Adjoint Rewriting

Neil Ghani

Ph.D.

University of Edinburgh

Graduation November 1995

Abstract

This thesis concerns rewriting in the typed λ-calculus. Traditional categorical models of typed λ-calculus use concepts such as functor, adjunction and algebra to model type constructors and their associated introduction and elimination rules, with the natural categorical equations inherent in these structures providing an equational theory for λ-terms. One then seeks a rewrite relation which, by transforming terms into canonical forms, provides a decision procedure for this equational theory. Unfortunately the rewrite relations which have been proposed, apart from for the most simple of calculi, either generate the full equational theory but contain no decision procedure, or contain a decision procedure but only for a subtheory of that required.

Our proposal is to unify the semantics and reduction theory of the typed λ-calculus by generalising the notion of model from categorical structures based on term equality to categorical structures based on term reduction. This is accomplished via the addition of a pre-order to each of the hom-sets of the category which is used to reflect the reduction of one term to another. Rewrite relations, whose associated equational theory matches that suggested by the traditional semantics, may then be derived from the natural categorical constructions on these ordered categories.

Rewrite relations derived in this fashion typically consist of a contractive β-rewrite rule and an expansionary η-rewrite rule for each type constructor. Although confluent, the presence of expansionary η-rewrite rules means the rewrite relation is not strongly normalising and so cannot in itself be used as the basis of a decision procedure. Instead, decidability of the equational theory is proved by a variety of term rewriting techniques which will necessarily vary from calculus to calculus. These techniques are developed in three case studies: the simply typed λ-calculus with unit, product and exponential types; the linear λ-calculus containing the tensor from linear logic; and the bicartesian closed calculus obtained by adding coproducts to the simply typed λ-calculus.

Acknowledgements

First and foremost I would like to express my gratitude to my supervisor Barry Jay who not only introduced me to the applications of category theory in term rewriting and provided many insights and suggestions, but also gave me a lot of help and advice in dealing with the ups and downs of postgraduate life.

I would also like to thank my other supervisors Don Sannella and Stefan Kahrs for their help throughout my time in Edinburgh and especially for patiently reading and correcting numerous drafts of my thesis. Thanks also to the numerous people who have listened patiently to my ideas and given me some rather better ones of their own. These include Roberto Di Cosmo, Christoph Lüth, Alex Simpson, Martin Hoffman, Peter Sewell and my internal examiner Alan Smaill. I have been financially supported by a SERC grant and also benefited from the use of Paul Taylors Diagrams package.

Finally, I would like to thank all the people who've helped me in other ways over the years. My parents have always supported me and made numerous sacrifices on my behalf. I also want to thank Simon McPartlin, Roope Kaivola, Brian McConnell, Julian Richardson, David Keeble and numerous others who have kept me company at various times. Without the efforts of Kevin Keegan and Alex Miller the world would surely not have been so pleasant a place. Finally Maria and Dan are a bit special.

Declaration

I hereby declare that this thesis has been composed by myself, that the work is my own, and that the ideas and results that I do not attribute to others are due to myself. The basic results of Chapter 3 have been published in [49] while the basic results of Chapter 5 have been published in [29].

Table of Contents

Chapter 1

Introduction

The λ-calculus has become one of the cornerstones of theoretical computer science because it provides a simple yet elegant formalism capable of representing exactly those functions which are computable by a Turing machine. Each computable function has many different representations as terms of the λ-calculus and, in order to reason about such functions, two natural questions must be asked:

- When do different terms of the λ-calculus represent the same function, i.e. which equations should the terms of the λ-calculus satisfy?

- Given such an equational theory, is there a decision procedure to tell if two terms are equal?

Although the problem is known to be undecidable in general, the search for decidable fragments of the λ-calculus has been the focus of much research. This thesis develops the techniques required to prove decidability for various typed variants of the λ-calculus, although features such as impredicativity, type dependency and recursive types are postponed for future research. The main insight of this thesis is that the process by which the terms of our example λ-calculi are formed can be described by a mathematical structure known as an adjunction, and that these adjunctions contain not only an implicit equality on terms, but also a rewrite relation which forms the basis of a decision procedure for this equality.

The first of our example calculi is the simply typed λ-calculus where η-conversion has been traditionally interpreted as a contraction:

$$\lambda x.tx \quad \Rightarrow \quad t \quad \text{if } t\!:\!A\!\rightarrow\! B \tag{1.1}$$

The resulting rewrite relation (including the usual β-reductions) is confluent and strongly normalising, and so reduction to normal form provides an effective procedure for deciding the associated equational theory. Unfortunately the addition of further datatypes typically causes these properties to fail, e.g. even the introduction of a unit type (necessary for defining types with given constants, such as booleans and lists) causes confluence to be lost. Specifically, if $* : 1$ is the given constant of unit type, with η-rewrite rule:

$$a \Rightarrow * \qquad \text{if } a\!:\!1$$

and if f is any variable of type $A\!\rightarrow\!1$, then the term $\lambda x.fx$ has two normal forms:

$$\lambda x.* \;\; \Leftarrow \;\; \lambda x.fx \;\; \Rightarrow \;\; f$$

Similar problems arise if we try to enrich the λ-calculus with extra rewrite rules which may be confluent by themselves, but which when taken in conjunction with η-contraction fail to be confluent [16]. Recently several researchers [2,15,20,19,22, 49] have adopted older proposals [41,62,68] that η-conversion be interpreted as an expansion:

$$t \quad \Rightarrow \quad \lambda x.tx \quad \text{if } t\!:\!A\!\rightarrow\! B$$

and the resulting rewrite relation has been shown confluent. In these works infinite reduction sequences such as:

$$f \;\Rightarrow \lambda x.fx \;\Rightarrow \lambda x.(\lambda y.fy)x \;\Rightarrow \;\ldots \tag{1.2}$$

are prohibited by imposing syntactic restrictions to limit the possibilities for expansion: namely, λ-abstractions cannot be expanded and nor can terms which are applied. The resulting rewrite relation is confluent and strongly normalising, e.g. in reduction sequence 1.2 all but the first η-expansion is prevented from occurring. If $t \Rightarrow t'$ is an η-expansion which is prevented from occurring by these restrictions, then there is a β-rewrite $t' \Rightarrow t$ in the reverse direction and hence the equational theory of the restricted rewrite relation is the same as that of the unrestricted rewrite relation. Thus $\beta\eta$-equality can be decided by reduction to normal form in the restricted fragment, although there is a price to be paid for this decidability result: namely reduction is no longer a pre-congruence on terms.

This interpretation of η-conversion as an expansion, and the restrictions required to recover strong normalisation, have a mathematical explanation within categorical models of reduction [48,70,75] where types are represented as objects, terms as morphisms and reductions as 2-cells. In such models the introduction and elimination rules for the exponential type constructor form an adjoint pair of functors whose local unit and counit [33,45] correspond to η-expansion and β-contraction respectively. These rewrite rules are linked by local triangle laws and the restrictions on the applicability of the expansionary η-rewrite rule prevent exactly those expansions which occur in these triangle laws. This thesis demonstrates how these categorical methods can be extended to formally derive rewrite rules for a wide variety of type constructors, including other final type constructors such as the *product*, initial type constructors such as the *coproduct* and the *tensor* of linear logic, and even recursive types such as the *natural numbers* type.

After some categorical and rewriting preliminaries, we give a brief introduction to typed λ-calculus and show how an equational theory for such a calculus may be derived by using traditional semantic models based on categories with extra equational structure [44,55,73]. Then, by generalising the notion of a model from these categorical structures based on term equality to categorical structures based on term reduction, we show how rewrite relations may be derived whose associated equational theory matches that suggested by the traditional semantics. Rewrite relations derived in this fashion typically consist of a contractive β-rewrite rule and an expansionary η-rewrite rule for each type constructor. Thus the associated equational theory will be called *$\beta\eta$-equality* and include not just β- and η-equations for the exponentials but also β- and η-equations for all the other type constructors present.

One nice feature of this approach is its modularity in that the rewrite rules for a given type constructor are determined solely by the introduction and elimination rules of that type constructor and so remain unaffected by the presence or absence of other type constructors. I'm not sure whether this modularity can be taken further, eg by obtaining a general confluence proof for arbitrary rewrite rules derived from our categorical methods, but recent developments in modular term rewriting [54] may provide a starting point.

Although confluent, the presence of expansionary η-rewrite rules means that these rewrite relations contain infinite reduction sequences, e.g. reduction sequence 1.2, and so cannot in themselves be used as a decision procedure. Instead decidability of the associated equational theories, e.g. via the construction of canonical elements of each equivalence class, must be proved by a variety of term rewriting techniques which will necessarily vary from calculus to calculus. These techniques are developed in three case studies, each of which is of strictly greater complexity than its predecessors, although each case study may be read independently of the others.

The Simply Typed λ-Calculus, $\Lambda^{1,\times,\rightarrow}$

The type constructors we study fall into two distinct varieties: namely, *final* type constructors such as the *product* and *exponential*, and *initial* type constructors such as the *coproduct* and the *tensor* of linear logic. For reasons which will become apparent later, the η-rewrite rules for final type constructors are significantly simpler than those for initial type constructors, and so the first of our case studies is a calculus which contains only final type constructors.

After defining the simply typed λ-calculus, we show how categorical models of reduction may be used to derive an expansionary η-rewrite rule and contractive β-rewrite rule for each type constructor and that the associated equational theory is sound and complete for models in cartesian closed categories. The second half of the chapter defines a subrelation by prohibiting those expansions occurring in the triangle laws inherent in the adjunctions defining the rewrite rules. This restricted rewrite relation generates the same equational theory as the unrestricted fragment and is proved strongly normalising and confluent by modifying the traditional proofs so as to cope with the presence of expansions and the non-congruent nature of reduction. Finally we show that the normal forms of the restricted rewrite relation are exactly Huet's *long $\beta\eta$-normal forms* [41] and that these normal forms may be calculated by first contracting all β-redexes and then performing any remaining η-expansions or, vice versa, by performing all η-expansions and then contracting any remaining β-redexes.

Historically the use of η-expansions, as opposed to η-contractions, can be traced back to Prawitz [68] and Huet [41]. The formulation of the restrictions on expansion required to recover strong normalisation were originally proposed by Mints [62] although it is only recently that several researchers [2,15,19,22,49], using different proof strategies, have proved confluence and strong normalisation.

The Linear λ-Calculus, $\Lambda^{I,\otimes,\rightarrow}$

Although substantial agreement exists on the nature of rewriting for final type constructors such as those contained in the simply typed λ-calculus, rewrite relations for initial type constructors remain the subject of much research with only limited results so far. Our second case study has two important features: firstly there are two initial type constructors called the *tensor* and *unit* and secondly, the structural rules *weakening* and *contraction* are omitted so as to permit a clearer account of the general properties of rewriting for initial type constructors unhindered by mathematical technicalities caused by the structural rules.

Categorical models of reduction are again used to derive a rewrite relation consisting of an expansionary η-rewrite rule and a contractive β-rewrite rule for each type constructor. Although originally thought to be a straightforward extension of previous results, the η-rewrite rules associated to the initial type constructors turn out to be substantially more complex than those previously studied. In addition to a facility for expanding terms similar to that possessed by the η-rewrite rule for the exponential, the proposed η-rewrite rules for the tensor and unit also possess the ability to permute the order in which different subterms of tensor and unit type occur.

These different aspects of η-conversion for initial type constructors are reflected in the decomposition of the full rewrite relation into two fragments. The first fragment mirrors the restricted rewrite relation of the simply typed λ-calculus in containing all β-reductions and limited possibilities for η-expansion. The relation is strongly normalising, confluent and has normal forms which satisfy similar structural properties to the long $\beta\eta$-normal forms of the simply typed λ-calculus.

The second part of the decomposition is called the *conversion relation* and is new to the field. The conversion relation permutes the order in which certain subterms of an initial type occur and, because each term typically has a non-empty and finite set of such permutations, terms cannot be rewritten to unique normal forms in the conversion relation. Instead we associate to each term a set of *quasi-normal forms*, one for each possible permutation, and prove that terms equivalent in the conversion relation have the same set of quasi-normal forms. Finally the full equational theory is proved decidable by appropriately embedding it into the conversion relation.

Almost Bicartesian Closed λ-Calculus, $\Lambda^{1,\times,\rightarrow,+}$

The final case study investigates the technical complications that are caused by the re-introduction of the structural rules *weakening* and *contraction*. "Almost Bicartesian Closed λ-calculus" is the calculus obtained by adding coproducts, also called *sums*, to the simply typed λ-calculus and — as coproducts are an initial type constructor — our analysis follows the same general pattern as that for the linear λ-calculus. Once a rewrite relation has been deduced from general categorical principles, it is again decomposed into two fragments which are shown to satisfy the same abstract properties as their counterparts of the last chapter. However the non-linear use of variables significantly complicates the mathematics required to establish these results.

Coproducts have received considerable attention recently [29,22,23] with several authors attempting to extend the results concerning the simply typed λ-calculus to this calculus. This proved to be substantially more difficult than first expected and so I chose first to study the linear λ-calculus as a simpler calculus which nevertheless contains initial type constructors. At the time of writing, the research presented here remains the only proof of decidability of the theory of coproducts. One other interesting result is [23] which extends [25] in providing a proof system for deriving a set of equations which is sound and complete for all "set-theoretic" models of a λ-calculus with exponentials and coproducts. This theory is then proved decidable by proving it is equivalent to the one presented here.

1.1 Notational Preliminaries

Categorical Preliminaries

The reader is assumed familiar with the categorical concepts of *category, functor, natural transformation, adjoint, monad* and the various uses of *algebra* in the literature. We also assume the reader is familiar with the use of categories to provide a class of models, and hence an equational theory, for various typed λ-calculi. Standard references are [6,55,59]. The following (large) categories will be needed later:

Sets	The category of all (small) sets.
Cat	The category of all (small) categories.
SMC	The category of all (small) symmetric monoidal closed categories.
CCC	The category of all (small) cartesian closed categories.
ABCC	The category of all (small) cartesian closed categories with binary coproducts.

Category theory is also used in this thesis to derive rewrite relations for typed λ-calculi by generalising the notion of a model from categorical structures based on term equality to categorical structures based on term reduction. Just as the natural constructions on traditional models may be used to derive an equational theory for the terms of the λ-calculus, so the natural categorical constructions on these ordered categories [33,39,48,47,75,72] implicitly contain rewrite relations whose equational theory matches that suggested by the traditional semantics. Although these structures, and the more general *enriched categories* [24,50,51, 56], are beginning to gain wider acceptance, their use is still limited enough to warrant formal definitions. These are given here, although a discussion of the use of ordered categories in modelling term rewriting in the λ-calculus is postponed until the next chapter.

Definition 1.1.1 *An* ordered category *is a category $\mathcal{C}$ such that for every pair of objects A, B of $\mathcal{C}$, the hom-set $\mathcal{C}(A, B)$ is equipped with a pre-order $\leq$ with respect to which composition is monotonic.*

If t and t' are morphisms with domain A and codomain B and such that $t \leq t'$, then we often write $t \Rightarrow t' : A \rightarrow B$ or use the following diagrammatic presentation:

$$\begin{tikzcd} A \arrow[r, bend left, "t"] \arrow[r, bend right, "t'"'] \arrow[r, phantom, "\Downarrow"] & B \end{tikzcd}$$

The reader may gain insight into these ordered categories by considering their use in modelling term rewriting systems [37,48,58,66,70,75,79]. In such models the objects of the category are contexts, morphisms are (tuples of) terms and the order structure is given by sequences of reductions. Composition in these models is given by substitution and thus the identity morphisms are tuples of variables. Every ordered category has an underlying category obtained by forgetting the order structure. The natural definition of an ordered functor is a functor between the underlying categories which preserves the hom-order. In general, enriched natural transformations are required to satisfy extra properties, but these extra conditions turn out to be vacuous for ordered natural transformations.

Definition 1.1.2 *An* ordered-functor *$\mathcal{F} : \mathcal{C} \rightarrow \mathcal{D}$ is a functor between the underlying categories of $\mathcal{C}$ and $\mathcal{D}$ such that each of the maps $\mathcal{F}_{A,B} : \mathcal{C}(A, B) \rightarrow \mathcal{D}(\mathcal{F}A, \mathcal{F}B)$ is monotonic. An* ordered-natural transformation *between two ordered-functors is just a natural transformation between the underlying functors.*

As adjunctions are defined in terms of a natural isomorphism between certain hom-functors, the notion of ordered adjunctions, and hence ordered limits, are implicit in these definitions. However we shall also give an explicit definition of an ordered-product.

Definition 1.1.3 *An ordered category has* ordered binary products *iff the underlying category has binary products and given morphisms*

$$\begin{tikzcd} Y_1 & X \arrow[l, shift right, "t_1"'] \arrow[l, shift left, "u_1"] \arrow[l, phantom, "\Downarrow"] \arrow[r, shift left, "t_2"] \arrow[r, shift right, "u_2"'] \arrow[r, phantom, "\Downarrow"] & Y_2 \end{tikzcd}$$

the respective mediating morphisms are related in the order, i.e.

$$\begin{tikzcd} X \arrow[r, shift left, "\langle t_1, t_2 \rangle"] \arrow[r, shift right, "\langle u_1, u_2 \rangle"'] \arrow[r, phantom, "\Downarrow"] & Y_1 \times Y_2 \end{tikzcd}$$

Finally, as we shall explain later, ordered monoidal categories form the basis of models of rewriting. The monoidal functor is used to model context concatenation and so the unit will be just the empty context.

Definition 1.1.4 *An* ordered monoidal category *consists of the same data as a monoidal category, except that this data must be order enriched. That is there is an ordered category $\mathcal{C}$, and ordered functor $\circ : \mathcal{C}^2 \to \mathcal{C}$, an object I of $\mathcal{C}$ called the unit and the following ordered natural isomorphisms:*

- *A natural isomorphism $\alpha_{X,Y,Z} : (X \circ Y) \circ Z \to X \circ (Y \circ Z)$*
- *A natural isomorphism $l_X : I \circ X \to X$*
- *A natural isomorphism $r_X : X \circ I \to X$*

These natural isomorphisms are required to satisfy two coherence conditions which may be found in [7,46,50,57,59].

Sequences

Let $\mathcal{N}$ be the set of natural numbers and $\mathcal{N}^*$ be the set of sequences of natural numbers with the empty sequence denoted ϵ, while $u \cdot v$ denotes the concatenation of u with v. If $u \neq \epsilon$, then u^+ is the sequence obtained by omitting the last element of u, while u^- is the sequence obtained by omitting the first element. The *prefix partial ordering* is defined $u \leq v$ iff there is a w such that $v = u \cdot w$ and in such a case define $(u \cdot w)/u = w$. Two sequences u and v are *independent* iff $u \not\leq v$ and $v \not\leq u$. These operations on sequences are extended pointwise to sets of sequences e.g. $X/u = \{w \mid u \cdot w \in X\}$.

Relations

Relations are a convenient framework to define some abstract properties of term rewriting systems and the reader may consult [21,42,43,53]. A *homogeneous relation* on a set X is a subset $R \subseteq X \times X$. If $(x, x') \in R$ then x is called the *redex*, x' the *reduct* and we say that x *rewrites* or *reduces* to x'. This is often written $x \Rightarrow_R x'$ and the subscript R is omitted if the relation is clear from the context. Given a homogeneous relation R on a set X, define the following:

- $R^=$ is the reflexive closure of R.
- R^{-1} is the inverse of R.
- R^+ is the transitive closure of R.
- R^* is the reflexive, transitive closure of R.
- If $x \in X$ then $x/R = \{x' \mid x \Rightarrow_R x'\}$ is the set of one-step R-reducts of x.
- An *R-normal form* is an element x such that $x/R = \emptyset$.
- A relation R is said to *preserve* S-normal forms iff whenever t is an S-normal form and $t \Rightarrow_R t'$, then t' is also an S-normal form.

The *equational theory* generated by a homogeneous relation R is the least equivalence relation containing R and if two elements x and x' are related in this theory we say x and x' are *R-equivalent* and write $x =_R x'$. If R is actually an equivalence relation, then the equivalence class of an element x is denoted $[x]_R$.

The technical results of this thesis proves that the equational theory generated by certain relations is decidable. There are two properties of homogeneous relations which are traditionally used to accomplish this:

- *Strong Normalisation:* An element t is *strongly normalising* iff there are no infinite sequences of reductions

$$t \Rightarrow t_1 \Rightarrow t_2 \ldots \Rightarrow t_n \Rightarrow \ldots$$

  If t is strongly normalising and the relation is finitely branching, which (up to α-equivalence) all the rewrite relations we study are, then there is a least natural number $|t|_R$ such that any reduction sequence starting from t has length at most $|t|_R$. This number is called the *normalisation rank* of t and is often used as the basis of induction arguments. A relation is strongly normalising iff all the elements of the underlying set are strongly normalising.

- *Confluence:* An *R-span* consists of a pair of reductions $t \Rightarrow_R t_1$ and $t \Rightarrow_R t_2$ whose redexes are the same, while an *R-cospan* is simply an R^{-1}-span. A relation satisfies the *diamond property* iff for every span with reducts t and u, there is an co-span with redexes t and u, while a relation is *confluent* iff its reflexive, transitive closure satisfies the diamond property. A relation R is *locally confluent* iff for every R-span with reducts t and u, there is an R^*-co-span with redexes t and u. There are many techniques to prove confluence — in this thesis a relation R will be shown confluent either by showing R is both locally confluent and strongly normalising, or by showing there is a relation S such that $R \subseteq S \subseteq R^*$ and S satisfies the diamond property.

Confluent and strongly normalising relations are important because every element in such a system has a unique normal form and every reduction strategy will find it. In addition those relations we study are *decidable*, i.e. the one-step reducts of an element are always enumerable and thus reduction to normal form provides a decision procedure for the equational theory generated by such a rewrite relation, i.e. if $t =_R t'$ then these terms share the same unique normal form.

A rewrite relation R is *R-weakly normalising* iff for every term t, there is a reduction sequence $t \Rightarrow_R^* t'$ such that t' is an R-normal form. Confluence and weak normalisation are sufficient to ensure that the equational theory generated by a rewrite relation is decidable. However, the absence of a normalisation rank complicates reasoning about such relations and about their interaction with other relations. Thus we shall concentrate on strong normalisation as the key property of interest in our work.

Sometimes strong normalisation cannot be proved because of the existence of certain looping reductions and in these circumstances we use the more categorical notion of a *quasi-normal form*. If R is a homogeneous relation, then an *R-quasi-normal form* is an element t such that if there is an element t' such that $t \Rightarrow_R^* t'$ there is a reduction sequence $t' \Rightarrow_R^* t$. In the second and third of our case studies, the equational theory associated to each reduction relation will be proved decidable by constructing for each term its set of quasi-normal forms and showing that terms equivalent in the equational theory have the same set of quasi-normal forms.

Term Rewriting

Let Λ be the terms of some calculus. These terms are built from *variables* by various *term constructors* and a term belonging to Λ is often denoted $\mathcal{T}(t_0, \ldots, t_n)$ where $\mathcal{T}$ represents the outer term constructor and $t_0, \ldots, t_n$ represent the immediate subterms of the term. If two terms t and u are syntactically equal we write $t \equiv u$. A homogeneous relation on Λ is called a *pre-congruence* iff whenever there is a rewrite $t \Rightarrow u$ and $\mathcal{T}(t_0, \ldots, t, \ldots, t_n)$ is a term, then there is also a rewrite

$$\mathcal{T}(t_0, \ldots, t, \ldots, t_n) \;\Rightarrow\; \mathcal{T}(t_0, \ldots, u, \ldots, t_n)$$

Many of the rewrite relations we study are defined as the least pre-congruence containing a set of basic reductions and in such a case a reduction is called *top-level* iff the reduction belongs to the generating set of reductions.

Occurrences are sequences of natural numbers which form a convenient algebra for dealing with the subterms of a term. If t is a term of some calculus then its set of *occurrences* is denoted $\mathtt{O}(t) \subseteq \mathcal{N}^*$, while the subterm indexed at occurrence σ is denoted t/σ. These are defined as follows:

- If t is a variable, then $\mathtt{O}(t) = \{\epsilon\}$ and $t/\epsilon = t$
- If $t = \mathcal{T}(t_0, \ldots, t_n)$, then $\mathtt{O}(t) = \{\epsilon\} \cup \{i \cdot \sigma \mid i \leq n, \sigma \in \mathtt{O}(t_i)\}$ and

$$t/\epsilon = t \quad \text{and} \quad \mathcal{T}(t_0, \ldots, t_n)/(i \cdot \sigma) = t_i/\sigma$$

When no danger of confusion exists, the distinction between an occurrence and the subterm so indexed may be blurred. Some of the rewrite relations we consider are not left linear in that different occurrences indexing the same subterm in the redex may be mapped to the same occurrence in the reduct. For this reason, define a set X of occurrences to be *consistent* iff given any elements σ, σ' of X then $t/\sigma = t/\sigma'$, and if X is non-empty the subterm so indexed is denoted t/X. A *replacement* of subterms consists of a partial function $\theta : \mathtt{O}(t) \rightharpoonup \Lambda$ and the result of applying a replacement to a term is defined as follows:

$$t\theta = \begin{cases} u & \text{if } \theta\epsilon = u \\ \mathcal{T}(t_0\theta_0, \ldots, t_n\theta_n) & \text{if } \theta\epsilon \text{ is undefined and } t = \mathcal{T}(t_0, \ldots, t_n) \end{cases}$$

where in the last clause $\theta_i(\sigma) = \theta(i.\sigma)$. The term $t\theta$ is also written $t[\sigma \leftarrow \theta(\sigma)]_{\sigma\in\Theta}$ where Θ is the domain of θ. There is a rich algebra relating occurrences to substitution etc. and the reader may wish to consult [42]. As presented occurrences are defined for calculi without features such as variable binding, but the appropriate generalisations are obvious. Finally we define a function $size : \Lambda \to \mathcal{N}$ which maps terms to natural numbers and is often used as an induction rank. The *size* of a term is defined recursively as follows:

$$size(t) = \begin{cases} 1 & \text{if } t \text{ is a variable} \\ 1 + \Sigma_{i=0}^{n} \; size(t_i) & \text{if } t = \mathcal{T}(t_0, \ldots, t_n) \end{cases}$$

Chapter 2

Typed λ-Calculus

This chapter contains a basic introduction to both the *untyped λ-calculus* and the *typed λ-calculus* and demonstrates how categorical models of typed λ-calculus can be used to provide not just an equational theory for typed λ-terms but also rewrite relations which form the basis of a decision procedure for this equality. There are several standard references: for the untyped λ-calculus one may consult [4,38] while [32,55] are good references for typed λ-calculus.

2.1 Untyped λ-Calculi

The untyped λ-calculus was first introduced by Church in the 1930s and has been widely studied by logicians and computer scientists as it is a particularly simple yet elegant formalism capable of representing exactly those functions which are computable by a Turing machine. The untyped λ-calculus is a formal theory for deriving syntactic entities called *terms* which are intended to represent recursive functions. These terms are built up from *variables* by two basic operations, namely *application* and *abstraction*. Application of a term t to a term u, usually written by juxtaposition

$$tu$$

is taken to represent the result of supplying the input u to the function t. There is however no distinction between terms representing functions and terms representing their arguments and so application is a total operation, i.e. any expression (considered as a function) may be applied to every other expression (considered as an argument). The ability to define *fixpoint* operators in the untyped λ-calculus, and hence of representing all recursive functions, relies on this self-applicability of untyped λ-terms.

The second operation is *abstraction*. If $t \equiv t[x]$ is an expression (possibly) containing the variable x, then the term $\lambda x.t[x]$ is called an *abstraction* and is taken to represent the function

$$x \mapsto t[x]$$

The variable x is regarded as a formal parameter to the function $\lambda x.t$, signifying how an argument to this function is to be used, and as such is said to be *bound*; variables which are not bound are called *free*.

When a function defined by an abstraction, say $\lambda x.t$, is applied to an argument, say u, the result may be calculated by *substitution* of the argument for the free occurrences of the variable bound by the λ-abstraction. Thus for example:

$$(\lambda x.x^2 + 1)3 = (x^2 + 1)[3/x] = 3^2 + 1$$

In general, if $t[u/x]$ represents the term obtained by substituting the term u for the free occurrences of x in t, then the equation

$$(\lambda x.t)(u) = t[u/x]$$

is taken to hold. A formal definition of substitution must take care to avoid variable capture and in general implicit changes in bound variables may be required, i.e. we work modulo α-equivalence. Further details may be found in [4].

A natural question is whether the above equation is the only equation that should be imposed on λ-terms? If so why, and if not, then which other equations should be added? These are essentially the questions which we seek to answer, although we pose them in a slightly different setting, namely that of typed λ-calculi.

2.2 Typed λ-Calculi

As mentioned above, the ability to treat expressions simultaneously as data and functions allows the full power of general recursion to be represented in the untyped λ-calculus. While having obvious merits, the resulting problems arising from partiality etc. make the meta-theory more complex than may be necessary. *Typed λ-calculi*, originally introduced by Curry and Church, remove problems like non-termination etc. by assigning syntactic entities called *types* to λ-terms in order to classify their nature and hence limit the expressivity of the language.

Types are built up from *base types*, such as the *natural numbers* and *booleans*, by *type constructors*. For example, if A and B are types, then the exponential type $A \to B$ is understood to have as terms functions which map data of type A to data of type B. Of course there is no absolute distinction between functions and data as functions may be supplied as data to other functions, although see 2.1 for the difference with the untyped λ-calculus.

To assign a type to a term one must first assign types to the free variables occurring in the term, i.e. provide a *context* in which the type assignment occurs. A *context* is a list of pairs of variables and types, written $x_1 : T_1, \ldots, x_n : T_n$ such that the variables are pairwise distinct. The variables in a context Γ are denoted $\mathtt{dom}(\Gamma)$ and two contexts Γ and Δ are *disjoint* iff $\mathtt{dom}(\Gamma) \cap \mathtt{dom}(\Delta) = \emptyset$. The concatenation of contexts Γ and Δ is written Γ, Δ and in order to ensure that the variables of Γ, Δ are pairwise distinct we shall always require Γ and Δ to be disjoint.

Typed λ-calculi consist of inference rules for deriving not just terms, but *judgements* which are of the form

$$\Gamma \vdash t : A$$

and may be read as "the term t has type A in context Γ". We shall use $\Gamma, \Delta, \ldots$ to range over contexts, $A, B, T, U, V \ldots$ to range over types and $t, u, v, \ldots$ to range over terms. By analogy with the untyped λ-calculus, there are two basic methods in the typed λ-calculus of deriving typing judgements, one describing the formation of functions and one their use. The typed version of the abstraction rule is

$$\frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x.t : A \to B}$$

while the typed application rule is

$$\frac{\Gamma \vdash u : A \to B \quad \Delta \vdash v : A}{\Gamma, \Delta \vdash uv : B} \tag{2.1}$$

Notice how the types ensure that an argument may be applied to a function only if the type of the argument is the same as that of the domain of the function. This idea of using the types of various subterms to make sure the term as a whole is well formed forms the basis of *type checking*. For example in our example calculi, one cannot assign a type to expressions of the form tt as the type assigned to the term t would have to be a strict sub-expression of itself. Thus the fixpoint combinators of the untyped calculus are not valid terms of the calculi we study and hence these calculi cannot represent all recursive functions. Unless otherwise stated, henceforth we shall restrict ourselves to the more limited, but better behaved typed λ-calculi.

Identity, Structural and Logical Rules

The traditional presentation of typed λ-calculi as a unified system masks the fact that several separate processes are involved — we follow [32] in identifying three different processes. The first type of inference rules are the *identity rules* and consist of *axiom* and *cut*:

$$\frac{x \in \mathtt{Var}}{x : A \vdash x : A}\ axiom \qquad\qquad \frac{\Gamma \vdash t : A \quad \Delta, x : A \vdash u : B}{\Gamma, \Delta \vdash u[t/x] : B}\ cut \tag{2.2}$$

where `Var` is the set of variables and *cut* requires Γ and Δ to be disjoint to ensure the context in the conclusion is well defined. These inference rules are of fundamental importance, corresponding to the reflexivity and transitivity of logical consequence and the identities and composition of a category, and so will be present in all of the example calculi studied.

The second set of inference rules are called *structural rules* and constrain the calculus by describing how variables and contexts may be manipulated. The importance of these rules is often underplayed but the recent development of linear logic [1, 5,7,31,76] has brought them into clearer focus by considering calculi lacking some of these rules. The structural rules *weakening* and *contraction* are:

$$\frac{\Gamma \vdash t : A}{\Gamma, x : B \vdash t : A}\ \textit{weakening} \qquad \frac{\Gamma, x : B, y : B \vdash u : A}{\Gamma, z : B \vdash u[z/x, z/y] : A}\ \textit{contraction} \tag{2.3}$$

where *weakening* requires the side condition $x \notin \mathtt{dom}(\Gamma)$ and *contraction* requires $z \notin \mathtt{dom}(\Gamma)$ so as to ensure the contexts in the conclusions of these rules are well-defined. These structural rules permit variables to be discarded and to be used on multiple occasions and are, depending on the properties the calculus is expected to have, sometimes omitted. When present these rules considerably complicate the rewriting theory of initial type constructors and so *weakening* and *contraction* are absent in the second of our example calculi. The final structural rule is *exchange*

$$\frac{\Gamma, x : A, y : B, \Delta \vdash t : C}{\Gamma, y : B, x : A, \Delta \vdash t : C}\ \textit{exchange}$$

which gives contexts set-like, as opposed to list-like, properties. Apart from requiring the semantic interpretation of context union to be a symmetric functor, the structural rule exchange has little effect on meta-theory of our case studies and we will tend to ignore *exchange* in future.

The final group of inference rules are the *logical rules* which describe the different mechanisms present in the λ-calculus for constructing compound types. Although there is an enormous variety of such *type constructors*, with exponentiation being one of the simplest, they all conform to the same basic principles. Each type constructor has an associated set of inference rules for deriving associated term judgements. These typically fall into two classes: (i) *introduction rules* which describe the circumstances under which terms of the compound type may be formed and; (ii) *elimination rules* which describe how terms of the compound type may be used to form terms of other types. In the case of exponentiation there is one introduction rule, namely the abstraction rule, and one elimination rule, namely application. In this thesis we shall consider several other type constructors called *product, coproduct, tensor* and two different *units* (one for the product and one for the tensor) and show how rewrite rules for these constructors can be derived in a uniform way.

The *identity* and *structural* inference rules will be collectively called *non-logical* rules. For many applications the subtle effects of the interaction between non-logical and logical inference rules may be overlooked, but, since it is our goal to derive rewrite relations for type constructors which are based solely on their introduction and elimination rules, it is important to remove implicit uses of the non-logical rules from their definition. For example, the application rule presented in 3.1 differs from the more traditional variant in equation 2.1 in not containing implicit uses of the inference rules *contraction* and *cut*. Of course, in the presence of *cut*, the traditional elimination rule for the exponential is a derived inference rule.

Propositions as Types

The interpretation of λ-terms as recursive functions is not the only possibility and several researchers [11,40] have commented upon the apparent similarity between the λ-calculus and formal theories used to formalise the process by which logical propositions are proved. This work is best known as the *Curry-Howard correspondence* or the *Propositions-as-Types* interpretation and demonstrates how a judgement $\Gamma \vdash t : A$ of the typed λ-calculus may be considered as representing a proof t of the proposition A under the assumptions Γ. The type constructors of a λ-calculus correspond to logical connectives, where their associated introduction and elimination rules determine the meaning of these connectives by specifying how compound propositions may be proved from simpler propositions and how these compound propositions may be used to prove other propositions.

This correspondence may be extended to another level if rewriting in the λ-calculus is viewed as part of the process of transforming proofs into canonical forms, e.g. cut-elimination theorems invariably construct a specific reduction sequence to a cut free proof and so correspond to weak normalisation of β-reduction in the λ-calculus. Our search for a decidable equational theory for terms of the λ-calculus may thus be viewed as an attempt to decide when two proofs contain the same "proof-theoretic content" [68,80].

Church versus Curry

There are two different families of systems, one regarding the typed λ-calculus as essentially a subcalculus of the untyped version, while the other views the typed λ-calculus as separated from the untyped version. Curry λ-terms are those of the type free theory and to each term a set (possibly empty, singleton or even infinite) of types is assigned. On the other hand Church λ-terms are terms of the untyped calculus which have been decorated with type information. An illustration of these differences is given by the identity which in a Curry-style definition is an untyped term

$$\vdash_{\text{Curry}} (\lambda x.x) : A \to A$$

which can be assigned the type $A \to A$ for all types A. However the Church identity

$$\vdash_{\text{Church}} (\lambda x : A.x) : A \to A$$

consists of an untyped term decorated with type information which constrains the types which can be assigned to the term — the Church identity can only be assigned the type $A \to A$ for the particular A which occurs within the term. Fortunately there is no need to choose one or the other formulation as most typed λ-calculi, including ours, may be formulated either way and our arguments work in both settings. In particular if $|\_|$ is the function mapping Church λ-terms to Curry λ-terms by simply erasing the type information, then

$$\Gamma \vdash_{\text{Church}} t : A \quad \text{implies} \quad \Gamma \vdash_{\text{Curry}} |t| : A$$

and

$$\Gamma \vdash_{\text{Curry}} t : A \quad \text{implies} \quad \exists t'. \Gamma \vdash_{\text{Church}} t' : A \text{ and } |t'| = t$$

In order to avoid overbearing notation, Curry-style definitions will be given although, as stated before, our results are applicable to calculi formulated in either approach.

2.3 Semantics of Typed λ-Calculus

Apart from directly reasoning about the terms of the λ-calculus, one can construct models of the λ-calculus in other mathematical structures and use results about these models to make inferences about the λ-calculus. For example, once an appropriate notion of model has been chosen, there are two properties which an equality $=_\lambda$ on terms is expected to satisfy and which uniquely characterise such an equality:

- *Soundness*: Given any model M, if $t =_\lambda t'$ then $M[t] =_M M[t']$
- *Completeness*: If for all models M, $M[t] =_M M[t']$ then $t =_\lambda t'$

where $M[t]$ is the interpretation of the term t in the model M and $=_M$ is the equality in the model. Given an appropriate notion of model, soundness is usually easy to prove while completeness is much harder and usually relies on the construction of particular models from the syntax of the language.

There are of course a wide variety of different mathematical structures within which models of the typed λ-calculus may be constructed. Categorical models [7, 55,73] form one of the most satisfactory semantic approaches to the λ-calculus for several reasons: (i) the link between the syntactic rules and the categorical structure used to interpret them is particularly clear; (ii) there are a large number of "naturally" occurring categorical models of considerable practical interest which have been studied extensively and, (iii) categorical models may be generalised to provide semantics for more complex calculi [3,44,64,74] while this is not always the case for other approaches [69].

Typically these categorical models consist of a category $\mathcal{C}$ together with a map interpreting types as objects of $\mathcal{C}$ and term judgements as morphisms of $\mathcal{C}$. The various type constructors and their associated introduction and elimination rules are reflected by certain extra properties which the category is expected to satisfy: cartesian closure in the case of the simply typed λ-calculus, symmetric monoidal closure in the case of the linear λ-calculus, etc. Once type constructors and their associated introduction and elimination rules have been interpreted via categorical concepts, the natural categorical equations inherent in these structures become equations between λ-terms. For example, if $\mathcal{C}(\Gamma; A)$ is the hom-set used to interpret terms which can be assigned the type A in context Γ, then the introduction and elimination rules of the exponential are interpreted as being natural isomorphisms:

$$\mathcal{C}(\Gamma, x\!:\!A; C) \underset{(\_)(x)}{\overset{\lambda x.\_}{\underset{\longleftarrow}{\overset{\longrightarrow}{\cong}}}} \mathcal{C}(\Gamma; A\!\to\!C)$$

Requiring the functions $t \mapsto \lambda x.t$ and $u \mapsto ux$ to be mutually inverse isomorphisms corresponds to imposing the following pair of equations:

$$(\lambda x.t)x = t \text{ and } \lambda x.tx = t$$

In addition, if $t \in \mathcal{C}(\Gamma; A\!\to\!C)$ then $x \notin \mathtt{FV}(t)$ as otherwise $\Gamma, x : A$ would not be a context. Thus we can derive from the categorical analysis not only the usual β- and η- equations for the exponential, but also the traditional side-condition for η-equality, $x \notin \mathtt{FV}(t)$.

This categorical approach of modelling the λ-calculus in terms of categories with extra structure is so strong that it is possible to go in the reverse direction, i.e. to deduce λ-calculi as the *internal language* of certain structured categories. Although a formal mathematical explanation is beyond the scope of this work, an illustrative outline of the main ideas can be gained through an analogy with universal algebra. Universal algebra concerns itself with the study of sets equipped with certain operations which satisfy certain universal equations, e.g. monoids, groups etc. In general such structured sets can be described as being the *algebras* of a certain *monad* on the category **Sets** of sets [6,59,60]. Similarly, once the notions of operation and equation have been suitably generalised, one can consider categories equipped with equational structure as also being the algebras of a monad, although now the base category is the category **Cat** of (small) categories [8,67]. The categories **CCC**, **SMC** and **ABCC** used in the semantics of the three case studies are all examples of categories with equational structure.

One important consequence of these remarks is the existence of free algebras, that is, the forgetful functors

$$U_0 : \mathbf{CCC} \to \mathbf{Cat} \quad U_1 : \mathbf{SMC} \to \mathbf{Cat} \quad U_2 : \mathbf{ABCC} \to \mathbf{Cat}$$

have left adjoints. In the same way that free monoids/groups are constructed by writing down the appropriate syntax and then quotienting by the equations of these structures, so free cartesian closed categories, symmetric monoidal closed categories etc. are also constructed from syntax quotiented by the relevant equations, and this syntax is nothing more than a λ-calculus. For example, the free cartesian closed category can be described as having objects the singleton contexts of the simply typed λ-calculus with each hom-set given by $\beta\eta$-equivalence classes of terms:

$$\mathcal{C}(x : X, y : Y) = \{\ [t]_{\beta\eta}\ \mid\ x : X \vdash t : Y\}$$

The identity morphisms are generated by variables while composition is given by substitution. In this categorical setting a model in a category $\mathcal{C}'$ is a *cartesian closed functor* from this free category to $\mathcal{C}'$, which of course can be shown to coincide with the traditional definition. There are alternative constructions of free categories where objects are either types or contexts and morphisms are appropriately typed terms or tuples of terms, but the resulting categories are isomorphic and so categorically indistinguishable. The construction of these free categories also gives completeness of $\beta\eta$-equality for models in all cartesian closed categories because, if the interpretations of two terms are equal in all models, then their interpretations in the free model are equal and thus the terms are $\beta\eta$-equal.

A different problem is to seek sound and complete equational theories for more restricted classes of models. Friedman [25] proved that $\beta\eta$-equality is complete for all models in the category **Sets**. In [77] simple conditions on an arbitrary cartesian closed category $\mathcal{C}$ are given to ensure $\beta\eta$-equality is sound and complete for all models in $\mathcal{C}$, while in [23] a proof system is given for deriving a sound and complete theory for all "set-theoretic" models of the simply typed λ-calculus extended with coproducts. This later theory is equivalent to our axiomatisation of the equational theory for coproducts and therefore also decidable.

So category theory can help decide which equality is appropriate for a typed λ-

calculus, but can category theory also provide rewrite relations to aid in the construction of a decision procedure for this equality? The answer to this is again positive. Just as equations were derived by considering natural categorical constructions on categorical models, once the notion of a model has been generalised to include rewrites between terms, the natural categorical structure on these generalised models will implicitly contain rewrite relations.

2.4 Categorical Models of Rewriting

Soundness of the traditional categorical models outlined above means that $\beta\eta$-equal terms have the same interpretation in such models and so these structures are too limited to provide a basis for the study of rewriting in the λ-calculus. Instead the notion of model must be generalised from categories based on term equality to categories based on term reduction.

The last decade has seen the development of categorical models of rewriting which add a pre-order/categorical structure to each of the hom-sets to reflect reduction [58,48,66,70,75,72]. Diagrammatically the reduction of a term t to t' may be represented:

$$\begin{tikzcd} X \arrow[r, bend left, "t"{name=U}] \arrow[r, bend right, "t'"'{name=D}] & Y \arrow[Rightarrow, from=U, to=D, shorten=2pt] \end{tikzcd}$$

Categories with extra structure on their hom-sets are known as *enriched categories* [24,50,51] and have been studied for almost 30 years now with several possibilities for generalising standard categorical concepts such as functor, adjoint and algebra to enriched categories having been proposed [33,39,47,45,50,51,75,72,78]. The basic structures used in this thesis to model term rewriting in the λ-calculus have been defined in the introduction but, for continuity, we repeat the definition of an ordered category.

Definition 2.4.1 *An ordered category is a category $\mathcal{C}$ such that for every pair of objects A, B of $\mathcal{C}$, the hom-set $\mathcal{C}(A, B)$ is equipped with a pre-order and composition is monotonic with respect to this order.*

Ordered categories may be constructed from term rewriting systems by taking as objects contexts and as morphisms

$$x_1\!:\!X_1,\ldots,x_n\!:\!X_n \longrightarrow y_1\!:\!Y_1,\ldots,y_m\!:\!Y_m$$

tuples of terms/substitutions $(e_1,\ldots,e_m)$ such that $x_1\!:\!X_1,\ldots,x_n\!:\!X_n \vdash e_i : Y_i$ for $i = 1,\ldots,m$. The order structure is given by

$$(e_1,\ldots,e_m) \leq (f_1,\ldots,f_m) \quad \text{iff} \quad \forall i.\ e_i \Rightarrow f_i$$

and composition is given by substitution while the identities are just tuples of variables.

Lemma 2.4.2 *The construction above defines an ordered category.*

Proof The proof is fairly straightforward and the basic ideas can be found in any of [37,48,70,75,72]. □

Notice that, because the order on the hom-sets is taken to be a pre-order, categorical models of reduction do not model the one-step reduction relation, but rather its reflexive transitive closure. It is surprising how many of the definitions of enriched category theory, although originally devised for reasons unconnected with term rewriting, are nevertheless extremely natural from this point of view. For instance, requiring composition to be monotonic means that if there are diagrams/rewrites of the form

$$\begin{tikzcd} w\!:\!W \arrow[r, "u"] & x\!:\!X \arrow[r, shift left, "t"] \arrow[r, shift right, "t'"'] \arrow[r, phantom, "\Downarrow"] & y\!:\!Y \end{tikzcd} \qquad \text{and} \qquad \begin{tikzcd} x\!:\!X \arrow[r, shift left, "t"] \arrow[r, shift right, "t'"'] \arrow[r, phantom, "\Downarrow"] & y\!:\!Y \arrow[r, "v"] & z\!:\!Z \end{tikzcd}$$

then, because composition is given by substitution, there must be diagrams, or equivalently, rewrites

$$\begin{tikzcd} w\!:\!W \arrow[r, shift left, "{t[u/x]}"] \arrow[r, shift right, "{t'[u/x]}"'] \arrow[r, phantom, "\Downarrow"] & y\!:\!Y \end{tikzcd} \qquad \text{and} \qquad \begin{tikzcd} x\!:\!X \arrow[r, shift left, "{v[t/y]}"] \arrow[r, shift right, "{v[t'/y]}"'] \arrow[r, phantom, "\Downarrow"] & z\!:\!Z \end{tikzcd}$$

i.e. reduction must be *stable* and *compatible* in the sense of [42]. Note that in this thesis we reserve the word *stable* for a different property.

An alternative approach is to model rewrite relations in 2-categories [51], i.e.

categories where each hom-set is itself equipped with categorical structure. The morphisms in these hom-categories are called *2-cells* and composition is required to be *functorial* on these 2-cells and satisfy certain extra axioms/equations which assert the equivalence of certain reduction sequences. Another variant are the *sesqui categories* [79] which are similar to 2-categories except that the "interchange law" [58,51,79] is not required to hold. The interchange law states that if there are reductions $t \Rightarrow t'$ and $u \Rightarrow u'$ then the two different reduction sequences from the term $F(t, u)$ to the term $F(t', u')$ are equal. Thus 2-categories may be thought of as models of parallel rewriting, while sesqui-categories may be thought of as modelling non-parallel rewriting.

Of all the equations required between reduction sequences in the construction of 2-categories, the interchange law is the only one which identifies reduction sequences of different length. Thus every 2-cell in a sesqui-categorical model of a term rewriting system has a *length* [63]. In other words, for every two cell f there is a natural number n such that f may be written as the composite of n non-identity morphisms, but not as the composite of more than n. See [79] for some ramifications of this.

The choice between ordered categories and 2-categories/sesqui-categories is of course determined by the issues to be resolved. The algebra of rewrites implicit in the 2-categorical approach, and the possibility of several different rewrites existing between the same terms, makes these structures more suitable for resolving questions regarding different reduction strategies or for determining whether a certain redex must be contracted to achieve a normal form etc. However the main focus of this thesis is the use of categorical constructions to formally derive rewrite relations and so the conceptually simpler ordered categories are preferred. Having said this, our arguments can of course be generalised to the 2-categorical setting.

Another possibility for modelling term rewriting systems is to generalise the traditional approach of functorial semantics [6,34,56] by considering a model of a term rewriting system as being a structure preserving functor from a free theory to a category with appropriate structure. This approach, although well suited to proving results about the category of models of a rewriting system, does not provide any intuition as to the actual choice of the free theory, in this context the rewrite relation, and so does not form the basis of our notion of model.

Just as the natural constructions on traditional models may be used to derive an equational theory for the terms of the λ-calculus, so the natural categorical constructions on these ordered categories allow us to formally derive rewrite relations whose equational theory matches that suggested by the traditional semantics. However before explaining what these constructions are, we must endow ordered categories with more structure to provide adequate models of rewriting. This extra structure is divided into two parts: (i) structure to model the algebra of contexts present in the non-logical inference rules; and (ii) structure to model the particular type constructors present in the λ-calculus. Context concatenation provides ordered categories which model term rewriting systems with a natural bifunctor and this bifunctor has a unit given by the empty context. This is summarised by saying a model of a term rewriting system must be a *strict ordered monoidal category* [7,46,50,57,59]. Finally, we cannot deal solely with contexts as type constructors are to build complex types from other types, and not from arbitrary contexts. However types can be thought of as singleton contexts and so are represented in a model by a subcategory which appropriately generates the rest of the category. In summary a model of rewriting is defined as follows:

Definition 2.4.3 *A rewrite category is a pair consisting of*

(i) An ordered monoidal category of contexts, tuples of terms and rewrites denoted $(\mathcal{C}, \circ, I)$.

(ii) A full subcategory $\mathcal{T}$ *of* $\mathcal{C}$*, called the* base *and with inclusion* $\Omega : \mathcal{T} \to \mathcal{C}$*, such that every object in* $\mathcal{C}$ *can be expressed as a finite multiplication of objects from the base.*

This definition is not ideal, e.g. one may take $\mathcal{T}$ to be the whole of $\mathcal{C}$, but it is sufficient for our purposes, namely showing how canonical constructions on these rewrite categories may be used to derive rewrite rules for a wide variety of type constructors.

Context concatenation is taken to be a monoidal functor rather than the more traditional strict product so that our framework is capable of dealing with theories such as the linear λ-calculus where the absence of certain structural rules means projections etc. cannot be defined. If however some of the structural rules are present then this is reflected in the models by the presence of certain natural transformations.

- Exchange corresponds to a symmetry on the monoidal functor, i.e. isomorphisms $s_{X,Y} : X \circ Y \to Y \circ X$ which are natural in X and Y.

- Weakening and contraction by X, an object of $\mathcal{T}$, are natural transformations

$$\mathbf{weak}^X : \_ \circ X \Rightarrow I \quad \text{and} \quad \mathbf{cont}^X : \_ \circ X \Rightarrow \_ \circ X \circ X.$$

It is conceivable, although no formal research has been conducted, that by replacing these transformations by lax-transformations, one may be able to reason about certain term rewriting problems caused by the non-linear use of variables.

In typed λ-calculi, the logical rules describe how terms of a compound type may be formed and how they may used to construct other terms. The semantic counterpart to this was described by Hagino [36] (and see [65]) using the concept of a *dialgebra.*

Definition 2.4.4 *Let $F, G : \mathcal{C} \to \mathcal{D}$ be functors. The category of F, G-dialgebras has as objects pairs (X, h) where*

$$FX \xrightarrow{h} GX$$

is a $\mathcal{D}$-morphism. A morphism $\theta : (X, h) \to (X', h')$ is an arrow $\theta : X \to X'$ such that

$$\begin{tikzcd} FX \arrow[r, "h"] \arrow[d, "F\theta"'] & GX \arrow[d, "G\theta"] \\ FX' \arrow[r, "h'"'] & GX' \end{tikzcd}$$

Those type constructors we call *final* are characterised in Hagino's calculus as being final dialgebras of certain functors, while our *initial* type constructors are characterised as being initial dialgebras. We shall incorporate rewrite rules directly into the actual definition of type constructors by generalising Hagino's calculus to ordered categories, replacing initial dialgebra by *locally initial lax-dialgebra* and final dialgebra by *locally terminal oplax dialgebra.* Although these definitions may seem daunting at first, the ease with which these categorical structures can be used to derive rewrite rules is quite remarkable.

Definition 2.4.5 *An object T of an ordered category $\mathcal{C}$ is locally terminal iff for every object X of $\mathcal{C}$, there is an arrow $!_X : X \to T$ which is terminal in its hom-order*

$$\begin{tikzcd} X \arrow[r, shift left=2, "e"] \arrow[r, shift right=2, "!_X"'] \arrow[r, phantom, "\Downarrow"] & T \end{tikzcd}$$

An object T is locally initial iff it is locally terminal in $\mathcal{C}^{op}$, i.e. I is locally initial iff for every object X of $\mathcal{C}$, there is an arrow $\epsilon_X : I \to X$ which is terminal in its hom-order

$$\begin{tikzcd} I \arrow[r, shift left=2, "e"] \arrow[r, shift right=2, "\epsilon_X"'] \arrow[r, phantom, "\Downarrow"] & X \end{tikzcd}$$

Locally initial (and terminal) objects X and Y are not defined up to isomorphism; rather there exist morphisms $f : X \to Y$ and $g : Y \to X$ such that gf and 1_X are related in the hom-order and similarly for fg and 1_Y. Thus when the hom-order is collapsed, i.e. the free category constructed, locally initial/terminal objects are mapped to isomorphic objects. Dialgebras can be generalised to ordered categories in several different ways; those of interest to us are the following:

Definition 2.4.6 *Let $F, G : \mathcal{C} \to \mathcal{D}$ be ordered functors. The ordered category of F, G-lax dialgebras, written $\mathcal{LD}(F, G)$, has as objects pairs (X, h) where*

$$FX \xrightarrow{h} GX$$

is a D-morphism. A morphism $\theta:(X,h)\to(X',h')$ *is an arrow* $\theta:X\to X'$ *such that*

$$\begin{tikzcd} FX \arrow[r, "h"] \arrow[d, "F\theta"'] & GX \arrow[d, "G\theta"] \\ FX' \arrow[r, "h'"'] \arrow[ur, phantom, "\Leftarrow"] & GX' \end{tikzcd}$$

and the order structure in $\mathcal{LD}(F,G)$ *is inherited from* $\mathcal{C}$*. The ordered category of oplax dialgebras, written* $\mathcal{OLD}(F,G)$*, is defined as for* $\mathcal{LD}(F,G)$ *except the 2-cell is in the other direction.*

The rest of this chapter will show how rewrite relations for a wide variety of type constructors may be formally derived from certain constructions on rewrite categories. This will involve dialgebras of functors $F,G:\mathcal{T}\to\mathcal{C}^n$ constructed from:

- The identity functor.
- The inclusion functor $\Omega:\mathcal{T}\to\mathcal{C}$.
- The constantly valued functors K_X.
- The monoidal functor $\circ$.

Example 1: The Unit

Rewrite rules for the unit of the simply typed λ-calculus may be derived by considering the locally terminal object of the category $\mathcal{OLD}(K_*,K_*)$ of oplax dialgebras, where $K_*:\mathcal{T}\to 1$ is the unique functor from $\mathcal{T}$ to the one object, one morphism, one rewrite ordered category. Such a dialgebra is exactly an object of $\mathcal{T}$, while a morphism between such dialgebras is just a morphism between the underlying objects. Thus a locally terminal oplax dialgebra is nothing more than a locally terminal object of $\mathcal{T}$.

Example 2: Products

Rewrite rules for the product may be derived by taking the product of two objects A and B in a rewrite category $\mathcal{C}$ to be a locally terminal object in the category $\mathcal{OLD}(\Omega^2, K_{A,B})$ of oplax dialgebras constructed from the functors $\Omega^2 : \mathcal{T} \to \mathcal{C}^2$ and $K_{A,B} : \mathcal{T} \to \mathcal{C}^2$. The dialgebra is a morphism in $\mathcal{C}^2$, i.e. a pair of morphisms in $\mathcal{C}$, which will be taken to be the projections:

$$(A \times B, A \times B) \xrightarrow{(\pi_0, \pi_1)} (A, B)$$

By terminality there must be a morphism between any other dialgebra $(t_0, t_1) : (Y, Y) \to (A, B)$ and the dialgebra (π_0, π_1). The underlying term of this mediating morphism will of course be the introduction term $\langle t_0, t_1 \rangle$, while the 2-cell will be the β-reduction rule

$$\begin{tikzcd}
\Omega(Y) \arrow[r, "{(t_0, t_1)}"] \arrow[d, "\Omega\langle t_0, t_1\rangle"'] \arrow[dr, phantom, "\Rightarrow"] & (A, B) \arrow[d, "id"] \\
\Omega(A \times B) \arrow[r, "{(\pi_0, \pi_1)}"'] & (A, B)
\end{tikzcd}$$

Finally because of local terminality, any other morphism between these dialgebras must reduce to the introduction term, which corresponds to the following conditional rewrite rule:

$$\frac{\pi_0 t \Rightarrow t_0 \quad \pi_1 t \Rightarrow t_1}{t \Rightarrow \langle t_0, t_1 \rangle} \tag{2.4}$$

Notice that this conditional η-rewrite rule is actually equivalent (in the presence of identity reductions) to the traditional, unconditional rewrite rule $t \Rightarrow \langle \pi_0 t, \pi_1 t \rangle$. Firstly the premises of equation 2.4 may be instantiated by identity reductions to derive the unconditional rewrite rule, while given the premises to equation 2.4

$$t \Rightarrow \langle \pi_0 t, \pi_1 t \rangle \Rightarrow \langle t_0, t_1 \rangle$$

Of course this is nothing more than expressing the natural isomorphism at the heart of an adjunction in terms of the associated unit and counit.

Example 3: The Tensor

The tensor found in linear logic is essentially the product with its projections and diagonal removed. The introduction and elimination rules of the tensor are:

$$\frac{\Gamma, z\!:\!X \otimes Y \vdash t\!:\, Z}{\Gamma, x\!:\!X, y\!:\!Y \vdash t[x \otimes y/z]\!:\!Z}\otimes int \qquad \frac{\Gamma, x\!:\!X, y\!:\!Y \vdash t\!:\!Z}{\Gamma, z\!:\!X \otimes Y \vdash \textbf{let}\, z\, \textbf{be}\, x \otimes y\, \textbf{in}\, t : Z}\otimes elim$$

where the elimination rule requires $z \notin \mathtt{dom}(\Gamma)$ and the introduction rules requires $x, y \notin \mathtt{dom}(\Gamma)$ so as to ensure the contexts in the conclusions are well defined.

Rewrite rules for the tensor of objects X and Y may be derived by considering the locally initial object in the category $\mathcal{LD}(K_{X,Y}, \Omega)$ of dialgebras constructed from the functors $K_{X,Y}, \Omega : \mathcal{T} \to \mathcal{C}$. Such a locally initial dialgebra will consist of a morphism

$$x\!:\!X, y\!:\!Y \xrightarrow{x \otimes y} z\!:\!X \otimes Y$$

which gives the introduction term, while the elimination term and β-reduction is induced by the mediating morphism to any other dialgebra:

$$\begin{tikzcd}
x\!:\!X, y\!:\!Y \arrow[r, "x \otimes y"] \arrow[d, "id"'] \arrow[dr, phantom, "\Leftarrow"] & z\!:\!X \otimes Y \arrow[d, "\textbf{let}\, z\, \textbf{be}\, x \otimes y\, \textbf{in}\, e"] \\
x\!:\!X, y\!:\!Y \arrow[r, "e"'] & w\!:\!C
\end{tikzcd}$$

i.e.

$$\textbf{let}\, x \otimes y\, \textbf{be}\, x \otimes y\, \textbf{in}\, e \quad \Rightarrow \quad e$$

Finally, as this mediating morphism is to be terminal amongst such morphisms, the η-rewrite rule for the tensor is the conditional rule

$$\frac{e'[x \otimes y/z] \Rightarrow e}{e' \Rightarrow \textbf{let}\, z\, \textbf{be}\, x \otimes y\, \textbf{in}\, e}$$

which, as with the product, has an equivalent unconditional formulation

$$e' \Rightarrow \textbf{let}\, z\, \textbf{be}\, x \otimes y\, \textbf{in}\, e'[x \otimes y/z]$$

Example 4: Natural Numbers

There seems to be little work in the literature on extensionality principles for recursive types such as the natural numbers (although see [71]). However Hagino's calculus is powerful enough to deal with not just the adjoint constructors already mentioned, but also recursive datatypes such as natural numbers and lists. Thus our ordered categorical version of Hagino's calculus may be used to derive interesting rewrite rules for such type constructors.

A natural numbers type, denoted N, has the introduction rules

$$\frac{}{\vdash 0\!:\!N} \qquad \frac{\Gamma \vdash t\!:\!N}{\Gamma \vdash st\!:\!N}$$

and elimination rule

$$\frac{\Gamma \vdash u\!:\!X \quad \Gamma, x\!:\!X \vdash v\!:\!X}{\Gamma, n\!:\!N \vdash \mathtt{It}(u, x.v, n)\!:\!X}$$

where the x in the iterator is a variable binder. Such a natural numbers type can be modelled in a rewrite category $\mathcal{C}$ as a locally initial lax dialgebra in the category $\mathcal{LD}((K_I, \Omega), (\Omega, \Omega))$, where I is the unit of the monoidal structure on $\mathcal{C}$.

$$\begin{tikzcd}
(K_I, N) \arrow[r, "{0, sn}"] \arrow[d, "{id, \mathtt{It}(u,x.v,n)}"'] \arrow[dr, phantom, "\Leftarrow"] & (N, N) \arrow[d, "{\Omega^2\ \mathtt{It}(u,x.v,n)}"] \\
(K_I, X) \arrow[r, "{u, v}"'] & (X, X)
\end{tikzcd}$$

The constants 0 and the successor operation are part of the dialgebra structure, while the iterator and two β-reduction rules are derived from the definition of a mediating morphism.

$$(\beta_{N,1}) \qquad \mathtt{It}(u, x.v, 0) \Rightarrow u$$

$$(\beta_{N,2}) \qquad \mathtt{It}(u, x.v, sn) \Rightarrow v[\mathtt{It}(u, x.v, n)/x]$$

Finally, local initiality translates into the following conditional η_N-rewrite rule for natural numbers:

$$\frac{h[0/n] \Rightarrow u \quad h[sn/n] \Rightarrow v[h/x]}{h \Rightarrow \mathtt{It}(u, x.v, n)}\ \eta_N$$

Although rewrite relations for recursive datatypes, such as that for the natural numbers derived above, have not been formally studied, a few preliminary remarks can be made. Firstly, unlike for adjoint type constructors such as the product and tensor, it is unclear whether the proposed equational theory for natural numbers is sufficient to define a *strong* natural numbers object or merely a *weak* natural numbers object [55], i.e. whether when the order relation is quotiented out, the mediating morphism represented by the iterator is unique or not. Another difference with the rewrite relations for adjoint types is the lack of an unconditional formulation of the proposed η_N-rewrite rule. Thirdly because all primitive recursive functions may be represented in a cartesian closed category with a natural numbers object, the goal of proving the associated equational theory decidable will be impossible if the equational theory matches the extensional equality of functions. These issues, although beyond the scope of this work, make recursive datatypes an intriguing area for further research.

The rest of this thesis is devoted to three case studies which develop the term rewriting techniques required to turn these categorical rewrite relations into decision procedures for the associated equational theory. These calculi contain no recursive type constructors and so we shall present alternative derivations of the rewrite rules for each type constructor in terms of the adjunctions formed by the associated introduction and elimination rules.

Chapter 3

The Simply Typed λ-Calculus

For reasons that will become clear in Chapters 4 and 5, the η-rewrite rules for final type constructors are significantly simpler than those for initial type constructors and so the first of our case studies is a calculus which contains only final type constructors. The simply typed λ-calculus contains three type constructors called the *unit*, *product* and *exponential* and as explained in the introduction, the presence of the unit type means that if η-conversion is interpreted as a contraction then confluence is lost and so an alternative approach to the decidability of $\beta\eta$-equality is required.

After defining the calculus we show how the introduction and elimination rules of each type constructor form an adjoint pair with the associated unit and counit forming an expansionary η-rewrite rule and a contractive β-rewrite rule. The rewrite relation defined by these β- and η-rewrite rules, although generating a sound and complete equational theory for models in cartesian closed categories, contains infinite sequences of rewrites and so cannot itself be used to decide this equational theory. Converting this rewrite relation into a decision procedure is the focus for the rest of this chapter and, as this calculus contains no initial type constructors, the decision procedure derived will be simpler than those for the other two case studies.

We define a subrelation by prohibiting those expansions which occur in the triangle laws inherent in the adjunctions defining each β- and η-rewrite rule. This subrelation generates the same equational theory as the unrestricted rewrite relation and is proved strongly normalising and confluent by adapting the traditional proofs to cope with the presence of expansionary rewrites. Thus $\beta\eta$-equality may be decided by reduction to normal form in the restricted rewrite relation, although the price of this decidability result is that the restricted rewrite relation is not a pre-congruence. We also prove that the normal forms of the restricted fragment coincide with Huet's *long $\beta\eta$ normal forms* [41], give a structural characterisation of these forms and show they may be calculated by first performing all β-reductions and then performing any remaining η-expansions or vice versa.

Although the calculus has been presented without a natural numbers type, the proof methods employed in this chapter are sufficiently robust to prove equivalent results for the calculus extended with such a natural numbers type and with associated iterator, successor, zero and β-reduction rules [49]. However, as mentioned at the end of the last chapter, the effect of the η-rewrite rule for the natural numbers remains a subject for further research.

3.1 The Calculus $\Lambda^{1,\times,\rightarrow}$

The *simply typed λ-calculus* over a set of base types $\mathcal{B}$ is denoted $\Lambda^{1,\times,\rightarrow}$ and consists of *types* freely generated from the base types by a nullary type constructor 1, called the *unit*, and two binary constructors $\times$ and $\rightarrow$ called the *product* and *exponential*

$$T := T \times T \mid T \rightarrow T \mid 1 \mid B$$

where $B \in \mathcal{B}$ is any base type. The *atomic* types are 1 and the base types. For each type T there is a set of constants $\mathtt{Con}(T)$, including the distinguished constant $* \in \mathtt{Con}(1)$, such that if $T \neq T'$, then $\mathtt{Con}(T) \cap \mathtt{Con}(T') = \emptyset$. There is also a set of variables $\mathtt{Var}$ which is disjoint from the constants. The *pre-terms* of the simply typed λ-calculus are:

$$t \coloneqq x \mid c \mid \langle t, t \rangle \mid \pi_0 t \mid \pi_1 t \mid tt \mid \lambda x.t$$

where x is a variable and c a constant. The free variables of a pre-term t are denoted $\mathtt{FV}(t)$ and the substitution of pre-terms for free variables is defined as expected.

The *term judgements* of the simply typed λ-calculus are of the form $\Gamma \vdash t\!:\!T$ where Γ is a context, t a pre-term and T a type, and are generated by the inference rules in Table 3-1. The side-conditions on the identity and structural rules may be found in equations 2.2 and 2.3. Note that the traditional elimination rule for the exponential can be derived from the inference rules presented in Table 3-1. Given

Table 3–1: Typing Rules for the Simply Typed λ-calculus

Structural and Identity Rules

$$\frac{x \in \mathtt{Var}}{x\!:\!A \vdash x\!:\!A}\ \textit{axiom} \qquad \frac{c \in Con(A)}{\vdash c\!:\!A}\ \textit{cons}$$

$$\frac{\Gamma \vdash t\!:\!A}{\Gamma, x\!:\!B \vdash t\!:\!A}\ \textit{weakening} \qquad \frac{\Gamma, x\!:\!B, y\!:\!B \vdash u\!:\!A}{\Gamma, z\!:\!B \vdash u[z/x, z/y]\!:\!A}\ \textit{contraction}$$

$$\frac{\Gamma, x\!:\!A, y\!:\!B, \Delta \vdash t\!:\!C}{\Gamma, y\!:\!B, x\!:\!A, \Delta \vdash t\!:\!C}\ \textit{exchange} \qquad \frac{\Gamma \vdash t\!:\!A \quad \Delta, x\!:\!A \vdash u\!:\!B}{\Gamma, \Delta \vdash u[t/x]\!:\!B}\ \textit{cut}$$

Logical Rules

$$\frac{\Gamma \vdash e\!:\!A \quad \Gamma \vdash e'\!:\!B}{\Gamma \vdash \langle e, e' \rangle\!:\!A \times B}\ \times\textit{int} \qquad \frac{\Gamma \vdash t\!:\!A_0 \times A_1}{\Gamma \vdash \pi_i t\!:\!A_i}\ \times\textit{elim}$$

$$\frac{\Gamma, x\!:\!A \vdash e\!:\!B}{\Gamma \vdash \lambda x.e\!:\!A \to B}\ \to\!\textit{int} \qquad \frac{\Gamma \vdash e\!:\!A \to B \quad x \notin \mathtt{dom}(\Gamma)}{\Gamma, x\!:\!A \vdash ex\!:\!B}\ \to\!\textit{elim}$$

any term judgement $\Gamma \vdash t\!:\!T$, we say t is a term of type T and when Γ is clear, or not important, this is written $t\!:\!T$. A term is called an *introduction* term if it is a λ-abstraction, a pair, or the constant $*$, while a term which is not an introduction term is called an *elimination* term. Finally, a subterm of a term occurs *negatively* iff it is either applied or projected.

3.2 Rewriting in $\Lambda^{1,\times,\rightarrow}$

A rewrite relation for terms of the simply typed λ-calculus is derived by taking the introduction and elimination rule for each connective as forming an adjoint pair; the associated unit then forms an expansionary η-rewrite rule while the counit forms a contractive β-rewrite rule.

For example, consider the product. If $\mathcal{C}(\Gamma; X)$ is the category whose objects are term judgements $\Gamma \vdash e : X$ and whose morphisms between $\Gamma \vdash e : X$ and $\Gamma \vdash e' : X$ are (quotiented, labelled) rewrites $r : e \Rightarrow e'$, then the introduction and elimination rules of the product may be regarded as functors between the categories shown below. As these functors are taken to constitute an adjoint pair

$$\begin{tikzcd} \mathcal{C}(\Gamma; A) \times \mathcal{C}(\Gamma; B) \arrow[r, shift left, "\langle \_, \_ \rangle"] \arrow[r, phantom, "\top"] & \mathcal{C}(\Gamma; A \times B) \arrow[l, shift left, "(\pi_0 \_, \pi_1 \_)"] \end{tikzcd}$$

rewrite rules can be obtained as counit and unit.

$$\begin{array}{llcll} (\beta_{\times,0}) & \pi_0\langle a, b\rangle & \Rightarrow & a & \\ (\beta_{\times,1}) & \pi_1\langle a, b\rangle & \Rightarrow & b & \quad (3.1) \\ (\eta_\times) & c & \Rightarrow & \langle \pi_0 c, \pi_1 c\rangle & \end{array}$$

As with adjunctions in general, there is an equivalent formulation of these rules in terms of representability/universal properties. In the product example the term $\langle e, e' \rangle$ satisfies the β-reduction shown in the diagram

$$\begin{tikzcd} & \Gamma \arrow[dl, "e"'] \arrow[d, "\langle e, e' \rangle"] \arrow[dr, "e'"] & \\ A & A \times B \arrow[l, "\pi_0"] \arrow[r, "\pi_1"'] & B \end{tikzcd}$$

and is in addition terminal amongst such terms. Thus there is a natural isomorphism of rewrites

$$t \Rightarrow \langle e, e' \rangle \text{ iff } \pi_0 t \Rightarrow e \text{ and } \pi_1 t \Rightarrow e'$$

which is of course equivalent (in the presence of identity rewrites) to the rewrites given in equation 3.1. Thus introduction terms may be regarded as being "defined up to isomorphism", i.e. their purpose is to represent certain possibilities for rewriting and this underlies their central role in the definition of $\beta\eta$-normal forms.

A similar approach is possible for the exponential type constructor where the introduction and elimination rules are again deemed to form an adjoint pair

$$\mathcal{C}(\Gamma, x\!:\!A\,;B) \underset{(\_)x}{\overset{\lambda x.\_}{\underset{\longleftarrow}{\overset{\longrightarrow}{\top}}}} \mathcal{C}(\Gamma; A\to B)$$

with the following associated rewrite rules

$(\beta_\to)$	$(\lambda x.b)x$	$\Rightarrow$	b	
$(\eta_\to)$	t	$\Rightarrow$	$\lambda x.tx$	$x \notin \mathtt{FV}(t)$

Notice that the side condition $x \notin \mathtt{FV}(t)$ is implicit in the requirement that $\Gamma, x\!:\!A$ be a context. Finally, rewrite rules for the unit may be derived in a similar fashion. If $\mathbf{1}$ is the terminal object in **Cat** and $!_{\mathcal{C}} : \mathcal{C}\to\mathbf{1}$ the unique functor from $\mathcal{C}$ to $\mathbf{1}$, then there is the adjunction

$$\mathbf{1} \underset{!}{\overset{*}{\underset{\longleftarrow}{\overset{\longrightarrow}{\top}}}} \mathcal{C}(\Gamma; 1)$$

which has a vacuous β-contraction and η-expansion given by $\eta_1 : t \Rightarrow *$ for terms $t\!:\!1$.

The *expansionary rewrite relation* is denoted $\Rightarrow$ and is the least pre-congruence on terms including the basic reductions in Table 3-2. Note that the η-rewrite rules

Table 3–2: Basic Reductions for the Expansionary Rewrite Relation

$(\beta_\to)$	$(\lambda x.b)a$	$\Rightarrow$	$b[a/x]$
$(\eta_\to)$	t	$\Rightarrow$	$\lambda x.tx$
$(\beta_{\times,1})$	$\pi_0\langle a, b\rangle$	$\Rightarrow$	a
$(\beta_{\times,2})$	$\pi_1\langle a, b\rangle$	$\Rightarrow$	b
$(\eta_\times)$	c	$\Rightarrow$	$\langle \pi_0 c, \pi_1 c\rangle$
(η_1)	a	$\Rightarrow$	$*$

have implicit type restrictions, e.g. $\eta_\to$ is restricted to terms of function type, and that $\eta_\to$ requires that x not be free in t. The equational theory generated by the expansionary rewrite relation is called *$\beta\eta$-equality*. The expansionary rewrite relation is well-defined, i.e. subject reduction holds.

Lemma 3.2.1 *If there is a term judgement $\Gamma \vdash t : A$ and a rewrite $t \Rightarrow t'$ then there is also a term judgement $\Gamma \vdash t' : A$.*

Proof The proof is by induction on the rewrite $t \Rightarrow t'$. $\square$

Lemma 3.2.2 *The $\beta\eta$-equational theory is sound and complete for models in cartesian closed categories.*

Proof Soundness is trivial while completeness follows from the construction of a free model. Consider the category $\mathcal{C}$ whose objects are the typed variables of $\Lambda^{1,\times,\to}$ and whose morphisms are

$$\mathcal{C}(x : X, y : Y) = \{ [t]_{\beta\eta} \mid x : X \vdash t : Y \}$$

with composition given by substitution and identities being variables. Completeness follows on showing that this structure is cartesian closed — we prove this category has binary products and leave the reader to verify the rest of the required structure. The product of objects $x : X$ and $y : Y$ is $z : X \times Y$ (for some appropriate choice of variable z) with projections given by the $\beta\eta$-equivalence classes of the terms $\pi_0 z$ and $\pi_1 z$. Given any product cone generated by judgements

$$w : W \vdash e : X \text{ and } w : W \vdash e' : Y$$

The mediating morphism is the $\beta\eta$-equivalence class of $\langle e, e' \rangle$ with β-reduction ensuring the required equations are satisfied while η-expansion guarantees uniqueness of the mediating morphism. $\square$

The majority of the rest of this chapter proves that $\beta\eta$-equality is decidable by placing restrictions on the applicability of η-expansion so as to recover strong normalisation, while preserving confluence and maintaining the strength of the equational theory. However, before, doing this we briefly comment on algebra of rewrites in the 2-categorical approach.

The Algebra of 2-cells

In [37] Hilken considers a 2-categorical approach to reduction in what he calls the 2-λ-calculus. This calculus is essentially the typed λ-calculus whose only type constructor is the exponential, augmented by a calculus of labelled rewrites defined in Table 3-3. Note that identity rewrites can be constructed for all terms from the identity rewrites on variables because the relation $\Rightarrow$ is a pre-congruence. In this

Table 3–3: Rewrites in the 2-λ-calculus

$$\frac{x\!:\!X \text{ is declared in } \Gamma}{\Gamma \vdash 1_x\!:\!x \Rightarrow x} \qquad \frac{\Gamma \vdash \alpha\!:\!t \Rightarrow u \quad \Gamma \vdash \alpha'\!:\!u \Rightarrow v}{\Gamma \vdash \alpha;\alpha'\!:\!t \Rightarrow v}$$

$$\frac{}{\Gamma \vdash \beta_{t,u}\!:\!(\lambda x.t)u \Rightarrow t[u/x]} \qquad \frac{t \text{ is of exponential type}}{\Gamma \vdash \eta_t\!:\!t \Rightarrow \lambda x.tx}$$

$$\frac{\Gamma \vdash \alpha\!:\!t \Rightarrow t' \quad \Gamma \vdash \alpha'\!:\!u \Rightarrow u'}{\Gamma \vdash \alpha\alpha'\!:\!tu \Rightarrow t'u'} \qquad \frac{\Gamma \vdash \alpha\!:\!t \Rightarrow t'}{\Gamma \vdash \lambda x.\alpha\!:\!\lambda x.t \Rightarrow \lambda x.t'}$$

work, the exponential is modelled as the 2-categorical generalisation of our interpretation of the exponential, namely as a *lax exponential*[37,75,72]. The equations on the rewrites of Table 3-3 required by this 2-categorical interpretation form an equational theory, denoted E, which is shown to have the following properties:

- The equational theory E is decidable.

- The equational theory is not inconsistent, i.e. there are rewrites $\alpha, \alpha'\!:\!t \Rightarrow t'$ such that $\alpha \neq_E \alpha'$. An example of such rewrites is

$$\beta_{x,\mathcal{I}(x)}\!:\!\mathcal{I}(\mathcal{I}(x)) \Rightarrow \mathcal{I}(x) \quad \text{and} \quad \mathcal{I}(\beta_{x,x})\!:\!\mathcal{I}(\mathcal{I}(x)) \Rightarrow \mathcal{I}(x)$$

  where $\mathcal{I}$ is an abbreviation for the term $\lambda x.x$.

- Confluence: If $\alpha_1 : t \Rightarrow u_1$ and $\alpha_2 : t \Rightarrow u_2$ then there exists a term v and rewrites $\gamma_1\!:\!u_1 \Rightarrow v$ and $\gamma_2\!:\!u_2 \Rightarrow v$ such that $\alpha_1;\gamma_1 =_E \alpha_2;\gamma_2$ in the equational theory on rewrites.

- Normal Forms: A term t is a long $\beta\eta$-normal form [41] iff given any rewrite $\gamma : t \Rightarrow t'$ there is a rewrite $\alpha : t' \Rightarrow t$ such that $\gamma; \alpha =_E 1_t$.

3.3 Decidability of $\beta\eta$-Equality

The restrictions on η-expansion required to recover strong normalisation were originally proposed by Mints [62], although he only proved weak normalisation and it is only recently that several authors [2,15,19,22,49], using a variety of different proof techniques, have proved both confluence and strong normalisation. There is an elegant explanation of Mints' restrictions on η-expansions in terms of categorical reduction theory.

The β- and η-rewrite rules were derived as counit and unit of an adjunction and, as with adjunctions in general, these rewrite rules are linked by triangle laws. For the exponential these are

$$\begin{array}{ccccc} \lambda x.t & \Rightarrow & \lambda y.(\lambda x.t)y & \Rightarrow & \lambda y.t[y/x] \equiv \lambda x.t \\ ta & \Rightarrow & (\lambda x.tx)a & \Rightarrow & ta \end{array} \tag{3.2}$$

while those for the product constructor are

$$\begin{array}{ccccc} \langle a, b\rangle & \Rightarrow & \langle \pi_0\langle a, b\rangle, \pi_1\langle a, b\rangle\rangle & \Rightarrow & \langle a, b\rangle \\ \pi_0 c & \Rightarrow & \pi_0\langle \pi_0 c, \pi_1 c\rangle & \Rightarrow & \pi_0 c \\ \pi_1 c & \Rightarrow & \pi_1\langle \pi_0 c, \pi_1 c\rangle & \Rightarrow & \pi_1 c \end{array} \tag{3.3}$$

where the latter pair of loops are thought of as a single loop in the product category. The triangle laws for the unit type are $* \Rightarrow *$ and the identity morphism in $\mathbf{1}$.

Any attempt to recover strong normalisation must "cut" these loops and the natural approach is to prevent those expansions which appear in the triangle laws. Thus terms of function type may be expanded provided they are neither λ-abstractions nor applied; terms of product type may be expanded if they are neither pairs nor projected and the rewrite rule η_1 may not be applied to the constant $*$ — in short we prohibit the expansion of subterms which are introduction terms or which occur negatively. In fact, these restrictions alone are enough to recover strong normalisation, i.e. it is precisely the loops created by the triangle laws which prevent normalisation.

A formal definition of Mints' reduction system, which we denote $\Rightarrow_{\mathcal{F}}$, requires a certain subtlety to encode the context sensitive nature of the restrictions imposed upon expansion. The basic idea is to simultaneously define a subrelation $\Rightarrow_{\mathcal{I}}$ of $\Rightarrow_{\mathcal{F}}$ which is guaranteed not to be a top-level η-expansion, and so a subterm which occurs negatively may be safely $\Rightarrow_{\mathcal{I}}$-rewritten when a $\Rightarrow_{\mathcal{F}}$-rewrite may cause a reduction loop. A term is called *expandable* iff it is not an introduction term, nor of base type. The rewrite relations $\Rightarrow_{\mathcal{F}}$ and $\Rightarrow_{\mathcal{I}}$ are formally defined by the inference rules in Table 3-4, where $\Rightarrow_{\mathcal{I}\cup\mathcal{F}}$ and $\Rightarrow_{\mathcal{I}\cap\mathcal{F}}$ are the union and intersection of $\Rightarrow_{\mathcal{I}}$

Table 3–4: The Restricted Rewrite Relation

$$\frac{t \Rightarrow_\beta t'}{t \Rightarrow_{\mathcal{I}\cap\mathcal{F}} t'} \qquad \frac{t \text{ expandable}}{t \Rightarrow_{\mathcal{F}} \eta(t)}$$

$$\frac{\phi \Rightarrow_{\mathcal{I}} \phi'}{\phi\psi \Rightarrow_{\mathcal{I}\cap\mathcal{F}} \phi'\psi} \qquad \frac{t \Rightarrow_{\mathcal{I}} t'}{\pi_0 t \Rightarrow_{\mathcal{I}\cap\mathcal{F}} \pi_0 t'}$$

$$\frac{\psi \Rightarrow_{\mathcal{I}\cup\mathcal{F}} \psi'}{\phi\psi \Rightarrow_{\mathcal{I}\cap\mathcal{F}} \phi\psi'} \qquad \frac{t \Rightarrow_{\mathcal{I}} t'}{\pi_1 t \Rightarrow_{\mathcal{I}\cap\mathcal{F}} \pi_1 t'}$$

$$\frac{t \Rightarrow_{\mathcal{I}\cup\mathcal{F}} t'}{\lambda x.t \Rightarrow_{\mathcal{I}\cap\mathcal{F}} \lambda x.t'} \quad \frac{u \Rightarrow_{\mathcal{I}\cup\mathcal{F}} u'}{\langle u, v\rangle \Rightarrow_{\mathcal{I}\cap\mathcal{F}} \langle u', v'\rangle} \quad \frac{v \Rightarrow_{\mathcal{I}\cup\mathcal{F}} v'}{\langle u, v\rangle \Rightarrow_{\mathcal{I}\cap\mathcal{F}} \langle u', v'\rangle}$$

and $\Rightarrow_{\mathcal{F}}$ respectively and $\eta(t)$ is the result of applying the appropriate expansion to t. The relation $\Rightarrow_{\mathcal{F}}$ is called the *restricted rewrite relation* and is clearly a subrelation of $\Rightarrow$ and, as intended, $\Rightarrow_{\mathcal{I}}$ is the subrelation of $\Rightarrow_{\mathcal{F}}$ obtained by preventing basic expansions, i.e. we may easily prove:

$$t \Rightarrow_{\mathcal{F}} t' \text{ iff } t \Rightarrow_{\mathcal{I}} t' \text{ or } t' = \eta(t) \text{ and } t \text{ is expandable}$$

The forward implication is a straightforward induction on the definition of $\Rightarrow_{\mathcal{F}}$ while the reverse implication is trivial. The following lemma allows results about the restricted rewrite relation to be lifted to the unrestricted one.

Lemma 3.3.1 *If $t \Rightarrow t'$ then either $t \Rightarrow_{\mathcal{F}} t'$ or $t' \Rightarrow^*_\beta t$. Thus confluence of $\Rightarrow_{\mathcal{F}}$ implies confluence of $\Rightarrow$.*

Proof The first half of the lemma holds because if $t \Rightarrow t'$ but $t \not\Rightarrow_{\mathcal{F}} t'$, then t' must be obtained from t by one of the triangular expansions in equations 3.2 and 3.3. As pointed out at the time, triangular expansions have β-reductions in the reverse direction and so $t' \Rightarrow^*_\beta t$.

For the second half of the lemma consider two $\Rightarrow$-equivalent terms. By the first part of this lemma these terms are also $\Rightarrow_{\mathcal{F}}$-equivalent and so, by $\Rightarrow_{\mathcal{F}}$-confluence, have a common $\Rightarrow^*_{\mathcal{F}}$-reduct. But since $\Rightarrow_{\mathcal{F}} \subseteq \Rightarrow$, we have constructed a common $\Rightarrow$-reduct for these terms and hence $\Rightarrow$ is confluent. □

3.3.1 Substitutivity, $\Rightarrow_{\mathcal{F}}$-Local Confluence

In the rest of this chapter we shall prove that the relation $\Rightarrow_{\mathcal{F}}$ is strongly normalising and confluent. An alternative development of these results may be found in [17].

Traditional proofs of strong normalisation and confluence rely heavily on reduction being *substitutive*, that is if there are reductions $t \Rightarrow t'$ and $u \Rightarrow u'$ then there is also a reduction $t[u/x] \Rightarrow^* t'[u'/x]$. However the rewrite relations $\Rightarrow_{\mathcal{I}}$ and $\Rightarrow_{\mathcal{F}}$ are not pre-congruences and so we must characterise when substitutivity fails and in these instances develop other proof techniques. Typically these consist of exhibiting reduction sequences to a common reduct or a reduction sequence with a different redex but with the same reduct.

Lemma 3.3.2 *Let t, t', u and u' be terms such that $t \Rightarrow_{\mathcal{R}} t'$ and $u \Rightarrow_{\mathcal{R}} u'$, where $\mathcal{R} \in \{\mathcal{I}, \mathcal{F}\}$. Then*

- *There is a rewrite $t[u/x] \Rightarrow_{\mathcal{R}} t'[u/x]$ unless u is an introduction term and t' is obtained by expanding an occurrence of x in t. In this case there are reduction sequences $t[\eta(u)/x] \Rightarrow^*_\beta t'[u/x] \Rightarrow^*_\beta t[u/x]$.*

- *There is a rewrite $t[u/x] \Rightarrow^*_{\mathcal{I}} t[u'/x]$ unless $u' = \eta(u)$ and either $t = x$ or there are negative occurrences of x in t. In this latter case $t[u'/x]$ and $t[u/x]$ have a common $\Rightarrow^*_{\mathcal{I}}$-reduct.*

Proof The two parts of the lemma are proved separately by induction on t. The first part follows because if u is an introduction term, then $\eta(u) \Rightarrow^*_{\beta} u$, while the reduct mentioned in the second half is constructed from $t[u/x]$ by expanding those occurrences of u which do not occur negatively. The required reductions to this term are easily verified. □

The obvious next step would be to hypothesise that both $\Rightarrow_{\mathcal{I}}$ and $\Rightarrow_{\mathcal{F}}$ are locally confluent. Unfortunately this is not the case, e.g. there are the following counterexamples:

$$\begin{tikzcd} (\lambda x.t)u \arrow[r, "\mathcal{I}"] \arrow[d, "\mathcal{I}"'] & (\lambda x.\eta(t))u \arrow[d, "\mathcal{I}"] \\ t[u/x] \arrow[r, "\mathcal{F}"'] & \eta(t)[u/x] \end{tikzcd} \quad \text{and} \quad \begin{tikzcd} \pi_0\langle u, v\rangle \arrow[r, "\mathcal{I}"] \arrow[d, "\mathcal{I}"'] & \pi_0\langle \eta(u), v\rangle \arrow[d, "\mathcal{I}"] \\ u \arrow[r, "\mathcal{F}"] & \eta(u) \end{tikzcd}$$

and similarly for $\pi_1\langle u, v\rangle$. In these examples the bottom arrow is $\Rightarrow_{\mathcal{F}}$, but not $\Rightarrow^*_{\mathcal{I}}$, and so $\Rightarrow_{\mathcal{I}}$ is not locally confluent. However local confluence of $\Rightarrow_{\mathcal{F}}$ can be proved in conjunction with a slight variant for $\Rightarrow_{\mathcal{I}}$.

Lemma 3.3.3 *The relation $\Rightarrow_{\mathcal{F}}$ is locally confluent and given any span $t \Rightarrow_{\mathcal{I}} t_i$ (where $i = 1, 2$), there is a term t' such that either $t_1 \Rightarrow^*_{\mathcal{I}} t'$ or $t_1 \Rightarrow_{\mathcal{F}} t'$ and similarly either $t_2 \Rightarrow^*_{\mathcal{I}} t'$ or $t_2 \Rightarrow_{\mathcal{F}} t'$.*

Proof The proof is by simultaneous induction on the redex. The base case is trivial as there are no $\Rightarrow_{\mathcal{I}}$-reducts of a variable and at most one $\Rightarrow_{\mathcal{F}}$-reduct. The $\Rightarrow_{\mathcal{I}}$-spans of introduction terms are caused by $\Rightarrow_{\mathcal{F}}$-spans of proper subterms, which by the induction hypothesis have $\Rightarrow^*_{\mathcal{F}}$-co-spans which may in turn be used to construct $\Rightarrow^*_{\mathcal{I}}$-co-spans to the original span. As introduction terms cannot be expanded, this analysis suffices for $\Rightarrow_{\mathcal{F}}$-spans of introduction terms as well.

Finally, consider $\Rightarrow_{\mathcal{I}}$-spans of an elimination term, say tu. Firstly given an $\Rightarrow_{\mathcal{I}}$-

span

$$t_1u \Leftarrow tu \ \Rightarrow t_2u$$

by the induction hypothesis there is a term t_3 which is either a $\Rightarrow^*_{\mathcal{I}}$-reduct, or an η-expansion, of both t_1 and t_2. Either way $\Rightarrow^*_{\mathcal{I}}$-co-spans may be given to one of t_1u, t_2u or t_3u. Secondly if a $\Rightarrow_{\mathcal{I}}$-span includes a top-level reduction $(\lambda x.t_0)u \Rightarrow t_0[u/x]$ then the result follows by lemma 3.3.2, while any other $\Rightarrow_{\mathcal{I}}$-span has a trivial $\Rightarrow^*_{\mathcal{I}}$-co-span by the induction hypothesis.

Now consider an $\Rightarrow_{\mathcal{F}}$-span of the form

$$\eta(t) \Leftarrow t \ \Rightarrow_{\mathcal{I}} \ t'$$

Again by lemma 3.3.2, $\eta(t) \Rightarrow^*_{\mathcal{I}} \eta(t')$, and since this term reduces to t', or vice versa, appropriate $\Rightarrow^*_{\mathcal{F}}$-co-spans may be constructed. All other $\Rightarrow_{\mathcal{F}}$-spans are either trivial or are $\Rightarrow_{\mathcal{I}}$-spans and so have already been considered. □

3.3.2 Strong Normalisation of $\Rightarrow_{\mathcal{F}}$

The relation $\Rightarrow_{\mathcal{F}}$ is shown to be strongly normalising to the long $\beta\eta$-normal forms. As $\Rightarrow_{\mathcal{F}}$ is not a pre-congruence care must be taken in establishing results which are usually taken as trivial in traditional term rewriting. One such example is the following:

Lemma 3.3.4 *If $\pi_0 t$ and $\pi_1 t$ are $\Rightarrow_{\mathcal{F}}$-strongly normalisable then so is t. Similarly if $x : A$ is a variable not free in u and ux is $\Rightarrow_{\mathcal{F}}$-strongly normalisable then so is u.*

Proof For the first part of the lemma it suffices to show, by induction on the sum of the normalisation ranks of the two projections, that all the 1-step reducts of t are strongly normalisable. The result $\langle \pi_0 t, \pi_1 t\rangle$ of a basic expansion is strongly normalisable since its only reductions arise from those of its components, while any reduction $t \Rightarrow_{\mathcal{I}} t'$ induces another such of $\pi_0 t$ and $\pi_1 t$. The projections of t' are thus strongly normalisable and so by the induction hypothesis t' is strongly normalisable. The second half of the lemma is proved similarly as reductions of λ-abstractions are all obtained by reductions of their body. □

The relation $\Rightarrow_{\mathcal{F}}$ is proved strongly normalising by constructing a termination order built inductively over the type structure and the rewrite relation $\Rightarrow_{\mathcal{I}}$. Given a type T, the set of *valid* terms of that type is denoted $V(T)$ and defined to be those terms of type T which can be shown valid by the inference rules in Table 3-5. Notice that this definition assumes w.l.o.g. that variables, and hence terms, have unique types. Because the relation $\Rightarrow_{\mathcal{I}}$ is finitely branching, we may associate

Table 3–5: Definition of Validity

$$\frac{t \text{ is not an intro. term} \quad t/\Rightarrow_{\mathcal{I}} \subseteq V(X)}{t \in V(X)}$$

$$\frac{u_i \in V(X_i) \quad \langle u_0, u_1\rangle/\Rightarrow_{\mathcal{I}} \subseteq V(X_0 \times X_1)}{\langle u_0, u_1\rangle \in V(X_0 \times X_1)}$$

$$\frac{\forall u \in V(X).t[u/x] \in V(Y) \quad \lambda x.t/\Rightarrow_{\mathcal{I}} \subseteq V(X \to Y)}{\lambda x.t \in V(X \to Y)}$$

$$\frac{}{* \in V(1)}$$

to each valid term t a natural number, called the *validity rank* of t and denoted $\nu(t)$, which is often used as an induction rank in reasoning about valid terms. The validity rank is defined as follows:

$$\nu(t) = 1 + max\{\nu(t') \mid t \Rightarrow_{\mathcal{I}} t'\}$$

An important remark is that, as variables have no $\Rightarrow_{\mathcal{I}}$-reducts, the valid terms of a given type contain all the variables of that type. This observation will be needed to prove that the valid terms of a given type satisfy the following *validity predicates* defined over sets of terms P:

V1: If $t \in P$ then t is $\Rightarrow_{\mathcal{F}}$-strongly normalising

V2: If $t \in P$ and $t \Rightarrow_{\mathcal{F}} t'$ then t' is valid

V3: If $t \in P$ then $\eta(t) \in P$

Induction on the type structure is used to show that the valid terms satisfy the validity predicates V1-3.

Exponential Types

Lemma 3.3.5 *Assume the valid terms of type X and Y satisfy the validity predicates V1-3. If for all valid terms u of type X, the term $t[u/x]$ is valid, then $\lambda x.t$ is also valid.*

Proof As remarked earlier if x is a variable of type X, then x is valid. Thus by assumption $t[x/x]$ is valid and, by V1 for type Y, t is $\Rightarrow_{\mathcal{F}}$-strongly normalising. Thus the $\Rightarrow_{\mathcal{F}}$-normalisation rank of t is used to prove the $\Rightarrow_{\mathcal{I}}$-reducts of $\lambda x.t$ are all valid. These reducts are of the form $\lambda x.t'$, where $t \Rightarrow_{\mathcal{F}} t'$, and so given any term $u \in V(X)$ we must prove that $t'[u/x]$ is valid. By lemma 3.3.2 there is either a reduction sequence $t[u/x] \Rightarrow^*_{\mathcal{F}} t'[u/x]$ or one $t[\eta(u)/x] \Rightarrow^*_{\mathcal{F}} t'[u/x]$ and as both $t[u/x]$ and $t[\eta(u)/x]$ are valid, the term $t'[u/x]$ is also valid. Thus $\lambda x.t'$ is valid by the induction hypothesis and hence so is $\lambda x.t$. □

Lemma 3.3.6 *Assume the valid terms of type X and Y satisfy the validity predicates V1-3. If $t \in V(X \to Y)$ and $u \in V(X)$, then $tu \in V(Y)$.*

Proof We must prove the $\Rightarrow_{\mathcal{I}}$-reducts of tu are valid and this is done by induction on the sum of the validity rank of t and the $\Rightarrow_{\mathcal{F}}$-normalisation rank of u. The $\Rightarrow_{\mathcal{I}}$-reducts of tu are: (i) induced by an $\Rightarrow_{\mathcal{F}}$-reduction of u or an $\Rightarrow_{\mathcal{I}}$-reduction of t and so are valid by the induction hypothesis or; (ii) if t is an introduction term, say $\lambda x.t_0$, then the validity of $t_0[u/x]$ follows from the validity of $\lambda x.t_0$ and the validity of u. □

Lemma 3.3.7 *Assume the valid terms of type X and Y satisfy the validity predicates V1-3 then so do the valid terms of type $X \to Y$.*

Proof If $t \in V(X \to Y)$ is valid then, given a variable $x : X$, the term tx is valid by lemma 3.3.6. Thus tx is $\Rightarrow_{\mathcal{F}}$-strongly normalising and hence by lemma 3.3.4 t is also $\Rightarrow_{\mathcal{F}}$-strongly normalising. All $\Rightarrow_{\mathcal{I}}$-reducts of a valid term are valid by definition, while the result of a basic expansion is valid by V3. By lemma 3.3.5, to prove $\lambda x.tx$ is valid it suffices to show that for every valid term u of type X, the term tu is valid. But this is just lemma 3.3.6. □

Product Types

Lemma 3.3.8 *Assume the valid terms of type X and Y satisfy the validity predicates V1-3. If $u{:}X$ and $v{:}Y$ are valid, then so is $\langle u, v\rangle$.*

Proof The proof is by induction on the sum of the $\Rightarrow_{\mathcal{F}}$-normalisation ranks of u and v. The one step $\Rightarrow_{\mathcal{I}}$-reducts of $\langle u, v\rangle$ are all induced by $\Rightarrow_{\mathcal{F}}$-reductions of either u or v and thus are valid by the induction hypothesis. □

Lemma 3.3.9 *Assume the valid terms of type X and Y satisfy the validity predicates V1-3. If $t{:}X \times Y$ is valid, then so are the terms $\pi_0 t$ and $\pi_1 t$.*

Proof The lemma is proved by induction on the validity rank of t. The one-step $\Rightarrow_{\mathcal{I}}$-reducts of $\pi_i t$ induced by $\Rightarrow_{\mathcal{I}}$-reductions of t are valid by the induction hypothesis, while, if t is a pair, the result of a basic $\beta_\times$-reduction is valid because t is valid. □

Lemma 3.3.10 *Assume the valid terms of type X and Y satisfy the validity predicates. Then so do the valid terms of type $X \times Y$.*

Proof If t is a valid term of type $X \times Y$ then by lemma 3.3.9 the terms $\pi_0 t$ and $\pi_1 t$ are valid, and so by assumption strongly normalising. Thus by lemma 3.3.4 the term t is also strongly normalising. All $\Rightarrow_{\mathcal{I}}$-reducts of a valid term are valid by definition, while the result of a basic expansion is valid by V3. Finally, if t is valid, we have already shown that $\pi_0 t$ and $\pi_1 t$ are valid and so by lemma 3.3.8 the term $\langle \pi_0 t, \pi_1 t\rangle$ is also valid. □

Lemma 3.3.11 *The set of valid terms of every type satisfy the three validity predicates V1, V2 and V3.*

Proof The proof is by induction on the type of a term. Terms of base type have no expansions and so are proved strongly normalising by induction on their validity, while the predicates V2 and V3 are trivial. The constant $*$ is automatically valid and so similar arguments hold for terms of unit type, while the validity predicates have just been established for terms of exponential and product types in lemmas 3.3.7 and 3.3.10. □

We may now show that all terms are valid and hence $\Rightarrow_{\mathcal{F}}$-strongly normalising.

Lemma 3.3.12 *Let $t : T$ be any term, with free variables among $x_i : X_i$ for $i = 1, \ldots, n$ and let $u_i : X_i$ be valid terms. Then $t[u_i/x_i]$ is valid.*

Proof The proof is by induction over the structure of the term. The cases involving variables, $*$, projection and application are all trivial, while pairing is handled by lemma 3.3.8. Finally, if $t = \lambda y.b : A \rightarrow B$ then $t[u_i/x_i]$ is valid iff $b[u_i/x_i][v/y]$ is valid for all valid $v : A$. But this follows by the induction hypothesis. □

Corollary 3.3.13 *Every term is valid, and so $\Rightarrow_{\mathcal{F}}$-strongly normalisable. Thus $\Rightarrow_{\mathcal{F}}$ is also confluent.*

Proof Apply lemma 3.3.12 with $u_i = x_i$. Confluence now follows from local confluence. □

These normalisation and confluence results mean $\beta\eta$-equality may be decided by reduction to $\Rightarrow_{\mathcal{F}}$-normal form.

Corollary 3.3.14 *Two terms are $\beta\eta$-equal iff they have the same $\Rightarrow_{\mathcal{F}}$-normal form.*

Proof The forward implication holds because of the first half of lemma 3.3.1, while the reverse implication is trivial as $\Rightarrow_{\mathcal{F}} \subseteq \Rightarrow$. □

This chapter finishes with further results concerning the nature and calculation of $\Rightarrow_{\mathcal{F}}$-normal forms.

Lemma 3.3.15 *The $\Rightarrow_{\mathcal{F}}$-normal form of a term may be calculated by first reducing the term to its β-normal form and then performing any remaining restricted*

η-expansions or, vice versa, by performing all restricted η-expansions and then reducing the resulting term to its β-normal form.

Proof The proof rests on showing that a restricted η-expansion preserves β-normal forms and vice versa that β-reduction preserves terms which are fully η-expanded. □

Notice that the traditional counterexample to this lemma [19]

$$t \Rightarrow \lambda x.tx \Rightarrow \lambda x.*$$

arises from the erroneous classification of η_1 as a β-redex.

Generalising these results to the subsequent case studies is difficult because the restricted η-rewrite rules for initial type constructors are not in themselves confluent. Thus we shall be forced to consider more complex normalisation strategies and will be guided by the generalisation of the following lemma. A term is a *long $\beta\eta$-normal form* [41] iff it is a β-normal form and all subterms are either of base type, introduction terms or occur negatively. A term is an *internal long $\beta\eta$-normal form* iff it is a β-normal form and all subterms, apart from the term itself, are either of base type, introduction terms or occur negatively.

Lemma 3.3.16 *A term is a long $\beta\eta$-normal form iff it is a $\Rightarrow_{\mathcal{F}}$-normal form.*

Proof Both directions of the lemma are easily established by induction on the structure of t, while simultaneously proving that a term is a $\Rightarrow_{\mathcal{I}}$-normal form iff it is an internal long $\beta\eta$-normal form. □

Chapter 4

Linear λ-Calculus

Although substantial agreement exists on the nature of rewriting for final type constructors such as those contained in the simply typed λ-calculus, rewriting for initial type constructors remains the subject of much research with only limited results so far. The linear λ-calculus contains two initial type constructors: a binary constructor called the *tensor* and a nullary type constructor called the *unit*. The relationship between the unit and the tensor is similar to that between the unit and the product of the simply typed λ-calculus, i.e. the semantic interpretation of these structures is intended to form part of a monoidal structure [7,46,50,57,59]. However, because the unit and the tensor of the linear λ-calculus are initial type constructors, their proposed η-rewrite rules are significantly more complex than those for the unit and product of the simply typed λ-calculus. The other interesting feature of the linear λ-calculus is the omission of the structural rules *weakening* and *contraction* which, by preventing the non-linear use of variables, permits a clearer account of the general properties of rewriting for initial type constructors unhindered by mathematical technicalities caused by these structural rules.

After defining the linear λ-calculus, a rewrite relation is derived by using the same basic principles as for final types, i.e. the introduction and elimination rules of each type constructor are interpreted as being adjoint functors and the associated unit and counit then form an expansionary η-rewrite rule and a contractive β-rewrite rule. The associated equational theory is shown sound and complete for models in symmetric monoidal closed categories.

Although decidability of this equational theory was originally thought to be a straightforward extension of the results of the previous chapter, the presence of initial type constructors poses substantial new technical problems. Not only is there a facility for expanding terms of tensor and unit type similar to that for expanding terms of product and exponential type, but also a new possibility to permute the order in which different subterms of initial type may occur.

These different aspects of η-conversion for initial type constructors are reflected in our analysis. The rewrite relation is decomposed into two fragments, the first of which is similar to the restricted rewrite relation of the simply typed λ-calculus in containing β-reductions, commuting conversions and limited possibilities for η-expansion. This fragment is strongly normalising, confluent and has normal forms which satisfy similar structural criteria to the long $\beta\eta$-normal forms of the simply typed λ-calculus. However, the restricted expansions on their own are not confluent and may introduce introduce new, non-looping, β-redexes. Thus lemma 3.3.15 does not generalise to this calculus and so we are forced to develop a different normalisation strategy.

The second part of the decomposition is called the *conversion relation* and allows permutation of the order in which certain subterms of tensor and unit type occur. For each term there is a (finite) set of such permutations, and so terms cannot always be rewritten to unique normal forms. Instead we construct for each term its set of *quasi-normal forms*, one for each possible permutation, and show that terms equivalent in the conversion relation have the same set of quasi-normal forms.

Finally, confluence and decidability of the full equational theory are proved by suitably embedding the full rewrite relation into the conversion relation; that is, terms equivalent in the full theory are shown to have unique normal forms in the first part of the decomposition which are equivalent in the conversion relation.

The linear λ-calculus has received considerable attention recently as its logical counterpart forms a core fragment of intuitionistic linear logic [1,7,32,31,76]. The adjective *linear* refers to the absence of the structural rules *weakening* and *contraction* which ensures that variables are 'used' exactly once. This linearity removes some of the technical problems which arise in more complex calculi such as the bicartesian calculus of the next chapter. In particular, reduction preserves the free variables of a term and the conversion relation is left-linear in that distinct occurrences in the redex are mapped to distinct occurrences in the reduct. The key technical contribution of this chapter is the idea of the conversion relation which is so named because the *commuting conversions* appearing in the literature [32,68] are special cases.

4.1 The Calculus $\Lambda^{I,\otimes,\rightarrow}$

The calculus studied in this chapter is a fragment of that presented in [1]. The *linear λ-calculus* over a set of base types $\mathcal{B}$ is denoted $\Lambda^{I,\otimes,\rightarrow}$ and consists of *types* freely generated from the base types by a nullary type constructor I, called the *unit*, and two binary constructors $\otimes$ and $\rightarrow$ called the *tensor* and *exponential*

$$T := T \otimes T \mid T \rightarrow T \mid I \mid B$$

where $B \in \mathcal{B}$ is any base type. The *atomic* types are I and the base types. For each type T there is a set of constants $\mathtt{Con}(T)$, including the distinguished constant $* \in \mathtt{Con}(I)$, such that if $T \neq T'$, then $\mathtt{Con}(T) \cap \mathtt{Con}(T') = \emptyset$. There is also a set of variables $\mathtt{Var}$ which is disjoint from the constants. The *pre-terms* of the linear λ-calculus are:

$$t \coloneqq x \mid c \mid t \otimes t \mid \mathbf{let}\, t\, \mathbf{be}\, *\, \mathbf{in}\, t \mid \mathbf{let}\, t\, \mathbf{be}\, x \otimes y\, \mathbf{in}\, t \mid tt \mid \lambda x.t$$

where x, y are variables and c a constant. In a pre-term $\mathbf{let}\, t\, \mathbf{be}\, x \otimes y\, \mathbf{in}\, t'$, the variables x and y are not subterms, but are rather variable binders which play a similar role to the x in $\lambda x.t$. The free variables of a pre-term t are denoted $\mathtt{FV}(t)$ and the substitution of pre-terms for free variables is defined as expected.

The *term judgements* of the linear λ-calculus are of the form $\Gamma \vdash t : T$ where Γ is a context, t a pre-term and T a type, and are generated by the inference rules in Table 4-1. As usual, the inference rule *cut* requires Γ and Δ to be disjoint so as to ensure the context in the conclusion of the rule is well defined. In addition, because the variables in a context are pairwise distinct, the variables x and y in a

Table 4–1: Typing Rules for the Linear λ-Calculus

Structural and Identity Rules

$$\frac{x \in \texttt{Var}}{x : A \vdash x : A}\ axiom \qquad \frac{c \in \texttt{Con}(A)}{\vdash c : A}\ cons$$

$$\frac{\Gamma, x : A, y : B, \Delta \vdash t : C}{\Gamma, y : B, x : A, \Delta \vdash t : C}\ exchange \qquad \frac{\Gamma \vdash t : A \quad \Delta, x : A \vdash u : B}{\Gamma, \Delta \vdash u[t/x] : B}\ cut$$

Logical Rules

$$\frac{}{\vdash * : I}\ I\ int \qquad \frac{\Gamma \vdash e : C \quad x \notin \texttt{dom}(\Gamma)}{\Gamma, x : I \vdash \mathbf{let}\, x\, \mathbf{be}\, *\, \mathbf{in}\, e : C}\ I\ elim$$

$$\frac{\Gamma \vdash e : A \to B \quad x \notin \texttt{dom}(\Gamma)}{\Gamma, x : A \vdash ex : B}\ {\to}elim \qquad \frac{\Gamma, x : A, y : B \vdash e : C \quad z \notin \texttt{dom}(\Gamma)}{\Gamma, z : A \otimes B \vdash \mathbf{let}\, z\, \mathbf{be}\, x \otimes y\, \mathbf{in}\, e : C}\ {\otimes}elim$$

$$\frac{\Gamma, x : A \vdash e : B}{\Gamma \vdash \lambda x.e : A \to B}\ {\to}int \qquad \frac{\Gamma, z : A \otimes B \vdash t : Y \quad x, y \notin \texttt{dom}(\Gamma)}{\Gamma, x : A, y : B \vdash t[x \otimes y/z] : Y}\ {\otimes}int$$

term $\mathbf{let}\, t\, \mathbf{be}\, x \otimes y\, \mathbf{in}\, t'$ are distinct and this will be required later to ensure that the associated β-redex is well defined.

Given any term judgement $\Gamma \vdash t : T$, we say that t is a term of type T and when Γ is clear, or unimportant, this is written $t : T$. The subterm occurrences of a term t are denoted $\texttt{O}(t)$. A term is called an *introduction* term if it is of the form $\lambda x.t$ or $u \otimes v$ or $*$, while a term which is not an introduction term is called an *elimination* term. Finally, a subterm occurs *negatively* if the subterm is either applied or is the first argument of a *let*-expression.

The linear λ-calculus differs from the simply typed λ-calculus in a crucial manner, namely the structural rules *weakening* and *contraction* which permit variables to be duplicated and/or discarded are absent in the linear λ-calculus. Thus in any judgement $\Gamma \vdash t : A$, the variables in the context Γ occur free in t exactly once, and, conversely, the free variables of t are exactly those occurring in the context

Γ. These structural properties are collectively referred to as *linearity* and, although considerably simplifying some of the mathematical technicalities, the need to preserve the linearity of the calculus complicates the meta-theory of the linear λ-calculus. The clearest example of this is the need to linearise the usual definition of substitution. The substitution of the terms t_i for the variables x_i into the term u is *linear* iff the variables x_i are pairwise distinct, each variable $x_i \in \mathtt{FV}(u)$ and the sets $\mathtt{FV}(t_i)$ and $\mathtt{FV}(u) \setminus \{x_1, \ldots x_n\}$ are also pairwise disjoint. Given a linear substitution the resulting term $u[t_1/x_1, \ldots, t_n/x_n]$ is defined as expected.

Lemma 4.1.1 *If the substitution of the terms t_i for the variables x_i in u is linear, then $u[t_1/x_1, \ldots, t_n/x_n]$ is a term of the linear λ-calculus.*

Proof The proof is by induction on the derivation of u as a term of the linear λ-calculus. □

Unless otherwise stated, all substitutions in this chapter are assumed linear, although for the sake of brevity non-linear substitutions may be used when the appropriate linear form is clear. The proposed η-rewrite rule for the tensor and unit considers terms expressed as substitutions, i.e. occurrences which index subterms whose free variables are not bound in the term as a whole. This is formalised by first defining the variables bound at an occurrence $\sigma \in \mathtt{O}(t)$ as follows:

$$\mathtt{BV}(\sigma, t) = \begin{cases} \emptyset & \text{if } \sigma = \epsilon \\ \{x\} \cup \mathtt{BV}(\sigma^-, t') & \text{if } t = \lambda x.t' \text{ and } \sigma \neq \epsilon \\ \{x, y\} \cup \mathtt{BV}(\sigma^-, t_1) & \text{if } t = \mathbf{let}\, t_0\, \mathbf{be}\, x \otimes y\, \mathbf{in}\, t_1 \text{ and } \sigma \geq 1 \\ \mathtt{BV}(\sigma^-, t/\sigma_0) & \text{otherwise, } \sigma = \sigma_0 \cdot \sigma^- \end{cases}$$

Given any term t its set of *free occurrences* are simply those occurrences which index subterms whose free variables are unbound at the occurrence, i.e.

$$\mathtt{FO}(t) = \{\sigma \in \mathtt{O}(t) \mid \mathtt{FV}(t/\sigma) \cap \mathtt{BV}(\sigma, t) = \emptyset\}$$

The reader is left to verify that free occurrences represent substitutions, i.e. that if $\sigma \in \mathtt{FC}(t)$, then:

$$t = t[\sigma \leftarrow x][(t/\sigma)/x]$$

where, unfortunately, the symbol / has been overloaded to be both the subterm indexed by an occurrence and also the result of a substitution.

4.2 Rewriting in $\Lambda^{I,\otimes,\multimap}$

A rewrite relation for terms of the linear λ-calculus is derived by taking the introduction and elimination rule for each type constructor as forming an adjoint pair; the associated unit then forms an expansionary η-rewrite rule while the counit forms a contractive β-rewrite rule. There is however a difference between the way rewrite rules are derived for final type constructors and for initial type constructors. For final type constructors introduction is right adjoint to elimination, but for initial type constructors the adjunction is the other way around, i.e. introduction is left adjoint to elimination. Thus the elimination terms and not the introduction terms are used to represent reductions and hence form the basis of quasi-normal forms for terms of initial type.

Consider first the tensor type constructor. If $\mathcal{C}(\Gamma; X)$ is the category whose objects are judgements of the form $\Gamma \vdash e : X$ and whose morphisms between $\Gamma \vdash e : X$ and $\Gamma \vdash e' : X$ are (quotiented, labelled) rewrites $r : e \Rightarrow e'$, then the introduction and elimination rules of the tensor may be regarded as functors between the categories shown below. As these functors are taken to constitute an adjoint pair,

$$\begin{tikzcd} \mathcal{C}(\Gamma, x:X, y:Y; A) \arrow[r, shift left=2, "\mathbf{let}\, z \,\mathbf{be}\, x \otimes y \,\mathbf{in}\, \_"] \arrow[r, phantom, "\top"] & \mathcal{C}(\Gamma, z:X \otimes Y; A) \arrow[l, shift left=2, "\_[x \otimes y/z]"] \end{tikzcd}$$

the associated unit and counit are the rewrite rules:

$$\begin{array}{lrcl} \beta_\otimes : & \mathbf{let}\, x \otimes y \,\mathbf{be}\, x \otimes y \,\mathbf{in}\, t & \Rightarrow & t \\ \eta_\otimes : & e & \Rightarrow & \mathbf{let}\, z \,\mathbf{be}\, x \otimes y \,\mathbf{in}\, e[x \otimes y/z] \end{array} \tag{4.1}$$

where in the $\eta_\otimes$-rewrite rule the side conditions $z \in \mathtt{FV}(e)$ and $x, y \notin \mathtt{FV}(e)$ are inferred from the linearity of the calculus and prevent variable capture. In terms of universal properties the elimination term satisfies the following $\beta_{\otimes}$-reduction:

$$\begin{tikzcd}
\Delta, x\!:\!X, y\!:\!Y \arrow[r, "x \otimes y"] \arrow[dr, "e"'] & \Delta, z\!:\!X \otimes Y \arrow[d, "\mathbf{let}\, z\, \mathbf{be}\, x \otimes y\, \mathbf{in}\, e"] \\
 & w\!:\!C
\end{tikzcd}$$

and in addition is terminal amongst such terms. Thus there is a natural isomorphism of rewrites:

$$e' \Rightarrow \mathbf{let}\, z\, \mathbf{be}\, x \otimes y\, \mathbf{in}\, e \quad \text{iff} \quad e'[x \otimes y/z] \Rightarrow e$$

with the side conditions that $z \in \mathtt{FV}(e')$ and $x, y \notin \mathtt{FV}(e')$. This natural isomorphism of rewrites is of course equivalent (in the presence of identity rewrites) to the rewrite rules in equation 4.1 above. Notice that it is elimination terms which are used to represent reductions and hence elimination terms form the basis for the construction of quasi-normal forms for terms of initial type. A similar analysis is applicable to the unit type where again the elimination rule is right-adjoint to introduction:

$$\begin{tikzcd}
\mathcal{C}(\Gamma; A) \arrow[r, shift left=1.5, "\mathbf{let}\, z\, \mathbf{be}\, *\, \mathbf{in}\, _"] \arrow[r, phantom, "\top"] & \mathcal{C}(\Gamma, z\!:\!I; A) \arrow[l, shift left=1.5, "_[*/z]"]
\end{tikzcd}$$

and the unit and counit are the following rewrite rules:

$$\begin{array}{llcl}
\beta_I : & \mathbf{let}\ *\ \mathbf{be}\ *\ \mathbf{in}\, t & \Rightarrow & t \\
\eta_I : & e & \Rightarrow & \mathbf{let}\, z\, \mathbf{be}\ *\ \mathbf{in}\, e[*/z]
\end{array}$$

where the η_I-rewrite rule has the side condition $z \in \mathtt{FV}(e)$. The reader is left to formulate these rewrite rules in terms of universal properties and a natural isomorphism of rewrites. Finally, as the rewrite rules for each type constructor are determined solely by their introduction and elimination rules, the rewrite rules previously derived for the exponential remain valid in this setting.

The *expansionary rewrite relation* is denoted $\Rightarrow$ and is the least pre-congruence on terms containing the basic reductions in Table 4-2. The $\eta_{\multimap}$-rewrite rule has the side condition $x \notin \mathtt{FV}(t)$ and the $\eta_{\otimes}$-rewrite rule has the side conditions $z \in \mathtt{FV}(e)$ and $x, y \notin \mathtt{FV}(e)$. Similar conditions are required for the η_I-rewrite rule. The

Table 4–2: Rewrite Rules for the Linear λ-Calculus

$(\beta_{\rightarrow})$	$(\lambda x.b)a$	$\Rightarrow$	$b[a/x]$
$(\eta_{\rightarrow})$	t	$\Rightarrow$	$\lambda x.tx$
$(\beta_{\otimes})$	$\mathbf{let}\, u \otimes v\, \mathbf{be}\, x \otimes y\, \mathbf{in}\, t$	$\Rightarrow$	$t[u/x, v/y]$
$(\eta_{\otimes})$	$e[e'/z]$	$\Rightarrow$	$\mathbf{let}\, e'\, \mathbf{be}\, x \otimes y\, \mathbf{in}\, e[x \otimes y/z]$
(β_I)	$\mathbf{let}\, * \,\mathbf{be}\, * \,\mathbf{in}\, t$	$\Rightarrow$	t
(η_I)	$e[e'/z]$	$\Rightarrow$	$\mathbf{let}\, e'\, \mathbf{be}\, * \,\mathbf{in}\, e[*/z]$

equational theory generated by the expansionary rewrite relation is called *$\beta\eta$-equality.*

The $\beta_{\otimes}$-rewrite rule is unambiguous because the variables in a context are pairwise distinct and hence the variables bound in the redex are also distinct. The linearity of the calculus ensures that the variable being substituted in the η_I- and $\eta_{\otimes}$-rewrite rules must occur freely exactly once and so the expansionary rewrite relation does not introduce any new free variables and is left-linear in the sense that distinct occurrences in the redex are mapped to distinct occurrences in the reduct. Given these remarks, we may prove that the expansionary rewrite relation is well-defined, i.e. that subject reduction holds.

Lemma 4.2.1 *If there is a term judgement $\Gamma \vdash t : A$ and a rewrite $t \Rightarrow t'$ then there is also a term judgement $\Gamma \vdash t' : A$.*

Proof The proof is by induction on the term judgement $\Gamma \vdash t : A$. □

The reason that the expansionary rewrite relation is important is that it generates a sound and complete equational theory for symmetric monoidal categories.

Lemma 4.2.2 *The $\beta\eta$-equality is sound and complete for models in symmetric monoidal closed categories.*

Proof Soundness is trivial while completeness follows from the construction of a free model. Consider a category $\mathcal{C}$ whose objects are typed variables and whose morphisms are $\beta\eta$-equivalence classes of terms:

$$\mathcal{C}(x\!:\!X, y\!:\!Y) = \{\ [t]_{\beta\eta}\ \mid x\!:\!X \vdash t\!:\!Y\}$$

with composition given by substitution and identities given by variables. Completeness is proved by showing that this category is a symmetric monoidal closed category. We define a bifunctor and leave the reader to verify the rest of the required structure. The bifunctor will map objects $x:X$ and $y:Y$ to an object $z:X \otimes Y$ (for an appropriate choice of variable z), and morphisms generated by terms

$$x\!:\!X \vdash t\!:\!X' \;\text{ and }\; y\!:\!Y \vdash u\!:\!Y'$$

are mapped to the morphism generated by the term $\mathbf{let}\, z\, \mathbf{be}\, x \otimes y\, \mathbf{in}\, t \otimes u$. The identity law holds as $z =_{\beta\eta} \mathbf{let}\, z\, \mathbf{be}\, x \otimes y\, \mathbf{in}\, x \otimes y$, while verifying the composition law for morphisms

$$x_i\!:\!X_i \vdash t_i\!:\!Y_i \;\text{ and }\; y_i\!:\!Y_i \vdash u_i\!:\!Z_i \qquad\qquad i = 0, 1$$

amounts to proving

$$\begin{aligned}&\mathbf{let}\,(\mathbf{let}\, w\, \mathbf{be}\, x_0 \otimes x_1\, \mathbf{in}\, t_0 \otimes t_1\,)\, \mathbf{be}\, y_0 \otimes y_1\, \mathbf{in}\, u_0 \otimes u_1 \;\; =_{\beta\eta}\\ &\qquad \mathbf{let}\, w\, \mathbf{be}\, x_0 \otimes x_1\, \mathbf{in}\, u_0[t_0/y_0] \otimes u_1[t_1/y_1]\end{aligned}$$

which may be done by an $\eta_\otimes$-expansion of the variable w followed by a series of $\beta_\otimes$-contractions. □

The $\eta_\otimes$-rewrite rule and the η_I-rewrite rule differ significantly from the η-rewrite rules for final types by permitting (free) subterms of tensor and unit type to be expanded to the head of the term. As terms typically contain many such subterms each term cannot be rewritten to a unique normal form; rather we associate to each term a set of quasi-normal forms, one for each of the different permutations in which subterms may be expanded. For example, the term

$$(\mathbf{let}\, u\, \mathbf{be}\, x \otimes y\, \mathbf{in}\, t\,) \otimes (\mathbf{let}\, u'\, \mathbf{be}\, x' \otimes y'\, \mathbf{in}\, t'\,)$$

has two quasi-normal forms:

$$\mathbf{let}\, u\, \mathbf{be}\, x \otimes y\, \mathbf{in}\,(\mathbf{let}\, u'\, \mathbf{be}\, x' \otimes y'\, \mathbf{in}\, t \otimes t'\,) \tag{4.2}$$

and

$$\textbf{let}\, u' \,\textbf{be}\, x' \otimes y' \,\textbf{in}\, (\textbf{let}\, u \,\textbf{be}\, x \otimes y \,\textbf{in}\, t \otimes t' \,) \tag{4.3}$$

depending on the order in which subterms are expanded. To accommodate this feature, the η_I-rewrite rule and the $\eta_\otimes$-rewrite rule are decomposed into two parts, the first of which is truly expansionary while the second deals with the permutations and so is more cyclic than expansionary.

Special cases of the η-rewrite rules for the tensor and the unit are:

$$t \Rightarrow \textbf{let}\, t \,\textbf{be}\, x \otimes y \,\textbf{in}\, x \otimes y \quad \text{and} \quad t \Rightarrow \textbf{let}\, t \,\textbf{be}\, * \,\textbf{in}\, *$$

These specialisations of $\eta_\otimes$ and η_I are more akin to the η-rewrite rules of final type constructors in that terms of tensor and unit type are converted into negatively occurring subterms of the reduct. Indeed, once suitable restrictions have been imposed upon the applicability of these expansions, and when taken together with the β-reductions and commuting conversions, the resulting relation is strongly normalising and confluent. In addition the normal forms of this relation satisfy structural criteria similar to the *long $\beta\eta$-normal forms* of the simply typed λ-calculus in being β-normal forms whose subterms are all either of base type, occur negatively or are a slight generalisation of an introduction term called a *quasi-introduction term*.

The second part of the decomposition is a new contribution to the field which generalises the *commuting conversions* of [32,68]. A *conversion* is a subterm of initial type which occurs negatively, e.g. as the first argument of a *let*-expression, and the *conversion relation* allows conversions to be rewritten to the head of a term. For example, the term in equation 4.2 may be rewritten to the term in equation 4.3 by expanding the conversion indexing the subterm u' to the head of the term. Confluence follows from parallelising the conversion relation and decidability is proved by constructing the finite set of quasi-normal forms of a term and showing that terms equivalent in the associated equational theory have the same set of quasi-normal forms. These confluence and decidability results are lifted to the full expansionary rewrite relation by showing that terms which are equivalent in the full rewrite relation have normal forms under the first part of the decomposition which are equivalent in the conversion relation.

4.3 The Conversion Relation

The $\eta_\otimes$- and η_I-rewrite rules extract a free subterm of tensor/unit type and insert an introduction term at its occurrence in the redex; if the original subterm occurs negatively, e.g. as the first argument of a *let*-expression, then a new β-redex is created by this process. The conversion relation restricts application of these η-rewrite rules to extract only those free subterms which occur negatively and then contracts the resulting β-redex. More precisely, the *conversions* of a term t are defined as follows:

$$\mathtt{C}(t) = \{\sigma \in \mathtt{O}(t) \mid \sigma \text{ is the first argument position of a } let\text{-expression}\}$$

Note that ϵ, the occurrence indexing the whole term, can never be a conversion. The tensor conversions are those conversions which index subterms of tensor type, i.e.

$$\mathtt{C}_\otimes(t) = \{\sigma \in \mathtt{C}(t) \mid t/\sigma \text{ has tensor type }\}$$

and unit conversions $\mathtt{C}_I(t)$ are defined similarly. The free conversions of t are simply those occurrences which are both free and conversions, $\mathtt{FC}(t) = \mathtt{C}(t) \cap \mathtt{FO}(t)$, with appropriate subscripts used to define free tensor conversions etc. Every conversion has a *binding* describing which, if any, variables are bound at the *let*-expression associated to the conversion, i.e.

$$\mathbf{bind}(\sigma, t) \quad = \quad \begin{cases} * & \text{if } t/\sigma^+ \equiv \mathbf{let}\, t_0 \,\mathbf{be}\, * \,\mathbf{in}\, t_1 \\ x \otimes y & \text{if } t/\sigma^+ \equiv \mathbf{let}\, t_0 \,\mathbf{be}\, x \otimes y \,\mathbf{in}\, t_1 \end{cases}$$

The binding of a conversion, which may be regarded as either a syntactic pattern or the pair of associated variables, is used to prevent variable capture in the definition of the conversion relation. If $\sigma \in \mathtt{FC}(t)$ is a free conversion of t, then the result of contracting the β-redex formed upon the insertion of the appropriate introduction term at σ is called the *residue*. This is denoted $t \setminus \sigma$ and has a particularly simple recursive definition:

$$t \setminus \sigma = \begin{cases} v & \text{if } t = \mathbf{let}\, u\, \mathbf{be}\, p\, \mathbf{in}\, v \text{ and } \sigma = 0 \\ \mathcal{T}(t'_0, \ldots, t'_n) & \text{if } \sigma \neq 0 \text{ and } t = \mathcal{T}(t_0, \ldots, t_n) \end{cases}$$

where in the last clause if $\sigma = j \cdot \sigma^-$ then $t'_j = t_j \setminus \sigma^-$ and for all other i, $t'_i = t_i$. In short the residue $t \setminus \sigma$ is obtained by textually replacing the *let*-expression associated to the conversion with its associated arm. Thus the size of the residue, and that of the subterm indexed by a conversion, will always be smaller than the term itself. This will be important in later inductive arguments.

The *linear conversion relation* is denoted $\Rightarrow_c$ and is defined as the least pre-congruence generated by the inference rule

$$\frac{\sigma \in \mathtt{FC}(t)}{t \;\Rightarrow_c\; \mathbf{let}\, t/\sigma\, \mathbf{be}\, p\, \mathbf{in}\, t \setminus \sigma}$$

where $p = \mathbf{bind}(\sigma, t)$ and, if σ is a tensor conversion, the following requirements are imposed to prevent variable capture:

$$\mathbf{bind}(\sigma, t) \cap \mathtt{FV}(t) = \emptyset \text{ and } \mathbf{bind}(\sigma, t) \cap \mathtt{BV}(\sigma^+, t) = \emptyset \tag{4.4}$$

Of course these conditions can always be met by renaming bound variables. The conversion relation is so named because the relation generalises the *commuting conversions* [32,68] whose basic redexes are formed by negatively occurring sub-terms which are *let*-expressions. In order to help the reader familiarise him/herself with the linear conversion relation, we prove the following commuting conversions (one for each of the possibilities for p and p') are instances of the linear conversion relation (assuming p doesn't bind $\mathtt{FV}(t)$):

$$\mathbf{let}\,(\mathbf{let}\, u\, \mathbf{be}\, p\, \mathbf{in}\, v\,)\, \mathbf{be}\, p'\, \mathbf{in}\, t \quad \Rightarrow_c \; \mathbf{let}\, u\, \mathbf{be}\, p\, \mathbf{in}\,(\mathbf{let}\, v\, \mathbf{be}\, p'\, \mathbf{in}\, t\,) \tag{4.5}$$

Let α be the above redex, α' its residual and consider the free conversion $(0 \cdot 0) \in \mathtt{FC}(\alpha)$ whose associated binding is p. Then

$$\begin{aligned} \alpha/(0 \cdot 0) &= (\mathbf{let}\,(\mathbf{let}\, u\, \mathbf{be}\, p\, \mathbf{in}\, v\,)\, \mathbf{be}\, p'\, \mathbf{in}\, t\,)/(0 \cdot 0) \\ &= \mathbf{let}\, u\, \mathbf{be}\, p\, \mathbf{in}\, v\,/0 \\ &= u \end{aligned}$$

and

$$\begin{aligned}
\alpha \setminus (0\cdot 0) &= (\mathbf{let}\,(\mathbf{let}\,u\,\mathbf{be}\,p\,\mathbf{in}\,v\,)\,\mathbf{be}\,p'\,\mathbf{in}\,t\,)\setminus(0\cdot 0)\\
&= \mathbf{let}\,(\mathbf{let}\,u\,\mathbf{be}\,p\,\mathbf{in}\,v\ \setminus 0)\,\mathbf{be}\,p'\,\mathbf{in}\,t\\
&= \mathbf{let}\,v\,\mathbf{be}\,p'\,\mathbf{in}\,t
\end{aligned}$$

and so there is a rewrite in the linear conversion relation

$$\begin{aligned}
\alpha &\Rightarrow_c \mathbf{let}\,(\alpha/(0\cdot 0)\,\mathbf{be}\,p\,\mathbf{in}\,\alpha\setminus(0\cdot 0)\\
&= \mathbf{let}\,u\,\mathbf{be}\,p\,\mathbf{in}\,(\mathbf{let}\,v\,\mathbf{be}\,p'\,\mathbf{in}\,t\,)
\end{aligned}$$

Rewrites such as those in equation 4.5 are extremely important in the construction of the quasi-normal forms of the conversion relation because they describe the irreversible process by which conversions embedded inside other conversions in the redex are rewritten so that they may become minimal conversions (in the prefix ordering) in the reduct. However, before such matters are addressed we shall use subject reduction of the expansionary rewrite relation to prove subject reduction for the linear conversion relation.

Lemma 4.3.1 *Let $\Gamma \vdash t\!:\!Z$ be a term judgement and $t \Rightarrow_c t'$. Then there is also a term judgement $\Gamma \vdash t'\!:\!Z$.*

Proof The lemma is proved by exhibiting a reduction $t \Rightarrow t'$ and appealing to lemma 4.2.1. Since both $\Rightarrow$ and $\Rightarrow_c$ are pre-congruences, we need only consider the case $t \Rightarrow_c \mathbf{let}\,t/\sigma\,\mathbf{be}\,p\,\mathbf{in}\,t\setminus\sigma$ where $\sigma \in \mathtt{FC}(t)$ has binding p. Assume w.l.o.g. that σ is a tensor conversion binding the variables x, y and that $z \notin \mathtt{FV}(t)$. Then

$$\begin{aligned}
t &= t[\sigma\leftarrow z][(t/\sigma)/z]\\
&\Rightarrow_\eta \mathbf{let}\,t/\sigma\,\mathbf{be}\,x\otimes y\,\mathbf{in}\,(t[\sigma\leftarrow z][x\otimes y/z])\\
&= \mathbf{let}\,t/\sigma\,\mathbf{be}\,x\otimes y\,\mathbf{in}\,(t[\sigma\leftarrow x\otimes y])\\
&\Rightarrow_\beta \mathbf{let}\,t/\sigma\,\mathbf{be}\,x\otimes y\,\mathbf{in}\,t\setminus\sigma
\end{aligned}$$

where, unfortunately, the symbol / has been overloaded in the first line to both extract the subterm at an occurrence and also the substitution of a term for a free variable. The first equality holds as σ is a free occurrence and $z \notin \mathtt{FV}(t)$, the

η-expansion is valid as the variables x, y are assumed not to be free in t, and the second equality holds as the variables x, y are not bound at σ^+ and so substitution is the same as syntactic replacement. Finally, the β-reduction follows as x, y were chosen as the variables bound at the conversion. □

Confluence

The conversion relation is shown confluent by embedding it in a relation which allows terms to be rewritten in parallel and which satisfies the diamond property. The quasi-normal forms of the conversion relation are characterised by two structural properties which provide the key to their construction, and hence decidability of the equational theory generated by the conversion relation. The *parallel conversion relation*, denoted $\twoheadrightarrow$, is the least relation on terms defined by the inference rules in Table 4-3, where in the clause (iv) of the definition, if σ is a tensor conversion, then the restrictions of equation 4.4 are required to prevent variable capture. Although identity derivations $t \twoheadrightarrow t$ are formally given only for variables,

Table 4–3: The Parallel Conversion Relation

(i) Identity: For any variable z

$$\frac{}{z \twoheadrightarrow z}$$

(ii) Pre-Congruence: One inference rule for each term constructor $\mathcal{T}$

$$\frac{u_0 \twoheadrightarrow u_0' \quad \ldots \quad u_n \twoheadrightarrow u_n'}{\mathcal{T}(u_0, \ldots, u_n) \twoheadrightarrow \mathcal{T}(u_0', \ldots, u_n')}$$

(iii) Substitution:

$$\frac{u \twoheadrightarrow u' \quad v \twoheadrightarrow v'}{u[v/z] \twoheadrightarrow u'[v'/z]}$$

(iv) The inclusion of the conversion relation:

$$\frac{\sigma \in \mathtt{FC}(t) \quad t/\sigma \twoheadrightarrow u \quad t \backslash \sigma \twoheadrightarrow v}{t \twoheadrightarrow \mathbf{let}\, u \,\mathbf{be}\, p \,\mathbf{in}\, v} \; p = \mathbf{bind}(\sigma, t)$$

the presence of pre-congruence rules for each term constructor ensures that parallel expansion is a reflexive relation. The first step towards proving that $\twoheadrightarrow$ satisfies the diamond lemma is to show that the substitution inference rule is an admissible rule in the system without it.

Lemma 4.3.2 *If there is a derivation whose only use of substitution is its last step then there is a derivation of the same rewrite which does not use substitution at all. Thus if there is a derivation $t \twoheadrightarrow t'$ then there is another derivation $t \twoheadrightarrow t'$ which does not involve substitution.*

Proof For the first part of the lemma, let the final substitution be:

$$\frac{u \twoheadrightarrow u' \quad t \twoheadrightarrow t'}{u[t/w] \twoheadrightarrow u'[t'/w]}$$

The proof is by induction on the height of the left-hand derivation $u \twoheadrightarrow u'$. If u is a variable the lemma is trivial to prove, while if the derivation $u \twoheadrightarrow u'$ ends in a pre-congruence, say

$$\frac{\dfrac{u_0 \twoheadrightarrow u_0' \ldots u_n \twoheadrightarrow u_n'}{\mathcal{T}(u_0,\ldots,u_n) \twoheadrightarrow \mathcal{T}(u_0',\ldots,u_n')} \quad t \twoheadrightarrow t'}{\mathcal{T}(u_0,\ldots,u_n)[t/w] \twoheadrightarrow \mathcal{T}(u_0',\ldots,u_n')[t'/w]}$$

then there is an alternative derivation

$$\frac{\dfrac{u_j \twoheadrightarrow u_j' \quad t \twoheadrightarrow t'}{u_j[t/w] \twoheadrightarrow u_j'[t'/w]} \quad \dfrac{}{u_i \twoheadrightarrow u_i'}\; i \neq j}{\mathcal{T}(u_0,\ldots,u_j[t/w],\ldots,u_n) \twoheadrightarrow \mathcal{T}(u_0',\ldots,u_j'[t'/w],\ldots,u_n')}$$

where u_j is the only immediate subterm of u containing w free (we assume w.l.o.g. that $\mathcal{T}$ does not bind any free variables of t and t'). The induction hypothesis may now be applied to the subderivation ending in substitution to obtain a substitution-free derivation. Finally, if the derivation ends

$$\frac{\dfrac{u/\sigma \twoheadrightarrow v' \quad u \backslash \sigma \twoheadrightarrow u'}{u \twoheadrightarrow \mathbf{let}\, v'\, \mathbf{be}\, p\, \mathbf{in}\, u'} \quad t \twoheadrightarrow t'}{u[t/w] \twoheadrightarrow (\mathbf{let}\, v'\, \mathbf{be}\, p\, \mathbf{in}\, u')[t'/w]}$$

with $\sigma \in \mathtt{FC}(u)$, then $\sigma \in \mathtt{FC}(u[t/w])$ and either $w \in \mathtt{FV}(u/\sigma)$ in which case there is the following derivation:

$$\frac{\dfrac{u/\sigma \twoheadrightarrow v' \quad t \twoheadrightarrow t'}{(u[t/w])/\sigma = (u/\sigma)[t/w] \twoheadrightarrow v'[t'/w]} \quad u[t/w] \backslash \sigma = u \backslash \sigma \twoheadrightarrow u'}{u[t/w] \twoheadrightarrow \mathbf{let}\, v'[t'/w]\, \mathbf{be}\, p\, \mathbf{in}\, u'}$$

or, on the other hand, if $w \in \mathtt{FV}(u \setminus \sigma)$, and assuming that w is not bound by p, there is the alternative derivation:

$$\frac{u[t/w]/\sigma = u/\sigma \twoheadrightarrow v' \quad \dfrac{u \setminus \sigma \twoheadrightarrow u' \quad t \twoheadrightarrow t'}{u[t/w] \setminus \sigma = (u \setminus \sigma)[t/w] \twoheadrightarrow u'[t'/w]}}{u[t/w] \twoheadrightarrow \mathbf{let}\, v' \,\mathbf{be}\, p \,\mathbf{in}\, u'[t'/w]}$$

In each case the height of the left-hand derivation of the new substitution is reduced and so can be eliminated.

The second part of the lemma may be proved by induction on the number of substitutions in the derivation and by using the first part of this lemma to remove successive substitutions. □

Lemma 4.3.3 *Parallel conversion is a subrelation of $\Rightarrow_c^*$ and hence satisfies subject reduction.*

Proof Let $t \twoheadrightarrow t'$. Then by lemma 4.3.2 there is a derivation of $t \twoheadrightarrow t'$ which does not use any substitutions and any such rewrite is clearly contained in $\Rightarrow_c^*$. The second part of the lemma is trivial. □

A term is called *small* iff it contains no free conversions, while a conversion $\sigma \in \mathtt{C}(t)$ is called *atomic* iff t/σ is small. The atomic free conversions of a term t, denoted $\mathtt{AFC}(t)$, are the key to proving confluence as they decompose a parallel conversion rewrite into one of the conversion and one of the associated residue. Note that if a term has a free conversion, then it has an atomic free conversion.

Lemma 4.3.4 *Let $t \twoheadrightarrow t'$ and $\sigma \in \mathtt{AFC}(t)$. Then there is a conversion $\sigma' \in \mathtt{FC}(t')$ such that*

$$t \setminus \sigma \twoheadrightarrow t' \setminus \sigma' \text{ and } t/\sigma \twoheadrightarrow t'/\sigma'$$

Proof The proof is by induction on the height of the derivation of $t \twoheadrightarrow t'$. Firstly t cannot be a variable as variables have no conversions. If the last rule was a pre-congruence, then either $\sigma = 0$, in which case set $\sigma' = 0$, or $\sigma^- \in \mathtt{AFC}(t/\sigma_0)$ and so by the induction hypothesis there is a $\sigma' \in \mathtt{FC}(t'/\sigma_0)$ satisfying the required reductions. As the conversion relation introduces no new free variables, $\sigma_0 \cdot \sigma'$ is a free conversion and so satisfies the requirements of the lemma. Finally, if the last rule is of the form:

$$\frac{\tau \in \mathtt{FC}(t) \quad t/\tau \twoheadrightarrow u \quad t \setminus \tau \twoheadrightarrow v}{t \twoheadrightarrow \mathbf{let}\, u\, \mathbf{be}\, p\, \mathbf{in}\, v}$$

then there are three subcases. Firstly if $\tau = \sigma$ then $0 \in \mathtt{FC}(t')$ is the required conversion. Secondly if $\tau < \sigma$, then $\sigma/\tau \in \mathtt{AFC}(t/\tau)$, and so by the induction hypothesis there is a conversion $\sigma' \in \mathtt{FC}(u)$ such that:

$$(t/\tau) \setminus (\sigma/\tau) \twoheadrightarrow u \setminus \sigma' \text{ and } (t/\tau)/(\sigma/\tau) \twoheadrightarrow u/\sigma'$$

Then $0 \cdot \sigma'$ satisfies the requirements of the lemma. As σ is atomic, the final subcase is that τ and σ are disjoint in which case there are descendants $\sigma' \in \mathtt{AFC}(t \setminus \tau)$ and $\tau' \in \mathtt{FC}(t \setminus \sigma)$ indexing the same subterms as their ancestors, and such that $(t \setminus \tau) \setminus \sigma' = (t \setminus \sigma) \setminus \tau'$. By the induction hypothesis there is a conversion $\sigma' \in \mathtt{FC}(t')$ and then $1 \cdot \sigma'$ is the required conversion. The relevant reductions may be deduced from the premises and the equations relating ancestors to descendants, while freeness follows from the conditions on the bindings of conversions in equation 4.4. □

Lemma 4.3.5 *The parallel conversion relation satisfies the diamond property and hence is confluent.*

Proof The proof is by induction on the size of the term at the head of the span. If there is a span $t \twoheadleftarrow z \twoheadrightarrow t'$, then both t and t' must be z and so there is a co-span $t \twoheadrightarrow z \twoheadleftarrow t'$. Given a span from a compound term $\mathcal{T}(t_0, \ldots, t_n)$, there are two possibilities:

(i) If $\mathcal{T}(t_0, \ldots, t_n)$ has no free conversions, then the span is necessarily of the form:

$$\mathcal{T}(t'_0, \ldots, t'_n) \twoheadleftarrow \mathcal{T}(t_0, \ldots, t_n) \twoheadrightarrow \mathcal{T}(t''_0, \ldots, t''_n)$$

where for each i, $t'_i \twoheadleftarrow t_i \twoheadrightarrow t''_i$. The induction hypothesis gives co-spans $t'_i \twoheadrightarrow s_i \twoheadleftarrow t''_i$ and hence an overall co-span exists:

$$\mathcal{T}(t'_0, \ldots, t'_n) \twoheadrightarrow \mathcal{T}(s_0, \ldots, s_n) \twoheadleftarrow \mathcal{T}(t''_0, \ldots, t''_n)$$

(ii) If $\mathcal{T}(t_0, \ldots, t_n)$ has a free conversion, then there is an atomic free conversion $\sigma \in \mathtt{AFC}(\mathcal{T}(t_0, \ldots, t_n))$. By lemma 4.3.4 any span $t' \twoheadleftarrow \mathcal{T}(t_0, \ldots, t_n) \twoheadrightarrow t''$ can then be factorised into two spans:

$$t'/\sigma' \twoheadleftarrow \mathcal{T}(t_0, \ldots, t_n)/\sigma \twoheadrightarrow t''/\sigma'' \text{ and } t' \setminus \sigma' \twoheadleftarrow \mathcal{T}(t_0, \ldots, t_n) \setminus \sigma \twoheadrightarrow t'' \setminus \sigma''$$

where $\sigma' \in \mathtt{FC}(t')$ and $\sigma'' \in \mathtt{FC}(t'')$. The redexes of these spans are of a strictly smaller size and so by the induction hypothesis there are co-spans $t'/\sigma' \twoheadrightarrow r \twoheadleftarrow t''/\sigma''$ and $t' \setminus \sigma' \twoheadrightarrow s \twoheadleftarrow t'' \setminus \sigma''$. Then there is the following co-span:

$$\frac{t'/\sigma' \twoheadrightarrow r \quad t' \setminus \sigma' \twoheadrightarrow s}{t' \twoheadrightarrow \mathbf{let}\, r\, \mathbf{be}\, p\, \mathbf{in}\, s} \qquad\qquad \frac{t''/\sigma'' \twoheadrightarrow r \quad t'' \setminus \sigma'' \twoheadrightarrow s}{t'' \twoheadrightarrow \mathbf{let}\, r\, \mathbf{be}\, p\, \mathbf{in}\, s}$$

where p is the binding of σ, and hence of σ' and σ''.

□

Corollary 4.3.6 *The conversion relation is confluent.*

Proof By lemma 4.3.5, the parallel conversion relation is confluent and by lemma 4.3.3 $\Rightarrow_c \subseteq \twoheadrightarrow \subseteq \Rightarrow_c^*$. Hence the conversion relation is also confluent. □

Quasi-Normal Forms

The conversion relation has been shown to be confluent, but the inherent choice in the order which conversions are expanded means that most terms cannot be rewritten to unique normal forms. However every term can be rewritten to a set of quasi-normal forms which are of the form:

$$\mathbf{let}\, u_1\, \mathbf{be}\, p_1\, \mathbf{in}\, (\mathbf{let}\, u_2\, \mathbf{be}\, p_2\, \mathbf{in}\, (\ldots \mathbf{let}\, u_n\, \mathbf{be}\, p_n\, \mathbf{in}\, t\,))$$

where the terms t and u_i are also quasi-normal forms and contain no free conversions. Quasi-normal forms may be axiomatised as follows: A term is *stable* iff all its conversions are atomic, while a term is *strongly stable* iff all its subterms are strongly stable, the term is stable, and either the term is small or a *let*-expression. We shall prove that the set of quasi-normal forms of the linear conversion relation coincides with the set of strongly stable terms and, by giving an algorithm for the construction of the set of quasi-normal reducts of a term, deduce decidability of the equational theory associated to the linear conversion relation. The crucial property of stable terms is that, unlike small terms, they are preserved by the conversion relation. This may be proved from the following strengthening of lemma 4.3.4.

Lemma 4.3.7 *Let t be stable and $t \twoheadrightarrow t'$. Then there is a bijection $r : \mathtt{FC}(t) \to \mathtt{FC}(t')$ such that*

$$t/\sigma \twoheadrightarrow t'/r(\sigma) \quad \textit{and} \quad t \setminus \sigma \twoheadrightarrow t' \setminus r(\sigma).$$

Thus if t is small and stable, then t' is also small.

Proof As all free conversions are atomic, the function r mapping $\mathtt{FC}(t)$ to $\mathtt{FC}(t')$ and the associated rewrites are exactly those constructed in the proof of lemma 4.3.4. In addition injectivity and surjectivity can be inferred from a close analysis of the proof. To prove the second part of the lemma assume t is small. Then $\mathtt{FC}(t) = \emptyset$ and so by the first half of the lemma $\mathtt{FC}(t') = \emptyset$, i.e. t' is also small. $\Box$

This bijection of conversions is crucial in proving that terms which are strongly stable are exactly the quasi-normal forms of the linear conversion relation. The first half of this containment can be proved immediately:

Lemma 4.3.8 *Let $t \twoheadrightarrow t'$. Then: (i) if t is stable then so is t'; and (ii) if t is strongly stable, then so is t' and there is a rewrite $t' \twoheadrightarrow t$.*

Proof The lemma is proved simultaneously by induction on the derivation of the rewrite $t \twoheadrightarrow t'$. If t is a variable the lemma is easily proved. If the last rule of the derivation is a pre-congruence rule and t is stable, then all the subterms of t' are stable and if $0 \in \mathtt{AFC}(t)$, then by lemma 4.3.7 $0 \in \mathtt{AFC}(t')$ and so t' is stable. Now assume t' is strongly stable. Again by the induction hypothesis all subterms of t' are strongly stable and there are two properties to check. If t is small, then by lemma 4.3.7 t' is also small, while if t is a *let*-expression then clearly so is t'. The rewrite $t' \twoheadrightarrow t$ follows by the induction hypothesis.

Finally, if the rewrite $t \twoheadrightarrow t'$ ends with an expansion:

$$\frac{t/\sigma \twoheadrightarrow u \quad t\backslash\sigma \twoheadrightarrow v}{t \twoheadrightarrow \mathbf{let}\, u\, \mathbf{be}\, p\, \mathbf{in}\, v}$$

then as the redexes in the premises are stable/strongly stable, so are u and v, while by lemma 4.3.7 u is small because t/σ is. Thus if t is stable/strongly stable, so is t'. To construct a rewrite in the reverse direction, note first that because t contains a free conversion, 0 must be an atomic conversion of t, say with binding p. Again by lemma 4.3.7 there is a $\tau \in \mathtt{AFC}(t')$ such that

$$t/0 \twoheadrightarrow t'/\tau \text{ and } t\backslash 0 \twoheadrightarrow t'\backslash\tau$$

and so by the induction hypothesis there are rewrites in the reverse direction. Thus there is also a rewrite $t' \twoheadrightarrow \mathbf{let}\, t/0\, \mathbf{be}\, p\, \mathbf{in}\, t\backslash 0$ and this latter term is of course t.

□

Thus strongly stable terms are $\Rightarrow_c$-quasi-normal forms. The reverse containment requires the construction of the quasi-normal reducts of a term and decidability follows by showing that terms equivalent in the conversion relation have the same set of quasi-normal forms. Given a term t, the set of terms $\mathcal{D}(t)$ is defined as in Table 4-4.

Table 4–4: The Operator $\mathcal{D}$

- If t is a variable, then:

$$\mathcal{D}(t) = \{t\}$$

- If t is not a variable and $\mathtt{FC}(t) \neq \emptyset$, then:

$$\mathcal{D}(t) = \bigcup_{\sigma \in \mathtt{AFC}(t)} \{\mathbf{let}\, u\, \mathbf{be}\, p\, \mathbf{in}\, v \mid u \in \mathcal{D}(t/\sigma), v \in \mathcal{D}(t\backslash\sigma)\}$$

  where p is the binding of σ.

- If t is not a variable and $\mathtt{FC}(t) = \emptyset$, then:

$$\mathcal{D}(t) = \{\mathcal{T}(t'_0, \ldots, t'_n) \mid t = \mathcal{T}(t_0, \ldots, t_n) \text{ and } t'_i \in \mathcal{D}(t_i)\}$$

Lemma 4.3.9 *For all terms t, the set $\mathcal{D}(t)$ is non-empty, finite, computable and if $t' \in \mathcal{D}(t)$ then $t \twoheadrightarrow t'$.*

Proof Trivial induction on the size of t. $\square$

Although members of $\mathcal{D}(t)$ are not necessarily strongly stable, this operator forms the basis of their creation.

Lemma 4.3.10 *The operator $\mathcal{D}$ maps stable terms to strongly stable terms.*

Proof Let t be stable and $t' \in \mathcal{D}(t)$. By lemmas 4.3.8 and 4.3.9 t' is stable, while if $\mathtt{FC}(t) \neq \emptyset$, then t' is a *let*-expression and the subterms of t' are strongly stable by the induction hypothesis. If however $\mathtt{FC}(t) = \emptyset$, then again by the induction hypothesis the subterms of t' are strongly stable and by lemma 4.3.7 t' is small. Thus t' is strongly stable. $\square$

The construction of strongly stable terms is thus reduced to the construction of stable terms, and this is accomplished by recursive application of the operator $\mathcal{D}$ and the rewrite relation $\Rightarrow_\mu$ defined by the commuting conversions in equation 4.5. Strong normalisation of $\Rightarrow_\mu$ follows from the strong normalisation result of the next section, while local confluence is easily proven. Thus every term t has a unique $\Rightarrow_\mu$-normal form which we shall denote $\mu(t)$. Note that if a term is stable it is its own $\Rightarrow_\mu$-normal form. The operators $\mathtt{NF}$ and $\mathtt{NF}^o$ are defined simultaneously in Table 4-5.

Lemma 4.3.11 *If t is a term, then the sets $\mathtt{NF}^o(t)$ and $\mathtt{NF}(t)$ are both non-empty, finite and if t' is a member of either of these sets, then $t \Rightarrow_c^* t'$.*

Proof Trivial induction on t and using lemma 4.3.9. $\square$

We shall prove that $\mathtt{NF}^o$ creates stable terms and use lemma 4.3.10 to prove that $\mathtt{NF}$ creates strongly stable terms. The key is the following lemma:

Lemma 4.3.12 *If u is strongly stable and v is stable, then $\mu(\mathbf{let}\, u \,\mathbf{be}\, p \,\mathbf{in}\, v)$ is stable.*

Table 4–5: The Operators $\mathtt{NF}$ and $\mathtt{NF}^o$

- If t is a variable, then:

$$\mathtt{NF}^o(t) = \{t\}$$

- If $t = \mathbf{let}\, u\, \mathbf{be}\, p\, \mathbf{in}\, v$, then:

$$\mathtt{NF}^o(t) = \{\mu(\mathbf{let}\, \alpha\, \mathbf{be}\, p\, \mathbf{in}\, \alpha'\,) \mid \alpha \in \mathtt{NF}(u) \text{ and } \alpha' \in \mathtt{NF}^o(v)\}$$

- If $t = \mathcal{T}(t_0, \ldots, t_n)$ is any other term, then:

$$\mathtt{NF}^o(t) = \{\mathcal{T}(\alpha_0, \ldots, \alpha_n) \mid \alpha_i \in \mathtt{NF}^o(t_i)\}$$

- The operator $\mathtt{NF}$ is defined as

$$\mathtt{NF}(t) = \bigcup_{t' \in \mathtt{NF}^o(t)} \mathcal{D}(t')$$

Proof The proof is by induction on u. If u contains no free conversions then $\mathbf{let}\, u\, \mathbf{be}\, p\, \mathbf{in}\, v$ is stable and hence its own $\Rightarrow_\mu$-normal form. The other possibility is that u is of the form $\mathbf{let}\, u_0\, \mathbf{be}\, p\, \mathbf{in}\, u_1$ where u_0 is a strongly stable term containing no free conversions, and u_1 is strongly stable. Thus

$$\begin{aligned} \mu(\mathbf{let}\, u\, \mathbf{be}\, p\, \mathbf{in}\, v\,) &= \mu(\mathbf{let}\, u_0\, \mathbf{be}\, p'\, \mathbf{in}\, (\mathbf{let}\, u_1\, \mathbf{be}\, p\, \mathbf{in}\, v\,)\,) \\ &= \mathbf{let}\, u_0\, \mathbf{be}\, p'\, \mathbf{in}\, \mu(\mathbf{let}\, u_1\, \mathbf{be}\, p\, \mathbf{in}\, v\,) \end{aligned}$$

By the induction hypothesis $\mu(\mathbf{let}\, u_1\, \mathbf{be}\, p\, \mathbf{in}\, v\,)$ is stable and hence so is the displayed term. □

Lemma 4.3.13 *If $\alpha \in \mathtt{NF}^o(t)$ then α is stable while if $\alpha' \in \mathtt{NF}(t)$ then α' is strongly stable.*

Proof The lemma is proved simultaneously by induction on term structure. If t is a *let*-expression, say $\mathbf{let}\, u\, \mathbf{be}\, p\, \mathbf{in}\, v$, then

$$\alpha = \mu(\mathbf{let}\, \alpha_0\, \mathbf{be}\, p\, \mathbf{in}\, \alpha_1\,)$$

for some $\alpha_0 \in \mathtt{NF}(u)$ and $\alpha_1 \in \mathtt{NF}^o(v)$. By the induction hypothesis α_0 is strongly stable while α_1 is stable and by lemma 4.3.12 these conditions are sufficient to ensure α is stable. If t is not a *let*-expression then α is calculated by applying $\mathtt{NF}^o$ to the immediate subterms of t and the result then follows by the induction hypothesis. The second part of the lemma then follows from lemma 4.3.10 as $\mathcal{D}$ maps stable terms to strongly stable terms. □

The reverse containment to lemma 4.3.8 can now be proved.

Lemma 4.3.14 *Quasi-normal forms are strongly stable terms.*

Proof Let t be a quasi-normal form and $t' \in \mathtt{NF}(t)$. Then by lemma 4.3.11 $t \Rightarrow_c^* t'$ and, as t is a quasi-normal form, there is a rewrite $t' \Rightarrow_c^* t$. Now by lemma 4.3.13 t' is strongly stable and hence by lemma 4.3.8 so is t. □

We have shown that the quasi-normal forms of the conversion relation coincide with the set of strongly stable terms, and thus that terms of the form $\mathtt{NF}(t)$ are quasi-normal forms. In fact all quasi-normal forms are of this form.

Lemma 4.3.15 *If t is strongly stable then $t \in \mathcal{D}(t)$ and hence $t \in \mathtt{NF}(t)$.*

Proof The two parts of the lemma are proved separately by induction on t. □

Finally, the equational theory generated by the conversion relation can be decided by simply comparing the quasi-normal forms of terms. Note that this proof uses confluence of the conversion relation, a result which cannot be proved independently of decidability in more complex calculi.

Lemma 4.3.16 *Let t be stable and $t \twoheadrightarrow t'$. Then the sets of terms $\mathcal{D}(t)$ and $\mathcal{D}(t')$ are equal.*

Proof The proof is by induction on t. If $\mathtt{FC}(t) = \emptyset$, then $\mathtt{FC}(t') = \emptyset$ and the result follows by the induction hypothesis. If however there is a $\sigma \in \mathtt{AFC}(t)$ such that $u \in \mathcal{D}(t/\sigma)$ and $v \in \mathcal{D}(t \setminus \sigma)$ and $\mathbf{let}\, u\, \mathbf{be}\, p\, \mathbf{in}\, v \in \mathcal{D}(t)$, then by lemma 4.3.7 there is a $\sigma' \in \mathtt{AFC}(t')$ such that $t \setminus \sigma \twoheadrightarrow t' \setminus \sigma'$ and $t/\sigma \twoheadrightarrow t'/\sigma'$. Thus by the induction hypothesis the *let*-expression is also in $\mathcal{D}(t')$. This shows that $\mathcal{D}(t) \subseteq \mathcal{D}(t')$ and, as lemma 4.3.7 speaks of a bijection, the same basic argument can be used to show that $\mathcal{D}(t') \subseteq \mathcal{D}(t)$. □

Corollary 4.3.17 *The equational theory generated by the conversion relation is decidable.*

Proof Decidability is proven by showing that if t and u are $\Rightarrow_c$-equivalent, then these terms have the same set of quasi-normal forms. Any terms $t' \in \mathtt{NF}^o(t)$ and $u' \in \mathtt{NF}^o(u)$ are stable terms which by confluence have a common reduct. Thus by lemma 4.3.16, the sets $\mathcal{D}(t')$ and $\mathcal{D}(u')$ are equal and hence so are $\mathtt{NF}(t)$ and $\mathtt{NF}(u)$. By lemma 4.3.11 these sets are finite and hence the equational theory generated by the conversion relation is decidable. □

4.4 An Extension of β-Reduction

A rewrite relation similar to the restricted rewrite relation of the simply typed λ-calculus is defined which, when taken together with the conversion relation of the previous section, generates the same equational theory as the full expansionary rewrite relation. For each type constructor of the calculus there is a β-redex, a pair of commuting conversions, and limited possibilities for η-expansion. This relation is strongly normalising and confluent and has normal forms which satisfy structural criteria similar to the long $\beta\eta$-normal forms of the simply typed λ-calculus.

Let $\Rightarrow_\beta$ be the least pre-congruence on terms containing the following three β-reductions and six commuting conversions:

$$
\begin{array}{lcl}
(\lambda x.b)a & \Rightarrow & b[a/x] \\
\mathbf{let}\, u \otimes v \,\mathbf{be}\, x \otimes y \,\mathbf{in}\, t & \Rightarrow & t[u/x, v/y] \\
\mathbf{let}\, * \,\mathbf{be}\, * \,\mathbf{in}\, t & \Rightarrow & t \\
(\mathbf{let}\, t \,\mathbf{be}\, p \,\mathbf{in}\, t')(e) & \Rightarrow & \mathbf{let}\, t \,\mathbf{be}\, p \,\mathbf{in}\, t'e \\
\mathbf{let}\, (\mathbf{let}\, t \,\mathbf{be}\, p \,\mathbf{in}\, t') \,\mathbf{be}\, p' \,\mathbf{in}\, u & \Rightarrow & \mathbf{let}\, t \,\mathbf{be}\, p \,\mathbf{in}\, (\mathbf{let}\, t' \,\mathbf{be}\, p' \,\mathbf{in}\, u)
\end{array}
$$

where p, p' are the bindings of the respective conversions, and the obvious restrictions concerning bound variables are imposed to prevent variable capture. Note that, although it may seem that there only two commuting conversions, each of p and p' can each be associated to terms of tensor or unit type, and this gives a total of six commuting conversions. To this relation limited possibilities for

η-conversion are added. The η-expansion of a term depends on its type and is defined as follows:

$$\eta(t) = \begin{cases} t & \text{if } t \text{ is of base type} \\ \lambda x.tx & \text{if } t\!:\!A \rightarrow B \\ \mathbf{let}\, t\, \mathbf{be}\, x \otimes y\, \mathbf{in}\, x \otimes y & \text{if } t\!:\!A \otimes B \\ \mathbf{let}\, t\, \mathbf{be}\, *\ \mathbf{in}\, * & \text{if } t\!:\!I \end{cases}$$

These η-expansions can be used to convert subterms of the redex into negatively occurring subterms of the reduct. If these subterms are of tensor or unit type, this creates conversions which may then be expanded by the conversion relation studied in the previous section. Thus when taken together, these η-expansions and the conversion relation have the full power of the η-rewrite rules given in Table 4-2.

Uncontrolled η-expansion is clearly not strongly normalising and so restrictions must be imposed on the scope of these rewrite rules, and in particular the expansions appearing in the triangle laws must be prohibited. As in the case of final type constructors, the triangle laws for the tensor and unit type constructors assert that expanding introduction terms and subterms which occur negatively cause looping reductions. However, prohibiting these expansions is as yet insufficient to obtain a strongly normalising relation as the η-expansion of *let*-expressions creates commuting conversions from which a reduction sequence to the original term may exist, e.g.

$$\begin{array}{lll} \mathbf{let}\, z\, \mathbf{be}\, x \otimes y\, \mathbf{in}\, x \otimes y & \Rightarrow_\eta & \eta(\mathbf{let}\, z\, \mathbf{be}\, x \otimes y\, \mathbf{in}\, x \otimes y\,) \\ & \Rightarrow_\beta & \mathbf{let}\, z\, \mathbf{be}\, x \otimes y\, \mathbf{in}\, \eta(x \otimes y) \\ & \Rightarrow_\beta & \mathbf{let}\, z\, \mathbf{be}\, x \otimes y\, \mathbf{in}\, x \otimes y \end{array} \tag{4.6}$$

Such terms are called *quasi-introduction* terms and the class of terms which may not be expanded must be enlarged to include them. The *quasi-introduction* terms of tensor type are defined as follows:

$$q := t \otimes t' \mid \mathbf{let}\, u\, \mathbf{be}\, p\, \mathbf{in}\, q$$

where u, t, t' are arbitrary terms and p is an appropriate binding. The *quasi-introduction* terms of unit type are defined as follows:

$$q := * \mid \mathbf{let}\, u\, \mathbf{be}\, p\, \mathbf{in}\, q$$

where again u is an arbitrary term and p is some binding. As the η-expansion of terms such as $\mathbf{let}\, u\, \mathbf{be}\, p\, \mathbf{in}\, \lambda x.t$ does not create a reduction loop such as in equation 4.6, the expansion of such terms is permitted, and so quasi-introduction terms of function type are defined to be exactly the introduction terms.

A term is called *expandable* iff it is neither a quasi-introduction term nor of base type. As with the simply typed λ-calculus, the context sensitive restrictions on expansion are encoded via a subrelation $\Rightarrow_{\mathcal{I}}$ of $\Rightarrow_{\mathcal{F}}$ which is guaranteed not to be a top-level expansion and so a negatively occurring subterm may be safely $\Rightarrow_{\mathcal{I}}$-rewritten when a $\Rightarrow_{\mathcal{F}}$-reduction may create a reduction loop. These rewrite relations are formally defined in Table 4-6, where $\Rightarrow_{\mathcal{I}\cup\mathcal{F}}$ and $\Rightarrow_{\mathcal{I}\cap\mathcal{F}}$ are the union and intersection of $\Rightarrow_{\mathcal{I}}$ and $\Rightarrow_{\mathcal{F}}$ respectively. As intended $\Rightarrow_{\mathcal{I}}$ is the subrelation

Table 4–6: The Restricted Linear Rewrite Relation

$$\frac{t \Rightarrow_\beta t'}{t \Rightarrow_{\mathcal{I}\cap\mathcal{F}} t'} \qquad \frac{u \Rightarrow_{\mathcal{I}\cup\mathcal{F}} u'}{u \otimes v \Rightarrow_{\mathcal{I}\cap\mathcal{F}} u' \otimes v} \qquad \frac{u \Rightarrow_{\mathcal{I}\cup\mathcal{F}} u'}{\mathbf{let}\, t\, \mathbf{be}\, p\, \mathbf{in}\, u \Rightarrow_{\mathcal{I}\cap\mathcal{F}} \mathbf{let}\, t\, \mathbf{be}\, p\, \mathbf{in}\, u'}$$

$$\frac{t \text{ expandable}}{t \Rightarrow_{\mathcal{F}} \eta(t)} \qquad \frac{v \Rightarrow_{\mathcal{I}\cup\mathcal{F}} v'}{u \otimes v \Rightarrow_{\mathcal{I}\cap\mathcal{F}} u \otimes v'} \qquad \frac{t \Rightarrow_{\mathcal{I}} t'}{\mathbf{let}\, t\, \mathbf{be}\, p\, \mathbf{in}\, u \Rightarrow_{\mathcal{I}\cap\mathcal{F}} \mathbf{let}\, t'\, \mathbf{be}\, p\, \mathbf{in}\, u}$$

$$\frac{\phi \Rightarrow_{\mathcal{I}} \phi'}{\phi\psi \Rightarrow_{\mathcal{I}\cap\mathcal{F}} \phi'\psi} \qquad \frac{\psi \Rightarrow_{\mathcal{I}\cup\mathcal{F}} \psi'}{\phi\psi \Rightarrow_{\mathcal{I}\cap\mathcal{F}} \phi\psi'} \qquad \frac{t \Rightarrow_{\mathcal{I}\cup\mathcal{F}} t'}{\lambda x.t \Rightarrow_{\mathcal{I}\cap\mathcal{F}} \lambda x.t'}$$

of $\Rightarrow_{\mathcal{F}}$ obtained by removing all basic expansions, i.e.

$$t \Rightarrow_{\mathcal{F}} t' \text{ iff } t \Rightarrow_{\mathcal{I}} t' \text{ or } t' = \eta(t) \text{ and } t \text{ is expandable}$$

Both implications are easily proved by induction on the rewrite in question. Quasi-introduction terms satisfy several closure properties which are relevant to both their use in proof of strong normalisation of $\Rightarrow_{\mathcal{F}}$ and in characterising $\Rightarrow_{\mathcal{F}}$-normal forms.

Lemma 4.4.1 *If t is a quasi-introduction term, then so are all $\Rightarrow_{\mathcal{F}}$-reducts of t.*

Proof First prove if t is a quasi-introduction term and the substitution of u for x in t is linear, then $t[u/x]$ is also a quasi-introduction term. The lemma then follows by induction on the term t. □

A term is called a *linear long $\beta\eta$-normal form* iff it is a β-normal form and each of its subterms is either of base type, occurs negatively or is a quasi-introduction term.

Lemma 4.4.2 *A term t is an $\Rightarrow_{\mathcal{F}}$-normal form iff it is a linear long $\beta\eta$-normal form.*

Proof The proof is the obvious generalisation of lemma 3.3.16. □

4.4.1 Substitutivity, Local Confluence and Normalisation

The rewrite relations $\Rightarrow_{\mathcal{I}}$ and $\Rightarrow_{\mathcal{F}}$ are not pre-congruences and this complicates traditional proofs of strong normalisation and confluence. In particular the class of non-expandable terms has been increased to include not just introduction terms, but also certain elimination terms, and this leads to further possibilities for the failure of substitutivity. As with the last chapter these instances are characterised and, by taking advantage of the linearity of this calculus, alternative reduction sequences are given.

Lemma 4.4.3 *Let t, t', u and u' be terms such that $t \Rightarrow_{\mathcal{R}} t'$ and $u \Rightarrow_{\mathcal{R}} u'$, where $\mathcal{R} \in \{\mathcal{I}, \mathcal{F}\}$.*

- *There is a rewrite $t[u/x] \Rightarrow_{\mathcal{R}} t'[u/x]$ unless the rewrite $t \Rightarrow t'$ expands the only free occurrence of x in t and u is a quasi-introduction term. In this case $t'[u/x] \Rightarrow_{\beta}^{*} t[u/x]$.*

- *There is a rewrite $t[u/x] \Rightarrow_{\mathcal{R}} t[u'/x]$ unless the only free occurrence of x in t occurs negatively and $u' = \eta(u)$. In this case there is a reduction $t[u'/x] \Rightarrow_{\beta}^{*} t[u/x]$.*

Proof The two parts of the lemma are proved separately by induction on t. □

The next step would be to hypothesise that both the relations $\Rightarrow_{\mathcal{I}}$ and $\Rightarrow_{\mathcal{F}}$ are locally confluent and attempt to prove this using the substitutivity results of lemma

4.4.3. However, this is not possible as the following counterexamples show:

$$
\begin{tikzcd}
(\lambda x.t)u \arrow[r, "\mathcal{I}"] \arrow[d, "\mathcal{I}"'] & (\lambda x.\eta(t))u \arrow[d, "\mathcal{I}"] \\
t[u/x] \arrow[r, "\mathcal{F}"'] & \eta(t)[u/x]
\end{tikzcd}
\quad \text{and} \quad
\begin{tikzcd}
\mathbf{let} * \mathbf{be} * \mathbf{in}\, t \arrow[r, "\mathcal{I}"] \arrow[d, "\mathcal{I}"'] & t \arrow[d, "\mathcal{F}"] \\
\mathbf{let} * \mathbf{be} * \mathbf{in}\, \eta(t) \arrow[r, "\mathcal{I}"'] & \eta(t)
\end{tikzcd}
$$

and similarly for the case of a $\beta_{\otimes}$-contraction. Each of these co-spans contains a $\Rightarrow_{\mathcal{F}}$-rewrite which is not a member of $\Rightarrow_{\mathcal{I}}$ and hence $\Rightarrow_{\mathcal{I}}$ is not locally confluent. Fortunately, local confluence of $\Rightarrow_{\mathcal{F}}$ may be proved using the same techniques as were used for the restricted rewrite relation of simply typed λ-calculus.

Lemma 4.4.4 *The relation $\Rightarrow_{\mathcal{F}}$ is locally confluent and given any span $t \Rightarrow_{\mathcal{I}} t_i$ (where $i = 1, 2$), there is a term t' such that either $t_1 \Rightarrow^*_{\mathcal{I}} t'$ or $t_1 \Rightarrow_{\mathcal{F}} t'$ and similarly either $t_2 \Rightarrow^*_{\mathcal{I}} t'$ or $t_2 \Rightarrow_{\mathcal{F}} t'$.*

Proof The proof is by induction on term structure and follows the same pattern as that for the simply typed λ-calculus given in lemma 3.3.3. □

Although the definition of the rewrite relation $\Rightarrow_{\mathcal{F}}$ mirrors its equivalent in the previous chapter, the presence of initial type constructors and especially their elimination rules complicates the proof of strong normalisation. In particular, unlike λ-abstractions where the variable abstracted is of "lower" type than that of the λ-abstraction, the variables bound by a ⊗-elimination are of arbitrarily complex type. This causes various technical problems in the proof of normalisation (explained in the next chapter) and so we prefer to state strong normalisation and claim that the methods developed in the next chapter are sufficient to also prove the result for $\Rightarrow_{\mathcal{F}}$.

Theorem 4.4.5 *The relation $\Rightarrow_{\mathcal{F}}$ is strongly normalising and thus confluent.*

Proof The proof is an adaptation of the normalisation proof contained in the next chapter. Confluence then follows from local confluence. □

4.5 Decidability of $\beta\eta$-Equality

The expansionary rewrite relation defined at the beginning of this chapter has been decomposed into the strongly normalising and confluent relation $\Rightarrow_{\mathcal{F}}$ and the confluent conversion relation $\Rightarrow_c$ whose equational theory has been shown to be decidable. The rest of this chapter proves that $\beta\eta$-equality is itself decidable by embedding it into the conversion relation, thus allowing the confluence and decidability results already proven in lemma 4.3.17 to be lifted to the full expansionary rewrite relation.

More explicitly we prove that if $t =_{\beta\eta} t'$ then the $\Rightarrow_{\mathcal{F}}$-normal forms of t and t' are equivalent in the conversion relation. Because $\beta\eta$-equality is the equational theory generated by the union of $\Rightarrow_{\mathcal{F}}$ and $\twoheadrightarrow$, this is equivalent to proving that if $t \twoheadrightarrow t'$ then the $\Rightarrow_{\mathcal{F}}$-normal forms of t and t' are equivalent in the conversion relation. The easiest proof strategy would be to consider the β-reductions of a term in isolation from the possibilities for η-expansion that exist within the term, i.e. to prove that η-expansion preserves β-normal forms or vice versa that β-reduction preserves η-normal forms, and that if $t \twoheadrightarrow t'$ then

$$\beta(t) \twoheadrightarrow \beta(t') \quad \text{and} \quad \eta(t) \twoheadrightarrow \eta(t')$$

where $\beta(t)$ denotes the β-normal form of t and $\eta(t)$ denotes the η-normal form of t. Unfortunately, unlike the restricted expansions of the simply typed λ-calculus, the restricted η-expansions contained in $\Rightarrow_{\mathcal{F}}$ do not form a confluent relation, e.g. the reducts of the span

$$\eta(\mathbf{let} * \mathbf{be} * \mathbf{in}\, x\,) \Leftarrow \mathbf{let} * \mathbf{be} * \mathbf{in}\, x \;\; \Rightarrow \mathbf{let} * \mathbf{be} * \mathbf{in}\, \eta(x)$$

cannot be rewritten to the same term by restricted η-expansions alone. Our solution is to pick a particular η-normal form for each term by increasing the class of non-expandable forms to prevent the η-expansion of all *let*-expressions. One side-effect of this is that η-expansion will now preserve β-normal forms and so we have achieved the aim of considering the effect of β-reduction on the parallel conversion in separation from the effect of η-expansion. In addition the conversion relation can map positive occurrences to negative occurrences, e.g. in the following rewrite the subterm x occurs positively in the redex but negatively in the reduct:

$$(\mathbf{let}\, z\, \mathbf{be}\, *\, \mathbf{in}\, x\,)y \twoheadrightarrow \mathbf{let}\, z\, \mathbf{be}\, *\, \mathbf{in}\, xy$$

Such reductions cannot be lifted to their η-normal forms, i.e. there is no reduction:

$$(\mathbf{let}\, z\, \mathbf{be}\, *\, \mathbf{in}\, \eta(x)\,)y \not\twoheadrightarrow \mathbf{let}\, z\, \mathbf{be}\, *\, \mathbf{in}\, xy$$

The solution to this problem is to increase those occurrences in a term which may be $\Rightarrow_{\mathcal{I}}$-rewritten but not $\Rightarrow_{\mathcal{F}}$-rewritten. Rather than present another series of relations, and associated confluence proofs, we define the fully η-expanded form of a term directly via the simultaneous definition of a pair of functions $\eta\mathcal{I}$ and $\eta\mathcal{F}$. The first step towards doing this is to define a function Δ^m which maps a variable to its $\Rightarrow_{\mathcal{F}}$-normal form. Note that we are assuming w.l.o.g. that variables, and hence terms, have unique types.

$$\begin{array}{lcll}
\Delta^m(z) & = & z & \text{if } z \text{ has base type} \\
\Delta^m(z) & = & * & \text{if } z \text{ has unit type} \\
\Delta^m(z) & = & \lambda x.\Delta^m(y)[z\Delta^m(x)/y] & \text{if } z \text{ has function type} \\
\Delta^m(z) & = & \mathbf{let}\, z\, \mathbf{be}\, x \otimes y\, \mathbf{in}\, \Delta^m(x) \otimes \Delta^m(y) & \text{if } z \text{ has tensor type}
\end{array}$$

Note that the superscript m in the function Δ^m is required to maintain consistency with the notation of the next chapter, where $\Delta(z)$ is defined to be the set of *all* $\Rightarrow_{\mathcal{F}}$-reducts of z.

Lemma 4.5.1 *The term $\Delta^m(z)$ is the $\Rightarrow_{\mathcal{F}}$-normal form of z.*

Proof The lemma is proved by induction over the type of z and follows the proof of lemma 5.4.6. □

The two functions $\eta\mathcal{I}$ and $\eta\mathcal{F}$, which map terms to terms, by induction on term size:

$$\begin{array}{rcl}
\eta\mathcal{I}(x) & = & x \\
\eta\mathcal{I}(u \otimes v) & = & \eta\mathcal{F}(u) \otimes \eta\mathcal{F}(v) \\
\eta\mathcal{I}(\mathbf{let}\, t\, \mathbf{be}\, p\, \mathbf{in}\, u\,) & = & \mathbf{let}\, \eta\mathcal{I}(t)\, \mathbf{be}\, p\, \mathbf{in}\, \eta\mathcal{I}(u)
\end{array}$$

$$\begin{aligned}\eta\mathcal{I}(uv) &= \eta\mathcal{I}(u)\eta\mathcal{F}(v)\\ \eta\mathcal{I}(\lambda x.t) &= \lambda x.\eta\mathcal{F}(t)\end{aligned}$$

and

$$\eta\mathcal{F}(t) = \begin{cases} \Delta^m(z)[\eta\mathcal{I}(t)/z] & \text{if } t \text{ is a variable/application} \\ \mathbf{let}\,\eta\mathcal{I}(u)\,\mathbf{be}\,p\,\mathbf{in}\,\eta\mathcal{F}(v) & \text{if } t \text{ is } \mathbf{let}\,u\,\mathbf{be}\,p\,\mathbf{in}\,v \\ t & \text{otherwise} \end{cases}$$

where in the definition of $\eta\mathcal{F}(t)$, z is some variable of the same type as t. To maintain the strength of the equational theory the η-expansion of *let*-expressions of function type is simulated by the rewrite relation $\Rightarrow_\delta$ which is defined to be the least pre-congruence generated by the redex

$$\mathbf{let}\,t\,\mathbf{be}\,p\,\mathbf{in}\,\lambda x.u \Rightarrow_\delta \lambda x.\mathbf{let}\,t\,\mathbf{be}\,p\,\mathbf{in}\,u \tag{4.7}$$

where $x \notin \mathtt{FV}(t)$ and x is not bound by p. The need for a new redex form is regrettable but unavoidable, because if the η-expansion of *let*-expressions of function type were permitted, then a rewrite

$$z(\mathbf{let}\,x\,\mathbf{be}\,*\,\mathbf{in}\,\lambda y.t\,) \twoheadrightarrow \mathbf{let}\,x\,\mathbf{be}\,*\,\mathbf{in}\,z\lambda y.t$$

would not induce a $\twoheadrightarrow$-rewrite of their η-normal forms, i.e. there would not be a $\twoheadrightarrow$-rewrite of the form:

$$z(\lambda w.(\mathbf{let}\,x\,\mathbf{be}\,*\,\mathbf{in}\,\lambda y.t\,)w) \not\twoheadrightarrow \mathbf{let}\,x\,\mathbf{be}\,*\,\mathbf{in}\,z\lambda y.t$$

The rewrite relation $\Rightarrow_\delta$ is strongly normalising and confluent and thus has unique normal forms which form part of a normalisation strategy for the calculation of $\Rightarrow_\mathcal{F}$-normal forms. Notice how this normalisation strategy gradually constructs $\Rightarrow_\mathcal{F}$-normal forms by ensuring that terms satisfy progressively more of the properties of linear long $\beta\eta$-normal forms as used in lemma 4.4.2.

Lemma 4.5.2 *The $\Rightarrow_\mathcal{F}$-normal form of t may be calculated by (i) calculating the β-normal form of t; (ii) applying the function $\eta\mathcal{F}$; and (iii) calculating the $\Rightarrow_\delta$-normal form of the result.*

Proof Let $\sharp t$ denote the result of applying the algorithm in the lemma to t. The lemma is proved by showing that $\sharp t$ is an $\Rightarrow_{\mathcal{F}}$-normal form and is also $\Rightarrow_{\mathcal{F}}$-equivalent to t. Firstly $\sharp t$ is a β-normal form as both $\eta\mathcal{F}$ and $\Rightarrow_\delta$ preserve β-normal forms, while induction on the structure of t is used to show that if t is a β-normal form, then $\sharp t$ is an $\Rightarrow_{\mathcal{F}}$-normal form. Secondly, because there is always a reduction sequence $t \Rightarrow_{\mathcal{F}} \eta\mathcal{F}(t)$ and a term is always $\Rightarrow_{\mathcal{F}}$-equivalent to its $\Rightarrow_\delta$-reducts, we may conclude that $t =_{\mathcal{F}} \sharp t$. $\square$

4.5.1 Embedding $\Rightarrow_c$ into β-Normal Forms

In this section $\beta(t)$ is used to denote the $\Rightarrow_\beta$-normal form of t. We show that for any rewrite $t \twoheadrightarrow t'$ there is also a rewrite $\beta(t) \twoheadrightarrow \beta(t')$. This is proven by implicitly using the structure of a conversion rewrite to construct a mapping between the β-redexes of the terms involved. However, the conversion relation may introduce new commuting conversions, e.g.

$$\frac{\lambda z.\mathbf{let} * \mathbf{be} * \mathbf{in}\, z \twoheadrightarrow \mathbf{let} * \mathbf{be} * \mathbf{in}\, \lambda z.z \qquad v \twoheadrightarrow v}{(\lambda z.\mathbf{let} * \mathbf{be} * \mathbf{in}\, z)v \twoheadrightarrow (\mathbf{let} * \mathbf{be} * \mathbf{in}\, \lambda z.z)v} \tag{4.8}$$

and these new commuting conversions must be contracted before the $\beta_{\rightarrow}$-redex in the reduct corresponding to the $\beta_{\rightarrow}$-redex in the original term can be contracted. The introduction of new commuting conversions is caused by $\twoheadrightarrow$-rewrites which do not expand conversions as much as possible. These rewrites come in two varieties: (i) if the last rule of the $\twoheadrightarrow$-rewrite is a pre-congruence associated to an elimination rule and the left-hand branch ends in an expansion; or (ii) if the last rule of the $\twoheadrightarrow$-rewrite is an expansion whose left-branch also ends in an expansion. For example, equation 4.8 is an example of the former possibility.

Reductions such as these may be prohibited by placing restrictions on the use of the expansion clause in the definition of $\twoheadrightarrow$ which resemble those employed in the definition of $\Rightarrow_{\mathcal{F}}$. A rewrite $t \twoheadrightarrow t'$ is said to be *full* iff it is generated from the inference rules in Table 4-7, where $t \twoheadrightarrow_c t'$ iff $t \twoheadrightarrow_f t'$ and the last rule of this derivation is not an expansion; in the last clause the same restrictions on bound variables are imposed as in the definition of the conversion relation.

Table 4–7: The Full Parallel Conversion Relation

(i) The identity:

$$\frac{}{t \twoheadrightarrow_f t}$$

(ii) For each introduction term constructor:

$$\frac{u_0 \twoheadrightarrow_f u'_0 \quad \ldots \quad u_n \twoheadrightarrow_f u'_n}{\mathcal{T}(u_0, \ldots, u_n) \twoheadrightarrow_f \mathcal{T}(u'_0, \ldots, u'_n)}$$

(iii) For each elimination term constructor:

$$\frac{u_0 \twoheadrightarrow_c u'_0 \quad u_i \twoheadrightarrow_f u'_i \ \ (i \neq 0)}{\mathcal{T}(u_0, \ldots, u_n) \twoheadrightarrow_f \mathcal{T}(u'_0, \ldots, u'_n)}$$

where u_0 is the negatively occurring immediate subterm of $\mathcal{T}(u_0, \ldots, u_n)$.

(iv) The inclusion of the conversion relation:

$$\frac{\sigma \in \mathtt{FC}(t) \quad t/\sigma \twoheadrightarrow_c u \quad t \backslash \sigma \twoheadrightarrow_f v}{t \twoheadrightarrow_f \mathbf{let}\, u\, \mathbf{be}\, p\, \mathbf{in}\, v}$$

where p is the binding of σ.

A quick induction shows that the restrictions imposed ensure that $\twoheadrightarrow_f$ preserves β-normal forms, i.e. if t is a β-normal form and $t \twoheadrightarrow_f t'$ then t' is also a β-normal form. In addition once any commuting conversions created by a $\twoheadrightarrow$-rewrite have been contracted, the reduct may also be obtained by a full $\twoheadrightarrow_f$-rewrite. This allows us to ignore the creation of new commuting conversions by $\twoheadrightarrow$ when proving the embedding lemma.

Lemma 4.5.3 *Let $t \twoheadrightarrow t'$. Then there is a term t'' such that $t' \Rightarrow^*_\beta t''$ and $t \twoheadrightarrow_f t''$.*

$$\begin{tikzcd}
t \arrow[r, two heads] \arrow[d, two heads, "f"'] & t' \arrow[dl, "\beta^*"] \\
t'' &
\end{tikzcd}$$

Proof Although it may seem that the lemma is proved by induction on the derivation of the rewrite $t \twoheadrightarrow t'$, the induction rank actually used is the size of t.

If the rewrite is of the form:

$$\frac{t/\sigma \twoheadrightarrow u' \quad t\setminus\sigma \twoheadrightarrow t'}{t \twoheadrightarrow \mathbf{let}\, u'\, \mathbf{be}\, p\, \mathbf{in}\, t'}$$

then apply the induction hypothesis to the left hand branch to obtain a full reduction $t/\sigma \twoheadrightarrow_f u''$. If this derivation does not end in an expansion apply the induction hypothesis to obtain a rewrite $t\setminus\sigma \twoheadrightarrow_f t''$ and a full rewrite from t may then be constructed to $\mathbf{let}\, u''\, \mathbf{be}\, p\, \mathbf{in}\, t''$. If, however, on applying the induction hypothesis to the left hand branch the resulting rewrite ends in an expansion, say of a free conversion $\tau \in \mathtt{FC}(t/\sigma)$

$$\frac{\dfrac{(t/\sigma)/\tau \twoheadrightarrow_c u_0 \quad (t/\sigma)\setminus\tau \twoheadrightarrow_f u_1}{t/\sigma \twoheadrightarrow_f \mathbf{let}\, u_0\, \mathbf{be}\, p'\, \mathbf{in}\, u_1} \qquad t\setminus\sigma \twoheadrightarrow t'}{t \twoheadrightarrow \mathbf{let}\,(\mathbf{let}\, u_0\, \mathbf{be}\, p'\, \mathbf{in}\, u_1\,)\, \mathbf{be}\, p\, \mathbf{in}\, t'}$$

then the commuting conversion is contracted and an alternative derivation formed by expanding the conversion $\sigma.\tau \in \mathtt{FC}(t)$:

$$\frac{t/(\sigma.\tau) = (t/\sigma)/\tau \twoheadrightarrow_c u_0 \qquad \dfrac{(t\setminus\sigma.\tau)/\sigma = (t/\sigma)\setminus\tau \twoheadrightarrow u_1 \quad (t/\sigma.\tau)\setminus\sigma = t\setminus\sigma \twoheadrightarrow t'}{t\setminus\sigma.\tau \twoheadrightarrow \mathbf{let}\, u_1\, \mathbf{be}\, p\, \mathbf{in}\, t'}}{t \twoheadrightarrow \mathbf{let}\, u_0\, \mathbf{be}\, p'\, \mathbf{in}\,(\mathbf{let}\, u_1\, \mathbf{be}\, p\, \mathbf{in}\, t'\,)}$$

Now use the induction hypothesis again to replace the right hand branch by a full derivation, making the resultant derivation full. Note that if induction on the height of the rewrite were being used, this step would fail as the induction rank has not necessarily been reduced. The required β-rewrites are clear.

Next, a derivation ending in an introduction pre-congruence rule may be converted to the required form by immediate application of the induction hypothesis. Finally, consider a derivation ending in an elimination pre-congruence, e.g. $uv \twoheadrightarrow u'v'$. Again apply the induction hypothesis to the major sub-derivation to obtain a rewrite $u \twoheadrightarrow_f u''$ where u'' is a β-reduct of u'. If this derivation does not end in an expansion, then apply the induction hypothesis to the other branches, while if it is of the form:

$$\frac{u/\sigma \twoheadrightarrow_c u_1 \quad u\setminus\sigma \twoheadrightarrow_f u_2}{u \twoheadrightarrow_f \mathbf{let}\, u_1\, \mathbf{be}\, p\, \mathbf{in}\, u_2}$$

then the resulting commuting conversion may be contracted and the following rewrite given:

$$\frac{(uv)/(0\cdot\sigma) \twoheadrightarrow_c u_1 \quad uv \setminus 0\cdot\sigma \twoheadrightarrow u_2 v'}{uv \twoheadrightarrow \mathbf{let}\, u_1\, \mathbf{be}\, p\, \mathbf{in}\, u_2 v'}$$

Finally, apply the induction hypothesis to the other branches. □

Lemma 4.5.4 *If $t \twoheadrightarrow_f t'$, then $\beta(t) \twoheadrightarrow_f \beta(t')$. Thus if $t \twoheadrightarrow t'$, then $\beta(t) \twoheadrightarrow \beta(t')$.*

Proof The first part of the lemma is proved by induction on firstly the β-normalisation rank of t and then the derivation of the $\twoheadrightarrow$-rewrite. If t is actually a β-normal form then so is t' and the lemma is trivial. If t is not a β-normal form then consider a rewrite of the form:

$$\frac{t/\sigma \twoheadrightarrow_c u \quad t\setminus\sigma \twoheadrightarrow_f v}{t \twoheadrightarrow_f \mathbf{let}\, u\, \mathbf{be}\, x\otimes y\, \mathbf{in}\, v}$$

If t/σ is not a β-normal form then there is a rewrite

$$\frac{\beta(t/\sigma) \twoheadrightarrow_f \beta(u) \quad t\setminus\sigma \twoheadrightarrow_f v}{t[\sigma\leftarrow\beta(t/\sigma)] \twoheadrightarrow \mathbf{let}\, \beta(u)\, \mathbf{be}\, x\otimes y\, \mathbf{in}\, v}$$

where $\beta(t/\sigma) \twoheadrightarrow_f \beta(u)$ follows by the induction hypothesis. Now by lemma 4.5.3 this rewrite can be extended to a full rewrite of lower rank and the lemma is then established by the induction hypothesis. If t/σ is a β-normal form and an introduction term, say $r\otimes s$ then, by fullness, u must be of the form $r'\otimes s'$ where $r \twoheadrightarrow_f r'$ and $s \twoheadrightarrow_f s'$. Thus we may contract the redex in t formed by the conversion σ and also its descendant

$$\begin{tikzcd}
t \arrow[r, twoheadrightarrow, "f"'] \arrow[d, "\beta"'] & \mathbf{let}\, r'\otimes s'\, \mathbf{be}\, x\otimes y\, \mathbf{in}\, v \arrow[d, "\beta"] \\
(t\setminus\sigma)[r/x, s/y] \arrow[r, twoheadrightarrow] & v[r'/x, s'/y]
\end{tikzcd}$$

where the bottom rewrite follows from the substitution closure of $\twoheadrightarrow$. This reduction is again extended to a full rewrite and then the induction hypothesis is applied. If t/σ is a *let*-expression a similar argument may be used, namely reduce the associated commuting conversion in t and at the top level of t', extend the resulting rewrite to a full rewrite and apply the induction hypothesis. Finally, if t/σ is a β-normal form, but neither an introduction term nor a *let*-expression, then an inductive argument proves that there is a conversion $\beta(\sigma) \in \mathtt{FC}(\beta(t))$ such that $\beta(t)/\beta(\sigma) = t/\sigma$ and $\beta(t) \setminus \beta(\sigma) = \beta(t \setminus \sigma)$. Thus

$$\frac{t/\sigma \twoheadrightarrow_f \beta(u) \quad \beta(t \setminus \sigma) \twoheadrightarrow_f \beta(v)}{\beta(t) \twoheadrightarrow \mathbf{let}\, \beta(u)\, \mathbf{be}\, x \otimes y\, \mathbf{in}\, \beta(v)}$$

where the induction hypothesis is used to derive the right-hand branch. Again this rewrite is extended to a full rewrite and the induction hypothesis applied.

If the last rule of the derivation $t \twoheadrightarrow_f t'$ is an introduction pre-congruence the lemma follows trivially by the induction hypothesis. For an elimination pre-congruence:

$$\frac{u_0 \twoheadrightarrow_c u_0' \quad u_i \twoheadrightarrow u_i' (i \neq 0)}{\mathcal{T}(u_0, \ldots, u_n) \twoheadrightarrow_f \mathcal{T}(u_0', \ldots, u_n')}$$

If there is a u_i which is not a β-normal form, by the induction hypothesis there is a rewrite

$$\frac{\beta(u_0) \twoheadrightarrow \beta(u_0'), \ldots, \beta(u_n) \twoheadrightarrow \beta(u_n')}{\mathcal{T}(\beta(u_0), \ldots, \beta(u_n)) \twoheadrightarrow \mathcal{T}(\beta(u_0'), \ldots, \beta(u_n'))}$$

which can be extended to a full rewrite and then the induction hypothesis applied once more. If however there is no such subterm, the only β-redexes in t are top level and by fullness these are preserved. Thus we need check only the nine basic β-reductions e.g. if $(\lambda x.u)v \twoheadrightarrow_f (\lambda x.u')v'$, then $u[v/x] \twoheadrightarrow u'[v'/x]$. As before, extend this to a full rewrite and apply the induction hypothesis to prove the lemma.

For the second part of the lemma consider a rewrite $t \twoheadrightarrow t'$. Then by lemma 4.5.3 there is a t'' such that $t \twoheadrightarrow_f t''$ and $t' \Rightarrow_\beta^* t''$. Now apply the first half of the lemma.

□

4.5.2 Embedding $\Rightarrow_c$ into $\Rightarrow_{\mathcal{F}}$-Normal Forms

The second part of the embedding theorem concerns the interaction between the conversion relation and the limited possibilities for η-expansion given by the functions $\eta\mathcal{F}$ and $\eta\mathcal{I}$. The key lemma is the following use of a conversion to factorise the construction of η-normal forms.

Lemma 4.5.5 *Given a conversion* $\sigma \in \mathtt{FC}(t)$ *there is another conversion* $\sigma' \in \mathtt{FC}(\eta\mathcal{R}(t))$ *such that:*

$$\eta\mathcal{R}(t) \setminus \sigma' = \eta\mathcal{R}(t \setminus \sigma) \textit{ and } \eta\mathcal{R}(t)/\sigma' = \eta\mathcal{I}(t/\sigma)$$

where $\mathcal{R} \in \{\mathcal{I}, \mathcal{F}\}$.

Proof The proof is by induction on the definition of the functions $\eta\mathcal{I}$ and $\eta\mathcal{F}$.

□

Lemma 4.5.6 *Let* $t \twoheadrightarrow t'$. *Then* $\eta\mathcal{R}(t) \twoheadrightarrow \eta\mathcal{R}(t')$ *where* $\mathcal{R} \in \{\mathcal{I}, \mathcal{F}\}$.

Proof The lemma is proved simultaneously by induction on the rewrite $t \twoheadrightarrow t'$. If t is a variable the lemma is trivial, while given a rewrite of the form:

$$\frac{t/\sigma \twoheadrightarrow u \quad t \setminus \sigma \twoheadrightarrow v}{t \twoheadrightarrow \mathbf{let}\, u\, \mathbf{be}\, p\, \mathbf{in}\, v}$$

there is a conversion $\sigma' \in \eta\mathcal{I}(t)$ such that:

$$\eta\mathcal{I}(t)/\sigma' = \eta\mathcal{I}(t/\sigma) \text{ and } \eta\mathcal{I}(t) \setminus \sigma' = \eta\mathcal{I}(t \setminus \sigma)$$

Then by the induction hypothesis

$$\frac{\eta\mathcal{I}(t)/\sigma' \twoheadrightarrow \eta\mathcal{I}(u) \quad \eta\mathcal{I}(t) \setminus \sigma' \twoheadrightarrow \eta\mathcal{I}(v)}{\eta\mathcal{I}(t) \twoheadrightarrow \mathbf{let}\, \eta\mathcal{I}(u)\, \mathbf{be}\, p\, \mathbf{in}\, \eta\mathcal{I}(v)}$$

A similar argument establishes the result for $\eta\mathcal{F}(t)$. The other possibility is for the rewrite $t \twoheadrightarrow t'$ to have as its last rule a pre-congruence and this case is trivially dealt with by the induction hypothesis. □

These results may now be combined to obtain decidability of $\beta\eta$-equality.

Theorem 4.5.7 *If two terms are equivalent in the conversion relation, then so are their* $\Rightarrow_\delta$*-normal forms. Thus the expansionary rewrite relation is confluent and* $\beta\eta$*-equality is decidable.*

Proof The first part of the lemma is trivial as $\Rightarrow_\delta$ is contained in the inverse of the conversion relation. Now if t and t' are $\beta\eta$-equivalent they are also equivalent in the rewrite relation defined as the union of $\Rightarrow_\mathcal{F}$ and the conversion relation, and thus their $\Rightarrow_\mathcal{F}$-normal forms are equivalent in the conversion relation which has already been shown to be decidable and confluent. □

Chapter 5

Almost Bicartesian Closed λ-Calculus

In the last chapter a sound and complete equational theory for terms of the linear λ-calculus was defined by interpreting the introduction and elimination rules of each type constructor as forming an adjoint pair. The key innovation distinguishing this work from other approaches in the literature is the development of a conversion relation which permutes the order in which negatively occurring subterms of tensor and unit type occur. The linear λ-calculus was chosen to investigate these ideas as the linearity of the calculus considerably simplified the mathematics involved, thus permitting a clearer account of the general properties of rewriting for initial types.

In this chapter we examine the effect of re-introducing the structural rules *weakening* and *contraction* by considering the calculus obtained by adding coproducts, also called sums, to the simply typed λ-calculus previously studied. As before, the introduction and elimination rules of each type constructor are interpreted as forming an adjoint pair, and the resulting unit and counit give rise to an expansionary η-rewrite rule and a contractive β-rewrite rule.

The associated rewrite relation is again decomposed into two fragments which are shown to satisfy the same abstract properties as their counterparts of the last chapter. However the non-linear use of variables significantly complicates the mathematics required to establish these results, e.g. confluence of the conversion relation cannot be proved directly and so cannot be assumed in the construction of quasi-normal forms. Instead confluence and decidability of the associated equational theory are proved in a two stage process: (i) firstly for the subrelation which preserves the layer structure on conversions induced by the prefix ordering; and (ii) secondly for the full conversion relation. Strong normalisation of the fragment of the rewrite relation containing all the β-reductions and restricted η-expansions is proved by extending the proof of strong normalisation for the simply typed λ-calculus to cope with the presence of initial type constructors. The final section of this chapter embeds the full rewrite relation into the conversion relation and this again is the natural generalisation of the proofs for the linear λ-calculus.

5.1 The Calculus $\Lambda^{1,\times,\rightarrow,+}$

Although this chapter is primarily concerned with the definition and decidability of $\beta\eta$-equality for initial types when the linearity principles of the previous chapter do not apply, in order to maintain continuity and to avoid certain trivial simplifications, a calculus which contains not only coproducts, but also products, a terminal object and exponentials is studied. This calculus is called Almost Bicartesian Closed as it corresponds to the internal language of bicartesian closed categories with the initial object omitted. We shall comment on this omission in greater detail later; at this point it suffices to say that I believe the techniques developed here are sufficient to cope with the addition of such an initial object.

The *Almost Bicartesian Closed λ-Calculus* over a set of base types $\mathcal{B}$ is denoted $\Lambda^{1,\times,\rightarrow,+}$ and consists of *types* freely generated from the base types by a nullary type constructor 1, called the *unit*, and three binary constructors $\times, +$ and $\rightarrow$ called the *product, coproduct* and *exponential*

$$T := T + T \mid T \times T \mid T{\rightarrow}T \mid 1 \mid B$$

where $B \in \mathcal{B}$ is any base type. The *atomic* types are 1 and the base types. For each type T there is a set of constants $\mathtt{Con}(T)$, including the distinguished constant $* \in \mathtt{Con}(1)$, such that if $T \neq T'$, then $\mathtt{Con}(T) \cap \mathtt{Con}(T') = \emptyset$. There is also a set of variables $\mathtt{Var}$ which is disjoint from the constants. The *pre-terms* of $\Lambda^{1,\times,\rightarrow,+}$ are:

$$t := x \mid c \mid \langle t, t\rangle \mid \pi_0 t \mid \pi_1 t \mid tt \mid \lambda x.t \mid \mathbf{in}_1(t) \mid \mathbf{in}_2(t) \mid \mathbf{case}(t, x.t, y.t)$$

where x, y are variables and c a constant. Note that in a pre-term $\mathbf{case}(t, x.u, y.v)$, the variables x and y are not subterms, but are rather variable binders which play a similar role to the x in $\lambda x.t$. The free variables of a pre-term t are denoted $\mathtt{FV}(t)$ and the substitution of pre-terms for free variables is defined as expected.

The *term judgements* of $\Lambda^{1,\times,\rightarrow,+}$ are of the form $\Gamma \vdash t\!:\!T$ where Γ is a context, t a pre-term and T a type, and are generated by the inference rules in Table 5-1. The side-conditions on the identity and structural rules may be found in equations 2.2 and 2.3, while new variables introduced by the logical rules are required to be distinct from those variables already present. Given any term judgement $\Gamma \vdash t\!:\!T$,

Table 5–1: Typing Rules for the Almost Bicartesian Closed λ-Calculus

Structural and Identity Rules

$$\frac{x \in \mathtt{Var}}{x\!:\!A \vdash x\!:\!A}\ \textit{axiom} \qquad \frac{c \in \mathtt{Con}(A)}{\vdash c\!:\!A}\ \textit{cons}$$

$$\frac{\Gamma \vdash t\!:\!A}{\Gamma, x\!:\!B \vdash t\!:\!A}\ \textit{weakening} \qquad \frac{\Gamma, x\!:\!B, y\!:\!B \vdash u\!:\!A}{\Gamma, z\!:\!B \vdash u[z/x, z/y]\!:\!A}\ \textit{contraction}$$

$$\frac{\Gamma, x\!:\!A, y\!:\!B, \Delta \vdash t\!:\!C}{\Gamma, y\!:\!B, x\!:\!A, \Delta \vdash t\!:\!C}\ \textit{exchange} \qquad \frac{\Gamma \vdash t\!:\!A \quad \Delta, x\!:\!A \vdash u\!:\!B}{\Gamma, \Delta \vdash u[t/x]\!:\!B}\ \textit{cut}$$

Logical Rules

$$\frac{\Gamma \vdash e\!:\!A \quad \Gamma \vdash e'\!:\!B}{\Gamma \vdash \langle e, e' \rangle\!:\!A \times B}\ \times\textit{int} \qquad \frac{\Gamma \vdash t\!:\!A_0 \times A_1}{\Gamma \vdash \pi_i t\!:\!A_i}\ \times\textit{elim}$$

$$\frac{\Gamma, z\!:\!A_1 + A_2 \vdash t\!:\!C}{\Gamma, x\!:\!A_i \vdash t[\mathbf{in}_i(x)/z]\!:\!C}\ +\textit{int} \qquad \frac{\Gamma, x\!:\!A \vdash u\!:\!C \quad \Gamma, y\!:\!B \vdash v\!:\!C}{\Gamma, z\!:\!A + B \vdash \mathbf{case}(z, x.u, y.v)\!:\!C}\ +\textit{elim}$$

$$\frac{\Gamma, x\!:\!A \vdash e\!:\!B}{\Gamma \vdash \lambda x.e\!:\!A \rightarrow B}\ \rightarrow\textit{int} \qquad \frac{\Gamma \vdash e\!:\!A \rightarrow B}{\Gamma, x\!:\!A \vdash ex\!:\!B}\ \rightarrow\textit{elim}$$

we say t is a term of type T and when Γ is clear, or not important, this is written $t\!:\!T$. A term is called an *introduction* term if it is a λ-abstraction, pair, injection or the constant $*$. If a term is not an introduction term, then it is called an *elimination* term. A subterm of a term occurs *negatively* iff it is applied, projected or is the first argument of a *case*-expression.

The proposed η_+-rewrite rule considers terms expressed as substitutions, i.e. occurrences which index free subterms. The variables bound at an occurrence $\sigma \in \mathtt{O}(t)$ are:

$$\mathtt{BV}(\sigma, t) = \begin{cases} \emptyset & \text{if } \sigma = \epsilon \\ \{x\} \cup \mathtt{BV}(\sigma^-, t') & \text{if } t = \lambda x.t' \text{ and } \sigma \neq \epsilon \\ \{x_1\} \cup \mathtt{BV}(\sigma^-, v_1) & \text{if } t = \mathbf{case}(u, x_1.v_1, x_2.v_2) \text{ and } \sigma \geq 1 \\ \{x_2\} \cup \mathtt{BV}(\sigma^-, v_2) & \text{if } t = \mathbf{case}(u, x_1.v_1, x_2.v_2) \text{ and } \sigma \geq 2 \\ \mathtt{BV}(\sigma^-, t/\sigma_0) & \text{otherwise} \end{cases}$$

When the term involved is clear from the context, we simply refer to $\mathtt{BV}(\sigma)$. The *free occurrences* of a term t are defined as follows:

$$\mathtt{FO}(t) = \{\sigma \in \mathtt{O}(t) \mid \mathtt{FV}(t/\sigma) \cap \mathtt{BV}(\sigma, t) = \emptyset\}$$

5.2 Rewriting in $\Lambda^{1,\times,\rightarrow,+}$

In the first of our case studies, rewrite rules for the product, unit and exponential were derived by constructing categorical models of reduction and interpreting the introduction and elimination rules of each type constructor as being adjoint functors. When applied to coproducts this approach again generates a contractive β-rewrite rule and an expansionary η-rewrite rule, although, as explained in the previous chapter, the elimination rules are left-adjoint for final type constructors but right-adjoint for initial type constructors.

For example, if $\mathcal{C}(\Gamma; X)$ is defined as in the previous chapters to be the category whose objects are judgements of the form $\Gamma \vdash e : X$ and whose morphisms are rewrites between terms, then the introduction and elimination rules of the coproduct may be regarded as functors between the categories shown below. As these functors are taken to constitute an adjoint pair:

$$\mathcal{C}(\Gamma, x\!:\!A; C) \times \mathcal{C}(\Gamma, y\!:\!B; C) \underset{(\_[\mathbf{in}_1(x)/z],\, \_[\mathbf{in}_2(y)/z])}{\overset{\mathbf{case}(z,\, x.\_,\, y.\_)}{\underset{\longleftarrow}{\overset{\longrightarrow}{\top}}}} \mathcal{C}(\Gamma, z\!:\!A+B; C) \qquad (5.1)$$

rewrite rules can be obtained as counit and unit:

$$\begin{array}{lrcl}
(\beta_{+,1}) & \mathbf{case}(\mathbf{in}_1(x), x.u, y.v) & \Rightarrow & u \\
(\beta_{+,2}) & \mathbf{case}(\mathbf{in}_2(y), x.u, y.v) & \Rightarrow & v \\
(\eta_{+}) & e & \Rightarrow & \mathbf{case}(z, x.e[\mathbf{in}_1(x)/z], y.e[\mathbf{in}_2(y)/z])
\end{array}$$

where the η_+-rewrite rule has the side condition $x, y \notin \mathtt{FV}(e)$. As always this adjunction can be equivalently presented as a natural isomorphism of rewrites:

$$t \Rightarrow \mathbf{case}(z, x.u, y.v) \quad \text{iff} \quad t[\mathbf{in}_1(x)/z] \Rightarrow u \text{ and } t[\mathbf{in}_2(y)/z] \Rightarrow v \qquad (5.2)$$

As with the *let*-expressions of the linear λ-calculus, the *case*-expressions can be viewed as representing certain reductions and hence it will be the *case*-expressions which form the basis of quasi-normal forms for terms of sum type.

As the rewrite rules for each type constructor are determined solely by their introduction and elimination rules, the rewrite rules previously derived for the exponential, product and terminal object remain valid in this calculus. The *expansionary rewrite relation* is denoted $\Rightarrow$ and is the least pre-congruence on terms including the basic reductions in Table 5-2. In addition to the side conditions for the product and exponential mentioned in Chapter 3, the η_+-rewrite rule assumes $x, y \notin \mathtt{FV}(e)$. The equational theory generated by the expansionary rewrite relation is called *βη-equality*.

Although superficially similar in structure to the expansionary $\eta_\otimes$- and η_I-rewrite rules of the linear λ-calculus, the η_+-rewrite rule is actually substantially more complex. In particular the non-linearity of the calculus means that in the η_+-rewrite rule there is no restriction on the number of times the free variable z may occur in e. Thus, not only is η_+ not left-linear in that different occurrences in the redex may be mapped to the same occurrence in the reduct, but η_+ may also introduce new free variables if the variable z does not actually occur in e. This is generally thought of as an inadvisable property in term rewriting circles and means that subject reduction must be modified as follows:

Table 5–2: Rewrite Rules for the Almost Bicartesian Closed λ-Calculus

$(\beta_{\times,1})$	$\pi_0\langle a,b\rangle$	$\Rightarrow$	a
$(\beta_{\times,2})$	$\pi_1\langle a,b\rangle$	$\Rightarrow$	b
$(\eta_\times)$	c	$\Rightarrow$	$\langle \pi_0 c, \pi_1 c\rangle$
$(\beta_\rightarrow)$	$(\lambda x.t)u$	$\Rightarrow$	$t[u/x]$
$(\eta_\rightarrow)$	t	$\Rightarrow$	$\lambda x.tx$
(η_1)	a	$\Rightarrow$	$*$
$(\beta_{+,1})$	$\mathbf{case}(\mathbf{in}_1(t), x.u, y.v)$	$\Rightarrow$	$u[t/x]$
$(\beta_{+,2})$	$\mathbf{case}(\mathbf{in}_2(t), x.u, y.v)$	$\Rightarrow$	$v[t/y]$
(η_+)	$e[e'/z]$	$\Rightarrow$	$\mathbf{case}(e', x.e[\mathbf{in}_1(x)/z], y.e[\mathbf{in}_2(y)/z])$

Lemma 5.2.1 *If there is a term judgement $\Gamma \vdash t : A$ and a rewrite $t \Rightarrow t'$ then there is also a term judgement $\Gamma' \vdash t' : A$, where Γ' is some context extending Γ.*

Proof The proof is by induction on the term judgement $\Gamma \vdash t : A$. The only case where $\Gamma \neq \Gamma'$ occurs in the η_+-rewrite rule

$$e[e'/z] \Rightarrow \mathbf{case}(e', x.e[\mathbf{in}_1(x)/z], y.e[\mathbf{in}_2(y)/z])$$

if the variable z does not actually occur free in e. □

$\beta\eta$-equality is proved decidable by following the same approach as for the linear λ-calculus, that is the rewrite relation is decomposed into two parts: (i) a strongly normalising and confluent relation consisting of β-reduction, commuting conversions and limited possibilities for η-expansion; and (ii) a conversion relation which permutes the order in which negatively occurring subterms of sum type occur. Strong normalisation and confluence of the first part of the decomposition is proved by adapting the proofs for the simply typed λ-calculus to cope with the presence of coproducts. The non-linearity of $\Lambda^{1,\times,\rightarrow,+}$ means that the parallelised form of the conversion relation no longer satisfies the diamond property, and thus confluence cannot be proved in advance of decidability of the equational theory associated to the conversion relation. Instead quasi-normal forms are first constructed for the subrelation consisting of those rewrites in the conversion relation which preserve the layer structure on conversions induced by prefix ordering. As the breakdown of this layer structure is fully described by certain commuting conversions, quasi-normal forms are constructed for the conversion relation as a whole by synthesising the quasi-normal forms just constructed with the unique normal forms of the commuting conversions. This turns out to be a natural generalisation of the construction of quasi-normal forms for the linear conversion relation.

Before doing this we comment on two other issues.

Distributivity

As with the simply typed λ-calculus and the linear λ-calculus, a model of $\Lambda^{1,\times,\rightarrow,+}$ consists of a category with certain extra structure — in this case the category must be cartesian closed and also have binary coproducts. The interpretation of a term judgement $\Gamma \vdash t\!:\!A$ in such a category $\mathcal{C}$ is as a morphism:

$$[\![t]\!] : [\![\Gamma]\!] \longrightarrow [\![A]\!]$$

where $[\![X]\!]$ denotes the interpretation of a type X in $\mathcal{C}$, which is extended to contexts via the product structure of $\mathcal{C}$. Thus interpreting the sum elimination rule:

$$\frac{\Gamma, x\!:\!A \vdash u\!:\!C \quad \Gamma, y\!:\!B \vdash v\!:\!C}{\Gamma, z\!:\!A+B \vdash \mathbf{case}(z, x.u, y.v)\!:\!C}$$

amounts to constructing a morphism:

$$[\![\mathbf{case}(z, x.u, y.v)]\!] : [\![\Gamma]\!] \times ([\![A]\!] + [\![B]\!]) \longrightarrow [\![C]\!]$$

from morphisms:

$$[\![u]\!] : [\![\Gamma]\!] \times [\![A]\!] \longrightarrow [\![C]\!] \quad \text{and} \quad [\![v]\!] : [\![\Gamma]\!] \times [\![B]\!] \longrightarrow [\![C]\!]$$

Although one can use $[\![u]\!]$ and $[\![v]\!]$ to construct a morphism from $([\![\Gamma]\!] \times [\![A]\!]) + ([\![\Gamma]\!] \times [\![B]\!])$ to $[\![C]\!]$, this will not suffice as this morphism has the wrong domain. What is then required is a morphism of the form:

$$d : [\![\Gamma]\!] \times [\![A+B]\!] \longrightarrow ([\![\Gamma]\!] \times [\![A]\!]) + ([\![\Gamma]\!] \times [\![B]\!])$$

and, although there is a canonical morphism in the reverse direction, the existence of the morphism d does not follow from the laws defining products and coproducts.

A *distributive category* is a category with binary products and coproducts such that the canonical morphism alluded to above is a natural isomorphism. In order to give a model of the sum elimination rule in a category $\mathcal{C}$ one must explicitly require that $\mathcal{C}$ be distributive, but fortunately this is a very weak condition, e.g. any category which is cartesian closed and has binary coproducts is automatically distributive.

Lemma 5.2.2 *$\beta\eta$-equality is a sound and complete equality for models of $\Lambda^{1,\times,\rightarrow,+}$ in cartesian closed categories with binary coproducts.*

Proof The proof is by the same kind of construction as for the simply typed λ-calculus and the linear λ-calculus. □

Initial Objects

An initial object of a category $\mathcal{C}$ is an object 0 such that for every other object X, the hom-set $\mathcal{C}(0, X)$ has exactly one morphism. Before giving typing and rewrite rules for an initial object, we quote some simple results illustrating the kind of equational theory expected for an initial object.

Lemma 5.2.3 *Let $\mathcal{C}$ be a category with products and an initial object. If A is an object of $\mathcal{C}$ then either $\mathcal{C}(A, 0) = \emptyset$, or A is also an initial object and $\mathcal{C}(A, 0)$ contains a single morphism.*

Proof See [55]. □

An initial object can be added to $\Lambda^{1,\times,\rightarrow,+}$ via the elimination rule

$$\frac{}{x : 0 \vdash \epsilon_A(x) : A}$$

Then, using the same notation as before, an expansionary η-rewrite called η_0 may be derived by interpreting the elimination rule as right adjoint to the unique functor $!:\mathcal{C}(x:0;A)\to\mathbf{1}$

$$\begin{tikzcd} \mathbf{1} \arrow[r, shift left=1.5ex, "\epsilon_A(x)"] \arrow[r, phantom, "\top"] & \mathcal{C}(x:0;A) \arrow[l, shift left=1.5ex, "!"] \end{tikzcd}$$

Note that, as with the unit 1, the β_0-rewrite rule is the identity morphism in $\mathbf{1}$. A context Γ is *inconsistent* iff there is a term judgement $\Gamma\vdash e:0$. Lemma 5.2.3 shows that if two terms are typed in the same inconsistent context, then by completeness these terms must be η_0-equivalent, even if the terms themselves can be typed in a consistent subcontext. Thus the η_0-reducts of a term are determined more by the consistency of the context in which the term was typed, than by the actual term itself. This leads us to replace the presentation of rewrite relations as relations on the set of all terms, and instead consider rewrite relations on the set of term judgements. In this framework the η_0-rewrite rule is presented as follows:

$$\frac{\Gamma\vdash e':0 \quad \Delta,x:0\vdash e:A}{\Gamma,\Delta\vdash e[e'/x]\Rightarrow\epsilon_A(e'):A}$$

i.e. each rewrite is indexed by the domain and codomain of the redex. Note that subject reduction says $\Rightarrow$ can be viewed as a family of context and type-indexed homogeneous relations $\Rightarrow_{\Gamma;A}$ on the set $\{t \mid \Gamma\vdash t:A\}$. We now make some brief comments about the η_0-rewrite rule, although a formal investigation is postponed for future research [28]. For a discussion of empty types in a polymorphic setting see [61].

- Context inconsistency, term typability and other important issues in the study of the η_0-rewrite rule are decidable for calculi such as the ones we study in this thesis. However in more complex theories this may not be the case.

- If $\Gamma\vdash e:0$ and $\Gamma\vdash e':0$ then e and e' are $\Rightarrow_{\Gamma;0}$-equivalent. This is the syntactic counterpart of lemma 5.2.3.

- If Γ is inconsistent then the $\Rightarrow_{\Gamma;A}$-quasi-normal forms of any term t of type A in context Γ are given by:

$$\{\epsilon_A(e) \mid \Gamma\vdash e:0\}$$

This set is typically infinite.

- If Γ is not inconsistent then the quasi-normal forms of a term may be calculated by: (i) constructing quasi-normal forms for the rest of the rewrite relation; and (ii) replacing those subterms occuring in inconsistent contexts by their quasi-normal forms.

5.3 The Conversion Relation

In this section a conversion relation over the terms of $\Lambda^{1,\times,\rightarrow,+}$ is proved decidable and confluent by explicitly constructing the quasi-normal reducts of each term. There are two major differences with the linear conversion relation of the previous chapter: (i) the η_+-rewrite rule places no restrictions on the number of times the subterm being expanded occurs, and hence the conversion relation is no longer defined in terms of single conversions, but rather sets of conversions; and (ii) because there are two different types of introduction terms, i.e. left-injection and right-injection, there are not one, but two residues involved in the definition of the conversion relation.

Given a term t, its set of *conversions* is defined by:

$$\mathtt{C}(t) = \{\sigma \in \mathtt{O}(t) \mid \sigma \text{ is the first argument of a } \textit{case}\text{-expression } \}$$

The free conversions of t are simply those occurrences which are both free and conversions $\mathtt{FC}(t) = \mathtt{C}(t) \cap \mathtt{FO}(t)$. As with the conversions of the linear λ-calculus, every conversion $\sigma \in \mathtt{C}(t)$ has a *binding* $\mathbf{bind}(\sigma, t)$ consisting of the ordered pair of variables bound by the arms of the *case*-expression associated to the conversion. Whenever sets of conversions are considered, they are assumed to be independent and the subterms so indexed are all assumed to be of the same type and to have the same binding. If $X \subseteq \mathtt{C}(t)$ is a set of conversions, then the result of contracting the β-redexes formed upon insertion of injections at these occurrences are called the *first and second residues* and are defined recursively:

$$t \setminus_1 X = \begin{cases} t & \text{if } X = \emptyset \\ v_1 \setminus_1 X_1 & \text{if } 0 \in X \text{ and } t = \mathbf{case}(u, x.v_1, y.v_2) \\ \mathcal{T}(t_0 \setminus_1 X_0, \ldots, t_n \setminus_1 X_n) & \text{if } X \neq \emptyset,\ 0 \notin X \text{ and } t = \mathcal{T}(t_0, \ldots, t_n) \end{cases}$$

and

$$t \setminus_2 X = \begin{cases} t & \text{if } X = \emptyset \\ v_2 \setminus_2 X_2 & \text{if } 0 \in X \text{ and } t = \mathbf{case}(u, x.v_1, y.v_2) \\ \mathcal{T}(t_0 \setminus_2 X_0, \ldots, t_n \setminus_2 X_n) & \text{if } X \neq \emptyset,\ 0 \notin X \text{ and } t = \mathcal{T}(t_0, \ldots, t_n) \end{cases}$$

where $X_n = X/n$.

Lemma 5.3.1 *Given a set of conversions $X \subseteq \mathtt{C}(t)$ which bind the variables x_1 and x_2, then for $i = 1$ or 2 there is a series of β-reductions*

$$t[\sigma \leftarrow \mathbf{in}_i(x_i)]_{\sigma \in X} \ \Rightarrow^* t \setminus_i X$$

Proof A straightforward induction over the definition of residue. □

The *Almost Bicartesian Closed Conversion relation* is formally defined in terms of a calculus for deriving quadruples of the form $\langle \sigma, X \rangle : t \Rightarrow_c t'$ where σ is an occurrence of t representing where in t the rewrite takes place and X are conversions to be expanded, i.e. $X \subseteq \mathtt{FC}(t/\sigma)$. We call $\langle \sigma, X \rangle$ the label of the rewrite, and, when not required, the label part of a rewrite may be omitted. These quadruples are generated by the three inference rules in Table 5-3, where in the *expansion* clause x, y are the variables bound by each $\sigma \in X$ and to avoid variable capture we assume:

$$x, y \notin \mathtt{FV}(t) \text{ and } x, y \notin \mathtt{BV}(\sigma^+, t) \tag{5.3}$$

These conditions can always be met by a change of bound variables and are the exact analogue of those for the linear conversion relation.

The *expansion* clause requires the set X of conversions to be free and consistent so that the redex may be expressed in a form compatible with the η_+-rewrite rule. In addition this set is required to be non-empty to prevent expansions of the

Table 5–3: The Almost Bicartesian Closed Conversion Relation

Expansion	$\dfrac{X \subseteq \mathtt{FC}(t) \quad X \text{ consistent} \quad X \neq \emptyset}{\langle \epsilon, X\rangle : t \;\Rightarrow_c \mathbf{case}(t/X, x.t \setminus_1 X, y.t \setminus_2 X)}$
Weakening	$\dfrac{x \notin \mathtt{FV}(t) \quad y \notin \mathtt{FV}(t)}{\langle \epsilon, \emptyset\rangle : \mathbf{case}(u, x.t, y.t) \;\Rightarrow_c t}$
Pre-Congruence	$\dfrac{\langle \sigma, X\rangle : t_j \;\Rightarrow_c t'_j}{\langle j.\sigma, X\rangle : \mathcal{T}(t_0, \ldots, t_n) \;\Rightarrow_c \mathcal{T}(t_0, \ldots, t_n)[j \leftarrow t'_j]}$

form $t \Rightarrow \mathbf{case}(u, x.t, y.t)$ which would allow terms to grow arbitrarily large, new free variables to be introduced, etc. However these terms remain identified in the equational theory generated by the conversion relation because redex and reduct have been inverted and included under the *weakening* clause. The name of this rule is chosen to suggest the idea that the subterm u may be discarded as it contributes nothing to the value of the overall expression. In addition, if the structural rule *weakening* were not present then pre-terms of the form $\mathbf{case}(t, x.u, y.u)$, with $x, y \notin \mathtt{FV}(u)$, would not be typable.

Lemma 5.3.2 *Given a rewrite $\langle \sigma, X\rangle : t \Rightarrow_c t'$, then $t =_{\beta\eta} t'$ in the expansionary rewrite relation. In addition, the relation $\Rightarrow_c$ satisfies subject reduction.*

Proof Assume first that $\sigma = \epsilon$. If the rewrite is of the form $\mathbf{case}(u, x.t, y.t) \;\Rightarrow_c t$ then, given a variable z not free in t,

$$t \;=\; t[u/z] \;\Rightarrow_{\eta+}\; \mathbf{case}(u, x.t[\mathbf{in}_1(x)/z], y.t[\mathbf{in}_2(y)/z]) \;=\; \mathbf{case}(u, x.t, y.t)$$

If however X is non-empty then, given a variable z not free in t,

$$\begin{aligned} t \quad &= \quad t[\sigma \leftarrow z]_{\sigma \in X}[z := t/X] \\ &\Rightarrow_{\eta+} \quad \mathbf{case}(t/X, x.t[\sigma \leftarrow \mathbf{in}_1(x)]_{\sigma \in X}, y.t[\sigma \leftarrow \mathbf{in}_2(y)]_{\sigma \in X}) \\ &\Rightarrow^{*} \quad \mathbf{case}(t/X, x.t \setminus_1 X, y.t \setminus_2 X) \end{aligned}$$

where the first equality holds as by assumption $z \notin \mathtt{FV}(t)$ and X is a non-empty set of free conversions, the second line is just an η_+-expansion with the conditions in equation 5.3 ensuring that substitution is just replacement, and the last line is from lemma 5.3.1. Finally, if $\sigma \neq \epsilon$ then, as both relations are pre-congruences, the first part of the lemma follows by the induction hypothesis. Since the *expansion clause* insists that the set of conversions being expanded is non-empty, and subject reduction holds for reductions included under the *weakening* clause, subject reduction for $\Rightarrow_c$ follows from lemma 5.2.1. □

5.3.1 Tracking Conversions

The technical core of the analysis of the conversion relation is to use the structure of a rewrite, represented in the associated label, to describe its action on arbitrary conversions. More formally, to each rewrite $r : t \Rightarrow t'$ is associated a function

$$\overline{r} : \mathtt{C}(t) \rightarrow \mathcal{P}\mathtt{C}(t')$$

which maps a conversion in the redex, called an *ancestor*, to a set of conversions in the reduct, called *descendants*. Firstly note that a single conversion may have more than one descendant and the layer structure, formed by the embedding of conversions inside each other, is not necessarily preserved. Examples of these phenomena can be seen by the commuting conversion below where the conversion $0 \cdot 0$ will be mapped to $\{0\}$ while the conversion 0 is mapped to the set $\{1 \cdot 0, 2 \cdot 0\}$.

$$\begin{array}{lcr} \mu : & \mathbf{case}(\mathbf{case}(t, x.u, y.v), x'.u', y'.v') & \Rightarrow \\ \multicolumn{3}{l}{\mathbf{case}(t, x.\mathbf{case}(u, x'.u', y'.v'), y.\mathbf{case}(v, x'.u', y'.v'))} \end{array} \tag{5.4}$$

As with the linear conversion relation, this redex form is of great importance and so the rewrite relation $\Rightarrow_\mu$ is defined to be the least pre-congruence including these reductions. Another interesting rewrite, this time without an analogue in the linear conversion relation, is the following:

$$\langle \epsilon, \{0, 10\} \rangle : \mathbf{case}(t, x.\mathbf{case}(t, x.u, y.v), y.s) \Rightarrow \mathbf{case}(t, x.u, y.s) \tag{5.5}$$

which shows that a conversion, e.g. those inside v, may have no descendants, and that a conversion in the reduct may have more than one ancestor. However one important property of $\overline{r}$ is surjectivity, i.e. all conversions in a reduct have at least one ancestor conversion in the redex. This means that the possibilities for further rewriting, which are determined by the conversions of the reduct, may be traced back to the redex and hence a static analysis of the transitive closure of the conversion relation may be given. This forms the basis for confluence and decidability. As a measure of the increased complexity of this bicartesian λ-calculus over the linear λ-calculus, the equivalent conversion tracking function is actually a bijection between the conversions of the redex and those of the reduct.

The action of a (consistent) set of conversions $X \subseteq \mathtt{C}(t)$ on the term t is to produce two residues, namely $t \setminus_1 X$ and $t \setminus_2 X$. This action induces a partitioning of the set of conversions $\mathtt{C}(t)$ into: (i) those conversions which are subconversions of (unique) members of X; (ii) those conversions which have descendants in one or both of the residues; and (iii) those conversions which fit into neither of these categories and hence have no descendants. If the calculation of a residue is viewed as indicating a kind of path through the term, then conversions of this last type are those which are unreachable, e.g. conversions like those inside v in the reduction 5.5.

If a conversion σ has a descendant in the residue $t \setminus_i X$, then this residue will be unique and is given by the partial function Ω_i (which is undefined if the conversion has no descendent):

$$\Omega_i(X, \sigma) = \begin{cases} \Omega_i(X/i, \sigma^-) & \text{if } 0 \in X \text{ and } \sigma \geq i \\ \text{undefined} & \text{if } 0 \in X \text{ and } \sigma \not\geq i \\ 0 & \text{if } 0 \notin X \text{ and } \sigma = 0 \\ \sigma_0 \cdot \Omega_i(X/\sigma_0, \sigma^-) & \text{otherwise} \end{cases}$$

Notice that in general the domain of Ω_1 will differ from that of Ω_2. The following properties of the functions Ω_i will be used later.

Lemma 5.3.3 *Let X be a set of conversions. Given a conversion τ in the domain of $\Omega_i(X)$, the conversion $\Omega_i(X, \tau)$ indexes the subterm given by:*

$$(t \setminus_i X)/\Omega_i(X, \tau) = (t/\tau) \setminus_i (X/\tau)$$

In addition the function $\Omega_i(X, \_) : \mathtt{C}(t) \rightarrow \mathtt{C}(t \setminus_i X)$ is surjective, injective on its domain, maps free conversions to free conversions and is strictly monotonic, i.e.

for conversions τ, σ in its domain:

$$\tau > \sigma \quad \textit{iff} \quad \Omega_i(X, \tau) > \Omega_i(X, \sigma)$$

Proof The proof is a simple induction. □

The functions Ω_i are extended pointwise to calculate the action of a set of conversions on another set of conversions, that is given sets of conversions $X \subseteq \mathtt{C}(t)$ and $X' \subseteq \mathtt{C}(t)$ define

$$\Omega_1(X, X') = \{\Omega_1(X, \tau) \mid \tau \in X' \text{ and } \Omega_1(X, \tau) \text{ is defined }\}$$

and

$$\Omega_2(X, X') = \{\Omega_2(X, \tau) \mid \tau \in X' \text{ and } \Omega_2(X, \tau) \text{ is defined }\}$$

By lemma 5.3.3 if $X' \subseteq \mathtt{C}(t)$ is a disjoint and/or free set of conversions, then so is $\Omega_i(X, X')$. The following lemma forms a kind of local confluence result which states that given two different sets of conversions, the order in which the compound residue is formed is unimportant.

Lemma 5.3.4 *Let $X \subseteq \mathtt{C}(t)$ and $X' \subseteq \mathtt{C}(t)$ each be disjoint sets of conversions. Then*

$$(t \setminus_i X) \setminus_j \Omega_i(X, X') = (t \setminus_j X') \setminus_i \Omega_j(X', X)$$

where $i, j = 1$ or 2. If σ is in the domain of $\Omega_i(X)$ and the domain of $\Omega_j(X')$ then the following expressions are both defined and equal:

$$\Omega_j(\Omega_i(X, X'), \Omega_i(X, \sigma)) = \Omega_i(\Omega_j(X', X), \Omega_j(X', \sigma))$$

If X and X' have a non-empty intersection then the lemma still holds providing $i = j$.

Proof The lemma is another trivial induction on the structure of t. □

Note that if the sets X and X' were not disjoint the lemma would be false, e.g. consider the term $\mathbf{case}(t, x.u, y.v)$ with $X, X' = \{0\}$ and $i \neq j$. The second half of the lemma holds because asserting $i = j$ removes this implicit choice.

The conversion tracking function promised at the beginning of this subsection can now be constructed. Given a rewrite $r : t \Rightarrow t'$ and a conversion $\sigma \in \mathtt{C}(t)$, define its set of *descendants* $\overline{r}(\sigma) \subseteq \mathtt{C}(t')$ recursively as follows. If r is a *weakening* then set:

$$\overline{r}(\sigma) = \begin{cases} \emptyset & \text{if } \sigma \geq 0 \\ \{\sigma^-\} & \text{otherwise} \end{cases}$$

If however r is an expansion of the non-empty, free and consistent set of conversions X, then set:

$$\overline{r}(\sigma) = \begin{cases} \{0 \cdot \sigma/\omega\} & \text{if there is a } \omega \in X \text{ with } \sigma \geq \omega \\ \{1 \cdot \Omega_1(X, \sigma), 2 \cdot \Omega_2(X, \sigma)\} & \text{otherwise} \end{cases}$$

where in the first clause because X is consistent, ω is necessarily unique and in the second clause those functions which are undefined are deleted. Finally if r is a pre-congruence then set:

$$\overline{\langle j \cdot \omega, X \rangle}(\sigma) = \begin{cases} \{\sigma\} & \text{if } \sigma = 0 \text{ or } \sigma_0 \neq j \\ j \cdot \overline{\langle \omega, X \rangle}(\sigma^-) & \text{otherwise} \end{cases}$$

The function $\overline{r}$ is extended pointwise to sets of conversions. Rather lengthy, but tedious, inductive arguments show that $\overline{r}(\sigma)$ actually consists of conversions, and that $\overline{r}$ covers $\mathtt{C}(t')$, i.e. the *ancestors* of a conversion $\sigma \in \mathtt{C}(t')$ defined by

$$r^{-1}(\sigma) = \{\omega \in \mathtt{C}(t) \mid \sigma \in \overline{r}(\omega)\}$$

form a non-empty set.

5.3.2 Closure, Preservation and Factorisation

The conversion relation associated with the linear λ-calculus was shown confluent by factorising a rewrite $t \twoheadrightarrow t'$ into a pair of "smaller" rewrites

$$t/\sigma \twoheadrightarrow t'/\sigma' \quad \text{and} \quad t \setminus \sigma \twoheadrightarrow t' \setminus \sigma'$$

for appropriately chosen conversions σ, σ'. We shall try to prove a similar result for this calculus, i.e. find conditions under which a rewrite $r : t \Rightarrow_c t'$ and a set of conversions $X \subseteq \mathtt{C}(t)$ induce rewrites of the form

$$r/\sigma' : t/\sigma \Rightarrow_c t'/\sigma' \quad \text{and} \quad r \setminus_i X : t \setminus_i X \Rightarrow_c t' \setminus_i \overline{r}(X)$$

where $\sigma \in X$ and $\sigma' \in \overline{r}(\sigma)$. A rewrite cannot always be localised to a conversion, e.g. in reduction 5.4 the conversion 0 is mapped to the conversions $1 \cdot 0$ and $2 \cdot 0$ but no reduction exists between the corresponding subterms. This problem occurs as the conversion 0 is mapped into the residues while one of its sub-conversions is expanded to the head of the term and is thus 'removed' from the original conversion. The embedding of conversions inside each other generates a layer structure and the key to localising a rewrite to a conversion is to ensure that this layer structure is preserved. Thus a rewrite $\langle \tau, X \rangle : t \Rightarrow t'$ is said to *preserve* a conversion $\sigma \in \mathtt{C}(t)$ iff $\forall \omega \in X . \neg(\tau < \sigma < \omega)$, i.e. no subconversions of σ are expanded outside of σ. The subrelation consisting of those rewrites which preserve all conversions is defined as follows:

$$\Rightarrow_p = \{ r : t \Rightarrow t' \mid r \text{ preserves every } \sigma \in \mathtt{C}(t) \}$$

The commuting conversion in equation 5.4 is an example of a rewrite which does not preserve the conversion layer structure, e.g. the conversion 0 is not preserved. In fact this redex fully describes all the cases in which $\Rightarrow_c$-reduction fails to preserve a conversion:

Lemma 5.3.5 *The conversion relation may be decomposed as follows:*

$$\Rightarrow_c^* = (\Rightarrow_p \cup \Rightarrow_\mu)^*$$

Proof Easy induction. □

Lemma 5.3.6 *Let $r : t \Rightarrow t'$ preserve $\sigma \in \mathtt{C}(t)$. Then for all $\sigma' \in \overline{r}(\sigma)$ there is a rewrite:*

$$r/\sigma : t/\sigma \Rightarrow t'/\sigma'$$

If in addition r preserves all conversions in $\mathtt{C}(t)$ then r/σ preserves all conversions in $\mathtt{C}(t/\sigma)$.

Proof The proof is by induction on the rewrite r. Firstly if r is of the form $\langle \epsilon, X \rangle$ for a non-empty set X containing σ, then the identity rewrite suffices, while if $\sigma' = i \cdot \Omega_i(X, \sigma)$, then

$$t'/\sigma' = (t \setminus_i X)/\Omega_i(X, \sigma) = (t/\sigma) \setminus_i (X/\sigma)$$

but, as r preserves σ, the set X/σ is empty and so the identity rewrite again suffices. The identity also suffices if r is of the form $\langle \epsilon, \emptyset \rangle$. Finally, if r is a pre-congruence $\langle j \cdot \sigma, X \rangle$ with $\sigma_0 \neq j$ then again the identity is used, while if $\sigma = j = 0$ then take r/σ to be $\langle \sigma, X \rangle$. The final possibility follows from the induction hypothesis. The second part of the lemma is trivially proved by analysing the only non-trivial case of r/σ. □

The other half of this section focusses upon the conditions which must be satisfied by a set of conversions $X \subseteq \mathtt{C}(t)$ such that a rewrite $r : t \Rightarrow t'$ induces rewrites of the two residues $r \setminus_i X : t \setminus_i X \Rightarrow t' \setminus_i \overline{r}(X)$. If r is of the form $\mathbf{case}(t, x.u, y.u) \Rightarrow u$ and X contains any conversions occurring inside one of the arms of the *case*-expression then, in order to maintain the shape of the redex, X must also contain the 'sister' conversion which occurs inside the other arm. Similar considerations apply to an expansion $\langle \epsilon, Y \rangle$ where, if X contains a conversion occurring inside an element of Y, then, in order to maintain consistency, X must also contain the sister conversions occurring inside the other elements of Y. These conditions are caused by the non-left linear nature of the conversion relation and can be easily formalised in terms of the conversion tracking function. Given a rewrite $r : t \Rightarrow t'$, a set of conversions $X \subseteq \mathtt{C}(t)$ is said to be *r-closed* iff $X = r^{-1}\overline{r}(X)$. This condition is sufficient to ensure a rewrite between the residues exists, although there is a slight complication in the direction of the residual rewrites. Notice that the inclusion $X \subseteq r^{-1}\overline{r}(X)$ always holds.

Lemma 5.3.7 *Let $r : t \Rightarrow t'$ and X be an r-closed set of conversions. Then there exist rewrites of the form*

$$r \setminus_1 X : t \setminus_1 X \Rightarrow t' \setminus_1 \overline{r}(X) \text{ and } r \setminus_2 X : t \setminus_2 X \Rightarrow t' \setminus_2 \overline{r}(X)$$

or rewrites in the reverse direction. If in addition r preserves all conversions so do the residual rewrites.

Proof First consider a rewrite of the form $\mathbf{case}(t, x.u, y.u) \Rightarrow u$. If $0 \in X$ then the identity rewrite suffices, while if $0 \notin X$, then by closure $X/1 = X/2$ and so another weakening suffices.

Secondly if r is an expansion $\langle \epsilon, X' \rangle$ then there are two subcases. If X and X' are disjoint, then the obvious candidate for the residual rewrite is an expansion of the conversions $\Omega_i(X, X') \subseteq \mathtt{C}(t \setminus_i X)$. By lemma 5.3.3 these conversions are all free and, by closure, for any $\sigma, \sigma' \in X'$ $X/\sigma = X/\sigma'$. Thus the set of conversions $\Omega_i(X, X')$ is also consistent and so, providing this set is non-empty, there is a rewrite

$$\begin{aligned}
t \setminus_i X \quad &\Rightarrow \quad \mathbf{case}((t \setminus_i X)/\Omega_i(X, X'), x.(t \setminus_i X) \setminus_1 \Omega_i(X, X'), y.(t \setminus_i X) \setminus_2 \Omega_i(X, X')) \\
&= \quad \mathbf{case}((t/X') \setminus_i (\overline{r}(X)/0), x.(t \setminus_1 X') \setminus_i \Omega_1(X', X), y.(t \setminus_2 X') \setminus_i \Omega_2(X', X)) \\
&= \quad \mathbf{case}((t/X'), x.t \setminus_1 X', y.t \setminus_2 X') \setminus_1 \overline{r}(X)
\end{aligned}$$

where the first equality is derived from lemmas 5.3.3 and 5.3.4 and the definition of $\overline{r}$, while the second holds as the disjointness of X and X' imply $0 \notin \overline{r}(X)$ and hence the residue of the last line is calculated on the subterms. If however $\Omega_i(X, X')$ is empty, then there is a rewrite in the reverse direction. The second subcase occurs if X and X' are not disjoint, in which case

$$\begin{aligned}
\mathbf{case}(t/X', x.t \setminus_1 X', y.t \setminus_2 X') \setminus_i \overline{r}(X) \quad &= \quad t \setminus_i X' \setminus_i \Omega_i(X', X) \\
&= \quad t \setminus_i X \setminus_i \Omega_i(X, X') \\
&= \quad t \setminus_i X
\end{aligned}$$

where by closure $X' \subseteq X$ and hence $0 \in \overline{r}(X)$ and $\Omega_i(X, X')$ is empty. Finally, if r is a pre-congruence then either the residual rewrite can be taken to be the identity or the lemma follows by the induction hypothesis. If in addition r preserves all conversions, then by inspection of the construction of the residual rewrites, and the strict monotonicity of Ω_i proved in lemma 5.3.3, the residual rewrites also preserve all conversions. □

5.3.3 Decidability of $\Rightarrow_p$-Equivalence

Conversion equivalence may now be proved decidable but, because of the conditions required by preservation and closure, this is done firstly for the subrelation $\Rightarrow_p$. For each term we construct its finite set of $\Rightarrow_p$-quasi-normal forms and show that $\Rightarrow_p$-equivalent terms have the same set of $\Rightarrow_p$-quasi-normal forms.

If a term is a quasi-normal form containing free conversions, then the term must be a *case*-expression, as otherwise a rewrite to such a term would exist, but not one in the other direction. Thus the construction of normal forms is essentially a process of expanding as many conversions as possible. However as a non-free conversion may have a $\Rightarrow_p$-free descendant, we must consider not just free conversions but also *potentially free conversions* and secondly, as these quasi-normal forms are to be $\Rightarrow_p$-reducts, only *minimal* conversions are considered. The construction of $\Rightarrow_p$-quasi-normal forms also performs two other tasks, namely checking for possible applications of *weakening* and also ensuring that as much identification of conversions occurs as is possible.

The construction of $\Rightarrow_p$-quasi-normal forms is presented in Table 5-4 in terms of a recursively defined function $\mathtt{NF}_p$ which maps terms to sets of terms. In this table the following definitions are used:

- The set of *minimal conversions* of a term are given by:

$$\mathtt{MC}(t) = \{\sigma \in \mathtt{C}(t) \mid \neg\exists\sigma' \in \mathtt{C}(t).\sigma' < \sigma\}$$

- The set of *potentially free conversions* is denoted $\mathtt{PFC}(t)$ and defined by:

$$\{\sigma \in \mathtt{C}(t) \mid \forall u \in \mathtt{NF}_p(t/\sigma).\mathtt{BV}(\sigma, t) \cap \mathtt{FV}(u) = \emptyset\}$$

- The set of *minimal potentially free conversions* is $\mathtt{MPFC}(t) = \mathtt{MC}(t) \cap \mathtt{PFC}(t)$.

- $\mathtt{MPFC}(t)$ has an equivalence relation $\sim$ determining which conversions are to be identified. This is defined as follows:

$$\sigma_1 \sim \sigma_2 \text{ iff } \mathtt{NF}_p(t/\sigma_1) = \mathtt{NF}_p(t/\sigma_2)$$

Note that for any $\sigma \in \mathtt{C}(t), \sigma \neq \epsilon$ and so the size of t/σ is strictly smaller than that of t. Hence $\mathtt{PFC}(t)$, which is defined in terms of $\mathtt{NF}_p(t/\sigma)$, is well defined.

Lemma 5.3.8 *The set of terms $\mathtt{NF}_p(t)$ is non-empty, finite, and if $t' \in \mathtt{NF}_p(t)$, then t' is a $\Rightarrow_p^*$-reduct of t.*

Proof The proof is by induction on term size. □

Table 5–4: Definition of $\Rightarrow_p$-Quasi-Normal Forms

- It t is a variable

$$\mathtt{NF}_p(t) = \{t\}$$

- If t is a compound term and $\mathtt{MPFC}(t) = \emptyset$

$$\frac{t = \mathcal{T}(t_0, \ldots, t_n) \quad \alpha_i \in \mathtt{NF}_p(t_i)}{\mathcal{T}(\alpha_0, \ldots, \alpha_n) \in \mathtt{NF}_p\ \mathcal{T}(t_0, \ldots, t_n)}$$

- If t is a compound term and $\sigma \in \mathtt{MPFC}(t)$ then either

$$\frac{\mathtt{NF}_p(t \setminus_1 [\sigma]_\sim) = \mathtt{NF}_p(t \setminus_2 [\sigma]_\sim) \quad u \in \mathtt{NF}_p(t \setminus_1 [\sigma]_\sim)}{u \in \mathtt{NF}_p(t)}$$

or

$$\frac{\mathtt{NF}_p(t \setminus_1 [\sigma]_\sim) \neq \mathtt{NF}_p(t \setminus_2 [\sigma]_\sim) \quad \beta_i \in \mathtt{NF}_p(t \setminus_i [\sigma]_\sim) \quad \alpha \in \mathtt{NF}_p(t/\sigma)}{\mathbf{case}(\alpha, x.\beta_1, y.\beta_2) \in \mathtt{NF}_p(t)}$$

where x, y are the variables bound by the set of conversions $[\sigma]_\sim$.

Lemma 5.3.9 *Let $r : t \Rightarrow_p t'$.*

- *Let $\sigma \in \mathtt{C}(t)$ and $\sigma' \in \overline{r}(\sigma)$. Then $\sigma \in \mathtt{MPFC}(t)$ iff $\sigma' \in \mathtt{MPFC}(t')$ and for such a minimal potentially free conversion σ, $[\sigma]_\sim$ is r-closed and $\overline{r}([\sigma]_\sim) = [\sigma']_\sim$.*

- *The sets $\mathtt{NF}_p(t)$ and $\mathtt{NF}_p(t')$ are equal.*

Proof The lemma is proved simultaneously by induction on the sum of the sizes of the terms in question. That σ is minimal iff σ' is follows by induction on the definition of the function $\overline{r}$ and the fact that r preserves all conversions. By lemma 5.3.6 there is a rewrite $t/\sigma \Rightarrow_p t'/\sigma'$ and so by the induction hypothesis $\mathtt{NF}_p(t/\sigma) = \mathtt{NF}_p(t'/\sigma')$. Thus for any element u of $\mathtt{NF}_p(t/\sigma)$, $\mathtt{FV}(u) \subseteq \mathtt{FV}(t/\sigma) \cap \mathtt{FV}(t'/\sigma')$ and hence

$$\begin{aligned} x \in \mathtt{BV}(\sigma, t) \cap \mathtt{FV}(u) \quad &\text{iff} \quad x \in \mathtt{BV}(\sigma, t) \cap \mathtt{FV}(t'/\sigma') \\ &\text{iff} \quad x \in \mathtt{BV}(\sigma', t') \cap \mathtt{FV}(t'/\sigma') \\ &\text{iff} \quad x \in \mathtt{BV}(\sigma', t') \cap \mathtt{FV}(u) \end{aligned}$$

where the equality

$$\mathtt{BV}(\sigma, t) \cap \mathtt{FV}(t'/\sigma') = \mathtt{BV}(\sigma', t') \cap \mathtt{FV}(t'/\sigma')$$

may be proved by induction on the rewrite r. Thus σ is potentially free iff σ' is. Finally, given $\tau \in r^{-1}\overline{r}[\sigma]_\sim$, there is a $\tau' \in \mathtt{MPFC}(t')$ and a $\omega \in [\sigma]_\sim$ such that $t/\tau \Rightarrow_p t'/\tau'$ and $t/\omega \Rightarrow_p t'/\tau'$. Thus by the induction hypothesis

$$\mathtt{NF}_p(t/\tau) = \mathtt{NF}_p(t'/\tau') = \mathtt{NF}_p(t/\omega) = \mathtt{NF}_p(t/\sigma)$$

and so $[\sigma]_\sim$ is r-closed. The equation $\overline{r}([\sigma]_\sim) = [\sigma']_\sim$ may be proved similarly by direct calculation.

For the second half of the lemma there are two possibilities. Firstly if $\mathtt{MPFC}(t) = \emptyset$, then by the first part of this lemma $\mathtt{MPFC}(t') = \emptyset$ and so the lemma follows by the induction hypothesis. If however there is a $\sigma \in \mathtt{MPFC}(t)$ then, because $[\sigma]_\sim$ is r-closed, there are rewrites

$$t/\sigma \Rightarrow_p t'/\sigma' \quad \text{and} \quad t \setminus_1 [\sigma]_\sim \Rightarrow_p t' \setminus_1 \overline{r}([\sigma]_\sim) \quad \text{and} \quad t \setminus_2 [\sigma]_\sim \Rightarrow_p t' \setminus_2 \overline{r}([\sigma]_\sim)$$

where $\sigma' \in \overline{r}(\sigma)$ and the directions of the second and third reductions may be reversed. For each of these rewrites, the set of normal forms of the left hand side is the same as those of the right hand side. It is now routine to check that the sets of terms $\mathtt{NF}_p(t)$ and $\mathtt{NF}_p(t')$ are equal. □

Lemma 5.3.10 *The relation $\Rightarrow_p$ is confluent and has a decidable equational theory, while if $t' \in \mathtt{NF}_p(t)$ then t' is a $\Rightarrow_p$-quasi-normal form.*

Proof By lemmas 5.3.8 and 5.3.9 any $\Rightarrow_p^*$-span with redex t has a $\Rightarrow_p^*$-co-span to any element of $\mathtt{NF}_p(t)$. Thus $\Rightarrow_p$ is confluent while the associated equational theory may be decided by comparing the quasi-normal forms just constructed. Finally, if $\alpha \in \mathtt{NF}_p(t)$ and $\alpha \Rightarrow_p^* \alpha'$, then by lemma 5.3.9 $\mathtt{NF}_p(\alpha') = \mathtt{NF}_p(t)$ and by lemma 5.3.8 $\alpha' \Rightarrow_p^* \alpha$. Thus α is a $\Rightarrow_p$-quasi-normal form. □

5.3.4 Decidability of $\Rightarrow_c$-Equivalence

By lemma 5.3.5 any $\Rightarrow_c$-reduction can be expressed as a sequence of commuting conversions given in equation 5.4 and $\Rightarrow_p$-reductions. Thus the construction of

$\Rightarrow_c$-quasi-normal forms is a process of combining the $\Rightarrow_p$-quasi-normal forms just defined with the normal forms of the strongly normalising and confluent rewrite relation generated by the commuting conversions. This is a three stage process which recursively normalises all minimal conversions and then contracts all μ-commuting conversions. This produces terms which have only $\Rightarrow_p^*$-reducts and so the normalisation procedure may be finished off by using operator $\mathtt{NF}_p$ which constructs $\Rightarrow_p$-quasi-normal forms.

Recall that $\Rightarrow_\mu$ is the least pre-congruence on terms containing the reductions in equation 5.4. This relation is strongly normalising because it is a subrelation of $\Rightarrow_{\mathcal{F}}$ which is defined and shown to be strongly normalising in the next section, and local confluence of $\Rightarrow_\mu$ may be easily proved. Thus we may define $\mu(t)$ to be the unique $\Rightarrow_\mu$-normal form of t. The functions $\mathtt{NF}$ and $\mathtt{NF}^o$ which map terms to sets of terms are defined simultaneously in Table 5-5. The algorithm used to

Table 5–5: Definition of $\mathtt{NF}$ and $\mathtt{NF}^o$

- If t is a variable

$$\frac{}{\mathtt{NF}^o(t) = \{t\}}$$

- If t is a *case*-expression

$$\frac{\alpha \in \mathtt{NF}(u) \quad \beta_i \in \mathtt{NF}^o(v_i)}{\mu\mathbf{case}(\alpha, x.\beta_1, y.\beta_2) \in \mathtt{NF}^o(\mathbf{case}(u, x.v_1, y.v_2))}$$

- If t is not a *case*-expression or a variable

$$\frac{t = \mathcal{T}(t_0, \ldots, t_n) \quad \alpha_i \in \mathtt{NF}^o(t_i)}{\mathcal{T}(\alpha_0, \ldots, \alpha_n) \in \mathtt{NF}^o(t)}$$

- $\mathtt{NF}(t)$ is defined

$$\mathtt{NF}(t) = \bigcup_{\alpha \in \mathtt{NF}^o(t)} \mathtt{NF}_p(\alpha)$$

construct $\mathtt{NF}(t)$ is very similar to that used to construct the quasi-normal forms of the linear conversion relation, with the operator $\mathcal{D}$ being replaced by the more complex operator $\mathtt{NF}_p$. An alternative definition of $\mathtt{NF}^o(t)$, which will be used later, is the following:

$$\mathtt{NF}^o(t) = \{\mu t[\sigma\leftarrow\alpha_\sigma]_{\sigma\in\mathtt{MC}(t)} \mid \alpha_\sigma \in \mathtt{NF}(t/\sigma)\}$$

The structural properties which characterised the quasi-normal forms of the linear conversion relation are generalised to this calculus as follows. A term is *stable* iff $\forall\sigma \in \mathtt{C}(t).\mathtt{PFC}(t/\sigma) = \emptyset$ and a term is *strongly stable* iff it is stable, each of its subterms is strongly stable and either $0 \in \mathtt{C}(t)$ or $\mathtt{MPFC}(t) = \emptyset$.

Lemma 5.3.11 *All $\Rightarrow_p$-reducts of stable terms are stable. Thus if $\alpha \in \mathtt{NF}^o(t)$ then α is stable, while if $\alpha' \in \mathtt{NF}(t)$ then α' is strongly stable and a $\Rightarrow_c$-quasi-normal form.*

Proof Let t be stable. If $t \Rightarrow_p t'$ and there is a conversion in t' containing a potentially free sub-conversion, then there is also a conversion in t' containing a minimal potentially free sub-conversion. Thus by lemma 5.3.9 there is a conversion in t also containing a minimal potentially free conversion and so the redex can't be stable.

The stability of α is proved as in lemma 4.3.13. Since α' is an $\Rightarrow_p$-reduct of a stable term it is stable, and because it is a $\Rightarrow_p$-quasi-normal it must either be a *case*-expression or have no minimal potentially free conversions. Finally, all subterms of α' are strongly stable by the induction hypothesis and, because all $\Rightarrow_c$-reducts of stable terms are actually $\Rightarrow_p$-reducts, α' is a $\Rightarrow_c$-quasi-normal form. □

In lemma 4.3.15 it was proved that all quasi-normal forms of the linear conversion relation are of the form $\mathtt{NF}(t)$. This does not generalise as certain reductions duplicate conversions and so introduce possibilities for weakening or the further identification of conversions, e.g.

$$\begin{aligned}
&\mathbf{case}(t, x.\mathbf{case}(u, x'.r, y'.s), y.v) \\
&\quad\Rightarrow\quad \mathbf{case}(u, x'.\mathbf{case}(t, x.r, y.v), y'.\mathbf{case}(t, x.s, y.v)) \\
&\quad\Rightarrow\quad \mathbf{case}(t, x.\mathbf{case}(u, x'.r, y'.s), y.\mathbf{case}(u, x'.v, y'.v))
\end{aligned}$$

Notice the reduct contains possibilities for weakening not present in terms in the image of $\mathtt{NF}$.

Proving that if $t =_c t'$ then $\mathtt{NF}(t) = \mathtt{NF}(t')$ by explicitly considering each quasi-normal form of each term may be possible but is too time consuming. A simpler approach is to show that there are $\Rightarrow_p$-equivalent members of $\mathtt{NF}^o(t)$ and $\mathtt{NF}^o(t')$ and the key is the following technical lemma which relates commuting conversions to $\Rightarrow_c$-reduction.

Lemma 5.3.12 *If $t \Rightarrow_c t'$, then $\mu(t) \Rightarrow_c^* \mu(t')$.*

Proof The proof follows the same pattern as lemma 5.5.4. Note that the notion of a full parallel rewrite is modified to prevent only the introduction of new μ-redexes. □

Lemma 5.3.13 *Given a term t and two terms $\alpha, \alpha' \in \mathtt{NF}^o(t)$ then $\alpha \Rightarrow_p^* \alpha'$. Thus given terms $\alpha \in \mathtt{NF}^o(t)$ and $\alpha' \in \mathtt{NF}^o(t')$, if $\alpha =_p \alpha'$ then the sets $\mathtt{NF}(t)$ and $\mathtt{NF}(t')$ are equal.*

Proof The first part of the lemma is proved by induction on the term t with the only interesting part being if t is a *case*-expression, say $\mathbf{case}(t', x.u, y.v)$. Then α and α' are of the form

$$\alpha \equiv \mu(\mathbf{case}(\alpha_0, x.\beta_1, y.\beta_2)) \text{ and } \alpha' \equiv \mu(\mathbf{case}(\alpha_0', x.\beta_1', y.\beta_2'))$$

By the induction hypothesis $\beta_i \Rightarrow_p^* \beta_i'$ and $\alpha_0 =_p \alpha_0'$, and since α_0' is a $\Rightarrow_p$-quasi-normal form and $\Rightarrow_p$ is confluent, there is a reduction sequence $\alpha_0 \Rightarrow_p \alpha_0'$. The lemma now follows from lemma 5.3.12.

For the second half of the lemma, given a term $\alpha_0 \in \mathtt{NF}(t)$, there is a term $\alpha_1 \in \mathtt{NF}^o(t)$ such that $\alpha_0 \in \mathtt{NF}_p(\alpha_1)$. By the first part of this lemma $\alpha_1 =_p \alpha =_p \alpha'$ and so $\alpha_0 \in \mathtt{NF}(t')$. Thus we have shown that $\mathtt{NF}(t) \subseteq \mathtt{NF}(t')$ and the argument is symmetric and so the reverse containment also holds. □

Lemma 5.3.14 *The terms $\mathbf{case}(t, x.u, y.u)$ and u have the same set of normal forms.*

Proof If $\alpha_0 \in \mathtt{NF}(t)$ and $\alpha_1 \in \mathtt{NF}^o(u)$, then by lemma 5.3.12

$$\mu(\mathbf{case}(\alpha_0, x.\alpha_1, y.\alpha_1)) \Rightarrow_c^* \mu(\alpha_1) = \alpha_1$$

The redex is a member of $\mathtt{NF}^o(\mathbf{case}(t, x.u, y.u))$ and reduct is a member of $\mathtt{NF}^o(u)$. The lemma now follows from lemma 5.3.13. $\square$

Lemma 5.3.15 *The terms*

$$t_0 = \mathbf{case}(\mathbf{case}(t, x_1.u_1, x_2.u_2), y_1.v_1, y_2.v_2)$$

and

$$t_1 = \mathbf{case}(t, x_1.\mathbf{case}(u_1, y_1.v_1, y_2.v_2), x_2.\mathbf{case}(u_2, y_1.v_1, y_2.v_2))$$

have the same set of quasi-normal forms.

Proof Let $\alpha \in \mathtt{NF}^o(\mathbf{case}(t, x_1.u_1, x_2.u_2))$ and $\alpha_{v_i} \in \mathtt{NF}^o(v_i)$. Then given an $\alpha' \in \mathtt{NF}_p(\alpha)$, by lemma 5.3.12 there is a reduction

$$\mu(\mathbf{case}(\alpha, y_1.\alpha_{v_1}, y_2.\alpha_{v_2})) \Rightarrow_p^* \mu(\mathbf{case}(\alpha', y_1.\alpha_{v_1}, y_2.\alpha_{v_2}))$$

where the reduct is a member of $\mathtt{NF}^o(t_0)$. Now α must be of the form

$$\mu(\mathbf{case}(\alpha_t, x_1.\alpha_{u_1}, x_2.\alpha_{u_2}))$$

where $\alpha_t \in \mathtt{NF}(t)$ and $\alpha_{u_i} \in \mathtt{NF}^o(u_i)$. Again by lemma 5.3.12, if $\alpha'_{u_i} \in \mathtt{NF}_p(\alpha_{u_i})$ then there is, a $\Rightarrow_p^*$-reduction sequence:

$$\begin{aligned}
&\mu\mathbf{case}(\alpha, y_1.\alpha_{v_1}, y_2.\alpha_{v_2})\\
&\quad = \quad \mu\mathbf{case}(\mathbf{case}(\alpha_t, x_1.\alpha_{u_1}, x_2.\alpha_{u_2}), y_1.\alpha_{v_1}, y_2.\alpha_{v_2})\\
&\quad \Rightarrow_p^* \quad \mu\mathbf{case}(\alpha_t, x_1.\mathbf{case}(\alpha_{u_1}, y_1.\alpha_{v_1}, y_2.\alpha_{v_2}), x_2.\mathbf{case}(\alpha_{u_2}, y_1.\alpha_{v_1}, y_2.\alpha_{v_2}))\\
&\quad \Rightarrow_p^* \quad \mu\mathbf{case}(\alpha_t, x_1.\mathbf{case}(\alpha'_{u_1}, y_1.\alpha_{v_1}, y_2.\alpha_{v_2}), x_2.\mathbf{case}(\alpha'_{u_2}, y_1.\alpha_{v_1}, y_2.\alpha_{v_2}))
\end{aligned}$$

As the reduct of this sequence is a member of $\mathtt{NF}^o(t_1)$, we have shown that there are $\Rightarrow_p$-equivalent members of $\mathtt{NF}^o(t_0)$ and $\mathtt{NF}^o(t_1)$. Hence the normal forms of t_0 and t_1 are equal. $\square$

Lemma 5.3.16 *Let $X \subseteq \mathtt{MC}(t)$ be a non-empty consistent set of minimal free conversions. Then there is a term $\alpha \in \mathtt{NF}^o(t)$ and $\alpha' \in \mathtt{NF}^o(\mathbf{case}(t/X, x.t \setminus_1 X, y.t \setminus_2 X))$ such that $\alpha \Rightarrow_p^* \alpha'$. Thus these terms have the same set of quasi-normal forms.*

Proof Let $\alpha_X \in \mathtt{NF}(t/X)$ and define the term

$$t_0 = t[\sigma \leftarrow \alpha_\sigma]_{\sigma \Subset \mathtt{MC}(t)}$$

where α_σ are chosen members of $\mathtt{NF}(t/\sigma)$ such that if $\sigma \in X, \alpha_\sigma = \alpha_X$. Then $X \subseteq \mathtt{MC}(t_0)$ is a set of minimal, free consistent conversions and so there is a rewrite

$$t_0 \Rightarrow_p \mathbf{case}(\alpha_X, x.t_0 \setminus_1 X, y.t_0 \setminus_2 X)$$

and hence by lemma 5.3.12 there is also a reduction sequence

$$\mu t_0 \Rightarrow_c^* \mu\mathbf{case}(\alpha_X, x.\mu(t_0 \setminus_1 X), y.\mu(t_0 \setminus_2 X))$$

Now for any conversion $\tau \in \mathcal{C}(t)$, τ is minimal iff $\Omega_i(X,\tau)$ (when defined) is minimal and, providing this is the case, $(t \setminus_i X)/\Omega_i(X,\tau) = t/\tau$. Thus

$$\begin{aligned} t_0 \setminus_i X &= t[\sigma \leftarrow \alpha_\sigma]_{\sigma \Subset \mathtt{MC}(t)} \setminus_i X \\ &= (t \setminus_i X)[\Omega_i(X,\sigma) \leftarrow \alpha_\sigma]_{\sigma \Subset \mathtt{MC}(t)} \\ &= (t \setminus_i X)[\sigma' \leftarrow \alpha_{\sigma'}]_{\sigma' \Subset \mathtt{MC}(t \setminus_i X)} \end{aligned}$$

and hence $\mu(t_0 \setminus_i X) \in \mathtt{NF}^o(t \setminus_i X)$. Thus the term $\mu(t_0)$ is a member of $\mathtt{NF}^o(t)$ and rewrites to a member of $\mathtt{NF}^o(\mathbf{case}(t/X, x.t \setminus_1 X, y.t \setminus_2 X))$ and so the lemma is proven. □

Theorem 5.3.17 *If $r : t \Rightarrow_c t'$ then $\mathtt{NF}(t) = \mathtt{NF}(t')$. Hence the conversion relation is confluent and conversion-equivalence decidable.*

Proof Induction on the label r of the rewrite $r : t \Rightarrow_c t'$ is used to show that $\mathtt{NF}^o(t)$ and $\mathtt{NF}^o(t')$ have $\Rightarrow_p$-equivalent members. If the rewrite is a top-level rewrite then the lemma follows by lemmas 5.3.14,5.3.15 and 5.3.16. If however r is a rewrite of a strict subterm then the lemma follows easily by the induction hypothesis and lemma 5.3.12. Decidability follows as one can prove $t =_c t'$ by simply enumerating and comparing the sets $\mathtt{NF}(t)$ and $\mathtt{NF}(t')$, while given a span from t, a co-span can be constructed to any member of $\mathtt{NF}(t)$ and thus the conversion relation is confluent. □

5.4 An Extension of β-Reduction

A rewrite relation consisting of β-reductions, commuting conversions and restricted η-expansions is defined which, when taken together with the conversion relation of the previous section, generates the same equational theory as the full expansionary rewrite relation. While local confluence of this relation is a straightforward generalisation of the proof of local confluence for the restricted rewrite relation of the simply typed λ-calculus, several technical innovations are required to generalise the proof of strong normalisation.

The most difficult problem is caused by the *case*-expression which, unlike a λ-abstraction, binds variables whose type is unrelated to the type of the *case*-expression as a whole. Thus when we try to prove analogue of lemma 3.3.5 for *case*-expressions, if a reduction $t \Rightarrow_{\mathcal{I}} t'$ fails to induce a reduction $t[u/x] \Rightarrow_{\mathcal{I}} t'[u/x]$, one may still construct a reduction sequence $t[\eta(u)/x] \Rightarrow_{\mathcal{I}}^{*} t'[u/x]$, but since the variable x is of arbitrary type, the induction hypothesis can no longer guarantee the validity of $t[\eta(u)/x]$.

The solution adopted here is to remove the non-substitutive reduction sequences caused by the variable binders in the *case*-expression by altering the associated β_{+}-reductions. Firstly an operator $\mathcal{I}$, short for the identity, is introduced on terms

$$\mathcal{I}(t) = (\lambda x.x)t$$

and this operator is then used to redefine the β-reductions for the sum type as follows:

$$\begin{array}{lrcl} (\beta'_{+,1}) & \mathbf{case}(\mathbf{in}_1(t), x.u, y.v) & \Rightarrow & u[\mathcal{I}t/x] \\ (\beta'_{+,2}) & \mathbf{case}(\mathbf{in}_2(t), x.u, y.v) & \Rightarrow & v[\mathcal{I}t/y] \end{array}$$

Notice how these new reductions substitute application terms, which — because application terms are expandable forms — removes the problems associated with non-substitutive reductions. Since the $\beta_{\rightarrow}$-redex remains unchanged, there is a β-reduction $\mathcal{I}(t) \Rightarrow t$ and so these new reductions generate the same equational theory and have the same normal forms as the rewrite relation containing the more traditional β_+-reductions.

The second major difference lies in how the termination order defined in terms of $\Rightarrow_{\mathcal{I}}$ is extended to $\Rightarrow_{\mathcal{F}}$, i.e. how we prove that if a term is valid then so is its η-expansion. While this may be done immediately for final types such as the product and exponential, for initial types one must consider not just the η-expansion of a term but also all other terms of the form:

$$\Delta(t) = \Delta(z)[t/z]$$

where $\Delta(z)$ is the set consisting of all the $\Rightarrow^*_{\mathcal{F}}$ reducts of the variable z.

Finally, because the elimination term of initial type constructors is used to represent certain reduction sequences as described in equation 5.2, it is the *case*-expression which provides the basis of normal forms for terms of sum types. This is manifest in the proof of strong normalisation where the validity predicates are established first for certain *case*-expressions, called *quasi-introduction* terms, before being established for terms in general.

The reduction relation $\Rightarrow_{\beta'}$ is the least pre-congruence containing the basic reductions in Table 5-6. The rewrite rules $\beta_{\rightarrow}$, $\beta_{+,\rightarrow}$, $\beta'_{+,i}$ and $\beta_{+,+}$ may involve implicit

Table 5–6: Basic Reductions for $\Rightarrow_{\beta'}$

$\beta_{\times,i}$	$\pi_i\langle u_0, u_1\rangle$	$\Rightarrow$	u_i
$\beta_{+,\times}$	$\pi_i\mathbf{case}(t, x.u, y.v)$	$\Rightarrow$	$\mathbf{case}(t, x.\pi_i u, y.\pi_i v)$
$\beta_{\rightarrow}$	$(\lambda x.t)t'$	$\Rightarrow$	$t[t'/x]$
$\beta_{+,\rightarrow}$	$\mathbf{case}(t, x.u, y.v)t'$	$\Rightarrow$	$\mathbf{case}(t, x.ut', y.vt')$
$\beta'_{+,1}$	$\mathbf{case}(\mathbf{in}_1(t), x.u, y.v)$	$\Rightarrow$	$u[\mathcal{I}t/x]$
$\beta'_{+,2}$	$\mathbf{case}(\mathbf{in}_2(t), x.u, y.v)$	$\Rightarrow$	$v[\mathcal{I}t/y]$

$$\beta_{+,+} \qquad \mathbf{case}(\mathbf{case}(t, x.u, y.v), x'.u', y'.v') \quad \Rightarrow$$
$$\mathbf{case}(t, x.\mathbf{case}(u, x'.u', y'.v'), y.\mathbf{case}(v, x'.u', y'.v'))$$

α-conversion to avoid variable capture. The rewrite relation obtained by replacing the altered β'-reductions with the more traditional β-reductions is denoted $\Rightarrow_\beta$.

The following lemma shows that our new β'-reductions, which were introduced solely to facilitate the proof of strong normalisation, do not change the equational theory or normal forms.

Lemma 5.4.1 *The rewrite relations $\Rightarrow_{\beta'}$ and $\Rightarrow_{\beta}$ generate the same equational theory, share the same normal forms, and strong normalisation of $\Rightarrow_{\beta'}$ implies strong normalisation of $\Rightarrow_{\beta}$.*

Proof Both parts of the lemma follow upon observation that if $t \Rightarrow_{\beta} t'$ then there is a reduction sequence $t \Rightarrow^{+}_{\beta'} t'$ of non-zero length, while if $t \Rightarrow_{\beta'} t'$ then there is a term t'' and reduction sequences $t \Rightarrow^{+}_{\beta} t''$ and $t' \Rightarrow^{*}_{\beta} t''$. $\square$

The η-expansion of a term depends on its type and is defined as follows:

$$\eta(t) = \begin{cases} t & \text{if } t \text{ is of base type} \\ \langle \pi_0 t, \pi_1 t \rangle & \text{if } t \text{ is of product type} \\ \mathbf{case}(t, x.\mathbf{in}_1(x), y.\mathbf{in}_2(y)) & \text{if } t \text{ is of sum type} \\ * & \text{if } t \text{ is of unit type} \\ \lambda x.tx & \text{if } t \text{ is of function type} \end{cases}$$

The η-expansion of terms of sum type given above is a special case of the η_{+}-rewrite rule derived at the beginning of this chapter and converts a (sub)term of sum type into a conversion which may then be expanded by the conversion relation. Uncontrolled η-expansion is clearly not strongly normalising and so restrictions must be imposed on their scope and in particular the expansions appearing in the triangle laws must be prevented. For the sum type, these triangle laws assert that expanding an injection term or a subterm which occurs negatively, i.e. as the first argument of a *case*-expression, forms a looping reduction. However prohibiting these expansions is as yet insufficient to obtain a strongly normalising relation as the η-expansion of a *case*-expression of sum type reduces to the term obtained by expanding the arms of the *case*-expression, and, if these arms are injections, a reduction loop may be formed. For example:

$$\begin{aligned}\mathbf{case}(t, x.\mathbf{in}_1(x), y.\mathbf{in}_2(y)) &\Rightarrow_\eta \eta(\mathbf{case}(t, x.\mathbf{in}_1(x), y.\mathbf{in}_2(y)))\\ &\Rightarrow_{\beta'} \mathbf{case}(t, x.\eta(\mathbf{in}_1(x)), y.\eta(\mathbf{in}_2(y)))\\ &\Rightarrow^*_{\beta'} \mathbf{case}(t, x.\mathbf{in}_1(x), y.\mathbf{in}_2(y))\end{aligned} \tag{5.6}$$

Such terms are called *quasi-introduction* terms and the set of terms which may not be expanded is enlarged to include them. The *quasi-introduction* terms of sum type are defined by the syntax:

$$q := \mathbf{in}_1(t) \mid \mathbf{in}_2(t) \mid \mathbf{case}(t, x.q, y.q')$$

where t ranges over arbitrary terms and q, q' over arbitrary quasi-introduction terms of sum type. Given any quasi-introduction term of sum type, the introduction terms at its leafs are extracted by the function *Arm* which is defined as follows:

$$Arm(t) = \begin{cases} Arm(t_1) \cup Arm(t_2) & \text{if } t = \mathbf{case}(u, x.t_1, y.t_2) \\ \{t\} & \text{otherwise} \end{cases}$$

Because the η-expansion of terms such as $\mathbf{case}(u, x.\lambda x'.t_1, y.\lambda y'.t_2)$ do not create a reduction loop as in equation 5.6, such terms are regarded as expandable and so the quasi-introduction terms of product, exponential and unit type are defined to be just the introduction terms of that type.

A term is *expandable* providing it is neither a quasi-introduction term nor of base type. The reduction relation $\Rightarrow_{\mathcal{F}}$ is defined simultaneously with a subrelation $\Rightarrow_{\mathcal{I}}$ which is guaranteed not to be a top-level expansion — hence a negatively occuring subterm may be $\Rightarrow_{\mathcal{I}}$-rewritten when a $\Rightarrow_{\mathcal{F}}$-reduction may create a reduction loop. The definitions of these relations are contained in Table 5-7, where $\Rightarrow_{\mathcal{I}\cup\mathcal{F}}$ and $\Rightarrow_{\mathcal{I}\cap\mathcal{F}}$ are the union and intersection of $\Rightarrow_{\mathcal{I}}$ and $\Rightarrow_{\mathcal{F}}$ respectively. The following properties of the relations $\Rightarrow_{\mathcal{F}}$ and $\Rightarrow_{\mathcal{I}}$ will be needed later.

Lemma 5.4.2 *The following are true:*

- *If $t \Rightarrow_{\mathcal{F}} t'$ then either there is a reduction $t \Rightarrow_{\mathcal{I}} t'$ or $t' = \eta(t)$.*

Table 5–7: The Restricted Rewrite Relation

$$\frac{t \quad \text{expandable}}{t \Rightarrow_{\mathcal{F}} \eta(t)} \qquad \frac{t \Rightarrow_{\beta'} t'}{t \Rightarrow_{\mathcal{I}\cap\mathcal{F}} t'}$$

$$\frac{u \Rightarrow_{\mathcal{I}\cup\mathcal{F}} u'}{\langle u, v\rangle \Rightarrow_{\mathcal{I}\cap\mathcal{F}} \langle u', v\rangle} \qquad \frac{v \Rightarrow_{\mathcal{I}\cup\mathcal{F}} v'}{\langle u, v\rangle \Rightarrow_{\mathcal{I}\cap\mathcal{F}} \langle u, v'\rangle}$$

$$\frac{t \Rightarrow_{\mathcal{I}} t'}{\pi_0 t \Rightarrow_{\mathcal{I}\cap\mathcal{F}} \pi_0 t'} \qquad \frac{t \Rightarrow_{\mathcal{I}} t'}{\pi_1 t \Rightarrow_{\mathcal{I}\cap\mathcal{F}} \pi_1 t'}$$

$$\frac{t \Rightarrow_{\mathcal{I}} t'}{tu \Rightarrow_{\mathcal{I}\cap\mathcal{F}} t'u} \qquad \frac{u \Rightarrow_{\mathcal{I}\cup\mathcal{F}} u'}{tu \Rightarrow_{\mathcal{I}\cap\mathcal{F}} tu'}$$

$$\frac{t \Rightarrow_{\mathcal{I}\cup\mathcal{F}} t'}{\lambda x.t \Rightarrow_{\mathcal{I}\cap\mathcal{F}} \lambda x.t'} \qquad \frac{t \Rightarrow_{\mathcal{I}} t'}{\mathbf{case}(t, x.u, y.v) \Rightarrow_{\mathcal{I}\cap\mathcal{F}} \mathbf{case}(t', x.u, y.v)}$$

$$\frac{u \Rightarrow_{\mathcal{I}\cup\mathcal{F}} u'}{\mathbf{case}(t, x.u, y.v) \Rightarrow_{\mathcal{I}\cap\mathcal{F}} \mathbf{case}(t, x.u', y.v)} \qquad \frac{v \Rightarrow_{\mathcal{I}\cup\mathcal{F}} v'}{\mathbf{case}(t, x.u, y.v) \Rightarrow_{\mathcal{I}\cap\mathcal{F}} \mathbf{case}(t, x.u, y.v')}$$

$$\frac{t \Rightarrow_{\mathcal{I}\cup\mathcal{F}} t'}{\mathbf{in}_1(t) \Rightarrow_{\mathcal{I}\cap\mathcal{F}} \mathbf{in}_1(t')} \qquad \frac{t \Rightarrow_{\mathcal{I}\cup\mathcal{F}} t'}{\mathbf{in}_2(t) \Rightarrow_{\mathcal{I}\cap\mathcal{F}} \mathbf{in}_2(t')}$$

- *If t is a quasi-introduction term, then so are its $\Rightarrow_{\mathcal{F}}$-reducts and $\eta(t) \Rightarrow^*_{\beta'} t$.*

Proof Both parts of the lemma are trivial inductions on t. □

As with previous calculi we may structurally characterise the $\Rightarrow_{\mathcal{F}}$-normal forms.

Lemma 5.4.3 *The normal forms of $\Rightarrow_{\mathcal{F}}$ are exactly those terms which are β-normal forms and each of whose subterms are either of base type, occur negatively, or a quasi-introduction term.*

Proof The proof is the obvious generalisation of lemma 3.3.16. □

Substitutivity and Local Confluence

The relations $\Rightarrow_{\mathcal{I}}$ and $\Rightarrow_{\mathcal{F}}$ are not pre-congruences and this complicates traditional normalisation and confluence proofs. As with previous calculi we characterise how substitutivity may fail.

Lemma 5.4.4 *Let t, t', u and u' be terms such that $t \Rightarrow_{\mathcal{R}} t'$ and $u \Rightarrow_{\mathcal{R}} u'$, where $\mathcal{R} \in \{\mathcal{I}, \mathcal{F}\}$. Then*

- *There is a rewrite $t[u/x] \Rightarrow_{\mathcal{R}} t'[u/x]$ unless u is a quasi-introduction term and t' is obtained by expanding an occurrence of x in t. In this case there are reduction sequences $t[\eta(u)/x] \Rightarrow^*_{\mathcal{I}} t'[u/x] \Rightarrow^*_{\mathcal{I}} t[u/x]$.*
- *There is a rewrite $t[u/x] \Rightarrow^*_{\mathcal{I}} t[u'/x]$ unless $u' = \eta(u)$ and either $t = x$ or there are negative occurrences of x in t. In this latter case $t[u'/x]$ and $t[u/x]$ have a common $\Rightarrow^*_{\mathcal{I}}$-reduct.*

Proof Induction on the definition of the rewrites. □

As explained before care must be taken in establishing local confluence of $\Rightarrow_{\mathcal{F}}$ because of the restrictions on expansion and because $\Rightarrow_{\mathcal{I}}$ is not locally confluent.

Lemma 5.4.5 *The rewrite relation $\Rightarrow_{\mathcal{F}}$ is locally confluent.*

Proof The proof is a straightforward generalisation of that presented for the simply typed λ-calculus in lemma 3.3.3. □

The substitutivity lemma 5.4.4 also allows us to give a simple definition of the $\Rightarrow^*_{\mathcal{F}}$ reducts of a variable. Note that for the rest of this section we shall assume w.l.o.g. that variables, and hence terms, have unique types. A function Δ, which maps variables to sets of terms, is defined inductively over the type of the variable as follows:

$\Delta(z) = \{z\}$	z has base type
$\Delta(z) = \{z, *\}$	z has unit type
$\Delta(z) = \{z\} \cup \{\langle \alpha[\pi_0 z/x], \alpha'[\pi_1 z/y]\rangle \mid \alpha \in \Delta(x) \text{ and } \alpha' \in \Delta(y)\}$	z has prod. type
$\Delta(z) = \{z\} \cup \{\lambda x.\alpha'[z\alpha/y] \mid \alpha \in \Delta(x) \text{ and } \alpha' \in \Delta(y)\}$	z has fun. type

and if z has sum type, then

$$\Delta(z) = \{z\} \cup \{\mathbf{case}(z, x.\mathbf{in}_1(\alpha), y.\mathbf{in}_2(\alpha')) \mid \alpha \in \Delta(x) \text{ and } \alpha' \in \Delta(y)\}$$

where the variables x and y have the appropriate type. The function Δ is extended to terms by:

$$\Delta(t) = \{t_0[t/z] \mid t_0 \in \Delta(z)\}$$

Lemma 5.4.6 *The function Δ gives the rewrites of a variable, i.e. $\Delta(z) = z/\Rightarrow_{\mathcal{F}}^*$.*

Proof Two containments must be established, both by induction over type structure. The forward inclusion is proved first. If z is of base type or unit type, the lemma is trivial. If z is of sum type, then by the induction hypothesis there is the following reduction sequence:

$$z \Rightarrow_{\mathcal{F}} \mathbf{case}(z, x.\mathbf{in}_1(x), y.\mathbf{in}_2(y)) \Rightarrow_{\mathcal{I}}^* \mathbf{case}(z, x.\mathbf{in}_1(\alpha), y.\mathbf{in}_2(\alpha'))$$

while if z is of exponential type

$$z \Rightarrow_{\mathcal{F}} \lambda x.zx \Rightarrow_{\mathcal{I}}^* \lambda x.z\alpha \Rightarrow_{\mathcal{I}}^* \lambda x.\alpha'[z\alpha/y]$$

where the last part of the reduction sequence exists as $z\alpha$ is an expandable form and a similar argument holds for product types. The reverse containment is proved by showing that the set $\Delta(z)$ is closed under reduction. Again this is easy for base, unit and also for sum types. The one-step reducts of $\lambda x.\alpha'[z\alpha/y] \in \Delta(z)$ are all caused by $\Rightarrow_{\mathcal{F}}$ reducts of either α or α' and hence by the induction hypothesis all reducts of $\lambda x.\alpha'[z\alpha/y]$ are contained in the set $\Delta(z)$. A similar argument suffices for product types. □

5.4.1 Strong Normalisation of $\Rightarrow_{\mathcal{F}}$

The relation $\Rightarrow_{\mathcal{F}}$ is proved strongly normalising by extending the proof of strong normalisation for the restricted rewrite relation of the simply typed λ-calculus (corollary 3.3.13). A termination order is built inductively over the type structure and the relation $\Rightarrow_{\mathcal{I}}$ as follows. Given a type T, the set of *valid* terms of that type is denoted $V(T)$ and defined to be those terms of type T which can be shown valid by the inference rules in Table 5-9 at the end of this chapter. This definition of validity is the natural extension of that given for the simply typed λ-calculus, with the only new innovations being the side conditions for validity of *case*-expressions of function/product type which are required to deal with the commuting conversions. For the reasons outlined in Chapter 3, we may associate to each valid term t a natural number, called the *validity rank* of t and defined as follows:

$$\nu(t) = 1 + max\{\nu(t') \mid t \Rightarrow_{\mathcal{I}} t'\}$$

An important remark is that, as variables have no $\Rightarrow_{\mathcal{I}}$-reducts, the valid terms of a given type contain all the variables of that type. This observation will be needed to prove that the valid terms of a given type satisfy the following *validity predicates* defined over sets of terms P:

V1: If $t \in P$ then t is $\Rightarrow_{\mathcal{F}}$-strongly normalising

V2: If $t \in P$ and $t \Rightarrow_{\mathcal{F}} t'$ then t' is valid

V3: If $t \in P$ then $\Delta(t) \subseteq P$

V4: If $t \in P$ then $\mathcal{I}(t) \in P$

Although the predicate V4 has been included as a separate predicate, if P is the set of valid terms of some type, then V4 is actually a consequence of the first three validity predicates.

Lemma 5.4.7 *Assume the valid terms of a given type satisfy the validity predicates V1-3 and let t be a valid term of that type. Then the term $\mathcal{I}(t)$ is also valid.*

Proof We prove the stronger assertion that if x is any variable having the same type as t, then for any $\alpha \in \Delta(x)$ the term $(\lambda x.\alpha)(t)$ is valid. By V3 the term α is valid and so by V1 the terms α and t are strongly normalising and hence the sum of their normalisation ranks may be used as an induction rank. The $\Rightarrow_{\mathcal{I}}$-reducts of $(\lambda x.\alpha)(t)$ induced by reductions of α or t are valid by the induction hypothesis while the only other $\Rightarrow_{\mathcal{I}}$-reduct is $\alpha[t/x]$ which is a member of $\Delta(t)$ and so valid by V3. $\square$

Induction on the type structure is used to show that the valid terms satisfy the validity predicates V1-3.

Sum Types

The first two validity predicates are easily established for quasi-normal forms as such terms are non-expandable.

Lemma 5.4.8 *Assume $V(X)$ and $V(Y)$ satisfy V1, V2 and V3. Then the set of valid quasi-introduction terms of type $X+Y$ satisfies V1 and V2. Also if $u:X$ and $v:Y$ are valid, then so are $\mathbf{in}_1(u)$ and $\mathbf{in}_2(v)$.*

Proof Since quasi-introduction terms are non-expandable and closed under reduction, the first part of the lemma follows by induction on validity. The second half follows by induction on the sum of the normalisation ranks of u and v. $\square$

Lemma 5.4.9 *Assume $V(X)$ and $V(Y)$ satisfy the validity predicates V1-3 and let t be a valid term of type $X+Y$.*

- *If $t = \mathbf{case}(t_0, x.u, y.v)$, $\eta(u) \Rightarrow^*_{\mathcal{F}} \alpha$ and $\eta(v) \Rightarrow^*_{\mathcal{F}} \beta$ then $\mathbf{case}(t_0, x.\alpha, y.\beta)$ is valid.*
- *All terms $t' \in \Delta(t)$ are valid.*

Proof The proof is by simultaneous induction on the validity of t.

(i) By the induction hypothesis the terms $\eta(u)$ and $\eta(v)$ are both valid, quasi-introduction terms and hence so are α and β. Thus the sum of their normalisation ranks forms an inner induction rank. Those $\Rightarrow_{\mathcal{I}}$-reducts of $\mathbf{case}(t_0, x.\alpha, y.\beta)$ induced by reductions of proper subterms are valid by the induction hypothesis and this leaves two cases. If t_0 is an introduction term, and say $\mathbf{case}(\mathbf{in}_1(s), x.\alpha, y.\beta) \Rightarrow_{\mathcal{I}} \alpha[\mathcal{I}s/x]$ then by the induction hypothesis $\eta(u[\mathcal{I}s/x]) = \eta(u)[\mathcal{I}s/x]$ is a valid quasi-introduction term. As $\alpha[\mathcal{I}s/x]$ is a reduct of this term it is also valid. Similarly if t_0 is a *case*-expression, the result of a commuting conversion is shown valid by applying the induction hypothesis to the term obtained by contracting the top level commuting conversion in t.

(ii) We must prove that $\mathbf{case}(t, x.\mathbf{in}_1(u), y.\mathbf{in}_2(v))$ is valid where $u \in \Delta(x)$ and $v \in \Delta(y)$. By the induction hypothesis u and v are valid and strongly normalising and hence the sum of their normalisation ranks forms an inner induction rank. The validity of $\mathbf{in}_1(u)$ and $\mathbf{in}_2(v)$ follow from the validity of u and v while those $\Rightarrow_{\mathcal{I}}$-reducts induced by reductions of proper subterms are valid by the induction hypothesis. The result of a top-level commuting conversion is valid by the first half of this lemma and, finally, if t is an injection, say $\mathbf{in}_1(t_0)$, then there is a reduction

$$\mathbf{case}(\mathbf{in}_1(t_0), x.\mathbf{in}_1(u), y.\mathbf{in}_2(v)) \Rightarrow_{\mathcal{F}} \mathbf{in}_1(u)[\mathcal{I}t_0/x] = \mathbf{in}_1((u[\mathcal{I}t_0/x]))$$

Now t_0 is a valid term of type X and hence $\mathcal{I}t_0$ is also valid. Thus, by V3, $u[\mathcal{I}t_0/x]$ is valid and hence $\mathbf{in}_1(u)[\mathcal{I}t_0/x]$ is also valid.

□

Corollary 5.4.10 *Assume $V(X)$ and $V(Y)$ satisfy the validity predicates V1-3. Then so do the valid terms of type $X + Y$.*

Proof Let t be a term. The lemma is established by induction on the validity of t. All $\Rightarrow_{\mathcal{I}}$-reducts are valid and, by the induction hypothesis, strongly normalising. The only other reduct is a valid quasi-introduction term which is also strongly normalising. Thus all reducts of t are strongly normalising and hence so is t. The $\Rightarrow_{\mathcal{I}}$-reducts of a term are valid by definition, while the result of a basic expansion has already been shown valid. Finally, V3 has just been established in lemma 5.4.9. □

Exponential Types

Lemma 5.4.11 *Assume the validity predicates V1-3 hold for the set of terms $V(X)$ and $V(Y)$. If for all terms $u \in V(X), t[u/x] \in V(Y)$ then $\lambda x.t$ is valid.*

Proof The proof is exactly as in lemma 3.3.5. □

Lemma 5.4.12 *Assume the valid terms of type X and Y satisfy the validity predicates V1-3. If $t_0 \in V(X \to Y)$ and $u \in V(X)$, then $t_0 u \in V(Y)$.*

Proof The lemma is proved exactly as in lemma 3.3.6, except that we must also consider the possibility of a top-level commuting conversion. However the validity of such a reduct is guaranteed by the validity of t_0. □

Lemma 5.4.13 *Assume the valid terms of type X and Y satisfy the validity predicates V1-V3. Then so do the valid terms of type $X \to Y$.*

Proof If t is a valid term, then so is tx. Thus tx is strongly normalising and hence, just as for the simply typed λ-calculus, so is t. All $\Rightarrow_{\mathcal{I}}$-reducts of a valid term are valid by definition, while the result of a basic expansion is valid by V3.

To prove $\lambda x.u[tv/y]$ is valid where $v \in \Delta(x), u \in \Delta(y)$, consider a valid term s of type X. The term $v[s/x] \in \Delta(s)$ is a valid term by V3 and thus $tv[s/x]$ is also valid. Finally, since

$$(u[tv/y])[s/x] = u[tv[s/x]/y]$$

and this latter term is in $\Delta(tv[s/x])$, the proof is complete. □

Lemma 5.4.14 *The set of valid terms of every type satisfy the three validity predicates V1, V2 and V3.*

Proof The proof is by induction on the type of a term. Terms of base type are proved strongly normalising by induction on their validity and because such terms have no expansions the predicates V2 and V3 are also satisfied. Similar remarks apply to terms of unit type as their only expansion is the valid constant $*$ which is also a normal form and so strongly normalising. Terms of sum type and function type have just been shown to satisfy the validity predicates while the arguments for terms of product type are similar to those for the exponential types. □

In the course of showing that valid terms satisfy the predicates V1-3 the validity criteria for λ-abstractions, injections etc. have been simplified and, before showing all terms are valid, the same must be done for *case*-expressions.

Lemma 5.4.15 *The term* $\mathbf{case}(t, x.u, y.v)$ *is valid iff* t *is strongly normalising,* u, v *are valid, and in addition if* $t \Rightarrow_{\mathcal{I}}^{*} t'$ *and* $\mathbf{in}_1(\alpha) \in \mathrm{Arm}(t')$ *then* $u[\mathcal{I}\alpha/x]$ *is valid, with a similar condition for right injections.*

Proof The proof is by induction with rank the quadruple of numbers (a, b, c, d), where a is the complexity of the type of the *case*-expression, b is the $\Rightarrow_{\mathcal{F}}$-normalisation rank of t, c is the size of t and d is the sum of the $\Rightarrow_{\mathcal{F}}$-normalisation ranks of u and v. There are two proof obligations. Firstly if the *case*-expression is of sum or function type, the clauses pertaining to commuting conversions are easily established by the induction hypothesis. Secondly, those $\Rightarrow_{\mathcal{I}}$-reducts induced by reductions of subterms are valid by the induction hypothesis with the second part of the induction hypothesis following from substitutivity considerations, while a basic β-reduction has a valid reduct by assumption. Finally, if t is a *case*-expression then the result of a basic commuting conversion is shown valid by first using the induction hypothesis to prove the arms valid and then once more for the whole term. □

Finally, all terms are shown valid in the traditional manner:

Lemma 5.4.16 *Let* t *be a term and* u_i *be valid terms. Then the term* $t[u_i/x_i]$ *is a valid term.*

Proof The proof is by induction on t and follows the standard pattern. The only interesting part is for the term $\mathbf{case}(t', x.u, y.v)$. The terms $u[u_i/x_i]$, $v[u_i/x_i]$ and $t'[u_i/x_i]$ are valid and thus strongly normalising by the induction hypothesis. Thus so is any $t'' \in (t[u_i/x_i])/\Rightarrow_{\mathcal{I}}^{*}$ and hence any $\mathbf{in}_1(\alpha) \in \mathit{Arm}(t'')$. From this we may deduce α is also valid and thus by the induction hypothesis

$$(u[u_i/x_i])[\mathcal{I}\alpha/x] = u[u_i/x_i, \mathcal{I}\alpha/x]$$

is a valid term. $\square$

Theorem 5.4.17 *The relations $\Rightarrow_{\mathcal{F}}$ and $\Rightarrow_{\mathcal{I}}$ are strongly normalising and $\Rightarrow_{\mathcal{F}}$ is confluent.*

Proof As variables are valid, all terms are valid and hence $\Rightarrow_{\mathcal{F}}$-strongly normalising. Confluence now follows from local confluence and the strong normalisation property of $\Rightarrow_{\mathcal{F}}$. $\square$

5.5 Decidability of $\beta\eta$-Equality

The expansionary rewrite relation defined at the beginning of this chapter has been decomposed into the strongly normalising and confluent relation $\Rightarrow_{\mathcal{F}}$ and the decidable conversion relation. The rest of this chapter proves that the expansionary rewrite relation is itself decidable by showing that if two terms are $\beta\eta$-equivalent, then their $\Rightarrow_{\mathcal{F}}$-normal forms are equivalent in the conversion relation.

The easiest proof strategy would be to consider the β-reductions of a term in isolation from the possibilities for η-expansion that exist within the term and, as explained in the previous chapter, this involves removing those η-expansions which introduce new β-redexes and defining a couple of new reductions to simulate the effect of some of these lost expansions. Define two functions $\eta\mathcal{F}$ and $\eta\mathcal{I}$ on terms as follows:

$$
\begin{aligned}
\eta\mathcal{I}(x) &= x \\
\eta\mathcal{I}(*) &= * \\
\eta\mathcal{I}(uv) &= \eta\mathcal{I}(u)\eta\mathcal{F}(v) \\
\eta\mathcal{I}(\lambda x.t) &= \lambda x.\eta\mathcal{F}(t) \\
\eta\mathcal{I}(\pi_i t) &= \pi_i\eta\mathcal{I}(t) \\
\eta\mathcal{I}(\langle u, v\rangle) &= \langle \eta\mathcal{F}(u), \eta\mathcal{F}(v)\rangle \\
\eta\mathcal{I}(\mathbf{in}_i(t)) &= \mathbf{in}_i(\eta\mathcal{F}(t)) \\
\eta\mathcal{I}(\mathbf{case}(t, x.u, y.v)) &= \mathbf{case}(\eta\mathcal{I}(t), x.\eta\mathcal{I}(u), y.\eta\mathcal{I}(v))
\end{aligned}
$$

and

$$\eta\mathcal{F}(t) = \begin{cases} \mathbf{case}(\eta\mathcal{I}(t'), x.\eta\mathcal{F}(u), y.\eta\mathcal{F}(v)) & t \text{ is } \mathbf{case}(t', x.u, y.v) \\ \Delta^m(z)[\eta\mathcal{I}(t)/z] & t \text{ is a projection, application, variable} \\ t & \text{otherwise} \end{cases}$$

where in the definition of $\eta\mathcal{F}(t)$, z is any variable having the same type as t and $\Delta^m(z)$ is the $\Rightarrow_{\mathcal{F}}$-normal form of z (the superscript m is to distinguish the term $\Delta^m(z)$ from the set of terms $\Delta(z)$). Also define the rewrite relation $\Rightarrow_\delta$ to be the least pre-congruence containing the reductions

$$\begin{aligned} \mathbf{case}(t, x.\lambda x'.u, y.\lambda x'.v) &\Rightarrow \lambda x'.\mathbf{case}(t, x.u, y.v) \\ \mathbf{case}(t, x.\langle u_1, v_1\rangle, y.\langle u_2, v_2\rangle) &\Rightarrow \langle \mathbf{case}(t, x.u_1, y.u_2), \mathbf{case}(t, x.v_1, y.v_2)\rangle \end{aligned}$$

Lemma 5.5.1 *The $\Rightarrow_{\mathcal{F}}$-normal form of a term t may be calculated by: (i) calculating the β-normal form of t; (ii) applying the function $\eta\mathcal{F}$; and (iii) calculating the $\Rightarrow_\delta$-normal form of the result.*

Proof The proof is a straightforward generalisation of that given for the linear λ-calculus in lemma 4.5.2. □

5.5.1 Embedding $\Rightarrow_c$ into β-Normal Forms

For the rest of this chapter let $\beta(t)$ be the β-normal forms of t. We shall define a relation $\twoheadrightarrow$ by parallelising the conversion relation and prove that if $t \twoheadrightarrow t'$ then $\beta(t) \twoheadrightarrow \beta(t')$. In order to prove this, the parallelised conversion relation must permit the expansion of empty sets of conversions and so a parallelised version of the *weakening* clause is not needed. *Parallel conversion*, denoted $\twoheadrightarrow$, is defined to be the least pre-congruence defined by the inference rules in Table 5-8. The *full parallel conversion* relation is defined exactly as the full linear parallel conversion relation was in Table 4-7, i.e. the left-branch of any instance of expansion or a elimination pre-congruence must not itself be an expansion. As with the linear parallel conversion relation, these restrictions are sufficient to prevent $\twoheadrightarrow$ destroying β-redexes by introducing new commuting conversions.

Table 5–8: Parallel Conversion

- Identity

$$\overline{z \twoheadrightarrow z}$$

- Expansion

$$\frac{X \subseteq \mathtt{FC}(t) \text{ is consistent} \quad t/X \twoheadrightarrow u \quad t \setminus_i X \twoheadrightarrow v_i}{t \twoheadrightarrow \mathbf{case}(u, x.v_1, y.v_2)}$$

where x, y are the variables bound by X. If however X is empty, then $x, y \not\in \mathtt{FV}(t)$ and u is any term of the appropriate type.

- A pre-congruence rule for each term constructor

$$\frac{u_0 \twoheadrightarrow u_0', \ldots, u_n \twoheadrightarrow u_n'}{\mathcal{T}(u_0, \ldots, u_n) \twoheadrightarrow \mathcal{T}(u_0', \ldots, u_n')}$$

- A parallel conversion $t \twoheadrightarrow t'$ is said to be *full*, and written $t \twoheadrightarrow_f t'$, iff the left branch of any expansion or elimination pre-congruence does not itself end in an expansion.

Lemma 5.5.2 *If there is a term judgement $\Gamma \vdash t : A$ and a rewrite $t \twoheadrightarrow t'$ then there is also a term judgement $\Gamma \vdash t' : A$.*

Proof Induction on the rewrite $t \twoheadrightarrow t'$ is used to show that t and t' are equivalent in the conversion relation. The lemma then follows from lemma 5.3.2. □

Lemma 5.5.3 *Parallel expansion is closed under substitution, i.e. if $t \twoheadrightarrow t'$ and $u \twoheadrightarrow u'$ then $t[u/x] \twoheadrightarrow t'[u'/x]$ and there is a term t'' such that $t \twoheadrightarrow_f t''$ and $t' \Rightarrow_\beta^* t''$. In addition if t is a β-normal form and $t \twoheadrightarrow_f t'$ then t' is also a β-normal form.*

Proof The proofs are as in the previous chapter for the parallelised linear conversion relation. □

Note however that unlike the parallelised conversion relation of the linear λ-calculus, this relation does not satisfy the diamond property because one half of a span may identify two conversions while the other rewrites the terms indexed by the conversions to different subterms and hence loses consistency. This is of course another consequence of the non-linearity of the calculus.

Lemma 5.5.4 *Let $t \twoheadrightarrow t'$. Then $\beta(t) \twoheadrightarrow \beta(t')$.*

Proof The lemma is proved by generalising the proof of lemma 4.5.4. We prove that if $t \twoheadrightarrow_f t'$ then $\beta(t) \twoheadrightarrow_f \beta(t')$ by induction on firstly the β-normalisation rank of t and secondly the depth of the rewrite. The lemma then follows from lemma 5.5.3 since $\twoheadrightarrow$ can be embedded in $\twoheadrightarrow_f$.

If t is a β-normal form then again by lemma 5.5.3 t' is also a β-normal form. For the inductive step consider a parallel conversion of the form:

$$\frac{X \subseteq \mathtt{FC}(t) \text{ is consistent} \quad t/X \twoheadrightarrow_f u \quad t \setminus_i X \twoheadrightarrow_f v_i}{t \twoheadrightarrow_f \mathbf{case}(u, x.v_1, y.v_2)}$$

If X is empty, then by the induction hypothesis there are rewrites $\beta(t) \twoheadrightarrow_f \beta(v_i)$ and hence a parallel conversion

$$\frac{\beta(t) \twoheadrightarrow_f \beta(v_i)}{\beta(t) \twoheadrightarrow \mathbf{case}(u, x.\beta(v_1), y.\beta(v_2))}$$

where we rely on β-reduction not to introduce new free variables. If however X is non-empty there are four subcases. Firstly if t/X is not a β-normal form, then by the induction hypothesis there is a parallel rewrite $\beta(t/X) \twoheadrightarrow_f \beta(u)$ and hence a rewrite

$$\frac{\beta(t/X) \twoheadrightarrow_f \beta(u) \quad t \setminus_i X \twoheadrightarrow_f v_i}{t[X \leftarrow \beta(t/X)] \twoheadrightarrow \mathbf{case}(\beta(u), x.v_1, y.v_2)}$$

which embeds to a full parallel conversion and the lemma then follows by the induction hypothesis. If however t/X is a β-normal form then consider the case where t/X is an injection, say $\mathbf{in}_1(r)$. Then by fullness u must also be an injection, say $\mathbf{in}_1(r')$ where $r \twoheadrightarrow_f r'$. Now by lemma 5.3.1, the result of contracting the β_+-redexes in t associated to X is $(t \setminus_1 X)[r/x]$ and thus there are reductions

$$\begin{tikzcd} t \arrow[r, twoheadrightarrow, "f"'] \arrow[d, "\beta^*"'] & \mathbf{case}(\mathbf{in}_1(r'), x.v_1, y.v_2) \arrow[d, "\beta^*"'] \\ (t \setminus_1 X)[r/x] \arrow[r, twoheadrightarrow] & v_1[r'/x] \end{tikzcd}$$

where the bottom rewrite follows as parallel rewriting is closed under substitution. This rewrite can then be extended to a full rewrite to which the induction hypothesis may be applied. A third possibility is that t/X is a case expression, in which case a similar argument works — namely carry out the commuting conversions in t and at the top level of t' and then apply the induction hypothesis. Finally, if none of these cases are applicable, then an inductive argument proves there is free consistent set of conversions $\beta(X) \subseteq \mathtt{FC}(\beta(t))$ such that

$$\beta(t)/\beta(X) = \beta(t/X) = t/X \text{ and } \beta(t) \setminus_i \beta(X) = \beta(t \setminus_i X)$$

and thus there is a rewrite

$$\frac{\beta(t)/\beta(X) = \beta(t/X) \twoheadrightarrow \beta(u) \quad \beta(t) \setminus_i \beta(X) = \beta(t \setminus_i X) \twoheadrightarrow \beta(v_i)}{\beta(t) \twoheadrightarrow \mathbf{case}(\beta(u), x.\beta(v_1), y.\beta(v_2))}$$

may be converted to a full rewrite which proves the lemma. Note that if $\beta(X)$ is empty the proof is still valid as one of the conditions on the variables x, y bound by conversions $X \subseteq \mathtt{FC}(t)$ is that $x, y \notin \mathtt{FV}(t)$ and so $x, y \notin \mathtt{FV}(\beta(t))$. If however the parallel rewrite has as last rule a pre-congruence,

$$\frac{t_0 \twoheadrightarrow_f t'_0, \ldots, t_n \twoheadrightarrow_f t'_n}{\mathcal{T}(t_0, \ldots, t_n) \twoheadrightarrow_f \mathcal{T}(t'_0, \ldots, t'_n)}$$

then there are three possibilities. If there is an immediate subterm which is not a β-normal form, then the induction hypothesis may be used on each of the subterms so that $\mathcal{T}(\beta(t_0), \ldots, \beta(t_n)) \twoheadrightarrow \mathcal{T}(\beta(t'_0), \ldots, \beta(t'_n))$ and then the induction hypothesis invoked again. On the other hand if the only redex is a top level redex, then by fullness there is a also a redex at the top level of t'. There is a parallel rewrite between the terms obtained by performing these reductions and the lemma then follows by the induction hypothesis. □

5.5.2 Embedding $\Rightarrow_c$ into $\Rightarrow_{\mathcal{F}}$-Normal Forms

The second part of the embedding theorem concerns the interaction between the conversion relation and the η-expansions implicit in the function $\eta\mathcal{F}$. The key lemma is the following generalisation of lemma 4.5.5.

Lemma 5.5.5 *Given a conversion $\sigma \in \mathtt{FC}(t)$ there is a consistent set of free conversions $\sigma^{\mathcal{R}} \subseteq \mathtt{FC}(\eta\mathcal{R}(t))$ such that*

$$\eta\mathcal{R}(t)/\sigma^{\mathcal{R}} = \eta\mathcal{I}(t/\sigma) \text{ and } \eta\mathcal{R}(t)\setminus_i\sigma^{\mathcal{R}} = \eta\mathcal{R}(t\setminus_i\sigma)$$

where $\mathcal{R} \in \{\mathcal{I}, \mathcal{F}\}$.

Proof The lemma is proved by induction on the definition of the functions $\eta\mathcal{F}$ and $\eta\mathcal{I}$ and follows the proof of lemma 4.5.5. □

Note that the η-rule for products duplicates its argument and hence the requirement that $\sigma^{\mathcal{R}}$ be a set of conversions.

Lemma 5.5.6 *Assume $t \twoheadrightarrow t'$. Then $\eta\mathcal{R}(t) \twoheadrightarrow \eta\mathcal{R}(t')$ where $\mathcal{R} \in \{\mathcal{I}, \mathcal{F}\}$.*

Proof The lemma is proved simultaneously by induction on the rewrite $t \twoheadrightarrow t'$ and is again a trivial generalisation of lemma 4.5.6. □

Theorem 5.5.7 *If two terms are equivalent in the conversion relation, then so are their $\Rightarrow_\delta$ normal forms. Thus the expansionary rewrite relation is decidable and confluent.*

Proof The first half of the lemma is trivial as $\Rightarrow_\delta$ is contained in the inverse of the conversion relation. Thus terms equivalent in the equational theory have $\Rightarrow_{\mathcal{F}}$-normal forms which are equivalent in the conversion relation which has already been shown to be decidable and confluent. Thus the expansionary rewrite relation is confluent and $\beta\eta$-equality is decidable. □

Table 5–9: Definition of Validity

- If t is a variable, constant, application or projection

$$\frac{t/\Rightarrow_{\mathcal{I}} \subseteq V(X)}{t \in V(X)}$$

- If t is a pair

$$\frac{u_i \in V(X_i) \quad \langle u_0, u_1\rangle/\Rightarrow_{\mathcal{I}} \subseteq V(X_0 \times X_1)}{\langle u_0, u_1\rangle \in V(X_0 \times X_1)}$$

- If t is a λ-abstraction

$$\frac{\forall u \in V(X).t[u/x] \in V(Y) \quad \lambda x.t/\Rightarrow_{\mathcal{I}} \subseteq V(X \to Y)}{\lambda x.t \in V(X \to Y)}$$

- If t is an injection

$$\frac{t \in V(X_i) \quad \mathbf{in}_i(t)/\Rightarrow_{\mathcal{I}} \subseteq V(X_1 + X_2)}{\mathbf{in}_i(t) \in V(X_1 + X_2)}$$

- If $t = \mathbf{case}(t_0, x_1.u_1, x_2.u_2)$ is a *case*-expression of function type

$$\frac{\begin{array}{l} u_i \in V(X \to Y) \\ t/\Rightarrow_{\mathcal{I}} \subseteq V(X \to Y) \end{array} \quad \forall s \in V(X).\mathbf{case}(t_0, x_1.u_1 s, x_2.u_2 s) \in V(Y)}{\mathbf{case}(t_0, x_1.u_1, x_2.u_2) \in V(X \to Y)}$$

- If $t = \mathbf{case}(t_0, x_1.u_1, x_2.u_2)$ is a *case*-expression of product type

$$\frac{\begin{array}{l} u_i \in V(X \times Y) \\ t/\Rightarrow_{\mathcal{I}} \subseteq V(X \times Y) \end{array} \quad \mathbf{case}(t_0, x_1.\pi_i u_1, x_2.\pi_i u_2) \in V(X_i)}{\mathbf{case}(t_0, x.u, y.v) \in V(X_0 \times X_1)}$$

- If $t = \mathbf{case}(t_0, x_1.u_1, x_2.u_2)$ is a *case*-expression not of function or product type

$$\frac{u_i \in V(X) \quad t/\Rightarrow_{\mathcal{I}} \subseteq V(X)}{\mathbf{case}(t_0, x_1.u_1, x_2.u_2) \in V(X)}$$

Chapter 6

Conclusions and Further Work

Traditionally, rewrite relations have been used in the λ-calculus to transform terms into canonical forms which can then be used to decide the associated equational theory. Unfortunately, for all but the simplest calculi, the rewrite relations which have been proposed so far either generate the full equational theory but contain no decision procedure, or contain a decision procedure but only for a subtheory of that required. The aim of this thesis has been to use the algebraic methods of category theory to provide a new approach to rewriting in the typed λ-calculus. We have shown that:

- Rewriting in typed λ-calculus can be modelled in ordered categories and that the natural categorical constructions on these ordered categories can be used to formally derive rewrite rules for a wide variety of type constructors.

- In this framework, the introduction and elimination rules for each type constructor are modelled as forming an adjoint pair of functors, with the associated unit and co-unit corresponding to an expansionary η-rewrite rule and a contractive β-rewrite rule.

- Although confluent, the presence of expansionary rewrite rules means that the associated rewrite relation is not strongly normalising. Instead quasi-normal forms are constructed, and decidability of the associated equational theory proved, by a variety of term rewriting techniques developed in the course of three case studies.

- The η-rewrite rules for initial type constructors are considerably more complex than those for final type constructors. The work presented in Chapter 5 is currently the only decidable axiomatisation of the theory of coproducts.

The use of categorical models of reduction to formally derive rewrite relations for typed λ-calculus has proved largely successful for the three case studies contained in this thesis. Although some of the proofs have been rather lengthy and involved, this seems to be an unavoidable consequence of the complexity of the problem.

There are several directions in which this research may be taken forward, ranging from minor extensions for which the results of this thesis provides the basis of a solution, to wider questions which will need much further research and possibly the development of new proof techniques.

Firstly there remain a couple of questions relating to coproducts. Unlike the simply typed λ-calculus, we could not associate a unique canonical quasi-normal form to each term of $\Lambda^{1,\times,\rightarrow,+}$ because of the inherent parallel nature of the conversion relation — see the discussion around equations 4.2 and 4.3. Uniqueness may be obtained by generalising the elimination rules of initial type constructors to allow the elimination of several terms to occur in parallel. For the coproduct, the generalised parallel elimination rule would be of the form

$$\frac{\Gamma \vdash t_1 : A_1 + B_1 \quad \ldots \quad \Gamma \vdash t_n : A_n + B_n \quad \Gamma, \vec{x}_p \vdash u_p : C}{\Gamma \vdash \mathbf{case}(\vec{t}, \vec{p}.\vec{u}) : C}$$

where $\vec{t}$ is the tuple of length n formed by the terms $t_1, \ldots t_n$, and for every one of the 2^n elements $p \in \{A_1, B_1\} \times \ldots \times \{A_n, B_n\}$, $\vec{x}_p$ is an appropriate list of variable declarations and u_p is a term. I believe that, by defining translations between this calculus and the calculus containing the traditional elimination rule, one would be able to use the results already proved to show that the new calculus has a sound, complete and decidable equational theory with each equivalence class of terms having a unique canonical representative. The other main question is whether our axiomatisation of the theory of coproducts matches that in [23]. I would conjecture that this is so, although the main stumbling block seems to be to understand exactly which equations are provable in the equational theory presented there.

Another question for which I feel the basic research has already been completed is the effect of adding an empty type. As indicated in subsection 5.2, an expansionary η-rewrite rule may derived for such a type by using the same categorical techniques as for the other type constructors of this thesis. From a term rewriting perspective, the interesting point about this rewrite rule is that the reducts of a term are more dependent on the consistency of the context in which the term is typed, than on the actual term itself. This pushes us towards considering rewrite relations which act, not on terms, but on term judgements.

Thirdly, the term rewriting techniques developed in the study of the linear and bicartesian conversion relations seem applicable to certain problems involved in pattern matching [9,52]. For example, in [10] a sound and complete equational theory for decision trees was proved decidable, and we have been able to re-formulate these results in terms of a conversion relation.

Further afield there are of course other extensions for which entirely new techniques may be needed. Firstly, one would like to extend these results to polymorphic theories such as System F [32,30] and the Calculus of Constructions [13]. The equational theory on System F terms generated by the following basic reductions

$$\begin{array}{lrcl}
\beta_{\rightarrow} & (\lambda x.t)u & \Rightarrow & t[u/x] \\
\eta_{\rightarrow} & t & \Rightarrow & \lambda x.tx \\
\beta_{\Pi} & (\Lambda X.t)V & \Rightarrow & t[V/X] \\
\eta_{\Pi} & t & \Rightarrow & (\Lambda X.tX)
\end{array}$$

has already been shown to be decidable [14,27]. However this result is not totally satisfactory as the equational theory is not as strong as one would like: For example, if types such as products and coproducts are encoded in System F, then the encoding of a term and its β-reducts are related in the equational theory, but the same is not true of a term and its η-expansion. An example of this is the unit type which is encoded as the type $\Pi X.X \rightarrow X$. The normal form of a variable $z : \Pi X.X \rightarrow X$ is

$$\Lambda X.\lambda y.zXy$$

and this term is not equivalent in the equational theory to the canonical element of $\Pi X.X{\rightarrow}X$:

$$\Lambda X.(\lambda y.y)$$

It seems that new ideas will be required here.

When we consider dependent type theories, a new complication is added by *type conversion*. For instance if E is some equational theory on types, and type conversion is permitted by inference rules of the form:

$$\frac{\Gamma \vdash t\!:\!A \quad A =_E B}{\Gamma \vdash t\!:\!B}$$

then the type of a term, which determines which η-expansions may be applied to it, will not be unique and may vary during reduction — see [18] for more details. The solution may be to follow the same approach as indicated for the empty type and consider rewrite relations acting, not on terms, but on term judgements. However some partial results have been obtained by other researchers [12,26,35] (some using η-contractions) and we hope that we shall be able to put a case forward for η-expansions.

Another way to increase the expressivity of our example calculi is to add recursive types such as natural numbers and lists. The following expansionary η-rewrite rule was derived in Chapter 2 by characterising the natural numbers type as a locally initial lax dialgebra:

$$\frac{h[0/n] \Rightarrow u \quad h[sn/n] \Rightarrow v[h/x]}{h \Rightarrow \mathtt{It}(u, x.v, n)}\ \eta_N$$

As mentioned at the end of Chapter 2, I am as yet unsure of the strength of the equational theory associated to this rewrite rule, and in particular the equational theory may well be undecidable for recursion-theoretic reasons. Another type constructor worthy of consideration is the ! (bang) [7,31,76] operator of linear logic. As the semantics [7], and hence an equational theory, for this constructor become clearer, we should see if our methods are applicable.

Finally, this project is just one of a number of case studies investigating the use of category theory in term rewriting [58,79]. One important issue to be resolved is whether our results fit into the 2-categorical framework, and the work of Hilken [37] provides a starting point for future research.

On a more general note it is hoped that the research proposed above will benefit these other projects in categorical rewriting, and vice versa, that results obtained elsewhere may be of use in my work.

All in all there seems much to do!

Index

Bibliography

[1] S. Abramsky. Computational interpretations of linear logic. *Theoretical Computer Science*, 111:3–57, 1993.

[2] Y. Akama. On Mints' reduction for ccc-calculus. In *Typed λ-Calculus and Applications*, volume 664 of *Lecture Notes in Computer Science*, pages 1–12. Springer Verlag, 1993.

[3] E. S. Bainbridge, A. Scedrov, P. Freyd, and P. Scott. Functorial polymorphism. In G. Huet, editor, *Logical Foundations of Functional Programming*, chapter 14, pages 315–327. Addison Wesley, 1990.

[4] H. Barendregt. *The Lambda Calculus: Its Syntax and Semantics (revised edition)*. Number 103 in Studies in Logic and the Foundations of Mathematics. North Holland, 1984.

[5] M. Barr. *-autonomous categories. *Lecture Notes in Mathematics*, 752, 1979.

[6] M. Barr and C. Wells. *Toposes, Triples and Theories.* Number 278 in Grundlehren der Mathematischen Wissenschaften. Springer Verlag, 1985.

[7] G. Bierman. On intuitionistic linear logic. Technical Report 346, Univ. of Cambridge, 1994.

[8] R. Blackwell, G. M. Kelly, and A. J. Power. Two-dimensional monad theory. *Journal of Pure and Applied Algebra*, 59:1–41, 1989.

[9] V. Breazu-Tannen, D. Kesner, and L. Puel. A typed pattern calculus. In *LICS*, 1993.

[10] J. R. Cockett and J. A. Herrera. Decision tree reduction. *Journal of the ACM*, 37(4):815–842, 1990.

[11] T. Coquand. On the analogy between propositions and types. In G. P. Huet, editor, *Logical Foundations of Functional Programming*. Addison-Wesley, Reading, MA, 1989.

[12] T. Coquand. An algorithm for testing conversion in type theory. In G. P. Huet and G.D. Plotkin, editors, *Logical Frameworks*. Cambridge University Press, 1991.

[13] T. Coquand and G. Huet. The calculus of constructions. *Information and Computation*, 76(2/3):95–120, 1988.

[14] R. Di Cosmo and D. Kesner. Rewriting with polymorphic extensional λ-calculus. Submitted.

[15] R. Di Cosmo and D. Kesner. A confluent reduction for the extensional typed λ-calculus with pairs, sums, recursion and terminal object. In *ICALP*, volume 700 of *Lecture Notes in Computer Science*. Springer Verlag, 1993.

[16] R. Di Cosmo and D. Kesner. Combining first order algebraic rewrite systems, recursion and extensional λ-calculi. In *ICALP*, volume 820 of *Lecture Notes in Computer Science*, pages 462–472. Springer Verlag, 1994.

[17] R. Di Cosmo and D. Kesner. Simulating expansions without expansions. *Mathematical Structures in Computer Science*, 4:1–48, 1994.

[18] R. Di Cosmo and A. Piperno. Expanding extensional polymorphism. In *Typed λ-calculus and Applications*, volume 902 of *Lecture Notes in Computer Science*, pages 139–153. Springer Verlag, 1995.

[19] D. Cubric. Embedding of a free CCC into the category of sets. Journal of Pure and Applied algebra, to appear.

[20] P.L. Curien and R. Di Cosmo. A confluent reduction system for the λ-calculus with surjective pairing. In *ICALP*, volume 510 of *Lecture Notes in Computer Science*, pages 291–302. Springer Verlag, 1991.

[21] N. Dershowitz and J.P. Jouannaud. Rewrite systems. In *The Handbook of Theoretical Computer Science*, volume Vol. B: Formal Models and Semantics, chapter 6, pages 243–320. The MIT Press, 1990.

[22] D. Dougherty. Some λ-calculi with categorical sums and products. In *Rewriting Techniques and Applications*, volume 690 of *Lecture Notes in Computer Science*, pages 137–151. Springer Verlag, 1993.

[23] D. Dougherty and R. Subrahmanyam. Equality between functionals in the presence of coproducts. *To appear, LICS*, 1995.

[24] E. J. Dubuc. *Kan Extensions in Enriched Category Theory*, volume 145 of *Lecture Notes in Mathematics*. Springer Verlag, 1970.

[25] H. Friedman. Equality between functionals. *Logic Colloquium*, 1975.

[26] P. Gardner. On the construction of long $\beta\eta$-normal forms in dependent type theories. Personal communication.

[27] N. Ghani. Extensional polymorphism. unsubmitted.

[28] N. Ghani. Inconsistency and extensionality. In preparation.

[29] N. Ghani. $\beta\eta$-equality for coproducts. In *Typed λ-calculus and Applications*, number 902 in Lecture Notes in Computer Science, pages 171–185. Springer Verlag, 1995.

[30] J. Y. Girard. The system F of variable types - 15 years later. *Theoretical Computer Science*, 45:159–192, 1986.

[31] J. Y. Girard. Linear logic. *Theoretical Computer Science*, 50, 1987.

[32] J. Y. Girard, P. Taylor, and Y. Lafont. *Proofs and Types*, volume 7 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 1989.

[33] J. W. Gray. *Formal Category Theory: Adjointness for 2-Categories*. Number 391 in Lecture Notes in Mathematics. Springer Verlag, 1974.

[34] J. W. Gray. Order-enriched sketches for typed lambda calculi. In Carboni, Pedicchio, and Rosolini, editors, *Category Theory*, number 1488 in Lecture Notes in Mathematics, pages 105–130, Como, 1990. Springer Verlag.

[35] H. Guevers. The Church-Rosser property for $\beta\eta$-reduction in typed λ-calculi. In *LICS*, pages 453–460. IEEE, 1992.

[36] T. Hagino. *A Categorical Programming Language*. PhD thesis, University of Edinburgh, Department of Computer Science, 1987.

[37] B. P. Hilken. Towards a proof theory of rewriting: The simply typed 2-λ-calculus. Technical Report 336, Computer Lab., University of Cambridge, 1994.

[38] J. R. Hindley and J. P. Seldin. *Introduction to Combinators and λ-Calculus*. Number 1 in LMS Student Texts. Cambridge University Press, 1986.

[39] C. A. R. Hoare, H. Jifeng, and C. E. Martin. Pre-adjunctions in order enriched categories. Technical report, Oxford University Computing Laboratory, 1989.

[40] W. Howard. The formulae-as-types notion of construction. In *To H.B. Curry: Essays in Combinatory Logic, λ-Calculus and Formalism*, pages 479–490. Academic Press, 1980.

[41] G. Huet. *Résolution d'équations dans des langages d'ordre* $1, 2, \ldots, \omega$. Thèse d'Etat, Université de Paris VII, 1976.

[42] G. Huet. Confluent reductions: Abstract properties and applications to term rewriting systems. *Journal of the ACM*, 27:797–821, 1980.

[43] G. Huet and D. C. Oppen. Equations and rewrite rules: A survey. In *Formal Language Theory: Perspectives and Open Problems*. Academic Press, 1980.

[44] B. Jacobs. *Categorical Type Theory.* PhD thesis, Katholieke Universiteit Nijmegen, September 1991.

[45] C. B. Jay. Local adjunctions. *Journal of Pure and Applied Algebra*, 53:227–238, 1988.

[46] C. B. Jay. Languages for monoidal categories. *Journal of Pure and Applied Algebra*, 59:61–85, 1989.

[47] C. B. Jay. Extending properties of categories to partial maps. LFCS Report Series, University of Edinburgh, Department of Computer Science, 1990.

[48] C. B. Jay. Modelling reduction in confluent categories. In *Applications of Categories in Computer Science*, volume 177. London Mathematical Society Lecture Note Series, 1992.

[49] C. B. Jay and N. Ghani. The virtues of eta-expansion. *Journal of Functional Programming*, to appear.

[50] G. M. Kelly. *Basic Concepts of Enriched Category Theory.* Number 64 in London Mathematical Society Lecture Note Series. Cambridge University Press, 1982.

[51] G. M. Kelly and Ross Street. Review of the elements of 2-categories. In *Category Seminar Sydney 1972/73*, number 420 in Lecture Notes in Mathematics, pages 75–103. Springer Verlag, 1974.

[52] D. Kesner. Reasoning about layered, wildcard and product patterns. In *Proc. 4th Int. Conf. on Algebraic and Logic Programming*, number 850 in Lecture Notes in Computer Science. Springer Verlag, 1994.

[53] J. W. Klop. Combinatory reduction systems. *Mathematical Center Tracts*, 27, 1980.

[54] J. W. Klop, A. Middeldorp, Y. Toyama, and R. de Vrijer. A simplified proof of Toyama's theorem. Technical Report CS–R9156, Centrum voor Wiskunde en Informatica, Amsterdam, 1991.

[55] J. Lambek and P. J. Scott. *Introduction to Higher Order Categorical Logic.* Number 7 in Cambridge Studies in Advanced Mathematics. Cambridge University Press, 1986.

[56] F. W. Lawvere. Functorial semantics of algebraic theories. *In Proceedings of the National Academy of Sciences*, 50:869–872, 1963.

[57] F. W. Lawvere. Metric spaces, generalised logic and closed categories. *Rend. del. Sem. Mat. e Fis. di Milano*, 43:135–166, 1973.

[58] C. Lüth. Categorical compositional term rewriting. Forthcoming PhD thesis, Univ. of Edinburgh.

[59] S. MacLane. *Categories for the Working Mathematician.* Graduate Texts in Mathematics. Springer Verlag, 1971.

[60] E. G. Manes. *Algebraic Theories*, volume 26 of *Graduate Texts in Mathematics.* Springer Verlag, 1976.

[61] A. Meyer, J. Mitchell, E. Moggi, and R. Statman. Empty types in polymorphic λ-calculus. In *Logical Foundations of Functional Programming*, chapter 11, pages 273–284. Addision Wesley, 1990.

[62] G. E. Mints. Teorija categorii i teoria dokazatelstv.i. *Aktualnye problemy logiki i metodologii nauky*, pages 252–278, 1979.

[63] B. Mitchell. Rings with several objects. *Advances in Mathematics*, 8:1–161, 1972.

[64] W. Phoa. An introduction to fibrations, topos theory, the effective topos and modest sets. LFCS Report Series ECS-LFCS-92-208, University of Edinburgh, Department of Computer Science, April 1992.

[65] A. Poigné and J. Voss. On the implementation of abstract data types by programming language constructs. *Journal of Computer and System Science*, 34(2-3):340–376, April/June 1987.

[66] A. J. Power. An abstract formulation for rewrite systems. In D. H. Pitt, D. E. Rydeheard, P. Dybjer, A. M. Pitts, and A. Poigné, editors, *Category Theory and Computer Science*, number 389 in Lecture Notes in Computer Science, pages 300–312, Manchester, September 1989. Springer Verlag.

[67] A. J. Power. Why tricategories. Technical Report ECS–LFCS-94-289, University of Edinburgh, Department of Computer Science, 1994.

[68] D. Prawitz. Ideas and results in proof theory. In J.E. Fenstad, editor, *Proc. 2nd Scandinavian Logic Symposium*, pages 235–307. North Holland, 1971.

[69] J. Reynolds. Polymorphism is not set-theoretic. In *Semantics of Data Types*, number 173 in Lecture Notes in Computer Science. Springer Verlag, 1984.

[70] D. E. Rydeheard and J. G. Stell. Foundations of equational deduction: A categorical treatment of equational proofs and unification algorithms. In *Category Theory and Computer Science*, number 283 in Lecture Notes in Computer Science, pages 114– 139. Springer Verlag, 1987.

[71] P. Scott and M. Okada. Rewriting theory for uniqueness conditions. Manuscript.

[72] R. A. G. Seely. *Hypderdoctrines and Natural Deduction.* PhD thesis, Univ. of Cambridge, 1977.

[73] R. A. G. Seely. Locally cartesian closed categories and type theory. *Math. Proc. Cam. Phil. Soc.*, 95:33–48, 1984.

[74] R. A. G. Seely. Categorical semantics for higher order polymorphic λ-calculus. *Journal of Symbolic Logic*, 52(4):969–989, 1987.

[75] R. A. G. Seely. Modelling computations: A 2-categorical framework. In *Proc. 2nd Annual Symposium on Logic in Computer Science*. IEEE publications, 1987.

[76] R. A. G. Seely. Linear logic, *-autonomous categories and cofree algebras. In *Categories in Computer Science and Logic*, volume 92 of *Contemporary Mathematics*. American Mathematical Society, 1989.

[77] A. Simpson. Categorical completeness for the simply typed λ-calculus. In *Typed λ-Calculus and Applications*, number 902 in Lecture Notes in Computer Science, pages 428–443. Springer Verlag, 1995.

[78] M. B. Smyth and G. D. Plotkin. The category-theoretic solution of recursive domain equations. *SIAM*, 11(4):763–783, 1982.

[79] J. Stell. Modelling term rewriting systems by sesqui-categories. Technical Report TR94-02, Dept. Comp. Sci., University of Keele, 1994.

[80] M. E. Szabo. *Algebra of Proofs*, volume 88 of *Studies in Logic and the Foundations of Mathematics*. North Holland, 1978.