the back left face is a pushout. Summarising, we have assumed that the front right face is a pushout and deduced that both the back left and right faces are pushouts. It follows from the pushouts version of Lemma 1.3.2 that the front left face, (i), is a pushout. □

Finally, the following lemma has been used by Ehrig and König in their work on rewriting via borrowed contexts [25]. Here we prove that it holds in any adhesive category.

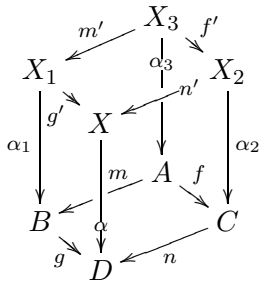
Lemma 5.2.9. Given the diagram below with the marked morphisms mono,



suppose that the left square is a pushout and the exterior is a pullback. Then the right square is a pullback.

Proof. Suppose we have an object X and morphisms $\alpha : X \to D$ and $\beta : X \to E$ such that $q\alpha = l\beta$. We will show that there exists $k : X \to C$ such that $nk = \alpha$ and $pk = \beta$. Notice that it suffices to show the existence of such a morphism, uniqueness follows since n is mono.

Construct the following cube by taking pullbacks.



Now $qg\alpha_1 = q\alpha g' = l\beta g'$, and we use the fact that (1)+(2) is a pullback to derive the existence of a unique morphism $h : X_1 \to A$ such that $mh = \alpha_1$ and $pfh = \beta g'$.

At this point, it shall be helpful to derive a couple of equations which will be useful later. First note that $m\alpha_3 = \alpha_1 m' = mhm'$, and using the fact that m is mono, $\alpha_3 = hm'$ (*). Also, $lp\alpha_2 = qn\alpha_2 = q\alpha n' = l\beta n'$. Since l is mono, we have that $p\alpha_2 = \beta n'$ (**).

We shall use the fact that the top face of the cube is a pushout to derive the existence of the required morphism. Indeed, we have $\alpha_2 f' = f\alpha_3 = fhm'$ where we used (*) to derive the last equality. Thus we get a unique $k : X \to C$ such that $kg' = fh$ and $kn' = \alpha_2$.

It remains to show that k satisfies the necessary properties, that is, $nk = \alpha$ and $pk = \beta$.

Indeed, we have $nkg' = nfh = gmh = g\alpha_1 = \alpha g'$ and $nkn' = n\alpha_2 = \alpha n'$. Using the fact that the top face of the cube is a pushout, and in particular, the uniqueness of the mediating morphism, we have $nk = \alpha$.

Similarly, $pkg' = pfh = \beta g'$ and $pkn' = p\alpha_2 = \beta n'$, where we used (**) to derive the last equality. This implies that $pk = \beta$ and we are finished. □

5.2.3 Rewriting theory

As explained in §5.2.2, adhesive categories with coproducts are high-level replacement categories. In particular, this implies that the rewriting theory worked out at the level of hlr-categories applies to adhesive categories. Here we shall recall and prove the local Church-Rosser [57,21] theorems and the concurrency theorem [23] in the setting of adhesive categories. While these theorems can be seen as a consequence of adhesive categories being hlr-categories, it is useful to prove them directly in this setting, both as an exercise in the theory of adhesive categories as well as in order to demonstrate their applicability to the problems associated with dpo graph transformation.

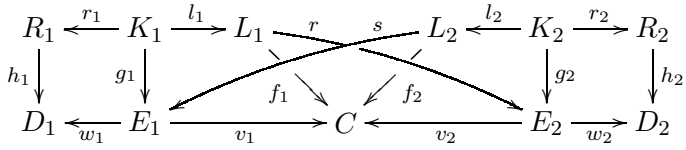
Local Church-Rosser. Before presenting this theorem we need to recall the notions of parallel-independent derivation and sequential-independent derivation. The reader may wish to consult [15] for a more complete presentation.

A *parallel-independent derivation* is a pair of derivations

$$C \Longrightarrow_{p_1,f_1} D_1 \qquad \text{and} \qquad C \Longrightarrow_{p_2,f_2} D_2$$

as illustrated in diagram (5.2) which satisfy an additional requirement, namely the existence of morphisms $r : L_1 \to E_2$ and $s : L_2 \to E_1$ which

render the diagram commutative, in the sense that $v_2 r = f_1$ and $v_1 s = f_2$.

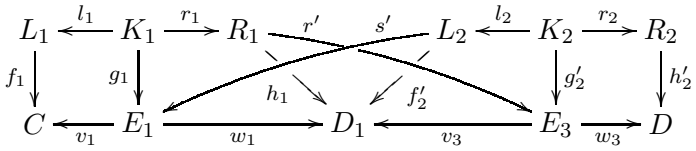


(5.2)

Similarly, a *sequential-independent derivation*, illustrated in diagram (5.3), is a derivation

$$C \Longrightarrow_{p_1,f_1} D_1 \Longrightarrow_{p_2,f_2'} D$$

where there additionally exist arrows $r' : R_1 \to E_3$ and $s' : L_2 \to E_1$ such that $w_1 s' = f_2'$ and $v_3 r' = h_1$.



(5.3)

The statement of the theorem below differs from those previously published in the literature in that we do not need coproducts to establish the equivalence of the first 3 items.

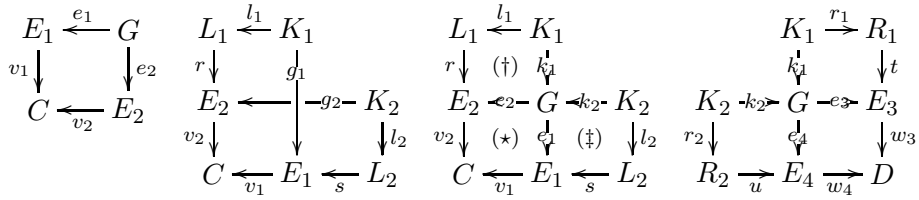
Theorem 5.2.10 (Local Church-Rosser). *The following are equivalent*

1. $C \Longrightarrow_{p_1,f_1} D_1$ *and* $C \Longrightarrow_{p_2,f_2} D_2$ *are parallel-independent derivations*
2. $C \Longrightarrow_{p_1,f_1} D_1$ *and* $D_1 \Longrightarrow_{p_2,f_2'} D$ *are sequential-independent derivations*
3. $C \Longrightarrow_{p_2,f_2} D_2$ *and* $D_2 \Longrightarrow_{p_1,f_1'} D$ *are sequential-independent derivations.*

If moreover $\mathbf{C}$ *is extensive then we may add the so-called* parallelism theorem

4. $C \Longrightarrow_{p_1+p_2,[f_1,f_2]} D$ *is a derivation.*

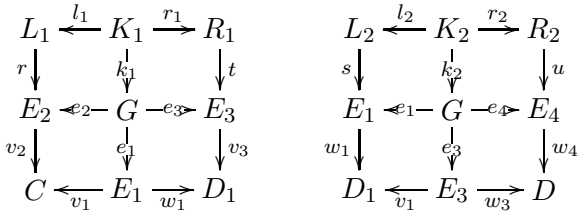
Proof. (1)⇒(2): Suppose that $C \Longrightarrow_{p_1,f_1} D_1$ and $C \Longrightarrow_{p_2,f_2} D_2$ are parallel-independent derivations. Form the pullback (i) below.



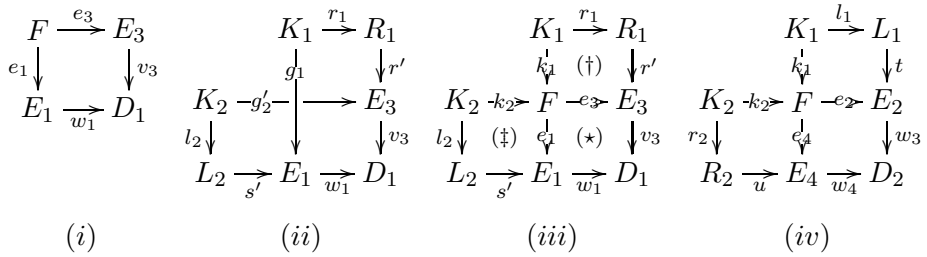
(i) (ii) (iii) (iv)

The two regions in diagram (ii) are pushouts [see diagram (5.3)]. Combining the two diagrams gives (iii), with k_1 and k_2 obtained by the universal property of (i) and satisfying $e_2k_1 = g_1$ and $e_1k_2 = g_2$. Regions (†), (‡), and $(\star)$ are pushouts by Lemma 5.1.19, and one now goes on to construct *(iv)* by taking successive pushouts.

The sequential-independent derivation $C \Longrightarrow_{p_1,f_1} D_1 \Longrightarrow_{p_2,f_2'} D$ may now be constructed with the pushout squares below.

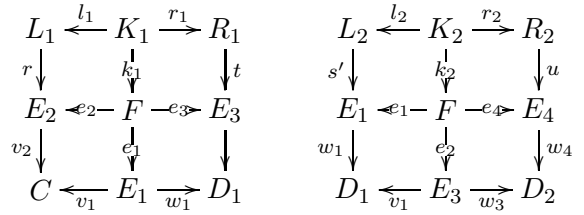


(2)⇒(1): Suppose that $C \Longrightarrow_{p_1,f_1} D_1$ and $D_1 \Longrightarrow_{p_2,f_2'} D$ are sequential-independent derivations. Form the pullback (i) below.



The two regions in diagram (ii) are pushouts [see diagram (5.3)]. Combining the two diagrams gives diagram (iii), with k_1 and k_2 obtained by the universal property of (i) and satisfying $e_3k_2 = g_2'$ and $e_1k_1 = g_1$. Regions (†), (‡), and $(\star)$ are pushouts by Lemma 5.1.19, and one now goes on to construct diagram (iv) by taking successive pushouts.

The parallel independent derivations $C \Longrightarrow_{p_1,f_1} D_1$ and $C \Longrightarrow_{p_2,f_2} D_2$ may now be constructed with the pushout squares below.



The proof of (1)⇔(3) is similar; the proof of (1)⇔(4) is a straightforward exercise in the theory of extensive categories. □

In fact, the proof that (1)⇒(2) remains valid more generally in the context of left-linear productions, but the proof of the converse requires linearity.

Concurrency theorem. The original concurrency theorems were proved for graph grammars [18] and later generalised to high-level replacement categories (see §5.2.2) in [23] which satisfy additional axiom sets, there called hlr2 and hlr2*. Roughly, the concurrency theorem states that given two derivations in a sequence, together with information about how they are related, one may construct a single derivation which internalises the two original derivations and performs them "concurrently". Moreover, one may reverse this process and deconstruct a concurrent derivation into two related sequential derivations. Here we state and prove the concurrency theorem for adhesive grammars without the need for extra axioms.

We shall first need to recall the notions of dependency relation, dependent derivation and concurrent production.

Definition 5.2.11 (Dependency Relation). Suppose that p_1 and p_2 are linear productions. A *dependency relation* for $\langle p_1, p_2 \rangle$ is an object X together with arrows $s : X \to R_1$ and $t : X \to L_2$ for which r_1, s, t, and l_2 can be incorporated into a diagram

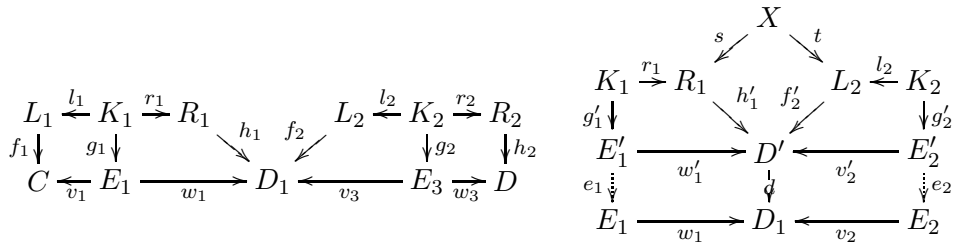
$$\begin{array}{ccccccc} & & & & X & & & & \\ & & & \swarrow^{s} & & \searrow^{t} & & & \\ K_1 & \xrightarrow{r_1} & R_1 & & & & L_2 & \xleftarrow{l_2} & K_2 \\ {\scriptstyle g_1'}\downarrow & & & \searrow^{h_1'} & & \swarrow_{f_2'} & & & \downarrow{\scriptstyle g_2'} \\ E_1' & & \xrightarrow[w_1']{} & & D' & & \xleftarrow[v_2']{} & & E_2' \end{array} \tag{5.4}$$

in which all three regions are pushouts.

Definition 5.2.12 (Dependent Derivation). Consider a derivation

$$C \Longrightarrow_{p_1,f_1} D_1 \Longrightarrow_{p_2,f_2} D$$

as illustrated in diagram (i) below, and a dependency relation X for $\langle p_1, p_2 \rangle$.



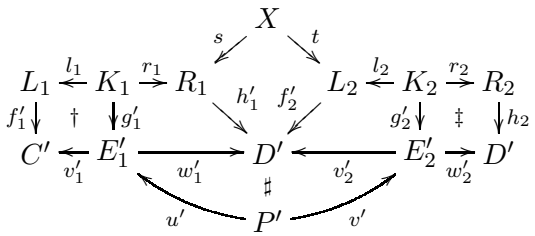
(i) (ii)

The derivation is said to be *X-dependent* if $h_1 s = f_2 t$ and there exist morphisms $e_1 : E_1' \to E_1$ and $e_2 : E_2' \to E_2$ satisfying $e_1 g_1' = g_1$ and $e_2 g_2' = g_2$, and if moreover the unique map $d : D' \to D_1$ satisfying $dh_1' = h_1$ and $df_2' = f_2$ also satisfies $dw_1' = w_1 e_1$ and $dv_2' = v_2 e_2$ [see diagram (ii)].

Definition 5.2.13 (Concurrent Production). Given a dependency relation X for $\langle p_1, p_2 \rangle$, the *X-concurrent production* $p_1 ;_X p_2$ is the span

$$C' \xleftarrow{v_1' u'} P' \xrightarrow{w_2' v'} D'$$

obtained by taking the bottom row of the following extension of Diagram (5.4)

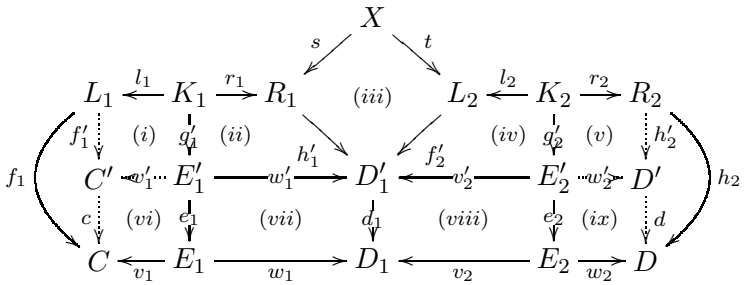


in which † and ‡ are pushouts and ♯ is a pullback.

Theorem 5.2.14 (Concurrency Theorem).

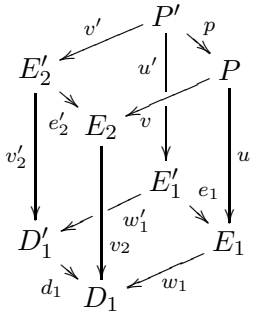
1. *Given an X-dependent derivation* $C \Longrightarrow_{p_1,f_1} D_1 \Longrightarrow_{p_2,f_2} D$ *there exists an X-concurrent derivation* $C \Longrightarrow_{p_1;_X p_2} D$
2. *Given an X-concurrent derivation* $C \Longrightarrow_{p_1;_X p_2} D$*, there exists an X-dependent derivation* $C \Longrightarrow_{p_1,f_1} D_1 \Longrightarrow_{p_2,f_2} D$.

Proof. 1. Suppose that we have an X dependent-derivation, as in the solid part of the diagram

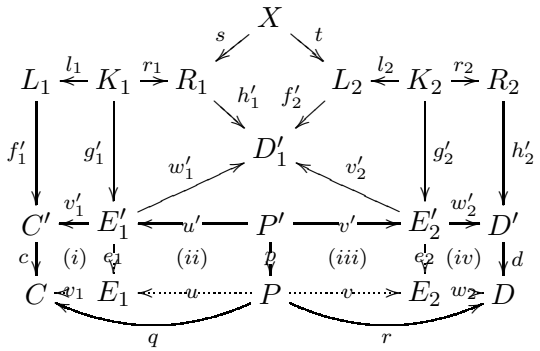


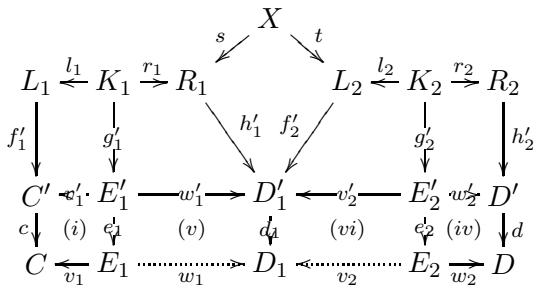
in which (iii), (i)+(vi), (ii)+(vii), (iv)+$(viii)$, (v)+(ix), (ii), and (iv) are pushouts, so that also (vii) and $(viii)$ are pushouts. Fill in the dotted parts of the diagram to obtain further pushouts (i), (v), (vi), and (ix).

By Lemma 5.1.15 both w_1' and v_2' are monomorphisms, and now by Lemma 5.1.16 both (vii) and $(viii)$ are pullbacks. Consider the cube in which bottom and front left faces are the pullbacks (vii) and $(viii)$, and the remaining faces are constructed so as to be pullbacks. Since the bottom face is also a pushout, so is the top face. Similarly, since the front left face is a pushout, so is the back right face.



2. Suppose that we have an X-concurrent derivation $C \Longrightarrow_{p_1;_X p_2,f} D$, as illustrated by the solid part of the first diagram below.





We construct pushouts (ii) and (iii) and obtain v_1 and w_2 using the universal properties. It now follows that (i) and (iv) are also pushouts. Now construct

the pushout (v); since (ii)+(v) and (iii) are pushouts, there is a unique map $v_2 : E_2 \to D_1$ so that (vi) is a pushout and (iii)+(vi) equals (ii)+(v). The diagram second diagram above now provides the required X-dependent derivation $C \longrightarrow\!\!\triangleright_{p_1, cf_1'} D_1 \longrightarrow\!\!\triangleright_{p_2, d_1 f_2'} D$. □

5.3 Quasiadhesive categories

The notion of adhesivity is too strong for some relevant examples, and therefore, it is useful to study weaker notions. Notably, such examples often arise in the theory of algebraic specifications.

Instead of requiring all pushouts along monomorphisms to be well behaved, quasiadhesive categories have well-behaved pushouts along *regular* monomorphisms, that is, only such pushouts are required to be VK-squares. Recall that a regular monomorphism is one that is an equaliser of two morphisms. Recently, Ehrig et al. [24] have considered categories where pushouts along a particular class of monomorphisms are van Kampen, which they have dubbed adhesive hlr-categories. Quasiadhesive categories fall under this general umbrella and cover many of the examples of adhesive hlr-categories.

After motivating the need for a weaker version of adhesivity in 5.3.1, we shall introduce the definition of quasiadhesive categories in 5.3.2. Finally, in 5.3.3 we shall consider some of their properties.

5.3.1 Structures and algebraic specifications

To motivate this development, we introduce the category of structures over a fixed set of *predicates* $\mathscr{P}$ [23]; that is a set with an associated arity function ar, which associates an arity $n \geq 0$ to each member of $\mathscr{P}$.

Definition 5.3.1. *A* structure R *is a pair* $\langle R_0, R_1 \rangle$ *where* R_0 *is a* set of atoms*, and* R_1 *is an arbitrary* set of formulas

$$R_1 \subseteq \{\, P(x_1, \ldots, x_n) \mid P \in \mathscr{P},\, ar(P) = n,\, x_1, \ldots, x_n \in R_0 \,\}.$$

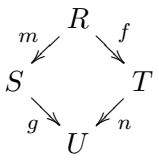
A structure morphism $f : R \to R'$ *is a function* $f_0 : R_0 \to R_0'$ *such that if* $P(x_1, \ldots, x_n) \in R_1$ *then* $P(f_0 x_1, \ldots, f_0 x_n) \in R_1'$.

Structures and structure morphisms form a category $\mathbf{Str}_{\mathscr{P}}$.

Structures can be considered as a kind of generalisation of (directed) graphs; one may consider graphs to be sets equipped with binary predicates for edges. The analogy is not perfect, however: if we take $\mathscr{P}$ to be the

singleton binary predicate $\{E\}$ then structures are graphs with at most one edge between two vertices.

It is easy to show that $\mathbf{Str}_{\mathscr{P}}$ has arbitrary pullbacks and pushouts. Indeed, in order to obtain a pushout diagram given structure morphisms $m : R \to S$ and $f : R \to T$,



let U_0, n_0 and g_0 be the morphisms induced by taking the pushout in **Set**. The set of formulas U_1 is taken to be the least collection which makes n and g into structure morphisms, explicitly:

$$U_1 = \{\, P(g_0x_1, \ldots, g_0x_n) \mid P(x_1, \ldots, x_n) \in S_0 \,\} \\ \cup \{\, P(n_0x_1, \ldots, n_0x_n) \mid P(x_1, \ldots, x_n) \in T_0 \,\}.$$

Similarly, in order to obtain a pullback in **Str** given structure morphisms $g : S \to U$ and $n : T \to U$, we first obtain R_0, m_0 and f_0 by taking the pullback in **Set**. We obtain the set of formulas R_1 by forming the greatest collection of formulas which makes m and f into structure morphisms. Thus,

$$R_1 = \{\, P(x_1, \ldots, x_n) \mid P(m_0x_1, \ldots, m_0x_n) \in S_1 \text{ and } P(f_0x_1, \ldots, f_1x_n) \in T_1 \,\}$$

Definition 5.3.2. A structure morphism $m : R \to S$ is *strict injective* [23] if it is injective and, additionally, if $P(m_0x_1, \ldots, m_0x_n) \in S_1$ then $P(x_1, \ldots, x_n)$ is in R_1.

Lemma 5.3.3. The class of regular monomorphisms $m : R \to S$ of $\mathbf{Str}_{\mathscr{P}}$ coincides with the class of strict injective morphisms.

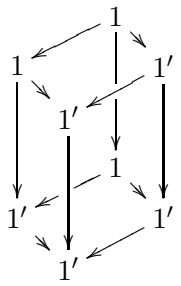
Proof. It is easy to check that taking an equaliser of two morphisms results in a strict injective morphism. Conversely, suppose we have a strict injective morphism $m : R \to S$. Let 2 denote the structure with a two element set $\{\top, \bot\}$ and all formulas. Let $\top : S \to 2$ be the structure morphism constant at $\top$ and let $[m] : S \to 2$ denote the structure morphism which takes $x \in S_0$ to $\top$ if x is the range of m, and $\bot$ otherwise.

$$R \xrightarrow{m} S \underset{\top}{\overset{[m]}{\rightrightarrows}} 2$$

It is easy to check that m is an equaliser of $[m]$ and $\top$. □

Notice that a simple corollary of Lemma 5.3.3 is that $\mathbf{Str}_{\mathscr{P}}$ is not a topos, since in toposes every mono is regular. In fact, the category $\mathbf{Str}_{\mathscr{P}}$ is not, in general, adhesive. For a simple counterexample, let $\mathscr{P}$ be the singleton containing a unary predicate P, let 1 denote the structure with a singleton set of atoms and no formulas, $1'$ to be the structure with a singleton set of atoms and a single formula $P(x)$.

Consider the cube below,



where all the morphisms are the unique morphisms between singleton sets. It is easily checked that the bottom face is a pushout, the back faces are pullbacks and the top face is a pushout. However, the front left face is not a pullback and, therefore, the bottom face is not a VK-square.

Indeed, it is immediately clear that the bottom face is not a VK-square because it is not a pullback (Lemma 5.1.16). On the other hand, if we restrict to pushouts along regular monomorphisms, we have:

Lemma 5.3.4. In $\mathbf{Str}_{\mathscr{P}}$, the following hold:

(i) pushouts along regular monomorphisms are pullbacks;

(ii) regular monomorphisms are stable under pushout.

In fact, as we shall prove in Lemma 5.3.6, in $\mathbf{Str}_{\mathscr{P}}$, pushouts along regular monomorphisms are VK-squares.

5.3.2 Quasiadhesive categories

The above example motivates the following definition.

Definition 5.3.5. A category $\mathbf{C}$ is said to be *quasiadhesive* if

(i) $\mathbf{C}$ has pushouts along regular monomorphisms;

(ii) $\mathbf{C}$ has pullbacks;

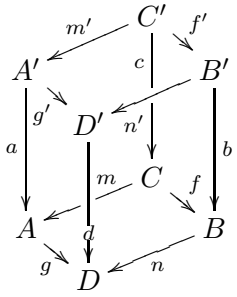
(iii) pushouts along regular monomorphisms are VK-squares.

Lemma 5.3.6. For any $\mathscr{P}$, $\mathbf{Str}_{\mathscr{P}}$ is quasiadhesive.

Proof. We have showed that $\mathbf{Str}_{\mathscr{P}}$ satisfies axioms (i) and (iii) of Definition 5.3.5. Regular monos are stable under pushout, as shown in Lemma 5.3.4.

To show that pushouts along regular morphisms are VK-squares, we first note that any pushout in $\mathbf{Str}_{\mathscr{P}}$ is stable under pullback.

It remains to show the following: given a cube



where the bottom and top faces are pushouts and the back faces are pullbacks, then the front faces are pullbacks. We shall show this for the front left face, the proof for the front right face is similar. In the following we write $\overline{x}$ for a sequence $x_1, \ldots, x_n$; thus for example $f\overline{x}$ denotes $fx_1, \ldots, fx_n$.

Suppose that, for some $\overline{x} \in A'$, we have $P(g'\overline{x}) \in D'_1$ and $P(a\overline{x}) \in A_1$ and $P(\overline{x}) \notin A'$. Then, since the top face is a pushout, there exists $\overline{x'} \in B'$ such that $P(\overline{x'}) \in B'_1$ and $n'\overline{x'} = g'\overline{x}$. Now, $P(b\overline{x'}) \in B_1$. Since the bottom square is a pullback, there exists $\overline{x''} \in C$ such that $P(\overline{x''}) \in C_1$, $m\overline{x''} = a\overline{x}$ and $f\overline{x''} = b\overline{x'}$. The fact that the back right face is a pullback implies that there exists $\overline{x'''} \in C'_0$, with $P(\overline{x'''}) \in C'_1$, $c\overline{x'''} = \overline{x''}$ and $f'\overline{x'''} = \overline{x'}$. Using the fact that the front left face is a pullback in **Set**, we have $m'\overline{x'''} = \overline{x}$. Then, since m' is a homomorphism and $P(\overline{x'''}) \in C'_1$, we have that $P(\overline{x}) \in A'_1$, a contradiction. □

5.3.3 Properties of quasiadhesive categories

Most of the properties of adhesive categories have a "quasi" version; with the proofs essentially the same. Here we state a few of the more interesting ones.

Proposition 5.3.7. If $\mathbf{C}$ is a quasiadhesive category then the following hold:

(i) pushouts along regular monos are pullbacks;

(ii) pushout complements of regular monos are unique up to isomorphism;

One also has the closure properties shown for adhesive categories. In particular:

Proposition 5.3.8.

(i) If $\mathbf{C}$ and $\mathbf{D}$ are quasiadhesive categories then so is $\mathbf{C} \times \mathbf{D}$;

(ii) If $\mathbf{C}$ is quasiadhesive then so is $\mathbf{C}/C$ for any object C of $\mathbf{C}$, if additionally $\mathbf{C}$ has equalisers then $C/\mathbf{C}$ is also quasiadhesive;

(iii) If $\mathbf{C}$ is quasiadhesive then so is any functor category $[\mathbf{X}, \mathbf{C}]$.

Proof. We rely on the facts that pushouts and pullbacks are constructed pointwise in all of the above cases, the class or regular monomorphisms in $\mathbf{C} \times \mathbf{D}$ is the product of the regular monomorphisms of $\mathbf{C}$ and $\mathbf{D}$, the regular monomorphisms of $\mathbf{C}/C$ coincide with those of $\mathbf{C}$. To show that the regular monomorphisms of $C/\mathbf{C}$ coincide with those of $\mathbf{C}$ we need the additional assumption that $\mathbf{C}$ has equalisers. The regular monos of $[\mathbf{X}, \mathbf{C}]$ are exactly those natural transformations of which every component is regular mono. This follows from the fact that limits in functor categories are calculated pointwise. □

Similarly, one may develop a double-pushout rewriting theory inside a quasiadhesive category. The only adjustment is made to the definition of linear production (Definition 5.2.1); we would require the components of the span to be regular monomorphisms. One may then define a quasiadhesive grammar in the obvious way (see Definition 5.2.4), restate and obtain Theorems 5.2.10 and 5.2.14 for quasiadhesive grammars. We leave it to the reader to fill in the details.

Chapter 6

Cospans as contexts

In this chapter we shall construct GRPOs in a general framework of abstract, uninterpreted contexts over an arbitrary adhesive base category $\mathbf{C}$: the cospan bicategory $\mathrm{Cospan}^{\cong}(\mathbf{C})$ with isomorphic 2-cells. Such bicategories have the same objects as $\mathbf{C}$, but the arrows are cospans $I \xrightarrow{\iota} C \xleftarrow{o} J$ of arrows in $\mathbf{C}$, which can be viewed as an object C enriched with an "input" interface ι and and "output" interface o. Intuitively, ι is the partial view of C attainable from its holes, while o is the restricted view of C afforded to the environment. Composition of cospans is performed by pushing out along the shared interfaces, which can be understood intuitively as glueing together an agent and its context along their common interface. Due to the nature of pushouts, composition is only associative up to a unique isomorphism.

The main result of this chapter is Theorem 6.2.5 which includes a construction of GRPOs in a class of cospan bicategories, which in turn allows the derivation of an lts for any reactive systems over such a bicategory. Specifically, we require a linearity condition on the input interfaces, namely, that ι is mono. Additionally, our cospans are over adhesive categories [59], which are categories in which pushouts along monomorphisms exist and are suitably well-behaved.

As we prove in the paper, adhesive categories have enough structure for the construction of GRPOs in our cospan bicategories.

In order to prove the relevance and usefulness of the construction, we shall treat two examples. Firstly, we apply it to derive lts for *double-pushout* (dpo) *graph-rewriting* systems. Graph rewriting is a well-established field of theoretical computer science [19], concerned with the extension of rewriting techniques from terms to graph structures. dpo graph rewriting can be generalised nicely to rewriting in arbitrary adhesive categories [59].

Since any arbitrary dpo graph-rewriting system can be seen as a reactive

systems on the bicategory Cospan$^{\cong}$(**Graph**), the bicategory of cospans over the (adhesive) category of graphs, we can derive an lts for any such graph rewriting system. This equips any arbitrary graph rewriting system with a contextual semantics and a corresponding coinduction principle, so as to allow for the transfer of concepts and techniques from the field of process algebra to graph-rewriting. In other words, this yields a behavioural equivalence based uniquely on the interactions of (concurrent) dynamic systems with their environment, while the presence of a well-behaved lts allows the use of coinduction to prove contextual equivalence.

When restricting cospans to purely linear (mono) maps, the concrete lts we derive agrees exactly with Ehrig and König's recently proposed approach, the so-called rewriting with borrowed contexts [25]. Consequently, Ehrig and König's congruence theorem can be understood as a corollary of the congruence theorem (Theorem 2.3.6) for GRPOs. Without the restriction, the application of reactive systems to graph rewriting extends the borrowed-context approach by considering graph contexts where the output interface need not be injective. In this application, therefore, the paper contributes in two ways. Firstly, it is an extension of the results of Ehrig and König; secondly, it provides a missing link between their work and the work of Leifer and Milner [66].

Our second application is the construction of GRPOs for a version of Milner's *bigraphs* [73]. Bigraphs have been recently proposed as a formalism to model mobility of communication channels, or links (as in the π calculus), together with spatial mobility of agents, or places (as in distributed calculi). We introduce the adhesive category of place-link graphs. The cospan bicategories over place-link graphs resemble Milner's bigraphs, with some differences imposed by the respective linearity conditions. The general construction of GRPOs provides reactive systems over our bigraphs with a labelled transition semantics.

We shall start in section 6.1 by extending the framework of reactive systems to bicategories. This does not introduce technical problems since GRPOs, being a type of bicolimit, are a natural bicategorical notion. In section 6.2 we shall present the main contribution of this chapter: the construction of GRPOs in arbitrary input-linear cospan bicategories over adhesive categories. Finally, in section 6.3 we shall present two immediate applications of this construction in the areas of dpo graph transformation and Milner's bigraphs.

6.1 Cospan bicategories

A *bicategory* [7] can be described, roughly, as a 2-category where associativity and identity laws of horizontal composition hold up to isomorphisms. We shall denote all associativity isomorphisms by a, as for example, $a : h(gf) \Rightarrow (hg)f$. The isomorphisms are required to respect the well-known coherence axioms.

We shall first recall the notion of cospan bicategory in §6.1.1 and proceed in §6.1.2 to extend the notion of reactive system so that it allows underlying bicategories of contexts.

6.1.1 Bicategories and cospans

We shall assume that $\mathbf{C}$ is an adhesive category with *chosen pushouts*. That is, for arrows $m : C \to A$ and $f : C \to B$, there exists a unique "chosen" object $A +_C B$ and arrows $i : A \to A +_C B$ and $i : B \to A +_C B$ such that the resulting square is a pushout. By the universality of pushouts, given any other object D and arrows $g : A \to D$ and $n : B \to D$ which render the resulting square a pushout, there exists a unique isomorphism $\alpha : A+_C B \to D$ such that $\alpha i_{A\to A+B} = g$ and $\alpha i_{B\to A+B} = n$. We shall adopt the convention of *always* labelling the morphisms into the chosen pushout by i in diagrams, and decorating them with the domain and codomain in the subscript when referring to them in algebra.

The bicategory of cospans Cospan($\mathbf{C}$) has the same objects as $\mathbf{C}$, but arrows from I_1 to I_2 are cospans.

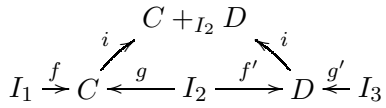
$$I_1 \xrightarrow{\ f\ } C \xleftarrow{\ g\ } I_2$$

We will denote such cospans $C^g_f : I_1 \to I_2$ or $C^{g:I_2}_{f:I_1}$, and omit f (respectively g) when I_1 (respectively I_2) is an initial object. We shall refer to I_1 and I_2 as the input and the output interfaces of C^g_f. Intuitively, we can think of a cospan as a generalised context, where C are the internals, (the image via g of) I_2 represents the public view of C, and (the image via f of) I_1 the view of C afforded to the 'holes' in it.

A 2-cell $h\colon C^g_f \Rightarrow C'^{g'}_{f'} : I_1 \to I_2$ is an arrow $h\colon C \to C'$ in $\mathbf{C}$ satisfying $hf = f'$ and $hg = g'$. The 2-cells that are iso (i.e. invertible) provide a canonical notion of "structural congruence." We shall denote the bicategory of cospans which has the 2-cells limited to isomorphisms by Cospan$^{\cong}$($\mathbf{C}$).

Given cospans $C^g_f : I_1 \to I_2$ and $D^{g'}_{f'} : I_2 \to I_3$, their composition $D^{g'}_{f'} \circ C^g_f : I_1 \to I_3$ is the cospan $(C +_{I_2} D)^{i_{D\to C+D}.g'}_{i_{C\to C+D}.f} : I_1 \to I_3$, as illustrated

by the pushout diagram below.



Note that in the resulting composition, I_2 is "forgotten." Composition is associative up to a unique isomorphism. It is easy to check that the associativity isomorphisms satisfy the coherence axioms, and thus yield a bicategory

In the construction of section 6.2 we shall need certain linearity restrictions. In particular, we shall work with input-linear cospans, as defined below.

Definition 6.1.1 (Linearity). A cospan C_m^g is said to be *input-linear* when m is a mono. A cospan C_m^n is said to be *linear* when both m and n are mono.

When working over an adhesive category, a simple corollary of the first part of Lemma 5.1.15 is that the composition of two input-linear cospans yields an input-linear cospan. Similarly, composition preserves linearity.

Definition 6.1.2 (Linear Cospans). Assuming that $\mathbf{C}$ is adhesive, let ILC($\mathbf{C}$) be the bicategory consisting of input-linear cospans and isomorphic 2-cells. Similarly, let LC($\mathbf{C}$) be the bicategory of linear cospans and isomorphic 2-cells.

6.1.2 Reactive systems over bicategories

Recall that the intuition behind the 2-dimensional structure is that, while arrows of the underlying category are viewed as contexts, the (isomorphic) 2-cells are thought of as "proofs of structural congruence" between contexts. In the particular case of the bicategory Cospan$^{\cong}$(**Graph**), the 2-cells are precisely graph isomorphisms which respect the input and output interfaces.

It is easy to extend the notion of reactive system so that it has an underlying bicategory.

Definition 6.1.3 (Reactive System). A *reactive system* $\mathbb{C}$ consists of

1. a bicategory $\mathbf{B}$;
2. a collection $\mathbf{D}$ of arrows of $\mathbf{B}$ called the *reactive contexts*; it is required to be closed under isomorphic 2-cells and composition-reflecting (see below);
3. a distinguished object $0 \in \mathbf{B}$;

4. a set of *reaction rules* $\mathcal{R}$, it consists of pairs of arrows $\langle l, r\rangle$ with domain 0. The members l, r of any given pair $\langle l, r\rangle \in \mathcal{R}$ have the same codomain.

Recall that the reactive contexts are those inside which evaluation may occur. Requiring $\mathbf{D}$ to be composition-reflecting means that $dd' \in \mathbf{D}$ implies d and $d' \in \mathbf{D}$, while the closure property means that given $d \in \mathbf{D}$ and an isomorphism $\rho\colon d \Rightarrow d'$ in $\mathbf{B}$ implies $d' \in \mathbf{D}$.

The reaction relation $\longrightarrow\!\triangleright$ is defined by taking $a \longrightarrow\!\triangleright dr$ if there is $\langle l, r\rangle \in \mathcal{R}$, $d \in \mathbf{D}$ and $\alpha\colon dl \Rightarrow a$. This represents that, up to structural congruence α, a is the left-hand side l of a reduction rule in a reaction context d.

The notion of bicolimit, as presented in Definition 3.1.7 of Chapter 3 is a bicategorical notion. Since GRPOs are a particular type of bicolimit, this means that we are free to speak about GRPOs and GIPOs in bicategories. In particular, we can still use the elementary presentation of Definition 2.2.9, taking care to add the coherent isomorphisms where necessary. Moreover, the proofs of the crucial properties of GRPOs and GIPOs in G-categories as presented in section 3.2 are easily extended to the more general setting of bicategories with isomorphic 2-cells, although we shall omit the details here.

Correspondingly, we obtain the concrete and abstract variants of labelled transition systems (Definitions 2.2.14 and 2.2.16) constructed using GIPOs, as defined in Chapter 2 Moreover, the congruence theorems for bisimilarity as well as trace and failures preorders follow

6.2 Constructing GRPOs for input-linear cospans

Let $\mathbf{C}$ be an adhesive category. In this section we shall prove that ILC($\mathbf{C}$) has GRPOs. In fact, we shall present a general construction of GRPOs in input-linear cospan bicategories. The result uses only pure category theory, yet as a consequence of the technology developed in Chapter 2, it has many applications, some of which we shall discuss in section 6.3. Indeed, any reactive system over ILC($\mathbf{C}$) has redex-GRPOs and therefore can be given a canonical labelled transition system semantics. We shall conclude this section with several examples of GRPOs in the bicategory ILC(**Graph**) – the bicategory of input-linear cospans over the category of directed graphs. The examples are ad-hoc and have no computational meaning, they are included so as to give an intuition to how GRPOs are constructed and to what they look like.

The main steps involved in the construction of a GRPO are outlined in Algorithm 6.2.2. The proof of correctness of the algorithm is quite technical and we shall present it in much detail, perhaps more detail than is customary for mathematical publications. This has been done in order to make the intended audience as broad as possible. At this point one should emphasise that the effort is justified; we are proving *one* theorem of pure category theory, instead of proving many small theorems about particular instances, say in the bicategory ILC(**Set**), ILC(**Graph**) or the category of bigraphs. Much of the detail is checking that various equations, required in the definition of GRPOs, hold. Many of these equations are shown to hold using routine "diagram chasing".

In order to increase its readability, the proof is divided into two sections. We shall start in §6.2.1 by proving Lemma 6.2.3 which states that the construction outlined in Algorithm 6.2.2 indeed results in a candidate. Secondly, in §6.2.2, we shall prove Lemma 6.2.4 which confirm that this candidate satisfies a universal property. In §6.2.3 we shall demonstrate a simple characterisation of GIPOs which is useful when actually deriving the labels of an lts.

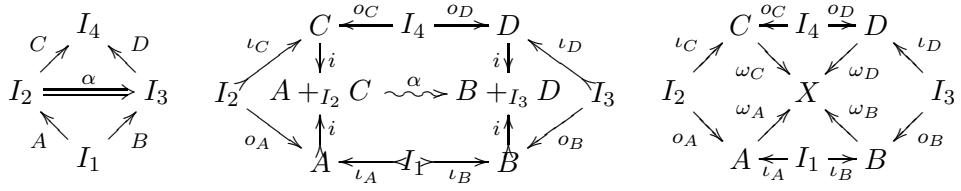
6.2.1 Construction

We shall, in Algorithm 6.2.2, outline a procedure for the construction of the desired minimal candidate of the arbitrary square illustrated in diagram (i). It is an algorithm in the sense that it gives a procedure of how, given diagram (i), to construct the minimal candidate. We shall not consider the computability or the efficiency of the individual steps in the construction.

The proof that applying the construction indeed results in a GRPO is divided into two lemmas. First, in Lemma 6.2.3 we shall prove that the components constructed in Algorithm 6.2.2 form a candidate for the redex square (i). Secondly, in Lemma 6.2.4, we shall prove that this candidate is indeed the minimal one; that is, we shall show that it satisfies the appropriate universal property, as specified in Definition 2.2.9.

An arbitrary square in ILC(**C**), as illustrated in diagram (i) below, amounts to a commutative diagram (ii) in **C**, with α an isomorphism. We shall adopt a loose convention of sometimes using $\rightsquigarrow$ to denote the iso-

morphisms of **C** which correspond to the 2-cells of ILC(**C**).



(*i*) (*ii*) (*iii*)

Recall that given an object X, a subobject $[\mu : Y \to X]$ is an equivalence class of monomorphisms into X, where the equivalence relation is generated from the canonical preorder on monomorphisms into X: $\mu \leq \mu'$ if there exists $k : Y \to Y'$ such that $\mu' k = \mu$. We shall abuse notation by confusing the subobject (equivalence classes of monos) with its representative μ (one mono).

Definition 6.2.1 (Frame of reference). Given a diagram (i) in ILC(**C**), by a *frame of reference* we shall mean an an arbitrary object X equipped with isomorphisms $\omega_l : A +_{I_2} C \to X$ and $\omega_r : B +_{I_3} D \to X$ such that $\omega_r^{-1}\omega_l = \alpha$. Let $\omega_A = \omega_l i_{A \to A+C} : A \to X$, $\omega_C = \omega_l i_{C \to A+C} : C \to X$, $\omega_B = \omega_r i_{B \to B+D} : B \to X$ and $\omega_D = \omega_r i_{D \to B+D} : D \to X$.

Clearly, a frame of reference always exists. For instance, one could let X equal $B +_{I_3} D$, let ω_r be identity and ω_l be α. Notice that:

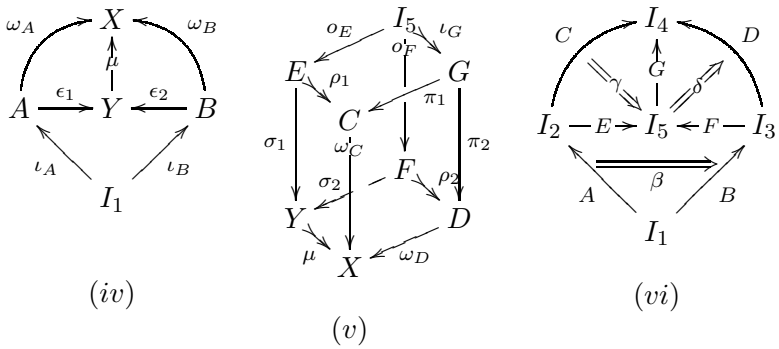
$$\omega_A \iota_A = \omega_l (i_{A \to A+C}) \iota_A = \omega_r \alpha (i_{A \to A+C}) \iota_A = \omega_r (i_{B \to B+D}) \iota_B = \omega_B \iota_B \text{ and}$$

$$\omega_C o_C = \omega_l (i_{C \to A+C}) o_C = \omega_r \alpha (i_{C \to A+C}) o_C = \omega_r (i_{D \to B+D}) o_D = \omega_D o_D,$$

which amounts to saying that diagram (iii) is commutative. Notice that both of the squares in diagram (iii) are actually pushouts.

Since ω_A and ω_B are readily seen to be mono, being pushouts of monos in an adhesive category (see Lemma 5.1.15), we are able to obtain the subobject union – an object $Y = A \cup B$ and monomorphisms $\mu : Y \to X$, $\epsilon_1 : A \to Y$ and $\epsilon_2 : B \to Y$ satisfying $\mu\epsilon_1 = \omega_A$ and $\mu\epsilon_2 = \omega_B$. Notice that since μ is mono and $\mu\epsilon_1\iota_A = \omega_A\iota_A = \omega_B\iota_B = \mu\epsilon_2\iota_B$ we have that $\epsilon_1\iota_A = \epsilon_2\iota_B$. We

obtain the commutative diagram (iv).



(iv) (v) (vi)

Algorithm 6.2.2 (GRPO Construction in ILC(**C**)**).** The construction of the components of the minimal candidate is outlined below. They are illustrated in diagrams (v) and (vi). We obtain:

1. G as the pullback of $\omega_C : C \to X$ and $\omega_D : D \to X$;

2. E as the pullback of $\mu : Y \to X$ and $\omega_C : C \to X$;

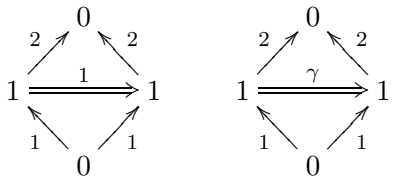
3. F as the pullback of $\mu : Y \to X$ and $\omega_D : D \to X$;

4. I_5 as the pullback of $\rho_2 \colon F \to D$ and $\pi_2 \colon G \to D$; Notice that due to the properties of pullbacks, we obtain a morphism $o_E : I_5 \to E$ such that all the faces of (v) are pullbacks.

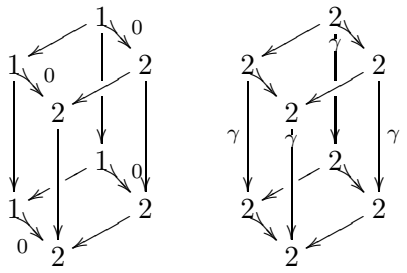
5. The input and output interfaces of E, F and G, as well as isomorphisms $\beta : A +_{I_2} E \to B +_{I_3} F$, $\gamma : C \to E +_{I_5} G$ and $\delta : F +_{I_5} G \to D$ which form the 2-cells of diagram (vi) follow in a canonical way. We shall show this in detail in the proof of Lemma 6.2.3.

Notice that it is *critical* that we fix a particular isomorphism $\alpha : A +_{I_2} C \to B +_{I_3} D$. Indeed, starting with two different isomorphism $\alpha_1, \alpha_2 : A +_{I_2} C \to B +_{I_3} D$, the construction of Algorithm 6.2.2 may give different results. This is to be expected since, as we argued in §2.2.1, the 2-dimensional structure is necessary if we are to have a satisfactory notion of minimal candidate. To illustrate this point, we shall consider a very simple example in the bicategory ILC(**Set**). Indeed, consider the following two

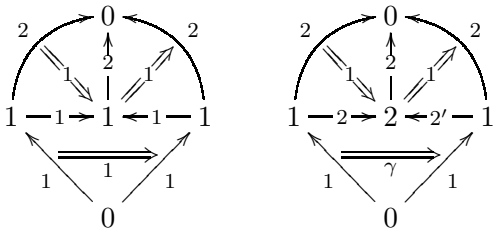
diagrams in ILC(**Set**)



where the arrow $0 \xrightarrow{1} 1$ is the cospan $0 \xrightarrow{\text{id}} 1 \xleftarrow{\text{id}} 1$ while the arrow $1 \xrightarrow{2} 0$ denotes the cospan $1 \xrightarrow{0} 2 \xleftarrow{!} 0$ where $1 \xrightarrow{0} 2$ "picks out" element $0 \in 2$. There are clearly two possible bijections $2 \to 2$, the identity and the map $\gamma : 2 \to 2$ which swaps the two elements. Notice that both the isomorphisms fit in as 2-cells in the diagrams; any isomorphism respects empty input and output interfaces. The GRPOs of the two diagrams are very different. First we illustrate the two resulting cubes which result from the steps outlined in Algorithm 6.2.2 and which correspond to diagram (v).



The two resulting candidates are illustrated below



where $1 \xrightarrow{1} 1$ is just the identity cospan $1 \xrightarrow{\text{id}} 1 \xleftarrow{\text{id}} 1$, $1 \xrightarrow{2} 2$ is the cospan $1 \xrightarrow{0} 2 \xleftarrow{\text{id}} 2$ and $1 \xrightarrow{2'} 2$ is the cospan $1 \xrightarrow{0} 2 \xleftarrow{\gamma} 2$.

Intuitively, in the first case, the two lower arrows (A and B) correspond to the same element via the contexts and the identity isomorphism. Thus the context contains redundant information which can be factored out resulting in a candidate which is just the identity context. Conversely, they

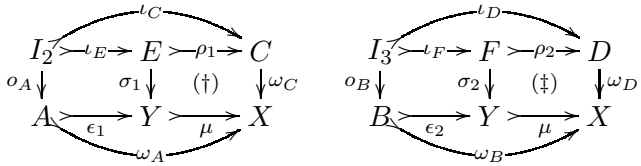
are different in the second case, meaning that the context has to be kept in the minimal candidate. The reader should compare this example to the situation in the G-category of bunch contexts, as illustrated in Example 4.2.8.

Lemma 6.2.3. The construction as outlined in Algorithm 6.2.2 provides a candidate for square (i).

Proof. In order to prove the lemma, we are required to:

1. demonstrate the existence of appropriate input and output interfaces for each of E, F and G;
2. show that there exist isomorphisms $\beta : A +_{I_2} E \to B +_{I_3} F$, $\gamma : C \to E +_{I_5} G$ and $\delta : F +_{I_5} G \to D$;
3. show that these isomorphisms paste together in way which results in a 2-cell equal to $\alpha : A +_{I_2} C \to B +_{I_3} D$.

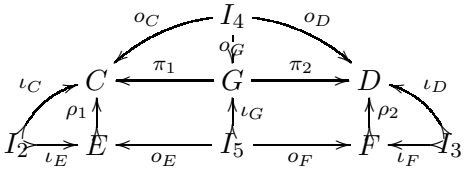
Recall that the components constructed in Algorithm 6.2.2 can be organised conveniently in the cube of pullbacks illustrated in diagram (v). The exterior rectangles of diagrams (vii) and $(viii)$ are pushouts (as illustrated in diagram (iii)) and as such they are commutative.



(vii) $(viii)$

Using the universal property of pullbacks, we obtain $\iota_E : I_2 \to E$ satisfying $\sigma_1\iota_E = \epsilon_1 o_A$ and $\rho_1\iota_E = \iota_C$. Similarly, we obtain $\iota_F : I_3 \to F$ which satisfies the analogous equations: $\sigma_2\iota_F = \epsilon_2 o_B$ and $\rho_2\iota_F = \iota_D$. We can now use the fact that we are working in an adhesive category: by Lemma 5.1.20 the left hand squares of diagrams (vii) and $(viii)$ are pushouts, which in turn implies that regions (†) and (‡) are pushouts, using ordinary pushout pasting (Lemma 1.3.2). But regions (†) and (‡) are, respectively, the front left face and the bottom face of diagram (v). Thus, since all the side faces of diagram (v) are pullbacks and the bottom face is a pushout, we can conclude by adhesiveness (Definition 5.1.7) that the top face is a pushout. Similarly, the fact that the front left face is a pushout implies that the back right face is a pushout.

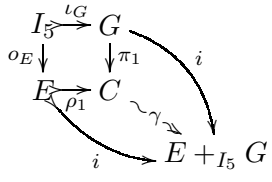
Using the fact that G was defined as the vertex of a pullback diagram, we obtain a unique morphism $o_G : I_4 \to G$ such that $\pi_1 o_G = o_C$ and $\pi_2 o_G = o_D$. Diagram (ix) illustrates the components of the candidate the existence of which we have derived above.



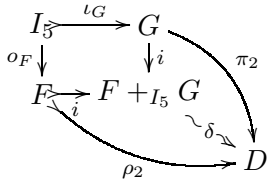
(ix)

Having constructed the basic building blocks of the candidate, we shall now return our attention to the bicategory ILC($\mathbf{C}$) and construct the appropriate isomorphic 2-cells β, γ and δ, as illustrated in diagram (vi).

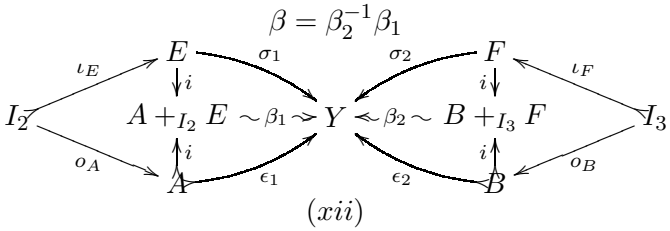
Let $E +_{I_5} G$ denote the chosen pushout of o_E and ι_G and let $\gamma : C \to E +_{I_5} G$ be the unique induced isomorphism, illustrated in diagram (x). Similarly, we obtain a unique isomorphism $\delta : F +_{I_5} G \to D$ as shown in diagram (xi). Referring back to diagrams (vii) and $(viii)$, we have $A +_{I_2} E \cong Y \cong B +_{I_3} F$. Let $\beta : A +_{I_2} E \to B +_{I_3} F$ denote the unique compatible isomorphism, illustrated in diagram (xii).



(x)



(xi)



(xii)

We first need to check that β, γ and δ are 2-cells in ILC($\mathbf{C}$), that is, that they preserve input and output interfaces.

First, using the fact that diagram (iv) is commutative, we have that $\epsilon_1 \iota_A = \epsilon_2 \iota_B$. We can use this equation to derive $\beta_1 (i_{A \to A+E}) \iota_A = \epsilon_1 \iota_A =$

$\epsilon_2 \iota_B = \beta_2(i_{B \to B+F})\iota_B$, which gives

$$\beta(i_{A \to A+E})\iota_A = (i_{B \to B+F})\iota_B. \tag{6.1}$$

Also, $\beta_1(i_{E \to A+E})o_E = \sigma_1 o_E = \sigma_2 o_F = \beta_2(i_{F \to B+F})o_F$, where we have used the fact that the back left face of diagram (v) is commutative. This gives

$$\beta(i_{E \to A+E})o_E = (i_{F \to B+F})o_F. \tag{6.2}$$

Equations (6.1) and (6.2) show, respectively, that β respects input and output interfaces and thus we can conclude that β is a 2-cell of ILC($\mathbf{C}$).

Checking γ, we have

$$\gamma \iota_C = \gamma \rho_1 \iota_E = (i_{E \to E+G})\iota_E \tag{6.3}$$

where we have used the fact that $\rho_1 \iota_E = \iota_C$ (see diagram (vii)). Also,

$$\gamma o_C = \gamma \pi_1 o_G = (i_{G \to E+G})o_G \tag{6.4}$$

where we have used the definition of o_G (diagram (ix)). Equations (6.3) and (6.4) imply that γ is a 2-cell of ILC($\mathbf{C}$). The fact that δ respects input and output interfaces and is, therefore, a 2-cell, follows by a similar argument.

It remains to show that the two cells of diagram (iii) paste together to give α, or precisely that:

$$(B +_{I_3} \delta)a(\beta +_{I_5} G)a(A +_{I_2} \gamma) = \alpha. \tag{6.5}$$

Recall that we use a to generically denote associativity isomorphisms. Rearranging this last equation, we shall show, equivalently, that:

$$a(\beta +_{I_5} G)a = (B +_{I_3} \delta^{-1})\alpha(A +_{I_2} \gamma^{-1}). \tag{6.6}$$

We shall first need to derive the following equation:

$$a(\beta_2^{-1} +_{I_5} G)(i_{Y \to Y+G}) = (B +_{I_3} \delta^{-1})\omega_r^{-1}\mu. \tag{6.7}$$

Since the left square of diagram $(viii)$ is a pushout, it suffices to check that the two sides of (6.7) are equal when precomposing with σ_2 and ϵ_2. Indeed, precomposing the left hand side with σ_2 yields

$$\begin{aligned} a(\beta_2^{-1} +_{I_5} G)(i_{Y \to Y+G})\sigma_2 &= a(i_{B+F \to (B+F)+G})\beta_2^{-1}\sigma_2 \\ &= a(i_{B+F \to (B+F)+G})(i_{F \to B+F}) \\ &= (i_{F+G \to B+(F+G)})(i_{F \to F+G}) \end{aligned}$$

using the definitions of β_2 (diagram (xii)) and the associativity isomorphism. Precomposing the right hand side with σ_2 and using the fact that $\mu\sigma_2 = \omega_D\rho_2$ (bottom face of diagram (v)) and the definitions of ω_r, ω_D and δ yields

$$\begin{aligned}(B +_{I_3} \delta^{-1})\omega_r^{-1}\mu\sigma_2 &= (B +_{I_3} \delta^{-1})\omega_r^{-1}\omega_D\rho_2 \\ &= (B +_{I_3} \delta^{-1})(i_{D\to B+D})\rho_2 \\ &= (i_{F+G\to B+(F+G)})\delta^{-1}\rho_2 \\ &= (i_{F+G\to B+(F+G)})(i_{F\to F+G}).\end{aligned}$$

Precomposing with ϵ_2 and using the definitions of β_2, associativity isomorphisms, ω_r and ω_B, and the commutativity of diagram (iv) respectively, allows us to derive

$$\begin{aligned}a(\beta_2^{-1} +_{I_5} G)(i_{Y\to Y+G})\epsilon_2 &= a(i_{B+F\to(B+F)+G})\beta_2^{-1}\epsilon_2 \\ &= a(i_{B+F\to(B+F)+G})(i_{B\to(B+F)}) \\ &= i_{B\to B+(F+G)} \\ &= (B +_{I_3} \delta^{-1})(i_{B\to B+D}) \\ &= (B +_{I_3} \delta^{-1})\omega_r^{-1}\omega_B \\ &= (B +_{I_3} \delta^{-1})\omega_r^{-1}\mu\epsilon_2.\end{aligned}$$

This concludes the verification of equation (6.7). Coming back to equation (6.6), we use the definitions of a and β_1 to obtain

$$\begin{aligned}&a(\beta +_{I_5} G)a(i_{A\to A+(E+G)}) \\ =&a(\beta_2^{-1} +_{I_5} G)(\beta_1 +_{I_5} G)a(i_{A\to A+(E+G)}) \\ =&a(\beta_2^{-1} +_{I_5} G)(\beta_1 +_{I_5} G)(i_{A+E\to(A+E)+G})(i_{A\to A+E}) \\ =&a(\beta_2^{-1} +_{I_5} G)(i_{Y\to Y+G})\beta_1(i_{A\to A+E}) \\ =&a(\beta_2^{-1} +_{I_5} G)(i_{Y\to Y+G})\epsilon_1. \quad (*)\end{aligned}$$

Now using

$$\begin{aligned}(B +_{I_3} \delta^{-1})\alpha(A +_{I_2} \gamma^{-1})(i_{A\to A+(E+G)}) &= (B +_{I_3} \delta^{-1})\alpha(i_{A\to A+C}) \\ &= (B +_{I_3} \delta^{-1})\omega_r^{-1}\omega_l(i_{A\to A+C}) \\ &= (B +_{I_3} \delta^{-1})\omega_r^{-1}\omega_A \\ &= (B +_{I_3} \delta^{-1})\omega_r^{-1}\mu\epsilon_1 \quad (**).\end{aligned}$$

Using equation (6.7) with equations (*) and (**), we get that

$$\begin{aligned} a(\beta +_{I_5} G)a(i_{A\to A+(E+G)}) &=_{(*)} a(\beta_2^{-1} +_{I_5} G)(i_{Y\to Y+G})\epsilon_1 \\ &=_{(6.7)} (B +_{I_3} \delta^{-1})\omega_r^{-1}\mu\epsilon_1 \\ &=_{(**)} (B + I_3\delta^{-1})\alpha(A +_{I_2} \gamma^{-1})(i_{A\to A+(E+G)}). \end{aligned}$$

which amounts to saying that the two sides of equation (6.6) are equal when precomposed with $i_{A\to A+(E+G)}$:

$$a(\beta +_{I_5} G)a(i_{A\to A+(E+G)}) = (B +_{I_3} \delta^{-1})\alpha(A +_{I_2} \gamma^{-1})(i_{A\to A+(E+G)}). \quad (6.8)$$

In order to complete the proof it remains only to verify that the two sides of equation (6.6) are equal when precomposed with $i_{E+G\to A+(E+G)}$. To obtain this equality we shall show that they are equal when precomposed with $(i_{E+G\to A+(E+G)})(i_{E\to E+G})$ and $(i_{E+G\to A+(E+G)})(i_{G\to E+G})$. Indeed, using the definitions of β and β_1 we have

$$\begin{aligned} a(\beta +_{I_5} G)a(i_{E+G\to A+(E+G)})(i_{E\to E+G}) &= a(i_{B+F\to(B+F)+G})\beta(i_{E\to A+E}) \\ &= a(i_{B+F\to(B+F)+G})\beta_2^{-1}\beta_1(i_{E\to A+E}) \\ &= a(i_{B+F\to(B+F)+G})\beta_2^{-1}\sigma_1 \\ &= a(\beta_2^{-1} +_{I_5} G)(i_{Y\to Y+G})\sigma_1 \end{aligned}$$

which, using equation (6.7), is equal to

$$\begin{aligned} (B +_{I_3} \delta^{-1})\omega_r^{-1}\mu\sigma_1 &= (B +_{I_3} \delta^{-1})\omega_r^{-1}\omega_C\rho_1 \\ &= (B +_{I_3} \delta^{-1})\alpha(i_{C\to A+C})\rho_1 \\ &= (B +_{I_3} \delta^{-1})\alpha(i_{C\to A+C})\gamma^{-1}(i_{E\to E+G}) \\ &= (B +_{I_3} \delta^{-1})\alpha(A +_{I_2} \gamma^{-1})(i_{E+G\to A+(E+G)})(i_{E\to E+G}) \end{aligned}$$

where we have used the fact that $\mu\sigma_1 = \omega_C\rho_1$ (front left face of diagram (v)), the definitions of ω_r and ω_C and the definition of γ.

Finally, using the definitions of δ, ω_D, ω_C and γ we can derive the

following:

$$\begin{aligned}
a(\beta +_{I_5} G)a(i_{E+G\to A+(E+G)})(i_{G\to E+G}) &= (i_{F+G\to B+(F+G)})(i_{G\to F+G}) \\
&= (i_{F+G\to B+(F+G)})\delta^{-1}\pi_2 \\
&= (B +_{I_3} \delta^{-1})(i_{D\to B+D})\pi_2 \\
&= (B +_{I_3} \delta^{-1})\omega_r^{-1}\omega_D\pi_2 \\
&= (B +_{I_3} \delta^{-1})\omega_r^{-1}\omega_C\pi_1 \\
&= (B +_{I_3} \delta^{-1})\alpha(i_{C\to A+C})\pi_1 \\
&= (B +_{I_3} \delta^{-1})\alpha(i_{C\to A+C})\gamma^{-1}(i_{G\to E+G}) \\
&= (B +_{I_3} \delta^{-1})\alpha(A +_{I_2} \gamma^{-1})(i_{E+G\to A+(E+G)})(i_{G\to E+G}).
\end{aligned}$$

The last two paragraphs allow us to conclude that

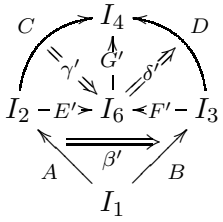
$$a(\beta+_{I_5}G)a(i_{E+G\to A+(E+G)}) = (B+\delta^{-1})\alpha(A+\gamma^{-1})(i_{E+G\to A+(E+G)}), \quad (6.9)$$

and we can conclude that that equation (6.6) holds using the fact that equations (6.8) and (6.9) both hold. □

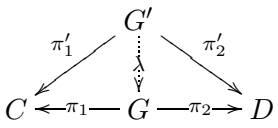
6.2.2 Universality

In this section, we shall show that the candidate constructed using Algorithm 6.2.2 is indeed the minimal candidate.

Lemma 6.2.4. The candidate constructed using Algorithm 6.2.2 satisfies the universal property of a GRPO. More precisely, given any other candidate for diagram (i), as illustrated in diagram $(xiii)$, there exists a mediating morphism:



$(xiii)$



(xiv)

a cospan $I_5 \xrightarrow{\iota_H} H \xleftarrow{o_H} I_6$ (an arrow $H : I_5 \to I_6$ of ILC($\mathbf{C}$)) and isomorphisms $\varphi : E' \to E +_{I_5} H$, $\psi : F +_{I_5} H \to F'$ and $\tau : H +_{I_6} G' \to G$ in

C which form 2-cells of ILC(**C**) and which satisfy the required equations:

$$(E +_{I_5} \tau)a(\varphi +_{I_6} G')\gamma' = \gamma, \quad \delta'(\psi +_{I_6} G')a(F +_{I_5} \tau^{-1}) = \delta$$

$$\text{and} \quad (B +_{I_3} \psi)a(\beta +_{I_5} H)a(A +_{I_2} \varphi) = \beta'.$$

Moreover, the mediating morphism $\langle H, \varphi, \psi, \tau \rangle$ is required to be essentially unique (see Definition 2.2.9). This means that for any other mediating morphism $\langle H', \varphi', \psi', \tau' \rangle$ which satisfies analogous equations, there exists a unique isomorphism $\xi : H \to H'$ which makes the two mediating morphisms compatible:

$$(E +_{I_5} \xi)\varphi = \varphi', \quad \psi(F +_{I_5} \xi^{-1}) = \psi' \quad \text{and} \quad \tau'(\xi +_{I_6} G') = \tau.$$

Proof. Consider another candidate, as illustrated in diagram (*xiii*). Then

$$(B +_{I_3} \delta')a(\beta' +_{I_6} G')a(A +_{I_2} \gamma') = \alpha. \tag{6.10}$$

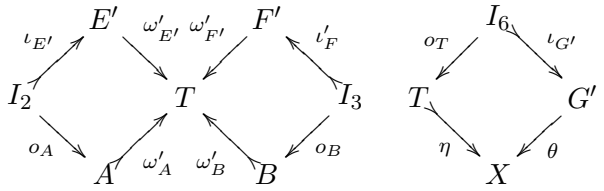
Letting $\pi_1' = {\gamma'}^{-1}(i_{G' \to E'+G'}) : G' \to C$ and $\pi_2{}' = \delta'(i_{G' \to F'+G'}) : G' \to D$, a simple diagram chase involving equation (6.10) yields $\alpha i_{C \to A+C} \pi_1' = i_{D \to B+D} \pi_2'$, which, using the definition of ω_C and ω_D can be written as $\omega_C \pi_1' = \omega_D \pi_2'$. Using the fact that the front right face of diagram (v) is a pullback, we obtain an arrow $\lambda : G' \to G$ such that $\pi_1 \lambda = \pi_1' : G' \to C$ and $\pi_2 \lambda = \pi_2' : G' \to C$, as illustrated in diagram (*xiv*). Note that since γ' is a 2-cell, it preserves interfaces and in particular $\gamma' o_C = (i_{G' \to E'+G'}) o_{G'}$ which in turn implies that $\pi_1' o_{G'} = o_C$. Similarly, using the fact that δ' is a 2-cell, we have $\pi_2' o_G' = o_D$. Since also $\pi_1 o_G = o_C$ and $\pi_2 o_G = o_D$ and the existence of o_G was inferred using the universal property of pullbacks, we use the fact the fact that such arrows are unique to obtain

$$\lambda o_{G'} = o_G. \tag{6.11}$$

Consider diagram (*xv*) below, where T, together with isomorphisms $\omega_l' : A +_{I_2} E' \to T$ and $\omega_r' : B +_{I_3} F' \to T$ is a frame of reference (Definition 6.2.1) for $\beta' : A +_{I_2} E' \to B +_{I_3} F'$ and ω_A', $\omega_{E'}'$, ω_B' and $\omega_{F'}'$ are defined in the standard way.

We have an isomorphism $\omega_r(B +_{I_3} \delta')a({\omega_r'}^{-1} +_{I_6} G') : T +_{I_6} G' \to X$, Let $\eta : T \to X$ and $\theta : G' \to X$ denote the corresponding morphisms derived by

precomposing with $i_{T\to T+G'}$ and $i_{G'\to T+G'}$.



(xv) (xvi)

Now notice that using equation (6.10) and the definitions of ω'_l, ω'_r and ω_A we have

$$\begin{aligned}\eta\omega'_A &= \omega_r(B +_{I_3} \delta')a(\omega'^{-1}_r +_{I_6} G')(i_{T\to T+G'})\omega'_l(i_{A\to A+E'})\\ &= \omega_r(B +_{I_e} \delta')a(\beta' +_{I_6} G')(i_{A+E'\to(A+E')+G'})(i_{A\to A+E'})\\ &= \omega_r(B +_{I_3} \delta')a(\beta' +_{I_6} G')a(A +_{I_2} \gamma')(i_{A\to A+C})\\ &= \omega_r\alpha(i_{A\to A+C})\\ &= \omega_A.\end{aligned}$$

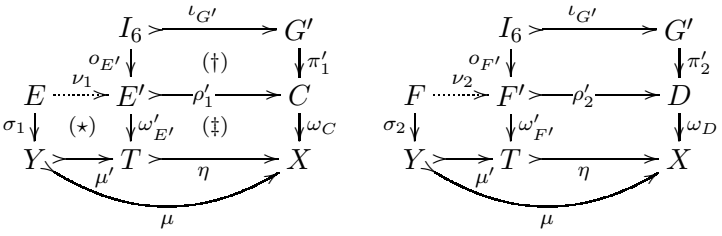
One can derive $\eta\omega'_B = \omega_B$ in a similar way. Recall from diagram (iv) that Y was defined as the subobject union of ω_A and ω_B. Thus, by definition of union, there exists a monomorphism $\mu' : Y \to T$ such that $\mu'\epsilon_1 = \omega'_A$ and $\mu'\epsilon_2 = \omega'_B$. It also follows that $\eta\mu' = \mu$.

In diagram $(xvii)$ below, we shall let ρ'_1 denote $\gamma'^{-1}(i_{E'\to E'+G'})$, while in diagram $(xviii)$ we shall use ρ'_2 to denote $\delta'(i_{F'\to F'+G'})$. Consider diagram $(xvii)$ again, and note that regions $(\dagger)$ and $(\dagger)+(\ddagger)$ are pushouts. Then if follows from ordinary pushout pasting (Lemma 1.3.2) that $(\ddagger)$ is also a pushout and, using the second part of Lemma 5.1.16, a pullback.

Using the fact that the front left face of diagram (v) is commutative, i.e. $\mu\sigma_1 = \omega_C\rho_1 : E \to X$ we can use the universal property of pullbacks to obtain a unique morphism $\nu_1 : E \to E'$ such that $\rho'_1\nu_1 = \rho_1$ and $\omega'_{E'}\nu_1 = \mu'\sigma_1$. A similar chain of reasoning involving diagram $(xviii)$ allows us to derive the existence of $\nu_2 : F \to F'$ which satisfies $\rho'_2\nu_2 = \rho_2$ and $\omega'_{F'}\nu_2 = \mu'\sigma_2$.

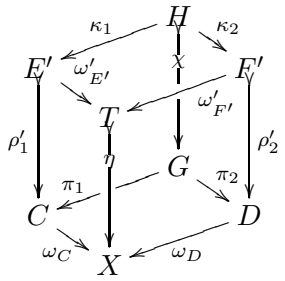
But then $(\star)+(\ddagger)$ is the front left face in the cube (v) the corresponding region in diagram $(xviii)$ is its bottom face. Since all the faces of diagram (v) are pullbacks, we can apply ordinary pullback pasting to infer that the left-most squares in the diagrams $(xvii)$ and $(xviii)$ are pullbacks, and therefore, that ν_1 and ν_2 are both mono, since they are pullbacks of mono μ' along,

respectively, $\omega'_{E'}$ and $\omega'_{F'}$.



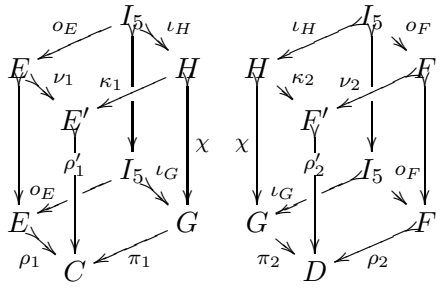
$(xvii)$ $(xviii)$

In diagram (xix), H and morphisms κ_1, κ_2 and χ are chosen so that the two rear faces are pullbacks. One could do this, e.g., by first taking the pullback of π_2 and ρ'_2 and obtaining κ_1 from the pullback property of the front left face.



(xix)

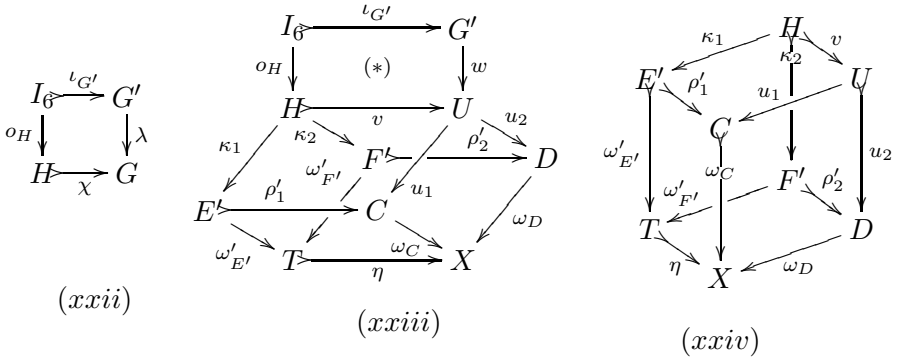
One may consider H as the mediating morphism from I_5 to I_6. Indeed, in diagrams (xx), (xxi) and we use adhesiveness in order to deduce that the top faces are pushouts. Notice that the front left face of diagram (xx) is a pullback precisely because $\rho'_1\nu_1 = \rho_1$ and ν_1 is a mono. The front right face of diagram (xxi) is a pullback for similar reasons.



(xx) (xxi)

Using the fact that β' is a 2-cell and therefore preserves the output interface, it is easy to verify that $\omega'_{E'} o_{E'} = \omega'_{F'} o_{F'}$. Because the top face of diagram (*xix*) is a pullback, we obtain a unique morphism $o_H : I_6 \to H$ such that $\kappa_1 o_H = o_{E'}$ and $\kappa_2 o_H = o_{F'}$.

We shall now show that diagram (*xxii*), below, is a pushout.



First notice that it is commutative, indeed, using respectively: the commutativity of the rear left face of diagram (*xix*), the definition of o_H, the commutativity of region (†) in diagram (*xvii*) and the definition of λ we obtain

$$\pi_1 \chi o_H = \rho'_1 \kappa_1 o_H = \rho'_1 o_{E'} = \pi'_1 \iota_{G'} = \pi_1 \lambda \iota_{G'}$$

and similarly, using respectively: the commutativity of the rear right face of diagram (*xix*), the definition of o_H, the commutativity of the upper rectangle of diagram (*xviii*) and the definition of λ we infer that

$$\pi_2 \chi o_H = \rho'_2 \kappa_2 o_H = \rho'_2 o_{F'} = \pi'_2 \iota'_G = \pi_2 \lambda \iota_{G'}.$$

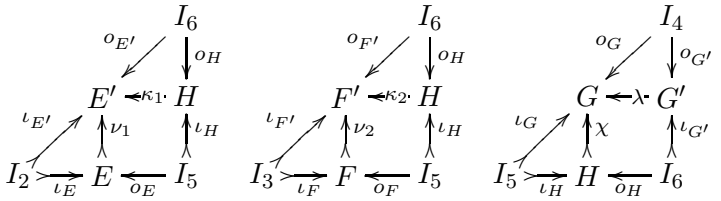
Using the universal property of the front right pullback of diagram (v) (which is also the bottom face of diagram (*xix*)) yields $\chi o_H = \lambda \iota_{G'}$.

In order to show that diagram (*xxii*) is a pushout we shall construct diagram (*xxiii*). We start by finding an object U and morphisms $v : H \to U$ and $w : G' \to U$ such that region $(*)$ is a pushout diagram. Now since $\kappa_1 o_H = o_{E'}$ and region (†) of diagram (*xvii*) is a pushout, there exists a unique morphism $u_1 : U \to C$ such that $u_1 w = \pi'_1$ and $u_1 v = \rho'_1 \kappa_1$. Similarly, using the fact that the corresponding region of diagram (*xv*) is a pushout, there exists a unique morphism $u_2 : U \to D$ such that $u_2 w = \pi'_2$ and $u_2 v = \rho'_2 \kappa_2$. Using the standard decomposition property of pushouts, the two newly constructed regions (the upper faces of the rotated cube within diagram (*xxiii*)) are pushouts. The two lower regions of diagram (*xxiii*) are

the pushouts which appeared as the two front faces of diagram (*xix*) and which we had originally obtained in diagrams (*xvii*) and (*xviii*). The left face of the rotated cube is the top face of (*xvi*) which is a pullback. Spinning the cube around into diagram (*xxiv*) we use the fact that the bottom and top faces are pushouts, and the back faces pullbacks to deduce that the front right face is a pullback.

Since also the bottom face of diagram (*xix*) is a pullback, we obtain a unique isomorphism $\zeta : G \to U$ such that $u_1\zeta = \pi_1$ and $u_2\zeta = \pi_2$. As we have assumed that $(*)$ is a pushout, it remains only to show that $\zeta\lambda = w$ and $\zeta\chi = v$ in order to prove that diagram (*xxii*) is a pushout.

We shall show that the required equalities hold using the fact that the front left face of diagram (*xxiv*) is a pullback. Indeed $u_1\zeta\lambda = \pi_1\lambda = \pi'_1 = u_1 w$ and $u_2\zeta\lambda = \pi_2\lambda = \pi'_2 = u_2 w$ which implies that $\zeta\lambda = w$. Also $u_1\zeta\chi = \pi_1\chi = \rho'_1\kappa_1 = u_1 v$ and $u_2\zeta\chi = \pi_2\chi = \rho'_2\kappa_2 = u_2 v$ which implies that $\zeta\chi = v$. This completes the task of verifying that $(*)$ is a pushout diagram.



(*xxv*) (*xxvi*) (*xxvii*)

Summarising our efforts so far, we illustrate the top faces of diagrams (*xx*) and (*xxi*) again in diagrams (*xxv*) and (*xxvi*). Also, the pushout $(*)$ of diagram (*xxiii*) is illustrated again in diagram (*xxvii*). We claim that these diagrams commute. Recall that $\kappa_1 o_H = o_{E'}$ by the definition of o_H. Thus in diagram (*xxv*) it suffices to check that $\nu_1\iota_E = \iota_{E'}$. Indeed, using the fact that γ' is a 2-cell we have

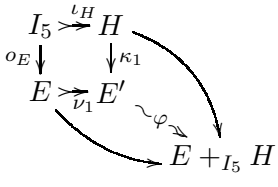
$$\rho'_1\iota_{E'} = \gamma'^{-1}(i_{E'\to E'+G'})\iota_{E'} = \iota_C,$$

But $\iota_C = \rho_1\iota_E$ as illustrated in diagram (*vii*). Using the fact that the front left face of diagram (*xx*) is a pullback, we obtain the equation $\nu_1\iota_E = \iota_{E'}$.

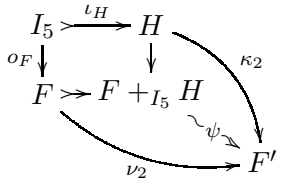
Proceeding in a similar way for diagram (*xxvi*), we use the definition of o_H to conclude that $\kappa_2 o_H = o_{F'}$. Similarly, we use the fact that the front right face of diagram (*xxi*) is a pullback, we derive $\nu_2\iota_F = \iota_{F'}$.

To show that diagram (*xxvii*) commutes, first notice that equation (6.11) gives $\lambda o_{G'} = o_G$. The fact that $\chi\iota_H = \iota_G$ follows from the fact that the back right face of diagram (*xvii*) commutes.

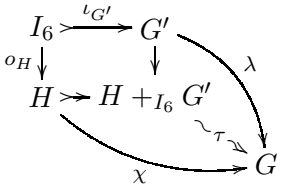
Let $\varphi : E' \to E +_{I_5} H$, $\psi : F +_{I_5} H \to F'$ and $\tau : H +_{I_6} G' \to G$ be the unique induced isomorphisms, as illustrated in diagrams (*xxviii*), (*xxivx*) and (*xxx*) below.



(*xxviii*)



(*xxix*)



(*xxx*)

We need to verify that φ, ψ and τ are 2-cells in ILC($\mathbf{C}$), that is, we need to check that they preserve input and output interfaces. Indeed, $\varphi o_{E'} = \varphi\kappa_1 o_H = i_{H\to E+H}$ and $\varphi\iota_{E'} = \varphi\nu_1\iota_E = i_{E\to E+H}$ using diagrams (*xxv*) and (*xxviii*). Similarly, $\psi(i_{H\to F+H})o_H = \kappa_2 o_H = o_{F'}$ and $\psi(i_{F\to F+H})\iota_F = \nu_2\iota_F = \iota_{F'}$, using diagrams (*xxvi*) and (*xxix*). Finally, $\tau(i_{G'\to H+G'})o_{G'} = \lambda o_{G'} = o_G$ and $\tau(i_{H\to H+G'})\iota_H = \chi\iota_H = \iota_G$, using diagrams (*xxvii*) and (*xxx*).

We shall now verify that φ, ψ and τ satisfy the three required equations:

$$(E +_{I_5} \tau)a(\varphi +_{I_6} G')\gamma' = \gamma \tag{6.12}$$

$$\delta'(\psi +_{I_6} G')a(F +_{I_5} \tau^{-1}) = \delta \tag{6.13}$$

$$(B +_{I_3} \psi)a(\beta +_{I_5} H)a(A +_{I_2} \varphi) = \beta'. \tag{6.14}$$

First we shall show that $\gamma'^{-1}(\varphi^{-1} +_{I_6} G') = \gamma^{-1}(E +_{I_5} \tau)a$, which in turn rearranges into equation (6.12). It suffices to check that the two sides are equal when precomposed with $i_{G'\to(E+H)+G'}$ and $i_{E+H\to(E+H)+G'}$. Checking the first of these, we use the definitions of λ, γ, τ and the associativity isomorphisms in order to derive

$$\begin{aligned}
\gamma'^{-1}(\varphi^{-1} +_{I_6} G')(i_{G'\to(E+H)+G'}) &= \gamma'^{-1}(i_{G'\to E'+G'}) \\
&= \pi'_1 \\
&= \pi_1\lambda \\
&= \gamma^{-1}(i_{G\to E+G})\lambda \\
&= \gamma^{-1}(i_{G\to E+G})\tau(i_{G'\to H'+G'}) \\
&= \gamma^{-1}(E +_{I_5} \tau)(i_{H+G'\to E+(H+G')})(i_{G'\to H+G'}) \\
&= \gamma^{-1}(E +_{I_5} \tau)a(i_{G'\to(E+H)+G'}).
\end{aligned}$$

To check that $\gamma'^{-1}(\varphi^{-1})(i_{E+H\to(E+H)+G'}) = \gamma^{-1}(E+_{I_5}\tau)a(i_{E+H\to(E+H)+G'})$ holds we shall precompose in turn with $i_{E\to E+H}$ and $i_{H\to E+H}$. Indeed, using the definitions of φ, ρ'_1 and the commutativity of the front left face of diagram (xx) we derive

$$\begin{aligned}\gamma'^{-1}(\varphi^{-1} +_{I_6} G')(i_{E+H\to(E+H)+G'})(i_{E\to E+H}) &= \gamma'^{-1}(i_{E'\to E'+G'})\varphi^{-1}(i_{E\to E+H}) \\ &= \gamma'^{-1}(i_{E'\to E'+G'})\nu_1 \\ &= \rho'_1\nu_1 \\ &= \rho_1 \\ &= \gamma^{-1}(i_{E\to E+G}) \\ &= \gamma^{-1}(E +_{I_5} \tau)a(i_{E+H\to(E+H)+G'})(i_{E\to E+H}).\end{aligned}$$

Also, using the definitions of γ', φ, the fact that the front right face of diagram (xx) is commutative and the definition of τ we obtain

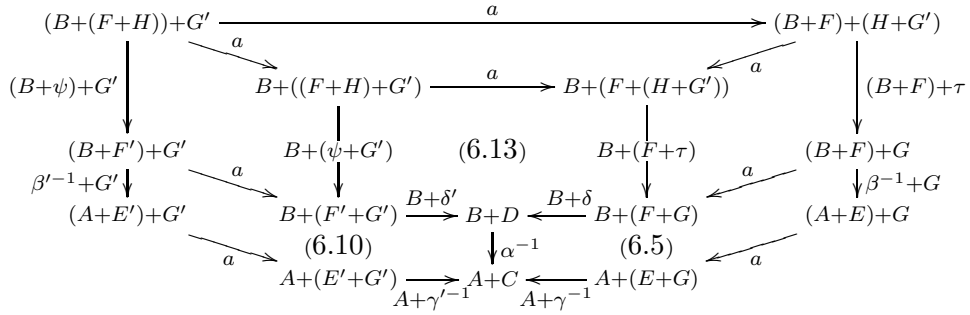
$$\begin{aligned}\gamma'^{-1}(\varphi^{-1} +_{I_6} G')(i_{E+H\to(E+H)+G'})(i_{H\to E+H}) &= \gamma'^{-1}(i_{E'\to E'+G'})\varphi^{-1}(i_{H\to E+H}) \\ &= \rho'_1\kappa_1 \\ &= \pi_1\chi \\ &= \gamma^{-1}(i_{G\to E+G})\chi \\ &= \gamma^{-1}(E +_{I_5} \tau)a(i_{E+H\to(E+H)+G'})(i_{H\to E+H}).\end{aligned}$$

This completes the verification of equation (6.12). By virtue of symmetry, equation (6.13) holds also.

In order to show that equation (6.14) holds, we shall show, equivalently,

$$(A +_{I_2} \varphi^{-1})a = \beta'^{-1}(B +_{I_3} \psi)a(\beta +_{I_5} H). \tag{6.15}$$

First notice that the using equations (6.10), (6.13) and (6.5) allows us to conclude that the diagram below



is commutative.

Composing the right hand side of equation (6.15) with the composite arrow

$$(A + \gamma'^{-1})a(i_{A+E' \to (A+E')+G'}) : A + E' \to A + C$$

allows us to derive

$$\begin{aligned}
&(A + \gamma'^{-1})a(i_{A+E' \to (A+E')+G'})\beta'^{-1}(B + \psi)a(\beta + H) \\
=&(A + \gamma'^{-1})a(\beta'^{-1} + G')((B + \psi) + G')(i_{B+(F+H) \to (B+(F+H))+G'})a(\beta + H) \\
=&(A + \gamma^{-1})a(\beta^{-1} + G)((B + F) + \tau)a(i_{B+(F+H) \to (B+(F+H))+G'})a(\beta + H) \\
=&(A + \gamma^{-1})(A + (E + \tau))a(i_{(A+E)+H \to ((A+E)+H)+G'}) \\
=&(A + \gamma'^{-1})(A + (\varphi^{-1} + G))a(i_{(A+E)+H \to ((A+E)+H)+G'}) \\
=&(A + \gamma'^{-1})(i_{A+E' \to (A+E')+G'})(A + \varphi^{-1})a,
\end{aligned}$$

where we use, respectively, the commutativity of the diagram and the fact that equation (6.12) holds. But $i_{A+E' \to (A+E')+G'}$ is mono, since it is the pushout of mono $\iota_{G'}$ in an adhesive category (Lemma 5.1.15). Canceling on the left results in equation (6.15), and therefore, equation (6.14).

Essential uniqueness of $H : I_5 \to I_6$ can be shown by using the third part of Lemma 5.1.18, applied to diagram (*xix*). In other words, we shall use the fact that we are working in an adhesive category and, therefore, the pair $\langle \iota_{G'}, \lambda \rangle$ has a unique pushout complement. Indeed, suppose that there exists $H' : I_5 \to I_6$ and $\varphi' : E' \to E +_{I_5} H'$, $\psi' : F +_{I_5} H' \to F'$ and $\tau' : H' +_{I_6} G' \to G$, so that equations (6.16), (6.17) and (6.18) hold.

$$(E +_{I_5} \tau')a(\varphi' +_{I_6} G')\gamma' = \gamma \tag{6.16}$$

$$\delta'(\psi' +_{I_6} G')a(F +_{I_5} \tau'^{-1}) = \delta \tag{6.17}$$

$$(B +_{I_3} \psi')a(\beta +_{I_5} H)a(A +_{I_2} \varphi') = \beta' \tag{6.18}$$

Consider the pushout shown in diagram (*xxxi*), where we let $\chi' = \tau'(i_{H' \to H'+G'})$.

$$\begin{array}{ccc}
I_6 & \overset{\iota_{G'}}{\rightarrowtail} & G' \\
{\scriptstyle o_{H'}}\downarrow & & \downarrow{\scriptstyle \tau' i} \\
H' & \underset{\chi'}{\rightarrowtail} & G
\end{array}$$

(*xxxi*)

We shall use the fact that diagram (*xxii*) is also a pushout in order to derive the existence of an isomorphism $\xi : H \to H'$. In order to apply this argument

we shall first need to verify that $\lambda = \tau' i_{G' \to H' + G'}$. Using a straightforward diagram chase on the diagram represented by equation (6.16) we obtain

$$\gamma'^{-1}(i_{G' \to E' + G'}) = \gamma^{-1}(i_{G \to E + G})\tau'(i_{G' \to H' + G'}),$$

which, using the definitions of γ and π_1', can be rewritten more concisely as

$$\pi_1' = \pi_1 \tau'(i_{G' \to H' + G'})$$

and since from the definition of λ we have that $\pi_1' = \pi_1 \lambda$, we may conclude that

$$\pi_1 \lambda = \pi_1 \tau'(i_{G' \to H + G'}). \tag{6.19}$$

A similar diagram chase on equation (6.17) results in

$$\delta'(i_{G' \to F' + G}) = \delta(i_{G \to F + G})\tau(i_{G' \to H' + G'}),$$

which, using the definitions of δ and π_2' amounts to saying that

$$\pi_2' = \pi_2 \tau(i_{G' \to H' + G'})$$

and again, since by the definition of λ we have that $\pi_2' = \pi_2 \lambda$, we have that

$$\pi_2 \lambda = \pi_2 \tau(i_{G' \to H' + G'}). \tag{6.20}$$

Using the universal property of the pullback which forms the bottom face of diagram (*xix*), equations (6.19) and (6.20) imply that we indeed have $\lambda = \tau'(i_{G' \to H' + G'})$, as required.

Using the conclusion of Lemma 5.1.18, we obtain an isomorphism $\xi : H \to H'$ which is the unique morphism which satisfies $\xi o_H = o_{H'}$ and $\chi' \xi = \chi$. To check that ξ defines a 2-cell, it suffices to show that $\xi \iota_H = \iota_{H'}$. Indeed, using the definition of ξ and the commutativity of the rear left face of diagram (*xxi*) we have that $\chi' \xi \iota_H = \chi \iota_H = \iota_G$. But since τ' is a 2-cell, it preserves its input interface. Thus we have equation $\iota_G = \tau'(i_{H' \to H' + G'})\iota_{H'}$, the right hand side of which is just $\chi' \iota_{H'}$. Since χ' is mono, being a composition of a mono $i_{H' \to H' + G'}$ (which is mono because we are working in an adhesive category and it is the pushout of a mono ι_H along $o_{H'}$) with an isomorphism τ', we have $\xi \iota_H = \iota_{H'}$ as required.

We shall now verify that ξ satisfies the required equations (6.21), (6.22) and (6.23). These equations ensure that ξ makes the two mediating morphisms $\langle H, \varphi, \psi, \tau \rangle$ and $\langle H', \varphi', \psi', \tau' \rangle$ compatible.

$$(E +_{I_5} \xi)\varphi = \varphi' \tag{6.21}$$

$$\psi(F +_{I_5} \xi^{-1}) = \psi' \tag{6.22}$$

$$\tau'(\xi +_{I_6} G') = \tau \tag{6.23}$$

We shall start by showing that equation (6.23) holds. First notice that

$$\begin{aligned}\tau'(\xi +_{I_6} G')(i_{H\to H+G'}) &= \tau'(i_{H'\to H'+G'})\xi \\ &= \chi'\xi = \chi = \tau(i_{H\to H+G'}),\end{aligned}$$

where the last equality follows from the definition of τ (diagram (xxx)). Similarly

$$\begin{aligned}\tau'(\xi +_{I_6} G')(i_{G'\to H+G'}) &= \tau'(i_{G'\to H'+G'}) \\ &= \lambda = \tau(i_{G\to H+G'})\end{aligned}$$

where the last equality again follows from the definition of τ. Equation (6.23) follows using the universal property of pushouts.

We shall now proceed with showing that equation (6.22) holds. We start by rearranging equation (6.17) in order to obtain

$$\delta'(\psi' +_{I_6} G') = \delta(F +_{I_5} \tau')a : (F +_{I_5} H') +_{I_6} G' \to F' +_{I_6} G'.$$

Precomposing the two sides of the above equation with $i_{F+H'\to(F+H')+G'}$ yields

$$\delta'(i_{F'\to F'+G'})\psi = \delta(F +_{I_5} \tau')a(i_{F+H'\to(F+H')+G'}).$$

Starting with the right hand side of the equation above, we use equation (6.23) in order to infer the following:

$$\begin{aligned}&\delta(F +_{I_5} \tau')a(i_{F+H'\to(F+H')+G'}) \\ &\quad=\delta(F +_{I_5} \tau)(F +_{I_5} (\xi^{-1} +_{I_6} G'))a(i_{F+H'\to(F+H')+G'}) \\ &\quad=\delta(F +_{I_5} \tau)a(i_{F+H\to(F+H)+G'})(F +_{I_5} \xi^{-1}).\end{aligned}$$

Continuing, we use equation (6.13) in order to obtain

$$\begin{aligned}&\delta(F +_{I_5} \tau)a(i_{F+H\to(F+H)+G'})(F +_{I_5} \xi^{-1}) \\ &\quad=\delta'(\psi +_{I_6} G')(i_{F+H\to(F+H)+G'})(F +_{I_5} \xi^{-1}) \\ &\quad=\delta'(i_{F'\to F'+G'})\psi(F +_{I_5} \xi^{-1}).\end{aligned}$$

Summarising, we have shown that

$$\delta'(i_{F'\to F'+G'})\psi' = \delta'(i_{F'\to F'+G'})\psi(F +_{I_5} \xi^{-1}).$$

Notice that, using the fact that we are in an adhesive category, $i : F' \to F' +_{I_6} G'$ is mono (see diagram (ix)). Equation (6.22) follows.

Similarly, using the fact that $i : E' \to E' +_{I_6} G'$ is mono, equations (6.16), (6.23) and (6.12) allows us to derive equation (6.21).

Finally, it is easy to check that ξ is the unique 2-cell which serves as a mediator between $\langle H, \varphi, \psi, \tau \rangle$ and $\langle H', \varphi', \psi', \tau' \rangle$. Indeed, suppose that there is another 2-cell $\xi' : H \to H'$ which satisfies equations analogous to (6.21), (6.22) and (6.23). Then in particular we have that

$$\tau'(\xi' +_{I_6} G') = \tau. \tag{6.24}$$

We shall show that ξ' fits in as a mediating morphism between the pushout complements illustrated in diagrams (*xxii*) and (*xxxi*) - these are guaranteed to be unique by the conclusion of Lemma 5.1.18. Because ξ' is a 2-cell, we get for free that $\xi' o_H = o_{H'}$. It remains to check that $\chi' \xi' = \chi$. Indeed,

$$\begin{aligned} \chi' \xi' = \tau'(i_{H' \to H' + G'}) \xi' &= \tau'(\xi' +_{I_6} G')(i_{H \to H + G'}) \\ &= \tau(i_{H \to H + G'}) = \chi \end{aligned}$$

where we have used, respectively, the definition of χ', equation (6.24) and the definition of τ. □

Summarising the technical content of the last two subsections, we are ready to state the main technical contribution of this chapter.

Theorem 6.2.5. ILC(**C**) *has GRPOs.*

Proof. We have showed how one constructs a candidate in Lemma 6.2.3. Such a candidate, as we have demonstrated in the proof of Lemma 6.2.4, satisfies the required universal property. □

Notice that we can give a simplified presentation of the construction of the minimal candidate if we assume that all the cospans are linear.

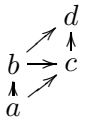
Algorithm 6.2.6 (GRPO Construction in LC(**C**)**).** When all the morphisms in diagram (*ii*) are mono the construction of Algorithm 6.2.2 is considerably simpler. This is because constructing a pullback of monos into X amounts to computing the intersection of subobjects of X. Indeed, we let:

1. $G = C \cap D$;
2. $E = Y \cap C$;

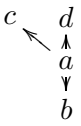
3. $F = Y \cap D$;

4. $I_5 = F \cap G = F \cap E = E \cap G$.

We shall finish this subsection with several simple examples. The examples have no computational meaning and are merely an exercise in applying Algorithm 6.2.2. We shall consider ILC(**Graph**) as our category of contexts. In the accompanying Figure 6.1, we label the nodes of the graphs in order to clarify the action of various graph morphisms, which we leave unlabelled. We also do not draw the 2-cells as the labelling on the nodes makes these clear. We shall make reference to the directed graphs below.

$X_1 =$ 

$X_2 =$ 

$X_3 =$ 

Example 6.2.7. Graph X_1 can be decomposed as illustrated by the exterior of of the upper left diagram of Figure 6.1. Here, all the graph morphisms are injective. The reader may wish to go through the steps of Algorithm 6.2.2 to construct the GRPO is this particular case, it is illustrated in the interior of the diagram.

Example 6.2.8. Graph X_2 can be broken up as illustrated by the exterior of the upper right diagram of Figure 6.1. Notice that o_A is not injective. The constructed GRPO is illustrated.

Example 6.2.9. A GRPO for a partition of X_3 is illustrated in the lower diagram of Figure 6.1. Notice that here both o_A and o_B are not injective. One can compare this "blowing up" of the output interface with the phenomenon of "forking" as described by Leifer [64, Figure 1.4].

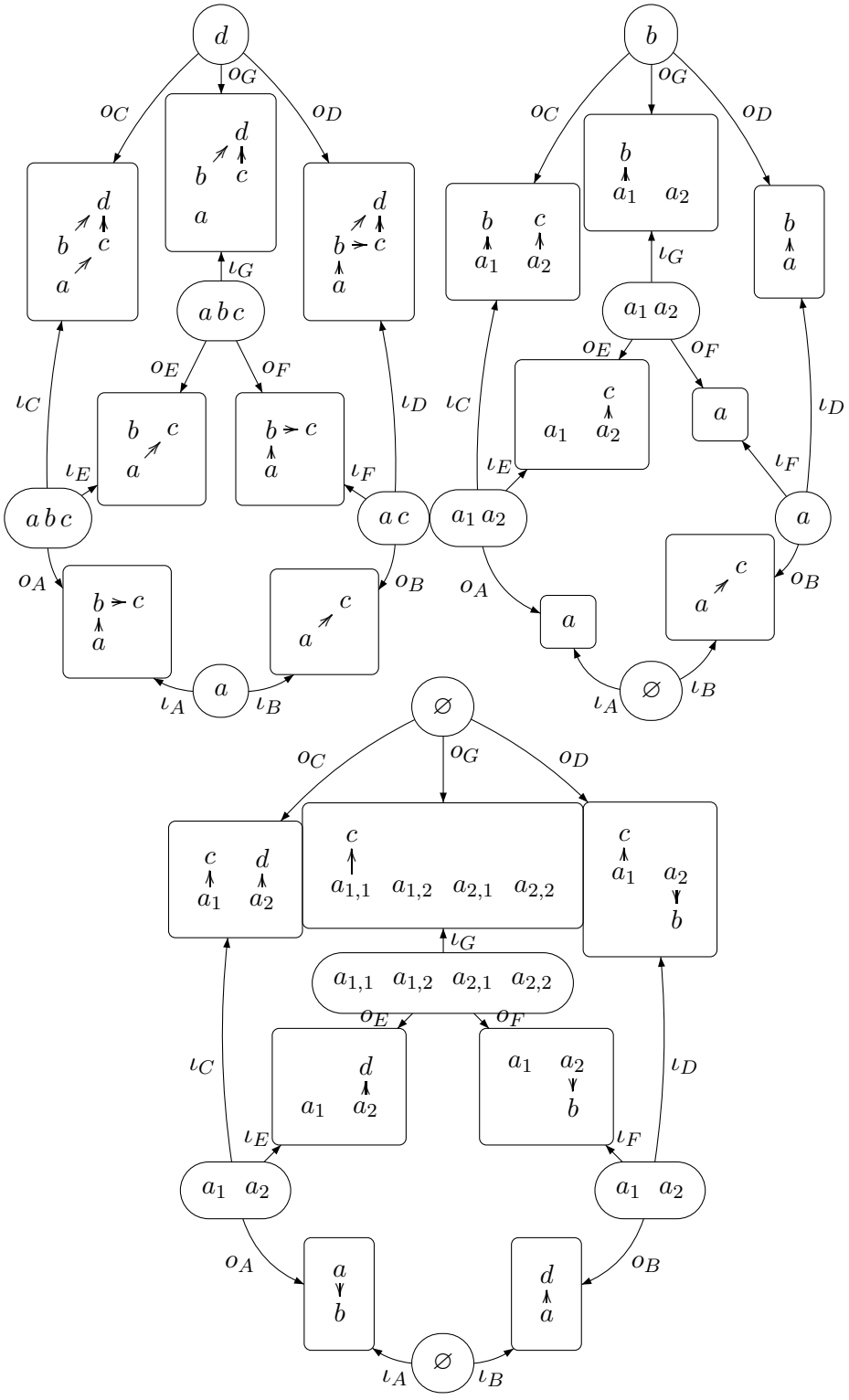
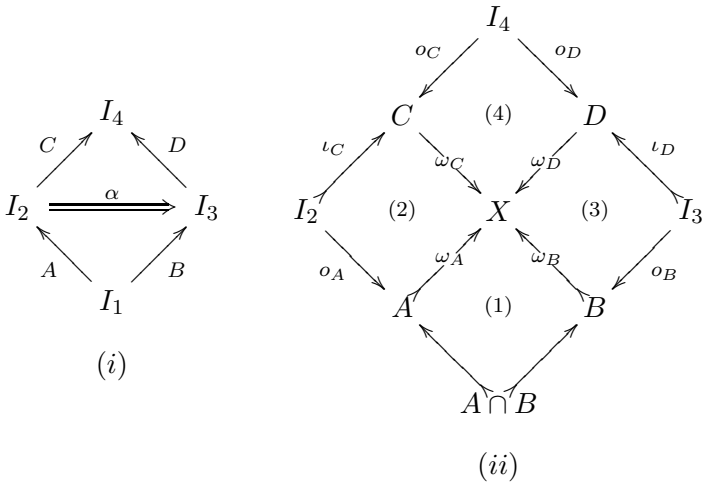


Figure 6.1: GRPOs in ILC(**Graph**).

6.2.3 Characterisation

As is the case in the G-categories of bunch contexts, there is a relatively simple characterisation of GIPOs. In the context of reactive systems, the characterisation of GIPOs provides a simple way to test whether a given redex square is minimal.

Lemma 6.2.10. A diagram (i) is a GIPO if and only if there exists an X and isomorphisms $\omega_l : A +_{I_2} C \to X$ and $\omega_r : B +_{I_3} D \to X$ such that $\omega_r^{-1}\omega_l = \alpha$ and so that in the resulting diagram (ii) we have:



- region (1) is both a pullback and a pushout;
- regions (2) and (3) are pushouts;
- region (4) is a pullback.

The proof that the characterisation is correct is quite simple, in particular it follows easily from the proof of Theorem 6.3.10.

6.3 Applications

In this section we shall introduce two immediate applications of our construction of GRPOs in input linear cospan bicategories.

First, after a brief review of the theory of double-pushout graph rewriting, we shall show that we may use the construction to derive congruences for graphs enriched with an output interface. Graph contexts here are input-linear cospans of graphs. The labelled transition system derived using our

technology admits labels which are the smallest contexts which allow a double-pushout rewrite. The results in this section are closely related to rewriting via borrowed contexts due to Ehrig and König [25]. In particular, in Theorem 6.3.10 we show that the labelled transition systems are essentially the same, the difference being that the nodes and labels of our transition system are quotiented by isomorphism.

Our results both shed light on and extend rewriting via borrowed contexts. Firstly, because borrowed contexts correspond to GIPOs, they satisfy a universal property. Secondly, we show that borrowed contexts fall within the framework of reactive systems [66, 93, 92] and therefore the various congruence properties and constructions carry over. In particular, Ehrig and König's congruence theorem can be seen as an application of the congruence theorem for reactive systems (Theorem 2.3.6). Finally, due to the generality of Theorem 6.2.5, we relax some of the technical conditions imposed by Ehrig and König and introduce the notion of extended borrowed contexts (Definition 6.3.12).

Our second application concerns Milner's theory of bigraphs [73]. We show how bigraphs can be represented by a cospan bicategory over an adhesive category. As a consequence, we derive labelled transition systems for reactive systems over an input-linear variant of bigraphs. Our input-linear bigraphs are different to Milner's variant in the sense that our formalism allows bigraphs which are not allowed in Milner's, and vice-versa. It appears that a closer correspondence to Milner's theory would be achieved by considering an output-linear variant. It is, therefore, an interesting line of future work whether one could derive a general construction of GRPOs for an interesting and general class of output-linear cospan bicategories.

Another consequence of representing bigraphs by cospans is that because of the close correspondence between double-pushout rewriting systems and reactive systems over cospan bicategories demonstrated by Lemma 6.3.5, one can view bigraphical reactive systems as certain double-pushout rewriting systems. In particular, this could mean that some of the theory and technology developed for the latter can perhaps be applied successfully to the former.

6.3.1 Double-pushout rewriting and borrowed contexts

Double-pushout (dpo) graph rewriting is a well known field with an impressive body of literature [19]. Introduced in [27], it has recently been generalised in [23, 59, 24]. We shall describe a variant of dpo-graph rewriting, working at the level of an arbitrary adhesive category $\mathbf{C}$. The reader

may of course, safely substitute the adhesive category **Graph** of graphs and graph homomorphisms for **C**. We start by relating dpo rewriting and reactive systems.

Definition 6.3.1 (Rewrite Rule). A rewrite rule p is a span

$$L \xleftarrow{l} K \xrightarrow{r} R \tag{6.25}$$

in **C**. We *do not* assume that either l or r are mono.

Here L and R represent respectively the left and right-hand side of the rule, while K is information which remains unaffected by the rewrite. A redex in an object C is identified by matching a rule's left-hand side, which is done via a morphism $f : L \to C$.

Definition 6.3.2 (Adhesive Grammar). An adhesive grammar G is a pair $\langle \mathbf{C}, \mathbf{P} \rangle$ where **C** is an adhesive category and **P** is a set of arbitrary rewrite rules.

Definition 6.3.3 (Rewrite Rule Application). Object C rewrites to D with rule p, in symbols $C \longrightarrow\!\!\!\!\blacktriangleright_{p,f} D$, if there exist an object E and morphisms so that the two squares in the following diagram are pushouts.

$$\begin{array}{ccccc} L & \xleftarrow{l} & K & \xrightarrow{r} & R \\ {\scriptstyle f}\downarrow & & \downarrow{\scriptstyle g} & & \downarrow{\scriptstyle h} \\ C & \xleftarrow[v]{} & E & \xrightarrow[w]{} & D \end{array} \tag{6.26}$$

We shall write $C \longrightarrow\!\!\!\!\blacktriangleright D$ if there exist $p \in P$ and $f : L \to C$ such that $C \longrightarrow\!\!\!\!\blacktriangleright_{p,f} D$.

Proposition 6.3.4. An adhesive grammar can be seen as a reactive system on $\mathrm{Cospan}^{\cong}(\mathbf{C})$. Let 0 denote the empty graph, and the set $\mathcal{R}$ contain for each rewrite rule p (6.25) a pair

$$\left\langle 0 \to L \xleftarrow{l} K,\ 0 \to R \xleftarrow{r} K \right\rangle.$$

We choose all arrows of $\mathrm{Cospan}^{\cong}(\mathbf{C})$ be reactive. Let $\longrightarrow\!\!\triangleright$ denote the resulting reaction relation.

It is useful to point out that the process described in Proposition 6.3.4 can be reversed. Starting with a reactive system over a cospan category

over $\mathbf{C}$ with chosen object the initial object $0 \in \mathbf{C}$, one can obtain a double-pushout rewriting system by adding a rewrite rule

$$L \xleftarrow{l} K \xrightarrow{r} R \quad \text{for every} \quad \left\langle\, 0 \to L \xleftarrow{l} K \;,\; 0 \to R \xleftarrow{r} K \,\right\rangle \in \mathcal{R}.$$

It turns out that these "encodings" are actually very well behaved. In this section we shall present Lemma 6.3.5 which shows that the dpo rewrite relation is exactly the reactive system reaction relation when it makes sense to compare them. The main result of this section, Theorem 6.3.10 makes the correspondence even stronger by relating the operational theories developed for the two approaches. Indeed, we shall show that it is equivalent to consider GIPOs in reactive systems over cospans on the one hand, and borrowed contexts, due to Ehrig and König [25], on the other.

The following lemma is similar to a previously published result [33] and can be considered folklore. As well as being simple to prove, it is a relatively little-known result; it is therefore worthwhile to state and prove it here. It is crucial for us because it serves as a foundation for relating the theory of dpo rewriting and the theory of reactive systems.

We use the shorthand $C \longrightarrow\!\!\!\triangleright\, D$ to mean that C and D are cospans with empty input and output contexts.

Lemma 6.3.5. $C \longrightarrow\!\!\!\blacktriangleright\, D$ iff $C \longrightarrow\!\!\!\triangleright\, D$.

Proof. If $C \longrightarrow\!\!\!\triangleright\, D$ then $C \cong E_g \circ L^l$ and $D \cong E_g \circ R^r$, which is equivalent to requiring that the two squares below are pushouts.

$$\begin{array}{ccccccc} & & & & 0 & & \\ & & & & \downarrow{\scriptstyle !} & & \\ & & C & \xleftarrow{v} & E & \xrightarrow{w} & D \\ & & {\scriptstyle f}\uparrow & & \downarrow{\scriptstyle g} & & \uparrow{\scriptstyle h} \\ 0 & \xrightarrow{!} & L & \xleftarrow[l]{} & K & \xrightarrow[r]{} & R & \xleftarrow{!} & 0 \end{array}$$

This means $C \longrightarrow\!\!\!\blacktriangleright\, D$, since the middle part of the diagram is a dpo rewrite as in (6.26). Note that the output interface of C and D is actually arbitrary, as long as it factors through E. □

The equivalence exhibited by Lemma 6.3.5 between the rewrite relation in a dpo rewriting system and the reaction relation of a reactive system over a cospan bicategory is a bridge which relates the two theories.

This may be compared with an easy lemma about term rewriting. In a ground term rewriting system over a signature Σ, one usually defines the

rewrite relation as follows: a term c rewrites to d, written $c \longrightarrow\!\!\!\!\blacktriangleright d$, if one can find l as a subterm of c, and replace it with r. On the other hand, one may recast such a term rewriting system as a reactive system over the free linear Lawvere theory $\mathbf{C}_\Sigma$ [93]. In $\mathbf{C}_\Sigma$, the objects are natural numbers, while arrows $m \to n$ are n-tuples of terms built up from Σ which contain exactly one occurrence each of m ordered variables. Composition in this category is substitution of terms, done in the obvious way. The reaction rules $\mathcal{R}$ consist of pairs $\langle l, r\rangle$, where $l, r : 0 \to 1$ are respectively the left and the right hand sides of a rewrite rule. One then defines the rewrite relation as follows: $c \longrightarrow\!\!\!\!\triangleright d$ if $c = c' \circ l$, $d = c' \circ r$ and $\langle l, r\rangle \in \mathcal{R}$. In other words, the reaction rules are closed under all linear contexts. It is easy to show that $c \longrightarrow\!\!\!\!\blacktriangleright d$ if and only if $c \longrightarrow\!\!\!\!\triangleright d$.

Indeed, Lemma 6.3.5 implies that reactive systems on $\mathrm{Cospan}^{\cong}(\mathbf{C})$ with all cospans reactive include all dpo rewriting systems over $\mathbf{C}$.

Lts semantics for dpo rewriting systems. In order to apply Theorem 6.2.5, we restrict to graph rewriting systems corresponding to bicategories of input-linear cospans.

The following notion, which we shall refer to as input-linear rewrite rule application is sometimes referred to in graph rewriting literature as rewriting with *injective matching*.

Definition 6.3.6 (Input-Linear Rewrite Application). Object C rewrites to D input-linearly with rule p, in symbols $C \longrightarrow\!\!\!\!\blacktriangleright^{il}_{p,f} D$, if $C \longrightarrow\!\!\!\!\blacktriangleright_{p,f} D$ and in addition f, g and h of (6.26) are mono.

Definition 6.3.7. For $G = \langle \mathbf{C}, \mathbf{P}\rangle$ an adhesive grammar with an input-linear rewrite relation $\longrightarrow\!\!\!\!\blacktriangleright^{il}$, we construct a reactive system $\mathbb{C}$ over the bicategory of input-linear cospans $\mathrm{ILC}(\mathbf{C})$ using the translation of Proposition 6.3.4.

Lemma 6.3.5 clearly specialises to double-pushout rewriting systems and reactive systems over input-linear cospan bicategories.

Proposition 6.3.8. Suppose that $\mathbf{C}$ is adhesive and consider an arbitrary adhesive grammar G, and let $\mathbb{C}$ be the corresponding reactive system over an *input-linear* cospan bicategory. Let $\longrightarrow\!\!\!\!\triangleright$ denote the reaction relation in $\mathbb{C}$. Then

$$C \longrightarrow\!\!\!\!\blacktriangleright^{il}_{p,f} D \text{ iff } C \longrightarrow\!\!\!\!\triangleright D.$$

We are now ready to examine the operational theory derived using GRPOs. Let LTS(G) be CTS($\mathbb{C}$) of Definition 2.2.14.[1] Using the congruence theorem (Theorem 2.3.6) for bisimulation on labelled transition systems generated by GIPOs [93], we obtain the following.

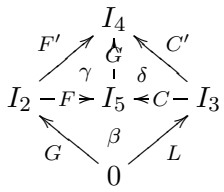
Corollary 6.3.9. *Bisimilarity on* LTS(G) *is a congruence.*

A rewrite rule (6.25) is called *linear* when both l and r are mono. If we restrict our view to dpo-rewriting systems with linear rewrite rules and input-linear rewrites then we are in a position to compare the resulting lts with rewriting via borrowed contexts.

Precisely, given an adhesive grammar $G = \langle \mathbf{C}, \mathbf{P} \rangle$, where $\mathbf{P}$ consists of linear rewrite rules, let RBC(G) be the lts derived via rewriting with borrowed contexts as in [25].

Theorem 6.3.10. LTS(G) = RBC(G).

Proof. (1.) From GIPOs to Borrowed Contexts. Suppose that $G^{o_G} \xrightarrow{F^{o_F}_{\iota_F}} H^{o_H}$ in LTS(G). then $F^{o_F}_{\iota_F}$ must be a part of an GIPO diagram. Since every GIPO can be constructed as a GRPO, we have a redex diagram



as the outside of the diagram, with $\langle L^{o_L}, R^{o_R} \rangle$ being a reaction rule, corresponding via the translation of Lemma 6.3.5 to the rewrite rule

$$L \xleftarrow{o_L} I_3 \xrightarrow{o_R} R.$$

The candidate $\langle I_5, F, C, G, \beta, \gamma, \delta \rangle$ illustrated above is the GRPO obtained via the construction of Algorithm 6.2.2.

[1] We consider the concrete version of the canonical lts in order to derive labels which are easily comparable with rewriting via borrowed contexts. For practical applications, the abstract lts ATS($\mathbb{C}$) would probably be more appropriate. Of course, the congruence theorems apply to both choices of lts.

We also have $H^{o_H} \cong C^{o_C}_{i_C} \circ R^{o_R}$, which in other words means that the diagram (i) below is a pushout

$$\begin{array}{ccc} I_3 & \xrightarrow{o_R} & R \\ {\scriptstyle i_C}\downarrow & & \downarrow{\scriptstyle \theta_1} \\ C & \xrightarrow[\theta_2]{} & H \end{array}$$

(i)

with $o_H : I_5 \to H$ equal to $\theta_2 \circ o_C$.

Recall that from the construction, we have that

$$\begin{array}{ccc} G \cap L & \rightarrowtail & G \\ \downarrow & & \downarrow{\scriptstyle \epsilon_1} \\ L & \xrightarrow[\epsilon_2]{} & Y \end{array} \qquad \begin{array}{ccc} I_5 & \xrightarrow{o_F} & F \\ {\scriptstyle o_C}\downarrow & & \downarrow{\scriptstyle \sigma_1} \\ C & \xrightarrow[\sigma_2]{} & Y \end{array} \qquad \begin{array}{ccc} I_2 & \rightarrowtail & F \\ {\scriptstyle o_G}\downarrow & & \downarrow{\scriptstyle \sigma_1} \\ G & \xrightarrow[\epsilon_1]{} & Y \end{array} \qquad \begin{array}{ccc} I_3 & \rightarrowtail & C \\ {\scriptstyle o_L}\downarrow & & \downarrow{\scriptstyle \sigma_2} \\ L & \xrightarrow[\epsilon_2]{} & Y \end{array}$$

(ii) (iii) (iv) (v)

diagram (ii) is a pushout, diagram (iii) is a pullback, diagrams (iv) and (v) are pushouts.

Notice that in diagrams (i) to (v) we have indicated which morphisms are assumed to be mono in the construction of Algorithm 6.2.2. While Ehrig and König assume that all morphisms are mono, it shall be useful for us here to indicate only the necessary ones as we shall use the extra generality to extend the notion of borrowed context in Definition 6.3.12.

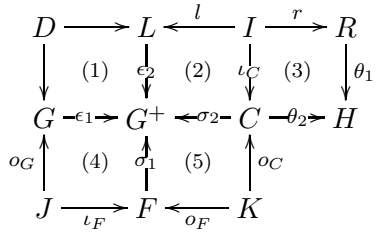
We can construct the following diagram from the five diagrams above:

$$\begin{array}{ccccccc} G \cap L & \xrightarrow{\epsilon'_1} & L & \xleftarrow{o_L} & I_3 & \xrightarrow{o_R} & R \\ {\scriptstyle \epsilon'_2}\downarrow & & \downarrow{\scriptstyle \epsilon_2} & & \downarrow{\scriptstyle \iota_C} & & \downarrow{\scriptstyle \theta_1} \\ G & \xrightarrow{\epsilon_1} & Y & \xleftarrow{\sigma_2} & C & \xrightarrow{\theta_2} & H \\ {\scriptstyle o_G}\uparrow & & \uparrow{\scriptstyle \sigma_1} & & \uparrow{\scriptstyle o_C} & & \\ I_2 & \xrightarrow[\iota_F]{} & F & \xleftarrow[o_F]{} & I_5 & & \end{array}$$

This is exactly the definition of $F^{o_F}_{\iota_F}$ constituting a borrowed context for G^{o_G}, i.e. $G^{o_G} \xrightarrow{F^{o_F}_{\iota_F}} H^{o_H}$ in RBC(G).

(2.) From Borrowed Contexts to GIPOs. In this section of the proof we shall use the notation of Ehrig and König [25]. We shall also follow Ehrig and König in assuming that all morphisms are mono. Given a borrowed

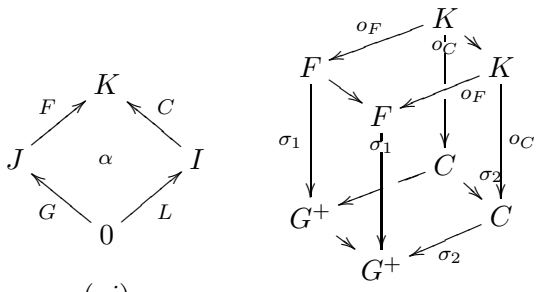
context diagram (*vi*) below,



(*vi*)

we shall show that one may construct a GIPO. First recall that a commutative diagram (*vi*) is a borrowed context diagram when squares (*1*), (*2*), (*3*) and (*4*) are pushouts, while square (*5*) is a pullback.

Indeed, consider the redex diagram (*vii*)



(*vi*) (*vii*)

and assume without loss of generality that $L +_I C = G^+$. Let $\alpha : F +_J G \to G^+$ be the unique isomorphism such that $\alpha i_i = \sigma_1 : F \to G^+$ and $\alpha i_2 = \epsilon_1 : G \to G^+$. Note that the isomorphism exists because square (*4*) in diagram (*vi*) is a pushout.

Since square (*1*) of diagram (*vi*) is a pushout of monos and we are in an adhesive category, it is also a pullback. This implies that $L \cup G = G^+$, because in adhesive categories one forms unions of subobjects by forming the pushout of their intersection. The central cube diagram from the construction of GRPOs is illustrated in (*vii*), in the construction we use only the fact that square (*5*) of diagram (*vi*) is a pullback. Thus, the GRPO of the redex square (*vi*) is $\langle K, F, C, \mathrm{id}, \mathrm{id}, \mathrm{id}, \alpha \rangle$, meaning that it is a GIPO. □

We say that two graphs $A^{o_A:I}$ and $A'^{o_{A'}:I}$ with the same output interface are isomorphic when there exists an isomorphism $\varphi : A \to A'$ such that $\varphi o_A = o_{A'}$. Similarly, we say that two transitions from $A^{o_A:I}$ to $B^{o_B:J}$, with

labels $C^{o_C:J}_{\iota_C:I}$ and $C'^{o_{C'}:J}_{\iota_{C'}:I}$, respectively, are isomorphic when there exists an isomorphism $\psi : C \to C'$ with $\psi\iota_C = \iota_{C'}$ and $\psi o_C = o_{C'}$.

Let $\mathrm{RBC}_{\cong}(G)$ be the lts with the nodes being the isomorphism classes of the nodes of $\mathrm{RBC}(G)$ and whose transitions are the isomorphism classes of transitions of $\mathrm{RBC}(G)$, and let $\mathrm{LTS}_{\cong}(G) = \mathrm{ATS}(\mathbb{C})$ of Definition 2.2.16. We have the following corollary as a straightforward consequence of Theorem 6.3.10.

Corollary 6.3.11. $\mathrm{RBC}_{\cong}(G) = \mathrm{LTS}_{\cong}(G)$.

The results of this section can be seen from two different perspectives. From the point of view of reactive systems, the borrowed context conditions of Ehrig and König [25], once extended by allowing the appropriate morphisms to be non-mono, can be seen as being an elegant and simple characterisation of GIPOs in a input-linear cospan bicategory. On the other hand, the results of this section show that borrowed contexts satisfy a universal property, this means that the congruence theorem for bisimilarity of Ehrig and König can actually be seen as a special case of Theorem 2.3.6. Similarly, other results and technology developed for reactive systems transfers to the setting of borrowed contexts, this includes the congruence theorems for equivalences other than bisimilarity as well as the elegant technique for deriving weak transition systems (in the sense of weak bisimilarity) developed by Jensen in his upcoming PhD thesis [45].

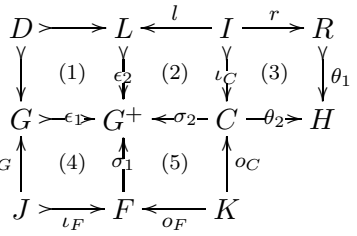
We end this section with an extension of borrowed contexts suggested by the linearity conditions imposed in the construction of Algorithm 6.2.2.

Definition 6.3.12 (Extended Borrowed Contexts). Given an adhesive grammar $G = \langle \mathbf{C}, \mathbf{P} \rangle$ with an *arbitrary* set of rules $\mathbf{P}$, we shall construct a labelled transition system with:

- Nodes: Graphs with output interfaces, $J \xrightarrow{o_G} G$ where o_G is arbitrary;
- Transitions: Cospans of graphs $J \xrightarrow{\iota_F} F \xleftarrow{o_F} K$ where ι_F is mono and o_F is arbitrary.

We derive a transition $[G^{o_G}] \xrightarrow{[F^{o_F}_{\iota_F}]} [H^{o_H}]$ if there exists a commutative

diagram as illustrated below, where



(*1*), (*2*), (*3*) and (*4*) are pushouts, while square (*5*) is a pullback. The indicated morphisms are assumed to be mono, the others are arbitrary.

As a corollary of the translation between borrowed contexts and GIPOs of Theorem 6.3.10 and the congruence Theorem 2.3.6, we have the following.

Corollary 6.3.13. Bisimulation on the labelled transition system resulting from Definition 6.3.12 is a congruence with respect to arbitrary input-linear graph contexts, that is cospans

$$J \xrightarrow{\iota_F} F \xleftarrow{o_F} K$$

where ι_F is mono and o_F is arbitrary.

6.3.2 Bigraphs as cospans

Bigraphs were introduced by Milner to model dynamic systems with independent locality and connectivity structures (cf. [73] for a comprehensive exposition), and have been used by Jensen and Milner [47] to model and derive an lts for the asynchronous π-calculus.

In this section, we recast the notion of bigraph in our approach, and obtain structures very similar to Milner's. Although we briefly discuss the differences between our product and Milner's bigraphs, we remark that a perfect match is not our objective. Rather, we aim at demonstrating that algebraic constructs relevant to the semantics of mobility and communication fall naturally within the realm of cospan bicategories over adhesive categories. To that end, we introduce the adhesive category of place-link graphs, which can be considered "bigraphs without interfaces." Interfaces will then be added when we consider the bicategory of cospans over the category of place-link graphs. We do not develop the notions of width, inactive sites, nor parametric reaction rules, which do not seem to have an effect on the actual construction of GRPOs for bigraphs. The construction shall be given by Algorithm 6.2.2.

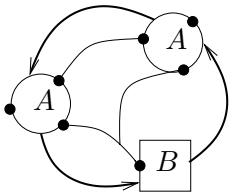


Figure 6.2: Typical place-link graph.

We define a *place-graph* to be a directed graph with nodes labelled over an alphabet Σ. Additionally, there is an arity function $ar : \Sigma \to \mathbb{N}$ (the natural numbers), and place-graph morphisms are directed graph morphisms which preserve node labelling. The intuition is that the elements of Σ are (names of) controls, each equipped with an ordered set of ports. The connectivity of each control is determined by the number of its ports. For a place-graph G with node-set V and labelling function $l : V \to \Sigma$, we denote the ith port of v by v_i, where $0 \leq i < ar(l(v))$. One can construct the total *set of ports of* G by taking the disjoint union:

$$P = \sum_{v \in V} \{\, v_i \mid 0 \leq i < ar(l(v)) \,\}.$$

Definition 6.3.14 (Place-Link Graphs). A place-link (pl) graph is a place-graph G over a set of controls Σ together with a set S and a link map $l : P \to S$, where P is the set of ports of G. A place-link morphism $\langle f_0, f_1, f_2 \rangle : G \to G'$ is a place-graph morphism $\langle f_0, f_1 \rangle$ together with a function $f_2 : S \to S'$ such that $l' f_p = f_2 l$, where $f_p : P \to P'$ is the morphism sending v_i to $f_0(v)_i$. Let $\mathbf{PLGraph}_\Sigma$ be the category of pl-graphs and pl-graph morphisms over a set of controls Σ.

The intuition is that S is the set of equivalence classes of ports in G. When two ports are in the same equivalence class (that is they map to the same element of S), we say that they are connected. Figure 6.2 illustrates a typical place-link graph with two kind of controls – A and B, respectively with three and two ports – where directed arcs represent the place structure, and the undirected ones are the elements of S.

It is easy to construct for each Σ a category $\mathbf{X}_\Sigma$ so that $\mathbf{PLGraph}_\Sigma \cong \mathbf{Set}^{\mathbf{X}_\Sigma}$, in other words, $\mathbf{PLGraph}_\Sigma$ is a presheaf category and, as such, adhesive.

Definition 6.3.15 (PL-Graphs with Interfaces). We shall refer to the bicategory $\mathrm{Cospan}^{\cong}(\mathbf{PLGraph}_\Sigma)$ as the bicategory of pl-graphs with inter-

faces. Restricting to input-linear cospans, we obtain the bicategory which we shall denote ILC($\mathbf{PLGraph}_\Sigma$).

Corollary 6.3.16. ILC($\mathbf{PLGraph}_\Sigma$) *has GRPOs, calculated using Algorithm 6.2.2.*

There are two aspects of our bicategory of pl-graphs with interfaces which generalise the theory of bigraphs. Firstly, the theory of bigraphs one traditionally considers only "discrete" interfaces, i.e., discrete place-graphs (which can be seen as strings over the alphabet Σ) together with *name sets* (cf. Definition 6.3.17). Secondly, place graphs are usually forests of trees, and their input (resp. output) interfaces reach only leaves (resp. roots). Fortunately, we can apply the necessary restrictions and still be able to perform the construction of Algorithm 6.2.2.

Definition 6.3.17 (Discrete PL-Graph). A discrete pl-graph $\langle m, X\rangle$, where m is a finite ordinal labelled over Σ and X is a finite *set of names*, is a pl-graph with m as its set of nodes and no edges. Its link map is the injection $P \to P + X$, for P the set of ports of m.

Let $\mathbf{TPLGraph}_\Sigma$ be the full subcategory of $\mathbf{PLGraph}_\Sigma$ consisting of pl-graphs with forests of trees as place-graphs.

Definition 6.3.18 (Bicategory of Bigraphs). The bicategory of bigraphs $\mathbf{Bigraph}_\Sigma$ over a set of controls Σ has:

- Objects: Discrete pl-graphs $\langle m, X\rangle$;
- Arrows: Input-linear cospans

$$\langle m, X\rangle \rightarrowtail G \longleftarrow \langle n, Y\rangle$$

  where $G \in \mathbf{TPLGraph}_\Sigma$; additionally, the left interface must reach only the leaves of G while the right interface must reach only the roots of G;
- 2-cells: isomorphisms between cospans.

Theorem 6.3.19. $\mathbf{Bigraph}_\Sigma$ *has GRPOs.*

Proof. It suffices to verify that Algorithm 6.2.2 preserves the conditions on cospans, and that I_5 is a discrete bigraph. □

Interestingly, a main difference between bigraphs as defined here and Milner's [73] is the effect of input-linearity on aliasing of names. Milner's formalism allows a bigraph which equates names from its input interface, which is disallowed by our input-linearity condition. Conversely, our formalism allows a bigraph which equates names within its output interface, an operation not allowed in Milner's. Observe that since (G)RPOs-derived contexts only exercise output interfaces, expressive power may be lost in some applications by not being able to equate ports in the input interface. Although this appears to be often compensated for by equating ports in output interfaces, we are currently investigating a generalised notion of relative pushout which acts at the same time on both interfaces.

In particular, in the link-graphs illustrated below,

the left one is not input-linear, and therefore, not allowed by our formalism. On the other hand, the right one, which is input-linear but not output-linear, is allowed in our formalism but not in Milner's.

It appears that an output-linear version of Definition 6.3.18 would capture the existing bigraphs almost exactly. We leave a general construction of GIPOs for output-linear cospan bicategories as future work.

Bibliography

[1] S. Abramsky. The lazy lambda calculus. In D. Turner, editor, *Research Topics in Functional Programming*, pages 65–117. Addison Wesley, 1990.

[2] L. Aceto, W. J. Fokkink, and C. Verhoef. Structural operational semantics. In J. Bergstra, A. Ponse, and S. Smolka, editors, *Handbook of Process Algebra*. Elsevier, 2002.

[3] P. Aczel and M. P. Mendler. A final coalgebra theorem. In *Category Theory and Computer Science CTCS '89*, volume 389 of *LNCS (Lecture Notes in Computer Science)*. Springer, 1989.

[4] P. Baldan, A. Corradini, H. Ehrig, M. Löwe, U. Montanari, and F. Rossi. Concurrent semantics of algebraic graph transformations. In H. Ehrig, H.-J. Kreowski, U. Montanari, and G. Rozenberg, editors, *Handbook of Graph Grammars and Computing by Graph Transformation*, volume 3, chapter 3, pages 107–187. World Scientific, 1999.

[5] H. Barendregt. *The Lambda Calculus: Its Syntax and Semantics*. North Holland, 1984.

[6] M. Barr. Terminal coalgebras in well-founded set theory. *Theoretical Computer Science*, 114:299–315, 1993.

[7] J. Bénabou. Introduction to bicategories. In *Midwest Category Seminar*, volume 42 of *Lecture Notes in Mathematics*, pages 1–77. Springer-Verlag, 1967.

[8] D. B. Benson. The basic algebraic structures in categories of derivations. *Information and Control*, 28(1):1–29, 1975.

[9] G. Berry and G. Boudol. The chemical abstract machine. *Theoretical Computer Science*, 96:217–248, 1992.

[10] B. Bloom, S. Istrail, and A. Meyer. Bisimulation can't be traced. *Journal of the ACM*, 42:232–268, 1995.

[11] R. Brown and G. Janelidze. Van Kampen theorems for categories of covering morphisms in lextensive categories. *J. Pure Appl. Algebra*, 119:255–263, 1997.

[12] A. Carboni, S. Lack, and R. F. C. Walters. Introduction to extensive and distributive categories. *Journal of Pure and Applied Algebra*, 84(2):145–158, February 1993.

[13] L. Cardelli and A. D. Gordon. Mobile ambients. In *Foundations of Software Science and Computation Structures, FoSSaCS '98*. Springer Verlag, 1998.

[14] G. Castagna and F. Zappa Nardelli. The Seal calculus revisited. In *Foundations of Software Technology and Theoretical Computer Science, FST&TCS '02*, volume 2556 of *LNCS (Lecture Notes in Computer Science)*, pages 85–96. Springer, 2002.

[15] A. Corradini, H. Ehrig, R. Heckel, M. Lowe, U. Montanari, and F. Rossi. Algebraic approaches to graph transformation part i: Basic concepts and double pushout approach. In G. Rozenberg, editor, *Handbook of Graph Grammars and Computing by Graph Transformation*, volume 1, pages 162–245. World Scientific, 1997.

[16] A. Corradini and F. Gadducci. A 2-categorical presentation of term graph rewriting. In *Category Theory and Computer Science, CTCS '97*, volume 1290 of *LNCS (Lecture Notes in Computer Science)*, pages 87–105. Springer, 1997.

[17] A. Corradini, R. Heckel, and U. Montanari. Compositional SOS and beyond: a coalgebraic view of open systems. *Theoretical Computer Science*, 280:163–192, 2002.

[18] H. Ehrig. Introduction to the algebraic theory of graph grammars. In *1st Int. Workshop on Graph Grammars*, volume 73 of *LNCS (Lecture Notes in Computer Science)*, pages 1–69. Springer Verlag, 1979.

[19] H. Ehrig, G. Engels, H.-J. Kreowski, U. Montanari, and G. Rozenberg, editors. *Handbook of Graph Grammars and Computing by Graph Transformation, Volumes 1-3*. World Scientific, 1997-1999.

[20] H. Ehrig, G. Engels, H.-J. Kreowski, and G. Rozenberg, editors. *Handbook of Graph Grammars and Computing by Graph Transformation, Volume 2: Applications, Languages and Tools*. World Scientific, 1999.

[21] H. Ehrig, M. Gajewsky, and F. Parisi-Presicce. High-level replacement systems with applications to algebraic specifications and Petri Nets. In H. Ehrig, H.-J. Kreowski, U. Montanari, and G. Rozenberg, editors, *Handbook of Graph Grammars and Computing by Graph Transformation*, volume 3, chapter 6, pages 341–400. World Scientific, 1999.

[22] H. Ehrig, A. Habel, H.-J. Kreowski, and F. Parisi-Presicce. From graph grammars to high level replacement systems. In *4th Int. Workshop on Graph Grammars and their Application to Computer Science*, volume 532 of *LNCS (Lecture Notes in Computer Science)*, pages 269–291. Springer Verlag, 1991.

[23] H. Ehrig, A. Habel, H.-J. Kreowski, and F. Parisi-Presicce. Parallelism and concurrency in high-level replacement systems. *Math. Struct. in Comp. Science*, 1, 1991.

[24] H. Ehrig, A. Habel, J. Padberg, and U. Prange. Adhesive high-level replacement categories and systems. In *International Conference on Graph Transformation ICGT '04*, 2004. To appear.

[25] H. Ehrig and B. König. Deriving bisimulation congruences in the dpo approach to graph rewriting. In *Foundations of Software Science and Computation Structures FoSSaCS '04*, volume 2987 of *LNCS (Lecture Notes in Computer Science)*, pages 151–166. Springer, 2004.

[26] H. Ehrig, H.-J. Kreowski, U. Montanari, and G. Rozenberg, editors. *Handbook of Graph Grammars and Computing by Graph Transformation, Volume 3: Concurrency, Parallelism and Distribution*. World Scientific, 1999.

[27] H. Ehrig, M. Pfender, and H. Schneider. Graph-grammars: an algebraic approach. In *IEEE Conf. on Automata and Switching Theory*, pages 167–180, 1973.

[28] U. Engberg and M. Nielsen. A calculus of communicating systems with label passing. Technical Report DAIMI PB-208, University of Aarhus, 1986.

[29] M. Fiore, G. L. Cattani, and G. Winskel. Weak bisimulation and open maps. In *Logic in Computer Science, LICS '99*, pages 67–76. IEEE Computer Society Press, 1999.

[30] M. P. Fiore and S. Staton. Comparing operational models of name-passing process calculi. In *Coalgebraic Methods in Computer Science CMCS '04*, ENTCS. Elsevier, 2004. To appear.

[31] M. P. Fiore and D. Turi. Semantics of name and value passing. In *Logic in Computer Science LICS '01*, pages 93–104. IEEE, 2001.

[32] C. Fournet and G. Gonthier. A hierarchy of equivalences for asynchronous calculi. In *International Colloquium on Automata, Languages and Programming, ICALP '98*, volume 1443 of *LNCS (Lecture Notes in Computer Science)*. Springer, 1998.

[33] F. Gadducci and R. Heckel. An inductive view of graph transformation. In *Recent Trends in Algebraic Development Techniques*, volume 1376 of *LNCS (Lecture Notes in Computer Science)*, pages 219–233. Springer, 1998.

[34] F. Gadducci and U. Montanari. Enriched categories as models of computations, 1996.

[35] F. Gadducci and U. Montanari. The tile model. In G. Plotkin, C. Stirling, and M. Tofte, editors, *Proof, Language and Interaction: Essays in honour of Robin Milner*. MIT Press, 2000.

[36] F. Gadducci and U. Montanari. A concurrent graph semantics for mobile ambients. In *Mathematical Foundations of Programming Semantics MFPS '01*, volume 45 of *ENTCS*. Elsevier, 2001.

[37] R. Gates. *On Extensive and Distributive Categories*. PhD thesis, University of Sydney, 1997.

[38] R. Godement. *Théorie des faisceaux*. Hermann, 1958.

[39] J. C. Godskesen, T. Hildebrandt, and V. Sassone. A calculus of mobile resources. In *International Conference on Concurrency Theory, Concur '02*, volume 2421 of *LNCS (Lecture Notes in Computer Science)*, pages 272–287. Springer, 2002.

[40] M. Hennessy. *Algebraic Theory of Process Languages*. MIT Press, 1988.

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

[42] C. A. R. Hoare. *Communicating Sequential Processes*. Prentice Hall, 1985.

[43] K. Honda and N. Yoshida. On reduction-based process semantics. *Theoretical Computer Science*, 151(2):437–486, 1995.

[44] D. J. Howe. Proving congruence of bisimulation in functional programming languages. *Information and Computation*, 124(2):103–112, 1996.

[45] O. H. Jensen. *TBA*. PhD thesis, University of Aalborg, 2004. To appear.

[46] O. H. Jensen and R. Milner. Bigraphs and mobile processes. Technical Report 570, Computer Laboratory, University of Cambridge, 2003.

[47] O. H. Jensen and R. Milner. Bigraphs and transitions. In *Principles of Programming Languages, POPL '03*. ACM Press, 2003.

[48] A. Joyal, M. Nielsen, and G. Winskel. Bisimulation from open maps. *Information and Computation*, 127(2):164–185, 1996.

[49] A. Joyal, R. Street, and D. Verity. Traced monoidal categories. *Mathematical Proceedings of the Cambridge Philosophical Society*, 119(3):447–468, 1996.

[50] P. Katis, N. Sabadini, and R. F. C. Walters. Bicategories of processes. *Journal of Pure and Applied Algebra*, 115:141–178, 1997.

[51] P. Katis, N. Sabadini, and R. F. C. Walters. Span(Graph):an algebra of transition systems. In *International Conference on Algebraic Methodology and Software Technology AMAST '97*, number 1349 in LNCS (Lecture Notes in Computer Science), pages 322–336. Springer, 1997.

[52] G. M. Kelly. *Basic Concepts of Enriched Category Theory*, volume 64 of *London Mathematical Society Lecture Notes*. Cambridge University Press, 1982.

[53] G. M. Kelly. Elementary observations on 2-categorical limits. *Bull. Austral. Math. Soc.*, 39:301–317, 1989.

[54] G. M. Kelly and R. H. Street. Review of the elements of 2-categories. *Lecture Notes in Mathematics*, 420:75–103, 1974.

[55] B. Klin. *An Abstract Coalgebraic Approach to Process Equivalence for Well-Behaved Operational Semantics*. PhD thesis, BRICS, University of Aarhus, 2004.

[56] B. Klin and P. Sobociński. Syntactic formats for free: An abstract approach to process equivalence. In *International Conference on Concurrency Theory Concur '03*, volume 2620 of *LNCS (Lecture Notes in Computer Science)*, pages 72–86. Springer, 2003.

[57] H.-J. Kreowski. Transformations of derivation sequences in graph grammars. In *LNCS (Lecture Notes in Computer Science)*, volume 56, pages 275–286, 1977.

[58] S. Lack and P. Sobociński. Van Kampen squares and adhesive categories. In preparation, 2003.

[59] S. Lack and P. Sobociński. Adhesive categories. In *Foundations of Software Science and Computation Structures FoSSaCS '04*, volume 2987 of *LNCS (Lecture Notes in Computer Science)*, pages 273–288. Springer, 2004.

[60] J. Lambek and P. J. Scott. *Introduction to higher order categorical logic*, volume 7 of *Cambridge studies in advanced mathematics*. Cambridge University Press, 1986.

[61] K. G. Larsen. A context dependent equivalence between processes. *Theoretical Computer Science*, 49:185–215, 1987.

[62] F. W. Lawvere. Functorial semantics of algebraic theories. *Proceedings, National Academy of Sciences*, 50:869–873, 1963.

[63] F. W. Lawvere. Some thoughts on the future of category theory. In *Category Theory*, volume 1488 of *Lecture Notes in Mathematics*, pages 1–13, Como, 1991. Springer-Verlag.

[64] J. Leifer. *Operational congruences for reactive systems*. Phd thesis, University of Cambridge, 2001.

[65] J. Leifer. Synthesising labelled transitions and operational congruences in reactive systems, parts 1 and 2. Technical Report RR-4394 and RR-4395, INRIA Rocquencourt, 2002.

[66] J. Leifer and R. Milner. Deriving bisimulation congruences for reactive systems. In *International Conference on Concurrency Theory Concur '00*, volume 1877 of *LNCS (Lecture Notes in Computer Science)*, pages 243–258. Springer, 2000.

[67] S. Mac Lane. *Categories for the Working Mathematician*. Graduate Texts in Mathematics. Springer Verlag, 2nd edition, 1992.

[68] S. Mac Lane and R. Paré. Coherence for bicategories and indexed categories. *Journal of Pure and Applied Algebra*, pages 59–80, 1985.

[69] D. Masulovic and J. Rothe. Towards weak bisimulation for coalgebras. *Electronic Notes in Theoretical Computer Science*, 68(1), 2003.

[70] R. Milner. *Communication and Concurrency*. Prentice Hall, 1989.

[71] R. Milner. Calculi for interaction. *Acta Informatica*, 33(8):707–737, 1996.

[72] R. Milner. *Communicating and Mobile Systems: the Pi-calculus*. Cambridge University Press, 1999.

[73] R. Milner. Bigraphical reactive systems. In *International Conference on Concurrency Theory Concur '01*, volume 2154 of *LNCS (Lecture Notes in Computer Science)*, pages 16–35. Springer, 2001.

[74] R. Milner, J. Parrow, and D. Walker. A calculus of mobile processes, (Parts I and II). *Information and Computation*, 100:1–77, 1992.

[75] R. Milner and D. Sangiorgi. Barbed bisimulation. In *9th Colloquium on Automata, Languages and Programming, ICALP92*, volume 623 of *LNCS (Lecture Notes in Computer Science)*, pages 685–695. Springer, 1992.

[76] U. Montanari and V. Sassone. Dynamic congruence vs. progressing bisimulation for CCS. *Fundamenta Informaticae*, XVI:171–199, 1992.

[77] U. Montanari and V. Sasssone. Dynamic bisimulation. Technical report, Univerisità di Pisa, 1990.

[78] M. Nygaard. *Domain Theory for Concurrency*. PhD thesis, BRICS, University of Aarhus, 2003.

[79] M. Nygaard and G. Winskel. HOPLA - A higher-order process language. In *International Conference on Concurrency Theory Concur '02*, volume 2421 of *LNCS (Lecture Notes in Computer Science)*, pages 434–448. Springer, 2002.

[80] M. Nygaard and G. Winskel. Full abstraction for HOPLA. In *International Conference on Concurrency Theory Concur '03*, volume 2761 of *LNCS (Lecture Notes in Computer Science)*, pages 378–392. Springer, 2003.

[81] M. Nygaard and G. Winskel. Domain theory for concurrency. *Theoretical Computer Science*, 316:152–190, 2004.

[82] D. Park. Concurrency on automata and infinite sequences. In P. Deussen, editor, *Conf. on Theoretical Computer Science*, volume 104 of *LNCS (Lecture Notes in Computer Science)*. Springer, 1981.

[83] G. D. Plotkin. LCF considered as a programming language. *Theoretical Computer Science*, 5:223–255, 1977.

[84] G. D. Plotkin. A structural approach to operational semantics. Technical Report FN-19, DAIMI, Computer Science Department, Aarhus University, 1981.

[85] E. P. Robinson and G. Rosolini. Categories of partial maps. *Information and Computation*, 79, 1988.

[86] A. W. Roscoe. *The Theory and Practice of Concurrency*. Prentice Hall, 1997.

[87] G. Rozenberg, editor. *Handbook of Graph Grammars and Computing by Graph Transformation, Volume 1: Foundations*. World Scientific, 1997.

[88] J. J. M. M. Rutten. Universal coalgebra: a theory of systems. *Theoretical Computer Science*, 249:3–80, 2000.

[89] D. Sangiorgi and D. Walker. *The π-calculus: a Theory of Mobile Processes*. Cambridge University Press, 2001.

[90] D. Sannella and A. Tarlecki. Essential concepts of algebraic specification and program development. *Formal Aspects of Computing*, 9:229–269, 1997.

[91] V. Sassone and P. Sobociński. Deriving bisimulation congruences: A 2-categorical approach. *Electronic Notes in Theoretical Computer Science*, 68(2), 2002.

[92] V. Sassone and P. Sobociński. Deriving bisimulation congruences: 2-categories vs. precategories. In *Foundations of Software Science and Computation Structures FoSSaCS '03*, volume 2620 of *LNCS (Lecture Notes in Computer Science)*. Springer, 2003.

[93] V. Sassone and P. Sobociński. Deriving bisimulation congruences using 2-categories. *Nordic Journal of Computing*, 10(2):163–183, 2003.

[94] V. Sassone and P. Sobociński. Congruences for contextual graph-rewriting. Technical Report RS-04-11, BRICS, University of Aarhus, June 2004.

[95] V. Sassone and P. Sobociński. Locating reaction with 2-categories. *Theoretical Computer Science*, 2004. To appear.

[96] S. H. Schanuel. Negative sets have Euler characteristic and dimension. In *Category Theory*, volume 1488 of *Lecture Notes in Mathematics*, pages 379–385, Como, 1991. Springer-Verlag.

[97] R. A. G. Seely. Modelling computations: a 2-categorical framework. In *Logic in Computer Science LICS '87*. IEEE Computer Society, 1987.

[98] P. Sewell. From rewrite rules to bisimulation congruences. *LNCS (Lecture Notes in Computer Science)*, 1466:269–284, 1998.

[99] P. Sewell. Working note PS14, March 2000. Unpublished.

[100] R. H. Street. Fibrations in bicategories. *Cahiers de topologie et géométrie différentielle*, XXI-2:111–159, 1980.

[101] R. H. Street. Categorical structures. In M. Hazewinkel, editor, *Handbook of algebra*, volume vol. 1, pages 529–577. North-Holland, 1996.

[102] D. Turi and G. Plotkin. Towards a mathematical operational semantics. In *Logic in Computer Science, LICS'97*, pages 280–291. IEEE Computer Society Press, 1997.

[103] R. J. van Glabbeek. The linear time - branching time spectrum II: The semantics of sequential processes with silent moves. In *International Conference on Concurrency Theory, Concur '93*, volume 715 of *LNCS (Lecture Notes in Computer Science)*, pages 66–81. Springer, 1993.

[104] R. J. van Glabbeek. The linear time – branching time spectrum I. In J. A. Bergstra, A. Ponse, and S. Smolka, editors, *Handbook of Process Algebra*. Elsevier, 1999.

[105] J. Vitek and G. Castagna. A calculus of secure mobile computations. In *IEEE Workshop on Internet Programming Languages*, 1998.

[106] R. F. C. Walters. Datatypes in distributive categories. *Bull. Austral. Math. Soc.*, 40:79–82, 1989.

[107] R. F. C. Walters. *Categories and Computer Science*. Carslaw Publications, 1991.

Recent BRICS Dissertation Series Publications

DS-04-6 **Paweł Sobocinski.** ***Deriving Process Congruences from Reaction Rules*****. December 2004. PhD thesis. xii+216 pp.**

DS-04-5 **Bjarke Skjernaa.** ***Exact Algorithms for Variants of Satisfiability and Colouring Problems*****. November 2004. PhD thesis. x+112 pp.**

DS-04-4 **Jesper Makholm Byskov.** ***Exact Algorithms for Graph Colouring and Exact Satisfiability*****. November 2004. PhD thesis.**

DS-04-3 **Jens Groth.** ***Honest Verifier Zero-knowledge Arguments Applied*****. October 2004. PhD thesis. xii+119 pp.**

DS-04-2 **Alex Rune Berg.** ***Rigidity of Frameworks and Connectivity of Graphs*****. July 2004. PhD thesis. xii+173 pp.**

DS-04-1 **Bartosz Klin.** ***An Abstract Coalgebraic Approach to Process Equivalence for Well-Behaved Operational Semantics*****. May 2004. PhD thesis. x+152 pp.**

DS-03-14 **Daniele Varacca.** ***Probability, Nondeterminism and Concurrency: Two Denotational Models for Probabilistic Computation*****. November 2003. PhD thesis. xii+163 pp.**

DS-03-13 **Mikkel Nygaard.** ***Domain Theory for Concurrency*****. November 2003. PhD thesis. xiii+161 pp.**

DS-03-12 **Paulo B. Oliva.** ***Proof Mining in Subsystems of Analysis*****. September 2003. PhD thesis. xii+198 pp.**

DS-03-11 **Maciej Koprowski.** ***Cryptographic Protocols Based on Root Extracting*****. August 2003. PhD thesis. xii+138 pp.**

DS-03-10 **Serge Fehr.** ***Secure Multi-Player Protocols: Fundamentals, Generality, and Efficiency*****. August 2003. PhD thesis. xii+125 pp.**

DS-03-9 **Mads J. Jurik.** ***Extensions to the Paillier Cryptosystem with Applications to Cryptological Protocols*****. August 2003. PhD thesis. xii+117 pp.**

DS-03-8 **Jesper Buus Nielsen.** ***On Protocol Security in the Cryptographic Model*****. August 2003. PhD thesis. xiv+341 pp.**

DS-03-7 **Mario José Cáccamo.** ***A Formal Calculus for Categories*****. June 2003. PhD thesis. xiv+151.**

DS-03-6 **Rasmus K. Ursem.** ***Models for Evolutionary Algorithms and Their Applications in System Identification and Control Optimization*****. June 2003. PhD thesis. xiv+183 pp.**