Turing Machines and Differential Linear Logic

James Clift

Supervisor: Daniel Murfet

Masters Thesis
The University of Melbourne
Department of Mathematics and Statistics

October 2017

Abstract

We describe the vector space semantics for linear logic in which the exponential modality is modelled by cofree coalgebras. There is a natural differentiable structure in this semantics arising from the primitive elements. We provide an investigation of this structure, through many examples of proofs and their derivatives. In particular, an encoding of Turing machines in linear logic is given in detail, based on work by Girard, but modified in order to be compatible with derivatives. It is shown that for proofs of the appropriate form, the derivatives obtained via the coalgebraic structure agree with those from elementary calculus, allowing one to write Taylor expansions of proofs. We provide an interpretation of the derivative of the Turing encoding via a simple example.

Contents

Acknowledgements: Many thanks to my wonderful supervisor Daniel Murfet for many hours of engaging conversation and invaluable advice. Thanks to my friends for helping me endure the stress which comes along with higher education. Lastly, thank you to my family, particularly my parents, for fostering my love of mathematics from a very young age.

1 Introduction

In the 1930s the notion of computability was given a rigorous definition in two different but equivalent forms. These are:

1. The λ-calculus, developed by Church, wherein computation is carried out by reducing terms in the language to a normal form.

2. Turing machines, developed by Turing, wherein computation is imagined as a finite state machine with unbounded memory following a simple set of rules.

In the terminology of computer science, the former is the prototypical example of *functional programming*, and the latter of *imperative programming*.

Given that the same class of functions $\mathbb{N} \to \mathbb{N}$ may be encoded by both λ-calculus and Turing machines, one could reasonably describe the question of *computability* as settled. In this thesis, we will consider the broader question of the *structure* of the set of all computer programs. Our model of computation is that of *linear logic* – introduced by Girard in the 1980s – into which simply typed λ-calculus embeds [8]. The logic restricts the use of the weakening and contraction rules of traditional intuitionistic logic to certain classes of formulas, so that the only formulas which may be duplicated or discarded are those preceded by a new operator '!', called the exponential. This gives a natural interpretation of the logic as being resource conscious: if a premise without the exponential appears in a logical argument, one can be sure that this premise is used precisely once; that is, in a *linear* way. The logic has the advantage of being able to compute almost any total function of integers one could possibly hope for, while still being *strongly normalising*; no computation can run forever.

A convenient lens through which one can study linear logic is its *semantics*: a way of mapping the syntax of proofs into familiar categorical structures [12, §2]. This is similar to the way in which representation theory provides an understanding of groups via linear maps of vector spaces. Originally, a semantics of linear logic in coherence spaces was provided by Girard [8], and since then a wide variety of other semantics have been proposed [3, 4]. Our particular focus will be a vector space semantics (the *Sweedler semantics*), which assigns to each logical formula A a vector space $[\![A]\!]$, and to each proof π of $\Gamma \vdash A$ a linear map $[\![\pi]\!] : [\![\Gamma]\!] \to [\![A]\!]$. In the Sweedler semantics, the exponential $!A$ is assigned the cofree coalgebra generated by $[\![A]\!]$, and the means to encode nonlinear maps is provided by the *grouplike* elements of the coalgebra [13].

Somewhat more interestingly, the cofree coalgebra comes with an intrinsic differentiable structure via the *primitive* elements. Precisely, a proof π of $!A \vdash B$ has, for each $\gamma \in [\![A]\!]$, an associated linear map

$$\partial_\gamma [\![\pi]\!] : [\![A]\!] \to [\![B]\!],$$

which is analogous to the derivative of a smooth map of manifolds at the point γ. It is important to emphasise that the differentiable structure is not artificially added to

the semantics; it naturally appears as soon as one considers cofree coalgebras. Given that proofs in linear logic correspond to computer programs which one would typically think of as being discrete objects, the appearance of differentiation – which seems to require continuity – comes as a surprise. We therefore consider it prudent to investigate this structure fully. The appearance of derivatives in linear logic is not new; the role of differentiable structure on the λ-calculus and linear logic has been motivated by its connection to head normal forms [20, 17]. However, in our view there is still a lack of elementary examples of proofs whose derivatives have clear computational meaning.

In this thesis we will attempt to fill this gap by providing such examples. Many familiar data types, including booleans and binary integers, can easily be encoded in linear logic. By this, we mean that there exists a formula $\mathbf{bint}_A$ in linear logic such that any binary sequence can be encoded as a proof of $\vdash \mathbf{bint}_A$. For a more involved example, we will show that there is an encoding of Turing machines into linear logic. One can encode the instantaneous configuration of Turing machine as a proof of

$$\mathbf{Tur}_A = !\mathbf{bint}_A \otimes !\mathbf{bint}_A \otimes !_n\mathbf{bool}_A,$$

and the state evolution of a Turing machine is described by a proof ${}_\delta\underline{\mathrm{step}}_A$ of

$$\mathbf{Tur}_{A^3} \vdash \mathbf{Tur}_A.$$

The idea of encoding Turing machines in linear logic is by no means a new development; originally this arose in connection with polynomial time complexity [25], and a variety of encodings exist in the literature [24, 19]. Our work is specifically based on an encoding due to Girard [9], but is novel in the sense that it does not require the use of second order quantifiers. Our justification for this change is twofold. Firstly, the Sweedler semantics has no obvious extension to second order linear logic, and so one cannot understand the logic algebraically; in particular, any link to differentiation is lost. Secondly, the use of second order quantification undermines the resource sensitivity that linear logic attempts to capture; in second order linear logic, there are ways to copy binary integers arbitrarily many times via a proof of $\mathbf{bint} \vdash !\mathbf{bint}$.

In Section 2 we define the syntax and terminology of linear logic, review the basic theory of coalgebras, and define the Sweedler semantics of linear logic. A wealth of examples of data types and functions which may be encoded in linear logic are given in Section 3. We provide a proof that the Sweedler semantics is injective on integers and binary integers. In Section 4, we explain how one can interpret the differentiable structure in the syntax by adjoining new deduction rules. In Section 5, we present our encoding of Turing machines in linear logic. An encoding of nondeterministic Turing machines is also provided. Connections to derivatives in the sense of elementary calculus are discussed in Section 6. We show that proofs of an appropriate form have denotations which can be expanded as a Taylor series. In particular this applies to ${}_\delta\underline{\mathrm{step}}_A$, and we give an interpretation of the derivative in terms of the operation of the machine.

$$\frac{}{A \vdash A}\ \text{axiom} \qquad \frac{\Gamma, A, B, \Delta \vdash C}{\Gamma, B, A, \Delta \vdash C}\ \text{exch} \qquad \frac{\Gamma \vdash A \qquad \Delta, A \vdash B}{\Gamma, \Delta \vdash B}\ \text{cut}$$

$$\frac{\Gamma, A \vdash C}{\Gamma, A \,\&\, B \vdash C}\ \&L_0 \qquad \frac{\Gamma, B \vdash C}{\Gamma, A \,\&\, B \vdash C}\ \&L_1 \qquad \frac{\Gamma \vdash A \qquad \Gamma \vdash B}{\Gamma \vdash A \,\&\, B}\ \&R$$

$$\frac{\Gamma, A, B \vdash C}{\Gamma, A \otimes B \vdash C}\ \otimes L \qquad \frac{\Gamma \vdash A \qquad \Delta \vdash B}{\Gamma, \Delta \vdash A \otimes B}\ \otimes R$$

$$\frac{\Gamma \vdash A \qquad \Delta, B \vdash C}{\Gamma, \Delta, A \multimap B \vdash C}\ \multimap L \qquad \frac{\Gamma, A \vdash B}{\Gamma \vdash A \multimap B}\ \multimap R$$

$$\frac{\Gamma, A \vdash B}{\Gamma, !A \vdash B}\ \text{der} \qquad \frac{!\Gamma \vdash A}{!\Gamma \vdash !A}\ \text{prom}$$

$$\frac{\Gamma \vdash B}{\Gamma, !A \vdash B}\ \text{weak} \qquad \frac{\Gamma, !A, !A \vdash B}{\Gamma, !A \vdash B}\ \text{ctr}$$

Figure 1.1: Sequent calculus for linear logic. The symbols A, B, C are formulas, and the symbols Γ, Δ are sequences of formulas, possibly empty. The notation $!\Gamma$ in the promotion rule 'prom' means that each formula in Γ is preceded by an exponential modality.

2 Linear logic

2.1 Preliminaries

We begin by introducing the terminology of first-order intuitionistic linear logic, hereafter referred to as linear logic. Fix a collection of atomic variables $\{x_1, x_2, ...\}$. The notion of a **formula** (or **type**) is defined recursively as follows: each atomic variable is a formula, and if A and B are formulas then so are $(A \,\&\, B)$, $(A \otimes B)$, $(A \multimap B)$ and $!A$. The operator '!' is called the exponential modality. A **sequent** is a non-empty sequence of formulas $(A_1, ..., A_n, B)$ where $n \geq 0$, written as $A_1, ..., A_n \vdash B$. The formulas $A_1, ..., A_n$ are called the **premises** and the formula B is called the **conclusion**, the idea being that the formula B can be deduced from the formulas $A_1, ..., A_n$. The symbol $\vdash$ is called the **turnstile**. A sequent may have any number of premises, even none, but must have precisely one conclusion. Sequences of formulas are usually denoted by upper case Greek letters.

The sequent calculus (Figure 1.1) for linear logic allows us to deduce the validity of some sequents from others. Proofs are inductively built by starting from axioms (tautological sequents of the form $A \vdash A$) and applying deduction rules. More precisely, a **proof** is a rooted plane tree whose vertices are labelled by sequents, such that all leaves are axioms and such that each node and its children correspond to one of the deduction rules in Figure 1.1. Typically when writing a proof we omit exchange rules to improve readability, treating the left hand side of a sequent as an unordered list.

Example 2.1. Consider the following proof:

$$\dfrac{\dfrac{}{A \vdash A} \quad \dfrac{\dfrac{}{B \vdash B} \quad \dfrac{}{C \vdash C}}{A, A \multimap B \vdash B}\multimap L}{\dfrac{A, A \multimap B, B \multimap C \vdash C}{A \multimap B, B \multimap C \vdash A \multimap C}\multimap R}\multimap L$$

If one thinks of a proof of $\vdash A \multimap B$ as being a function $A \to B$, the above proof may be interpreted as composition of the two inputs to obtain a single output of type $A \multimap C$.

Of particular significance is the 'cut' rule, which encapsulates modus ponens for the linear implication, $\multimap$. Perhaps surprisingly, this rule turns out to be unnecessary from the point of view of provability.

Theorem 2.2. (Hauptsatz.) If π is a proof of $\Gamma \vdash A$, then there exists a proof π' of $\Gamma \vdash A$ in which there is no occurrence of the cut rule.

Moreover, there is an effective procedure to transform π into π', outlined in [8, 12]. It is this cut elimination procedure which gives rise to the computational interpretation of linear logic; the cut rule should be understood as being composition of proofs, and performing cut elimination is analogous to performing β-reduction in the λ-calculus. It is the process that transforms π to π' which is important, not merely the existence of

π'. We write $\pi \to_{\text{cut}} \pi'$ to mean that π reduces to π' under cut elimination, and define $\sim_{\text{cut}}$ as the smallest equivalence relation containing $\to_{\text{cut}}$.

Let $\mathscr{C}$ be a symmetric monoidal category [16]. A **categorical semantics** [12] for linear logic assigns to each formula A an object of $\mathscr{C}$ denoted $[\![A]\!]$, and to each proof π of $A_1, ..., A_n \vdash B$ a morphism $[\![\pi]\!] : [\![A_1]\!] \otimes ... \otimes [\![A_n]\!] \to [\![B]\!]$, such that if $\pi \sim_{\text{cut}} \pi'$ then $[\![\pi]\!] = [\![\pi']\!]$. The objects and morphisms are called **denotations**.

Our aim is to define a vector space semantics for linear logic. As is typical, the bulk of the work is to obtain a suitable interpretation of the exponential $!A$. To do this, we first review the basic theory of coalgebras from [2]. Let k be a field.

Definition 2.3. A **coassociative counital coalgebra** over k is a k-vector space C equipped with linear maps $\Delta : C \to C \otimes C$ and $\epsilon : C \to k$, called the **coproduct** and **counit** respectively, such that the following diagrams commute:

$$\begin{array}{ccc} C & \xrightarrow{\Delta} & C \otimes C \\ {\scriptstyle\Delta}\downarrow & & \downarrow{\scriptstyle \mathrm{id}\otimes\Delta} \\ C \otimes C & \xrightarrow[\Delta\otimes\mathrm{id}]{} & C \otimes C \otimes C \end{array} \qquad \begin{array}{ccccc} & & C & & \\ & \overset{\cong}{\swarrow} & \downarrow{\scriptstyle\Delta} & \overset{\cong}{\searrow} & \\ C \otimes k & \xleftarrow[\mathrm{id}\otimes\epsilon]{} & C \otimes C & \xrightarrow[\epsilon\otimes\mathrm{id}]{} & k \otimes C \end{array}$$

The coalgebra C is **cocommutative** if the following diagram also commutes, where $\sigma : C \otimes C \to C \otimes C$ is the map $\sigma(a \otimes b) = b \otimes a$:

$$\begin{array}{ccc} C & \xrightarrow{\Delta} & C \otimes C \\ & {\scriptstyle\Delta}\searrow & \downarrow{\scriptstyle\sigma} \\ & & C \otimes C \end{array}$$

Given two coalgebras $(C_1, \Delta_1, \epsilon_1)$ and $(C_2, \Delta_2, \epsilon_2)$ over k, a **coalgebra morphism** from C_1 to C_2 is a k-linear map $\Phi : C_1 \to C_2$ such that the following diagrams commute:

$$\begin{array}{ccc} C_1 & \xrightarrow{\Delta_1} & C_1 \otimes C_1 \\ {\scriptstyle\Phi}\downarrow & & \downarrow{\scriptstyle\Phi\otimes\Phi} \\ C_2 & \xrightarrow[\Delta_2]{} & C_2 \otimes C_2 \end{array} \qquad \begin{array}{ccc} C_1 & \xrightarrow{\Phi} & C_2 \\ & {\scriptstyle\epsilon_1}\searrow \quad \swarrow{\scriptstyle\epsilon_2} & \\ & k & \end{array}$$

From here on, all coalgebras are coassociative, counital and cocommutative.

Example 2.4. If A is a finite-dimensional (unital) algebra with multiplication map $\nabla : A \otimes A \to A$ and unit $\eta : k \to A$, then its dual A^* can be equipped with the structure of a coalgebra. The coproduct Δ is the composite

$$A^* \xrightarrow{\nabla^*} (A \otimes A)^* \xrightarrow{\cong} A^* \otimes A^*.$$

The isomorphism $(A \otimes A)^* \cong A^* \otimes A^*$ is natural in A, but an explicit description of Δ requires a choice of basis. If $\{v_1, ..., v_n\}$ is a basis of A with dual basis $\{v^1, ..., v^n\}$, then

the value of $\Delta(\varphi)$ for $\varphi \in A^*$ is given by

$$\varphi \longmapsto \varphi \circ \nabla \longmapsto \sum_{i,j=1}^{n} \varphi(v_i v_j)(v^i \otimes v^j).$$

The counit of A^* is simply η^* followed by the canonical isomorphism $k^* \cong k$.

Definition 2.5. Let C be a coalgebra. An element $x \in C$ is called **grouplike** [22] if $\Delta(x) = x \otimes x$ and $\epsilon(x) = 1$. We denote the set of grouplike elements by $G(C)$. If $x \in G(C)$, an element $y \in C$ is called **primitive** over x if $\Delta(y) = x \otimes y + y \otimes x$. The set of primitive elements over x is denoted $P_x(C)$, and we write $P(C)$ for the union of all such $P_x(C)$.

Lemma 2.6. Coalgebra morphisms send grouplike elements to grouplike elements, and primitive elements to primitive elements.

Proof. Let $(C, \Delta_C, \epsilon_C)$ and $(D, \Delta_D, \epsilon_D)$ be coalgebras and $\Phi : C \to D$ a coalgebra morphism. If $x \in G(C)$ then $\Delta_D\Phi(x) = (\Phi \otimes \Phi)\Delta_C(x) = \Phi(x) \otimes \Phi(x)$ and $\epsilon_D\Phi(x) = \epsilon_C(x) = 1$, so $\Phi(x) \in G(D)$. Likewise, if $y \in P_x(C)$ then $\Delta_D\Phi(y) = (\Phi \otimes \Phi)\Delta_C(y) = \Phi(x) \otimes \Phi(y) + \Phi(y) \otimes \Phi(x)$, so $\Phi(y) \in P_{\Phi(x)}(D)$. □

Given a morphism of coalgebras $\Phi : C \to D$, there is therefore an induced function on grouplike elements $G(\Phi) : G(C) \to G(D)$, and so we have a functor

$$G : \mathbf{Coalg} \to \mathbf{Set}.$$

Similarly, for $x \in G(C)$ there is an induced map on primitive elements $P_x(\Phi) : P_x(C) \to P_{\Phi(x)}(D)$ which is also functorial.

Lemma 2.7. Let C be a coalgebra. There exist natural bijections:

- $\mathrm{Hom}_{\mathbf{Coalg}}(k, C) \to G(C)$ given by $\Phi \mapsto \Phi(1)$, and
- $\mathrm{Hom}_{\mathbf{Coalg}}((k[t]/t^2)^*, C) \to P(C)$ given by $\Phi \mapsto \Phi(t^*)$.

Proof. If $\Phi : k \to C$ is a morphism of coalgebras, then $\Phi(1)$ is grouplike in C by the previous lemma. Conversely, any grouplike element $x \in C$ induces a unique morphism of coalgebras $\Phi : k \to C$ given by $\Phi(a) = ax$, and hence $G(C) \cong \mathrm{Hom}_{\mathbf{Coalg}}(k, C)$.

Let $T = (k[t]/t^2)^*$, which has the dual basis $\{1^*, t^*\}$. From the discussion in Example 2.4, the coproduct Δ on T is given by

$$\Delta(1^*) = 1^* \otimes 1^*, \qquad \Delta(t^*) = 1^* \otimes t^* + t^* \otimes 1^*,$$

and the counit ϵ by

$$\epsilon(1^*) = 1, \qquad \epsilon(t^*) = 0.$$

Hence t^* is primitive over 1^*. If $\Phi : T \to C$ is a morphism of coalgebras, it follows that $\Phi(t^*)$ is primitive over $\Phi(1^*)$. Since any primitive element $y \in C$ over $x \in G(C)$ gives rise to a morphism of coalgebras $\Phi : T \to C$ via $\Phi(1^*) = x$, $\Phi(t^*) = y$, we conclude that $P(C) \cong \mathrm{Hom}_{\mathbf{Coalg}}(T, C)$. □

Let $\mathscr{V}$ be the category of vector spaces over k and $\mathscr{C}$ the category of (coassociative, counital, cocommutative) coalgebras over k. As proven in [2, Theorem 4.1], the forgetful functor $U : \mathscr{C} \to \mathscr{V}$ has a right adjoint $F : \mathscr{V} \to \mathscr{C}$.

Definition 2.8. Let V be a vector space. The coalgebra $F(V)$ is called the **cofree coalgebra** generated by V, and we write $!V$ for the underlying vector space $UF(V)$. The counit of adjunction $d : !V \to V$ is called the **dereliction map**. We will often write $!V$ to instead mean the coalgebra $F(V)$ itself; the meaning will always be clear from context.

The coalgebra $!V$ is unique up to unique isomorphism. The universal property of the dereliction map tells us that for any coalgebra C and any linear map $\varphi : C \to V$, there is a unique lifting of φ to a morphism of coalgebras $\Phi = \operatorname{prom} \varphi : C \to !V$ called the **promotion** of φ, such that $d \circ \Phi = \varphi$. Indeed, the adjunction between F and U implies that the map $\operatorname{Hom}_k(C, V) \to \operatorname{Hom}_{\mathbf{Coalg}}(C, !V)$ which sends φ to its promotion is a bijection.

In order to understand the structure of $!V$, our first step is to determine the grouplike and primitive elements. By Lemma 2.7 and the universal property of $!V$, we have a natural bijection

$$G(!V) \cong \operatorname{Hom}_{\mathbf{Coalg}}(k, !V) \cong \operatorname{Hom}_k(k, V) \cong V.$$

The grouplike element of $!V$ corresponding to $v \in V$ is denoted $|\emptyset\rangle_v$ and is called the **vacuum vector** at v. Explicitly, the linear map $\varphi : k \to V$ associated to $v \in V$ is $\varphi(a) = av$, the lifting is $\Phi(a) = a|\emptyset\rangle_v$, and hence

$$d(|\emptyset\rangle_v) = d(\Phi(1)) = \varphi(1) = v.$$

Similarly, for the primitive elements we have

$$P(!V) \cong \operatorname{Hom}_{\mathbf{Coalg}}(T, !V) \cong \operatorname{Hom}_k(T, V) \cong V \times V,$$

where $T = (k[t]/t^2)^*$. We denote the primitive element associated to $(v, w) \in V \times V$ by $|w\rangle_v \in !V$. The associated linear map $\varphi : T \to V$ is given by $1^* \mapsto v, t^* \mapsto w$, and the lifting $\Phi : T \to !V$ is $1^* \mapsto |\emptyset\rangle_v, t^* \mapsto |w\rangle_v$, and thus we have

$$d(|w\rangle_v) = d(\Phi(t^*)) = \varphi(t^*) = w.$$

Remark 2.9. The role of the dual numbers $k[t]/t^2$ gives a geometric interpretation of $|w\rangle_v$ as the *tangent vector* at v in the direction of w. This can be made precise via algebraic geometry; see [13, Appendix B]. Briefly, if X is a scheme and $p \in X$ is a closed point, the tangent space of X at p is $T_pX = (\mathfrak{m}/\mathfrak{m}^2)^*$, where $\mathfrak{m} \subseteq \mathcal{O}_{X,p}$ is the maximal ideal. It is well known [10, §VI.1.3] that specifying a closed point $p \in X$ and a tangent vector $w \in T_pX$ is equivalent to specifying a morphism of k-schemes $\operatorname{Spec}(k[t]/t^2) \to X$. In particular, if $X = \operatorname{Spec}(\operatorname{Sym}(V^*))$ then scheme morphisms $\Psi : \operatorname{Spec}(k[t]/t^2) \to X$

correspond to elements of $P(!V)$ (that is, pairs of vectors in V) via the following natural bijections [15]:

$$\begin{aligned}\mathrm{Hom}_{\mathrm{Sch}/k}(\mathrm{Spec}(k[t]/t^2), \mathrm{Spec}(\mathrm{Sym}(V^*)) &\cong \mathrm{Hom}_{\mathbf{Alg}}(\mathrm{Sym}(V^*), k[t]/t^2) \\ &\cong \mathrm{Hom}_k(V^*, k[t]/t^2) \\ &\cong \mathrm{Hom}_k\left((k[t]/t^2)^*, V\right) \\ &\cong P(!V).\end{aligned}$$

We now give an explicit description of $!V$ in the case where V is finite-dimensional, following [13, 14]. Assume that k is algebraically closed and of characteristic zero. By results of Sweedler [22, Chapter 8], any cocommutative coalgebra over an algebraically closed field can be written as a direct sum of its irreducible components, and each irreducible component contains a unique grouplike element. By the above, the irreducible components are therefore in bijection with elements of V. Given that k has characteristic zero, if $(!V)^\gamma$ denotes the irreducible component of $!V$ containing $|\emptyset\rangle_\gamma$, we have $(!V)^\gamma \cong \mathrm{Sym}(V)$ as coalgebras for each $\gamma \in V$, where $\mathrm{Sym}(V)$ is the symmetric coalgebra [14, Lemma 2.18]. We therefore have

$$!V = \bigoplus_{\gamma \in V} (!V)^\gamma \cong \bigoplus_{\gamma \in V} \mathrm{Sym}(V).$$

Generalising the notation already used for the grouplike and primitive elements:

Definition 2.10. The equivalence class of $\alpha_1 \otimes ... \otimes \alpha_n$ in the copy of $\mathrm{Sym}(V)$ associated to $\gamma \in V$ is denoted by a ket $|\alpha_1, ..., \alpha_n\rangle_\gamma$. The empty tensor at γ is denoted $|\emptyset\rangle_\gamma$.

With this notation, the coproduct and counit are given by

$$\Delta|\alpha_1, ..., \alpha_n\rangle_\gamma = \sum_{I \subseteq [n]} |\alpha_I\rangle_\gamma \otimes |\alpha_{I^c}\rangle_\gamma \qquad \text{and} \qquad \epsilon|\alpha_1, ..., \alpha_n\rangle_\gamma = \begin{cases} 1 & n = 0 \\ 0 & n > 0 \end{cases}$$

where $[n] = \{1, ..., n\}$ and for a (possibly empty) subset $I = \{i_1, ..., i_k\} \subseteq [n]$ we write $|\alpha_I\rangle_\gamma$ to mean $|\alpha_{i_1}, ..., \alpha_{i_k}\rangle_\gamma$. The dereliction map $d : !V \to V$ is given by

$$d|\alpha_1, ..., \alpha_n\rangle_\gamma = \begin{cases} \gamma & n = 0 \\ \alpha_1 & n = 1 \\ 0 & n > 1. \end{cases}$$

If $\varphi : !V \to W$ is a linear map then the promotion [14, Theorem 2.22] $\Phi : !V \to !W$ can be written explicitly as

$$\Phi|\alpha_1, ..., \alpha_n\rangle_\gamma = \sum_{P \in \mathscr{P}([n])} \left| \varphi|\alpha_{P_1}\rangle_\gamma, ..., \varphi|\alpha_{P_m}\rangle_\gamma \right\rangle_{\varphi|\emptyset\rangle_\gamma}$$

where $\mathscr{P}([n])$ denotes the set of partitions of $[n]$.

The explicit description above assumes that V is finite-dimensional. In the case where V is infinite-dimensional, one can write V as the direct limit of its finite-dimensional subspaces V_i, in which case $!V = \varinjlim !V_i$. In this case, any element of $!V$ can still be written as a finite sum of kets, and so the formulas for the coproduct, counit, dereliction map and promotion remain the same [14, Section 2.1].

2.2 The Sweedler semantics

Definition 2.11. We recursively define the **denotation** $[\![A]\!]$ of a formula A. For each atomic formula x_i, we choose a finite-dimensional k-vector space $[\![x_i]\!]$. For formulas A, B, define:

- $[\![A \,\&\, B]\!] = [\![A]\!] \oplus [\![B]\!]$,
- $[\![A \otimes B]\!] = [\![A]\!] \otimes [\![B]\!]$,
- $[\![A \multimap B]\!] = \mathrm{Hom}_k([\![A]\!], [\![B]\!])$,
- $[\![!A]\!] = ![\![A]\!]$.

If Γ is $A_1, ..., A_n$, then we define $[\![\Gamma]\!] = [\![A_1]\!] \otimes ... \otimes [\![A_n]\!]$.

Definition 2.12. Let π be a proof of $\Gamma \vdash B$. The **denotation** $[\![\pi]\!]$ is a linear map $[\![\pi]\!] : [\![\Gamma]\!] \to [\![B]\!]$ which is defined recursively on the structure of proofs. The proof π must match one of the proofs in the first column of the following table, and the second table defines its denotation. Here, d denotes the dereliction map, Δ the coproduct, and $\mathrm{prom}\,\varphi$ the promotion of φ as in Definition 2.8.

$\dfrac{}{A \vdash A}\ \text{axiom}$	$[\![\pi]\!](a) = a$
$\dfrac{\begin{array}{c}\pi_1 \\ \vdots \\ \Gamma, A, B, \Delta \vdash C\end{array}}{\Gamma, B, A, \Delta \vdash C}\ \text{exch}$	$[\![\pi]\!](\gamma \otimes b \otimes a \otimes \delta) = [\![\pi_1]\!](\gamma \otimes a \otimes b \otimes \delta)$
$\dfrac{\begin{array}{cc}\pi_1 & \pi_2 \\ \vdots & \vdots \\ \Gamma \vdash A & \Delta, A \vdash B\end{array}}{\Gamma, \Delta \vdash B}\ \text{cut}$	$[\![\pi]\!](\gamma \otimes \delta) = [\![\pi_2]\!](\delta \otimes [\![\pi_1]\!](\gamma))$
$\dfrac{\begin{array}{c}\pi_1 \\ \vdots \\ \Gamma, A \vdash C\end{array}}{\Gamma, A \,\&\, B \vdash C}\ \&L_0$	$[\![\pi]\!](\gamma \otimes (a, b)) = [\![\pi_1]\!](\gamma \otimes a)$

$\dfrac{\begin{array}{c}\pi_1\\ \vdots\\ \Gamma, B \vdash C\end{array}}{\Gamma, A \,\&\, B \vdash C}\ \&L_1$	$[\![\pi]\!](\gamma \otimes (a, b)) = [\![\pi_1]\!](\gamma \otimes b)$
$\dfrac{\begin{array}{cc}\pi_1 & \pi_2\\ \vdots & \vdots\\ \Gamma \vdash A & \Gamma \vdash B\end{array}}{\Gamma \vdash A \,\&\, B}\ \&R$	$[\![\pi]\!](\gamma) = ([\![\pi_1]\!](\gamma), [\![\pi_2]\!](\gamma))$
$\dfrac{\begin{array}{c}\pi_1\\ \vdots\\ \Gamma, A, B \vdash C\end{array}}{\Gamma, A \otimes B \vdash C}\ \otimes L$	$[\![\pi]\!](\gamma \otimes (a \otimes b)) = [\![\pi_1]\!](\gamma \otimes a \otimes b)$
$\dfrac{\begin{array}{cc}\pi_1 & \pi_2\\ \vdots & \vdots\\ \Gamma \vdash A & \Delta \vdash B\end{array}}{\Gamma, \Delta \vdash A \otimes B}\ \otimes R$	$[\![\pi]\!](\gamma \otimes \delta) = [\![\pi_1]\!](\gamma) \otimes [\![\pi_2]\!](\delta)$
$\dfrac{\begin{array}{cc}\pi_1 & \pi_2\\ \vdots & \vdots\\ \Gamma \vdash A & \Delta, B \vdash C\end{array}}{\Gamma, \Delta, A \multimap B \vdash C}\ \multimap L$	$[\![\pi]\!](\gamma \otimes \delta \otimes \varphi) = [\![\pi_2]\!](\delta \otimes \varphi \circ [\![\pi_1]\!](\gamma))$
$\dfrac{\begin{array}{c}\pi_1\\ \vdots\\ \Gamma, A \vdash B\end{array}}{\Gamma \vdash A \multimap B}\ \multimap R$	$[\![\pi]\!](\gamma) = \{a \mapsto [\![\pi_1]\!](\gamma \otimes a)\}$
$\dfrac{\begin{array}{c}\pi_1\\ \vdots\\ \Gamma, A \vdash B\end{array}}{\Gamma, !A \vdash B}\ \mathrm{der}$	$[\![\pi]\!](\gamma \otimes \overline{a}) = [\![\pi_1]\!](\gamma \otimes d(\overline{a}))$
$\dfrac{\begin{array}{c}\pi_1\\ \vdots\\ !\Gamma \vdash A\end{array}}{!\Gamma \vdash !A}\ \mathrm{prom}$	$[\![\pi]\!] = \mathrm{prom}[\![\pi_1]\!]$
$\dfrac{\begin{array}{c}\pi_1\\ \vdots\\ \Gamma \vdash B\end{array}}{\Gamma, !A \vdash B}\ \mathrm{weak}$	$[\![\pi]\!](\gamma \otimes \overline{a}) = [\![\pi_1]\!](\gamma)$
$\dfrac{\begin{array}{c}\pi_1\\ \vdots\\ \Gamma, !A, !A \vdash B\end{array}}{\Gamma, !A \vdash B}\ \mathrm{ctr}$	$[\![\pi]\!](\gamma \otimes \overline{a}) = [\![\pi_1]\!](\gamma \otimes \Delta(\overline{a}))$

Table 2.1: Denotations of proofs in the Sweedler semantics.

Since the exponential is modelled through the cofree coalgebra, we have the ability to encode nonlinear maps as follows:

Definition 2.13. Let π be a proof of $!A \vdash B$. There is an associated nonlinear map $[\![\pi]\!]_{\mathrm{nl}} : [\![A]\!] \to [\![B]\!]$ defined by $[\![\pi]\!]_{\mathrm{nl}}(\gamma) = [\![\pi]\!]|\emptyset\rangle_\gamma$.

Example 2.14. The fact that the function $\gamma \mapsto \gamma \otimes \gamma$ is nonlinear is captured by the fact that there is no proof of $A \vdash A \otimes A$ for a generic formula A in linear logic. There *is* however always a proof π of $!A \vdash A \otimes A$:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\overline{A \vdash A} \quad \overline{A \vdash A}}{A, A \vdash A \otimes A}\otimes R}{!A, A \vdash A \otimes A}\,\mathrm{der}}{!A, !A \vdash A \otimes A}\,\mathrm{der}}{!A \vdash A \otimes A}\,\mathrm{ctr}$$

The denotation of π is $[\![\pi]\!] = (d \otimes d) \circ \Delta$, and so the associated nonlinear map is

$$[\![\pi]\!]_{\mathrm{nl}}(\gamma) = (d \otimes d) \circ \Delta|\emptyset\rangle_\gamma = d|\emptyset\rangle_\gamma \otimes d|\emptyset\rangle_\gamma = \gamma \otimes \gamma.$$

In this sense, $[\![\pi]\!]_{\mathrm{nl}}$ validates our intuition that the proof π is 'copying' the input.

We can give an alternative description of $[\![\pi]\!]_{\mathrm{nl}}$. Let π' be the promotion of π to a proof of $!A \vdash !B$. Then $[\![\pi]\!]_{\mathrm{nl}}$ can be defined as the composite:

$$\begin{array}{ccccccc} [\![A]\!] & \xrightarrow{\cong} & G(![\![A]\!]) & \xrightarrow{G([\![\pi']\!])} & G(![\![B]\!]) & \xrightarrow{\cong} & [\![B]\!] \\ \gamma & \longmapsto & |\emptyset\rangle_\gamma & \longmapsto & |\emptyset\rangle_{[\![\pi]\!]|\emptyset\rangle_\gamma} & \longmapsto & [\![\pi]\!]|\emptyset\rangle_\gamma. \end{array}$$

The fact that this map is nonlinear is a consequence of the fact that for $c, d \in k$ and $\gamma, \delta \in [\![A]\!]$ we generally have:

$$c|\emptyset\rangle_\gamma + d|\emptyset\rangle_\delta \neq |\emptyset\rangle_{c\gamma + d\delta}.$$

There is also a map which corresponds to taking the image of the primitive elements of $![\![A]\!]$ under $[\![\pi']\!]$. If $\gamma \in [\![A]\!]$, and writing $\tilde{\gamma} = |\emptyset\rangle_\gamma$, consider the composite:

$$[\![A]\!] \xrightarrow{\cong} P_{\tilde{\gamma}}(![\![A]\!]) \xrightarrow{P_{\tilde{\gamma}}([\![\pi']\!])} P_{[\![\pi]\!](\tilde{\gamma})}(![\![B]\!]) \xrightarrow{\cong} [\![B]\!].$$

Denote this composite by $\partial_\gamma[\![\pi]\!]$. It is easily verified that $\partial_\gamma[\![\pi]\!]$ is linear[1]. This is reminiscent of ideas from differential geometry; recall that if M, N are smooth manifolds, $p \in M$ and $F : M \to N$ is a smooth map, the **derivative** of F at p (Figure 2.1) is a linear map $T_pF : T_pM \to T_{Fp}N$ between the tangent spaces, thought of as the best linear approximation to F at p. Given our interpretation of $|\alpha\rangle_\gamma$ as a tangent vector at γ in the direction of α (Remark 2.9), it is reasonable to understand the map $\partial_\gamma[\![\pi]\!]$ as being analogous $T_\gamma([\![\pi]\!]_{\mathrm{nl}})$. In summary:

[1]Note that the dependence on γ is generally nonlinear.

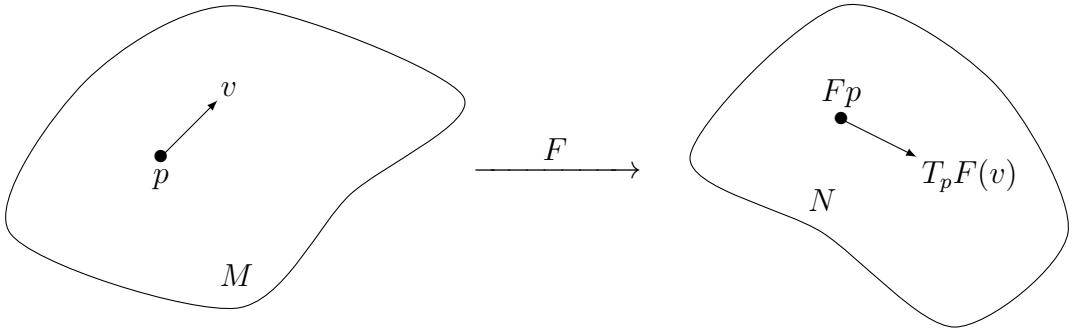


Figure 2.1: The derivative of a smooth map.

Definition 2.15. Let π be a proof of $!A \vdash B$, and $\gamma \in [\![A]\!]$. There is an associated linear map $\partial_\gamma[\![\pi]\!] : [\![A]\!] \to [\![B]\!]$ given by $\partial_\gamma[\![\pi]\!](\alpha) = [\![\pi]\!]|\alpha\rangle_\gamma$, called the **coalgebraic derivative** of $[\![\pi]\!]$ at γ. We also write $\partial[\![\pi]\!] : [\![A]\!] \times [\![A]\!] \to [\![B]\!]$ for the (nonlinear) function given by $\partial[\![\pi]\!](\alpha, \gamma) = [\![\pi]\!]|\alpha\rangle_\gamma$.

One of our goals in the next section is to justify this analogy, by showing that in many cases, the coalgebraic derivative really does behave as one would expect from elementary calculus.

3 Denotations and derivatives

Many simple data types such as booleans, integers and binary integers have a natural encoding as proofs in linear logic. We now provide a variety of examples of such proofs and their coalgebraic derivatives. In doing so, we hope the reader will develop a sense of the types of programming that is possible within linear logic.

3.1 Booleans

Definition 3.1. Let A be a formula. The type of **booleans on** A is

$$\mathbf{bool}_A = (A \,\&\, A) \multimap A.$$

The two values of a boolean correspond to the following proofs $\underline{0}_A$ and $\underline{1}_A$ of $\vdash \mathbf{bool}_A$:

$$\dfrac{\dfrac{\dfrac{}{A \vdash A}}{A \,\&\, A \vdash A}\,\&L_0}{\vdash \mathbf{bool}_A}\multimap R \qquad \dfrac{\dfrac{\dfrac{}{A \vdash A}}{A \,\&\, A \vdash A}\,\&L_1}{\vdash \mathbf{bool}_A}\multimap R$$

whose denotations are projection onto the zeroth and first coordinates respectively. Note that we are using the convention that the left introduction rules for & are indexed by 0 and 1, rather than by the more conventional choice of 1 and 2, in order to be consistent with the usual assignment of 0 as 'false' and 1 as 'true'.

A boolean can be used to 'choose' between two proofs π_0, π_1 of $\Gamma \vdash A$, via the following proof ρ:

$$\dfrac{\dfrac{\begin{array}{cc}\pi_0 & \pi_1\\ \vdots & \vdots\\ \Gamma \vdash A & \Gamma \vdash A\end{array}}{\Gamma \vdash A \,\&\, A}\,\&R \qquad \dfrac{}{A \vdash A}}{\Gamma, \mathbf{bool}_A \vdash A}\multimap L$$

The denotation of this proof is the map

$$[\![\rho]\!](\gamma \otimes \varphi) = \varphi([\![\pi_0]\!](\gamma), [\![\pi_1]\!](\gamma))$$

for $\gamma \in [\![\Gamma]\!]$ and $\varphi \in [\![\mathbf{bool}_A]\!]$. In particular, if $\varphi = [\![\underline{i}_A]\!]$, then this simplifies to

$$[\![\rho]\!](\gamma \otimes [\![\underline{i}_A]\!]) = [\![\pi_i]\!](\gamma).$$

The above construction easily generalises to provide an encoding of n-booleans in linear logic; that is, elements of the set $\{0, ..., n-1\}$. Let $A^n = A \,\&\, ... \,\&\, A$ where there are n copies of A.

Definition 3.2. Let $n \geq 2$. The type of **n-booleans on** A is ${}_n\mathbf{bool}_A = A^n \multimap A$.

The n possible values of an n-boolean correspond to the projection maps $\mathrm{proj}_i : [\![A^n]\!] \to [\![A]\!]$, where $i \in \{0, ..., n-1\}$. We denote by $\underline{i}_A$ the proof of $\vdash {}_n\mathbf{bool}_A$ whose denotation is proj_i; that is:

$$\dfrac{\dfrac{\dfrac{}{A \vdash A}}{A^n \vdash A}\,\&L_i}{\vdash {}_n\mathbf{bool}_A}\,\multimap R$$

Here, by $\&L_i$ $(0 \leq i \leq n-1)$ we mean the rule which introduces $n-1$ new copies of A on the left, such that the original A is at position i, indexed from left to right.

3.2 Integers

Another familiar data type which may be encoded in linear logic is that of integers. The idea is the same as that which is used in λ-calculus, in which the natural number n is encoded as the term $\lambda f.\lambda x.f^n x$; that is, the function $(f, x) \mapsto f^n(x)$. Of course, such a function has a nonlinear dependence on f, which suggests the following definition.

Definition 3.3. The type of **integers on** A is the formula

$$\mathbf{int}_A = !(A \multimap A) \multimap (A \multimap A).$$

Definition 3.4. Let $\underline{\mathrm{comp}}^0_A$ be the proof $\overline{A \vdash A}$. We recursively define $\underline{\mathrm{comp}}^n_A$ for $n \in \mathbb{N}$ as the proof

$$\dfrac{\dfrac{}{A \vdash A} \qquad \begin{array}{c}\underline{\mathrm{comp}}^{n-1}_A \\ \vdots \\ A, (n-1)\,(A \multimap A) \vdash A\end{array}}{A, n\,(A \multimap A) \vdash A}\,\multimap L$$

where $n\,(A \multimap A)$ is notation for n copies of $A \multimap A$.

An easy computation shows that $[\![\underline{\mathrm{comp}}^n_A]\!] : [\![A]\!] \otimes [\![A \multimap A]\!]^{\otimes n} \to [\![A]\!]$ is the map $a \otimes f_1 \otimes ... \otimes f_n \mapsto (f_n \circ ... \circ f_1)(a)$; note that the ordering of the f_i is reversed.

For $n \in \mathbb{N}$, there is a proof[2] $\underline{n}_A$ of $\vdash \mathbf{int}_A$ given by

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\begin{array}{c}\underline{\mathrm{comp}}^n_A \\ \vdots \\ A, n\,A \multimap A \vdash A\end{array}}{A, n\,!(A \multimap A) \vdash A}\,n\times \text{ der}}{A, !(A \multimap A) \vdash A}\,(n-1)\times \text{ ctr}}{!(A \multimap A) \vdash A \multimap A}\,\multimap R}{\vdash \mathbf{int}_A}\,\multimap R$$

[2]It will sometimes be convenient to exclude the final $\multimap R$ rule, instead writing $\underline{n}_A$ as a proof of $!(A \multimap A) \vdash A \multimap A$. This is reasonable since $\mathrm{Hom}_k(U \otimes V, W)$ and $\mathrm{Hom}_k(U, \mathrm{Hom}_k(V, W))$ are canonically isomorphic by currying, and semantically the $\multimap R$ rule corresponds to one direction of this isomorphism. We will usually not distinguish between proofs containing rules such as $\multimap L$ or $\otimes L$, and the corresponding proofs with those rules removed.

called the **Church numeral** for n. We occasionally drop the subscript A and simply write $\underline{n}$ when doing so will not cause confusion.

In order to describe the denotation of $\underline{n}_A$, for sets X, Y let $\mathrm{Inj}(X, Y)$ denote the set of all injective functions $X \to Y$.

Proposition 3.5. For $n > 0$ the denotation of $\underline{n}_A$ is the linear map

$$[\![\underline{n}_A]\!] : ![\![A \multimap A]\!] \to [\![A \multimap A]\!]$$

given by

$$[\![\underline{n}_A]\!]|\alpha_1, ..., \alpha_k\rangle_\gamma = \sum_{f \in \mathrm{Inj}([k],[n])} \Gamma_n^f \circ \cdots \circ \Gamma_1^f,$$

where $\alpha_1, ..., \alpha_k, \gamma \in [\![A \multimap A]\!] = \mathrm{End}([\![A]\!])$ and

$$\Gamma_i^f = \begin{cases} \alpha_{f^{-1}(i)} & i \in \mathrm{im}(f) \\ \gamma & i \notin \mathrm{im}(f). \end{cases}$$

In particular, $[\![\underline{n}_A]\!]$ vanishes on $|\alpha_1, ..., \alpha_k\rangle_\gamma$ if $k > n$.

Intuitively speaking, the value of $[\![\underline{n}_A]\!]|\alpha_1, ..., \alpha_k\rangle_\gamma$ is the sum over all ways of replacing k of the copies of γ in the composite γ^n with the maps $\alpha_1, ..., \alpha_k$.

Proof. It is easily seen that $[\![\underline{n}_A]\!] = [\![\underline{\mathrm{comp}}_A^n]\!] \circ d^{\otimes n} \circ \Delta^{n-1}$. The image of $|\alpha_1, ..., \alpha_k\rangle_\gamma$ under $n-1$ contractions is

$$\sum_{I_1,...,I_n} |\alpha_{I_1}\rangle_\gamma \otimes ... \otimes |\alpha_{I_n}\rangle_\gamma$$

where the sum ranges over all sequences of subsets $I_1, ..., I_n \subseteq [k]$ which are pairwise disjoint and such that $I_1 \cup ... \cup I_n = [k]$. Applying $d^{\otimes n}$ annihilates any term of the sum which contains a ket of length greater than one. Any summand which remains corresponds to a choice of k distinct elements of $[n]$, or equivalently an injective function $[k] \to [n]$. Finally, $\underline{\mathrm{comp}}_A^n$ composes the operators. □

Corollary 3.6. For $\alpha, \gamma \in [\![A \multimap A]\!]$, we have

$$[\![\underline{n}_A]\!]_{\mathrm{nl}}(\gamma) = \gamma^n \qquad \text{and} \qquad \partial_\gamma [\![\underline{n}_A]\!](\alpha) = \sum_{i=1}^n \gamma^{i-1} \alpha \gamma^{n-i}.$$

It is in this sense that the Church numeral $\underline{n}_A$ can be thought of as encoding the nonlinear function $(f, x) \mapsto f^n(x)$. Moreover, at least in this simple example, there is a clear connection between the coalgebraic derivative and the ordinary tangent map of smooth manifolds:

Example 3.7. With $k = \mathbb{C}$ and $[\![A]\!] = \mathbb{C}^n$, let $X = [\![\underline{2}_A]\!]_{\mathrm{nl}} : M_n(\mathbb{C}) \to M_n(\mathbb{C})$, which is the map $\gamma \mapsto \gamma^2$. For $1 \leq a, b \leq n$ let $X_{a,b} : M_n(\mathbb{C}) \to \mathbb{C}$ denote the component of X obtained by projecting onto the ath row and bth column. Explicitly, for $\gamma \in M_n(\mathbb{C})$ we have:

$$X_{a,b}(\gamma) = \sum_{i=1}^{n} \gamma_{a,i}\gamma_{i,b}.$$

This is a smooth map of real manifolds with tangent map $T_\gamma X : T_\gamma M_n(\mathbb{C}) \to T_{\gamma^2} M_n(\mathbb{C})$. Under the identification $T_\gamma M_n(\mathbb{C}) = T_{\gamma^2} M_n(\mathbb{C}) = M_n(\mathbb{C})$, for matrices $\alpha, \gamma \in M_n(\mathbb{C})$, the matrix $T_\gamma X(\alpha) \in M_n(\mathbb{C})$ has entries:

$$\begin{aligned}
(T_\gamma X)_{a,b}(\alpha) &= \sum_{1 \leq c,d \leq n} \alpha_{c,d} \frac{\partial X_{a,b}}{\partial \gamma_{c,d}} \\
&= \sum_{1 \leq c,d \leq n} \alpha_{c,d}(\delta_{a,c}\gamma_{d,b} + \delta_{b,d}\gamma_{a,c}) \\
&= \sum_{d=1}^{n} \alpha_{a,d}\gamma_{d,b} + \sum_{c=1}^{n} \alpha_{c,b}\gamma_{a,c} \\
&= (\alpha\gamma + \gamma\alpha)_{a,b},
\end{aligned}$$

where δ is the Kronecker delta. So the tangent map $T_\gamma X$ agrees with $\partial_\gamma [\![\underline{2}_A]\!]$, and by a similar computation one can show that the same holds for any Church numeral $\underline{m}_A$.

We will shortly present some examples of the kinds of functions of integers which can be encoded in linear logic. To justify their computational meaning, we will typically argue semantically rather than by actually performing the cut elimination procedure on the proofs themselves. A priori, this is not sufficient since the computational meaning of the proof is a syntactic notion, and just because two proofs have the same denotation does not guarantee that they are equivalent under cut elimination. Fortunately for integers, this turns out to be the case.

Proposition 3.8. Let A be a formula with $\dim [\![A]\!] > 0$. The set $\{[\![\underline{n}_A]\!]\}_{n \geq 0}$ is linearly independent in $[\![\mathbf{int}_A]\!]$.

Proof. Suppose that $\sum_{i=0}^{n} c_i [\![\underline{i}]\!] = 0$ for some scalars $c_0, ..., c_n \in k$. Let $p \in k[x]$ be the polynomial $p = \sum_{i=0}^{n} c_i x^i$. For $\alpha \in [\![A \multimap A]\!]$ we have

$$p(\alpha) = \sum_{i=0}^{n} c_i \alpha^i = \sum_{i=0}^{n} c_i [\![\underline{i}]\!] |\emptyset\rangle_\alpha = 0.$$

The minimal polynomial of α therefore divides p. Since this holds for any linear map $\alpha \in [\![A \multimap A]\!]$, it follows that p is identically zero, as k is characteristic zero. □

Corollary 3.9. The function $\mathbb{N} \to [\![\mathbf{int}_A]\!]$ given by $n \mapsto [\![\underline{n}_A]\!]$ is injective.

The upshot of this is that we may deduce facts about the behaviour of proofs of sequents of the form $\Gamma \vdash \mathbf{int}_A$ under cut elimination by considering their denotations in the Sweedler semantics. This is convenient, as in practice performing the cut elimination procedure for longer proofs is both laborious and unilluminating when presented directly, and indeed it seems that it is rarely carried out in the literature.

For the following examples, let A be a formula and write $E = A \multimap A$.

Example 3.10. The proof $\underline{\mathrm{add}}_A$ is

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\underline{\mathrm{comp}}^2_A}{\vdots}}{A, E, E \vdash A}}{E, E \vdash E}\multimap R}{!E \vdash !E \qquad !E, E, \mathbf{int}_A \vdash E}\multimap L}{!E \vdash !E \qquad !E, !E, \mathbf{int}_A, \mathbf{int}_A \vdash E}\multimap L}{!E, !E, \mathbf{int}_A, \mathbf{int}_A \vdash E}\,\mathrm{ctr}}{!E, \mathbf{int}_A, \mathbf{int}_A \vdash E}\multimap R}{\mathbf{int}_A, \mathbf{int}_A \vdash \mathbf{int}_A}$$

Let m, n be integers. If $\underline{\mathrm{add}}_A$ is cut against the Church numerals $\underline{m}_A$ and $\underline{n}_A$, the resulting proof is equivalent under cut elimination to $\underline{m+n}_A$.

Arguing semantically, we compute the denotation of $\underline{\mathrm{add}}_A$ when evaluated on the Church numerals $\underline{m}_A$ and $\underline{n}_A$ and then on $|\alpha_1, ..., \alpha_s\rangle_\gamma \in [\![!E]\!]$:

$$\begin{aligned}
[\![\underline{\mathrm{add}}_A]\!]([\![\underline{m}_A]\!] \otimes [\![\underline{n}_A]\!])|\alpha_1, ..., \alpha_s\rangle_\gamma &= [\![\underline{\mathrm{comp}}^2_A]\!](([\![\underline{m}_A]\!] \otimes [\![\underline{n}_A]\!])\Delta|\alpha_1, ..., \alpha_s\rangle_\gamma) \\
&= [\![\underline{\mathrm{comp}}^2_A]\!] \left(\sum_{I \subseteq [s]} [\![\underline{m}_A]\!]|\alpha_I\rangle_\gamma \otimes [\![\underline{n}_A]\!]|\alpha_{I^c}\rangle_\gamma \right) \\
&= \sum_{I \subseteq [s]} [\![\underline{n}_A]\!]|\alpha_{I^c}\rangle_\gamma \circ [\![\underline{m}_A]\!]|\alpha_I\rangle_\gamma.
\end{aligned}$$

By Proposition 3.5, the term $[\![\underline{n}_A]\!]|\alpha_{I^c}\rangle_\gamma \circ [\![\underline{m}_A]\!]|\alpha_I\rangle_\gamma$ is a sum over all ways of replacing $|I^c|$ copies of γ in γ^n by the entries in α_{I^c}, composed with all ways of replacing $|I|$ copies of γ in γ^m by the entries in α_I. By summing over all subsets $I \subseteq [s]$, this is equal to

$$[\![\underline{m+n}_A]\!]|\alpha_1, ..., \alpha_s\rangle_\gamma,$$

and so $[\![\underline{\mathrm{add}}_A]\!]([\![\underline{m}_A]\!] \otimes [\![\underline{n}_A]\!]) = [\![\underline{m+n}_A]\!]$.

Example 3.11. There is also a proof $\underline{\mathrm{mult}}_A$ in linear logic which multiplies integers. Consider the following intermediate proof π:

$$\dfrac{\dfrac{\dfrac{\dfrac{!E \vdash !E \qquad E \vdash E}{!E, \mathbf{int}_A \vdash E}\multimap L}{!E, !\mathbf{int}_A \vdash E}\,\mathrm{der}}{!E, !\mathbf{int}_A \vdash !E}\,\mathrm{prom}$$

Then $[\![\pi]\!] : !\,\mathrm{End}[\![A]\!] \otimes ![\![\mathbf{int}_A]\!] \to !\,\mathrm{End}[\![A]\!]$ is a morphism of coalgebras such that

$$[\![\pi]\!](x \otimes |\emptyset\rangle_\gamma) = |\emptyset\rangle_{\gamma(x)} \qquad \text{and} \qquad [\![\pi]\!](x \otimes |\alpha\rangle_\gamma) = |\alpha(x)\rangle_{\gamma(x)},$$

for any $\alpha, \gamma \in [\![\mathbf{int}_A]\!] = \mathrm{Hom}_k(!\,\mathrm{End}[\![A]\!], \mathrm{End}[\![A]\!])$. Using π, we can write down the proof $\underline{\mathrm{mult}}_A$ which corresponds to multiplying two integers:

$$\dfrac{\dfrac{\begin{array}{c}\pi\\ \vdots\\ !E, !\mathbf{int}_A \vdash !E\end{array} \qquad \overline{E \vdash E}}{!E, !\mathbf{int}_A, \mathbf{int}_A \vdash E} \multimap L}{!\mathbf{int}_A, \mathbf{int}_A \vdash \mathbf{int}_A} \multimap R$$

Now let l, m, n be integers and write $\gamma = [\![\underline{l}]\!]$ and $\alpha = [\![\underline{m}]\!]$. Then the coalgebraic derivative of $\underline{\mathrm{mult}}_A(-, n)$ at α in the direction of γ is (for $t \in [\![!E]\!]$):

$$[\![\underline{\mathrm{mult}}_A]\!](|\alpha\rangle_\gamma \otimes [\![\underline{n}]\!])(t) = [\![\underline{n}]\!](|\alpha(t)\rangle_{\gamma(t)}) = \sum_{i=1}^{n} \gamma(t)^{i-1}\alpha(t)\gamma(t)^{n-i}.$$

In the case where $t = |\emptyset\rangle_x$, this evaluates to:

$$\sum_{i=1}^{n} \gamma(t)^{i-1}\alpha(t)\gamma(t)^{n-i} = \sum_{i=1}^{n} x^{l(i-1)}x^m x^{l(n-i)} = nx^{l(n-1)+m}.$$

If we take $k = \mathbb{C}$, this result agrees with a more traditional calculus approach using limits:

$$\begin{aligned}
&\lim_{h\to 0} \frac{[\![\underline{\mathrm{mult}}_A]\!](|\emptyset\rangle_{[\![\underline{l}]\!]+h[\![\underline{m}]\!]} \otimes [\![\underline{n}]\!])|\emptyset\rangle_x - [\![\underline{\mathrm{mult}}_A]\!](|\emptyset\rangle_{[\![\underline{l}]\!]} \otimes [\![\underline{n}]\!])|\emptyset\rangle_x}{h}\\
=&\lim_{h\to 0} \frac{[\![\underline{n}]\!]|\emptyset\rangle_{x^l+hx^m} - [\![\underline{n}]\!]|\emptyset\rangle_{x^l}}{h}\\
=&\lim_{h\to 0} \frac{(x^l + hx^m)^n - x^{ln}}{h}\\
=&nx^{l(n-1)+m}.
\end{aligned}$$

Example 3.12. A somewhat more complicated example is that of the predecessor $\underline{\mathrm{pred}}_A$, which encodes $n \mapsto n - 1$. The construction we will use is from [9, §2.5.2]; the idea is to iterate the function

$$(a_0, a_1) \mapsto (a_1, f(a_1))$$

n times, and then project onto the first component. For this to be possible, we will need $\underline{\mathrm{pred}}_A$ to be a proof of the sequent $\mathbf{int}_{A^2} \vdash \mathbf{int}_A$, where $A^2 = A \,\&\, A$.

Define π as the following proof:

$$\dfrac{\dfrac{\dfrac{}{A \vdash A} \quad \dfrac{}{A \vdash A}\,\&R \qquad \dfrac{\dfrac{}{A \vdash A}}{A^2 \vdash A}\,\&L_0}{A \vdash A^2 \multimap A^2 \vdash A}\multimap L}{A, A^2 \multimap A^2 \vdash A}\multimap R}{A^2 \multimap A^2 \vdash A \multimap A}\multimap R$$

Then it is easily verified that $[\![\pi]\!](\varphi)(a) = \mathrm{proj}_0(\varphi(a,a))$, for $a \in [\![A]\!]$, $\varphi \in \mathrm{End}([\![A^2]\!])$. Now, let ρ be the following proof:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{A \vdash A}\,\&L_1}{A^2 \vdash A}\,\mathrm{weak}}{A^2, !(A \multimap A) \vdash A} \qquad \dfrac{\dfrac{\dfrac{\dfrac{}{A \vdash A} \quad \dfrac{}{A \vdash A}\multimap L}{A, A \multimap A \vdash A}\,\&L_1}{A^2, A \multimap A \vdash A}\,\mathrm{der}}{A^2, !(A \multimap A) \vdash A}\,\&R}{A^2, !(A \multimap A) \vdash A^2}\multimap R}{!(A \multimap A) \vdash A^2 \multimap A^2}\multimap R$$

The denotation $[\![\rho]\!]$ is, for $a_0, a_1 \in [\![A]\!]$ and $\alpha_1, ..., \alpha_s, \gamma \in \mathrm{End}([\![A]\!])$:

$$[\![\rho]\!](|\alpha_1, ..., \alpha_s\rangle_\gamma)(a_0, a_1) = (a_1, d(|\alpha_1, ..., \alpha_s\rangle_\gamma)(a_1)) = \begin{cases} (a_1, \gamma a_1) & s = 0 \\ (a_1, \alpha_1 a_1) & s = 1 \\ (a_1, 0) & s > 1. \end{cases}$$

Finally, define $\underline{\mathrm{pred}}_A$ as the following proof:

$$\dfrac{\dfrac{\dfrac{\rho}{\vdots}}{!(A \multimap A) \vdash A^2 \multimap A^2}\,\mathrm{prom}}{!(A \multimap A) \vdash !(A^2 \multimap A^2)} \qquad \dfrac{\dfrac{\pi}{\vdots}}{A^2 \multimap A^2 \vdash A \multimap A}\multimap L}{!(A \multimap A), \mathbf{int}_{A^2} \vdash A \multimap A}\multimap R}{\mathbf{int}_{A^2} \vdash \mathbf{int}_A}\multimap R$$

While a full description of the corresponding function is somewhat complicated, consider its evaluation on the Church numeral $[\![\underline{n}_{A^2}]\!]$. This gives an element of $[\![\mathbf{int}_A]\!]$, which if fed a vacuum vector $|\emptyset\rangle_\gamma$ will return the linear map

$$[\![\pi]\!](([\![\rho]\!]|\emptyset\rangle_\gamma)^n) : [\![A]\!] \to [\![A]\!].$$

Writing $\tilde{\gamma} = [\![\rho]\!]|\emptyset\rangle_\gamma$, which is the morphism $(a_0, a_1) \mapsto (a_1, \gamma a_1)$, we therefore have:

$$\begin{aligned} [\![\underline{\mathrm{pred}}_A]\!]([\![\underline{n}_{A^2}]\!])(a) &= [\![\pi]\!](\tilde{\gamma}^n)(a) \\ &= \mathrm{proj}_0(\tilde{\gamma}^n(a,a)) \\ &= \mathrm{proj}_0(\gamma^{n-1}(a), \gamma^n(a)) \\ &= \gamma^{n-1}(a) \end{aligned}$$

which indeed corresponds to the Church numeral $\underline{n-1}_A$.

3.3 Binary integers

Definition 3.13. The type of **binary integers on** A is

$$\mathbf{bint}_A = !(A \multimap A) \multimap (!(A \multimap A) \multimap (A \multimap A)).$$

For any $n \geq 0$ and any $T \in \{0,1\}^n$, there exists an associated proof $\underline{T}_A$ of $\vdash \mathbf{bint}_A$:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\mathrm{comp}_A^n}{\vdots}}{A, n\,(A \multimap A) \vdash A}}{A, n\,!(A \multimap A) \vdash A}\, n\times \text{ der}}{A, !(A \multimap A), !(A \multimap A) \vdash A}\, \text{ctr, possibly weak}}{!(A \multimap A), !(A \multimap A) \vdash A \multimap A}\, \multimap R}{!(A \multimap A) \vdash \mathbf{int}_A}\, \multimap R}{\vdash \mathbf{bint}_A}\, \multimap R$$

The contraction and weakening steps are determined as follows. Associate the ith copy of $!(A \multimap A)$ with the ith digit of the binary integer T reading from left to right, and perform contractions to collapse all the 0-associated copies of $!(A \multimap A)$ down to a single copy. Similarly perform contractions on the 1-associated copies. Once this is done, the result is two copies of $!(A \multimap A)$ on the left, unless T contains no zeroes or no ones in which case we introduce the missing copies by weakening. Finally, we move the 1-associated copy of $!(A \multimap A)$ across to the right of the turnstile, followed by the 0-associated copy.

Example 3.14. The proof associated to the binary integer $T = 0101$ is given below. The 0-associated copies are coloured blue, and the 1-associated copies are coloured red.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{A \vdash A}\quad \dfrac{\dfrac{}{A \vdash A}\quad \dfrac{}{A \vdash A}}{A, A \multimap A \vdash A}\, \multimap L}{A, A \multimap A, A \multimap A \vdash A}\, \multimap L}{A, A \multimap A, A \multimap A, A \multimap A \vdash A}\, \multimap L}{A, !(A \multimap A), !(A \multimap A), !(A \multimap A), !(A \multimap A) \vdash A}\, 4\times \text{ der}}{A, !(A \multimap A), !(A \multimap A), !(A \multimap A) \vdash A}\, \text{ctr}}{A, !(A \multimap A), !(A \multimap A) \vdash A}\, \text{ctr}}{A, !(A \multimap A), !(A \multimap A) \vdash A}\, \multimap R}{!(A \multimap A), !(A \multimap A) \vdash A \multimap A}\, \multimap R}{!(A \multimap A) \vdash \mathbf{int}_A}\, \multimap R}{\vdash \mathbf{bint}_A}\, \multimap R$$

The denotation of binary integers is as one would expect from Proposition 3.5; we omit the proof as it is essentially the same.

Proposition 3.15. For $T \in \{0,1\}^n$, let t_i be the ith digit of T and let $T_0 = \{i \mid t_i = 0\}$ and $T_1 = \{i \mid t_i = 1\}$. Then

$$[\![\underline{T}_A]\!](|\alpha_1, ..., \alpha_k\rangle_\gamma \otimes |\beta_1, ..., \beta_l\rangle_\delta) = \sum_{f \in \mathrm{Inj}([k], T_0)} \sum_{g \in \mathrm{Inj}([l], T_1)} \Gamma_n^{f,g} \circ \cdots \circ \Gamma_1^{f,g}.$$

where

$$\Gamma_i^{f,g} = \begin{cases} \alpha_{f^{-1}(i)} & i \in \mathrm{im}(f) \\ \beta_{g^{-1}(i)} & i \in \mathrm{im}(g) \\ \gamma & i \in T_0 \setminus \mathrm{im}(f) \\ \delta & i \in T_1 \setminus \mathrm{im}(g). \end{cases}$$

In particular, $[\![\underline{T}_A]\!]$ vanishes on $|\alpha_1, ..., \alpha_k\rangle_\gamma \otimes |\beta_1, ..., \beta_l\rangle_\delta$ if $k > |T_0|$ or $l > |T_1|$.

Note that the ordering of the composition is reversed; as an example, for the binary sequence 110 we have

$$[\![\underline{110}_A]\!]_{\mathrm{nl}}(\gamma \otimes \delta) = \gamma \circ \delta \circ \delta.$$

Example 3.16. Let $E = A \multimap A$. The proof $\underline{\mathrm{concat}}_A$ is the analogue of $\underline{\mathrm{add}}_A$ for binary integers:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\underline{\mathrm{comp}}^2_A}{\vdots}}{A, E, E \vdash A}}{E, E \vdash E} \multimap L}{!E \vdash !E \quad !E, E, \mathbf{int}_A \vdash E} \multimap L}{!E \vdash !E \quad !E, !E, E, \mathbf{bint}_A \vdash E} \multimap L}{!E \vdash !E \quad !E, !E, !E, \mathbf{int}_A, \mathbf{bint}_A \vdash E} \multimap L}{!E, !E, !E, !E, \mathbf{bint}_A, \mathbf{bint}_A \vdash E} 2\times \text{ ctr}}{!E, !E, \mathbf{bint}_A, \mathbf{bint}_A \vdash E} 2\times \multimap R}{\mathbf{bint}_A, \mathbf{bint}_A \vdash \mathbf{bint}_A}$$

Once again, the colours indicate which copies of $!E$ are contracted together. When cut against the proofs of binary integers $\underline{S}_A$ and $\underline{T}_A$, the resulting proof will be equivalent under cut elimination to $\underline{ST}_A$. We write $\underline{\mathrm{concat}}(S, -)$ for the proof of $\mathbf{bint}_A \vdash \mathbf{bint}_A$ obtained by cutting a binary integer $\underline{S}_A$ against $\underline{\mathrm{concat}}_A$ such that the first $\mathbf{bint}_A$ is consumed; meaning that $\underline{\mathrm{concat}}(S, -)$ prepends by S. Similarly define $\underline{\mathrm{concat}}(-, T)$ as the proof which appends by T.

Example 3.17. There is a proof $\underline{\mathrm{repeat}}_A$ of $!\mathbf{bint}_A \vdash \mathbf{bint}_A$ which repeats a binary sequence [15, Definition 3.13], in the sense that $[\![\underline{\mathrm{repeat}}_A]\!]|\emptyset\rangle_{[\![S]\!]} = [\![\underline{SS}]\!]$. It is given by

$$\dfrac{\dfrac{\dfrac{\underline{\mathrm{concat}}_A}{\vdots}}{\mathbf{bint}_A, \mathbf{bint}_A \vdash \mathbf{bint}_A} 2\times \text{ der}}{!\mathbf{bint}_A, !\mathbf{bint}_A \vdash \mathbf{bint}_A} \text{ ctr}}{!\mathbf{bint}_A \vdash \mathbf{bint}_A}$$

This is easily seen to have the required denotation by reading the above proof tree from bottom to top:

$$|\emptyset\rangle_{[\![\underline{S}]\!]} \xmapsto{\text{ctr}} |\emptyset\rangle_{[\![\underline{S}]\!]} \otimes |\emptyset\rangle_{[\![\underline{S}]\!]} \xmapsto{2\times \text{ der}} [\![\underline{S}]\!] \otimes [\![\underline{S}]\!] \xmapsto{[\![\underline{\text{concat}}_A]\!]} [\![\underline{SS}]\!].$$

Likewise, if S and T are binary sequences, we can compute the coalgebraic derivative of $\underline{\text{repeat}}_A$ at S in the direction of T as follows:

$$\begin{aligned} |[\![\underline{T}]\!]\rangle_{[\![\underline{S}]\!]} &\xmapsto{\text{ctr}} |[\![\underline{T}]\!]\rangle_{[\![\underline{S}]\!]} \otimes |\emptyset\rangle_{[\![\underline{S}]\!]} + |\emptyset\rangle_{[\![\underline{S}]\!]} \otimes |[\![\underline{T}]\!]\rangle_{[\![\underline{S}]\!]} \\ &\xmapsto{2\times \text{ der}} [\![\underline{T}]\!] \otimes [\![\underline{S}]\!] + [\![\underline{S}]\!] \otimes [\![\underline{T}]\!] \\ &\xmapsto{[\![\underline{\text{concat}}_A]\!]} [\![\underline{ST}]\!] + [\![\underline{TS}]\!]. \end{aligned}$$

Note that again this agrees with the usual definition of a derivative when $k = \mathbb{C}$:

$$\begin{aligned} &\lim_{h\to 0} \frac{[\![\underline{\text{repeat}}_A]\!](|\emptyset\rangle_{[\![\underline{S}]\!]+h[\![\underline{T}]\!]}) - [\![\underline{\text{repeat}}_A]\!](|\emptyset\rangle_{[\![\underline{S}]\!]})}{h} \\ =&\lim_{h\to 0} \frac{[\![\underline{\text{concat}}_A]\!](([\![\underline{S}]\!] + h[\![\underline{T}]\!]) \otimes ([\![\underline{S}]\!] + h[\![\underline{T}]\!])) - [\![\underline{SS}]\!])}{h} \\ =&\lim_{h\to 0} \frac{[\![\underline{SS}]\!] + h([\![\underline{ST}]\!] + [\![\underline{TS}]\!]) + h^2[\![\underline{TT}]\!] - [\![\underline{SS}]\!])}{h} \\ =&[\![\underline{ST}]\!] + [\![\underline{TS}]\!]. \end{aligned}$$

Here, to make sense of this limit, we are working in the finite-dimensional subspace of $[\![\mathbf{bint}_A]\!]$ spanned by the vectors $[\![\underline{SS}]\!]$, $[\![\underline{ST}]\!]$, $[\![\underline{TS}]\!]$ and $[\![\underline{TT}]\!]$. We will further investigate limits of this kind in Section 6.

We now investigate the question of whether distinct binary integers have linearly independent denotations. Surprisingly it turns out that this does not hold in general. In fact, for an atomic formula A it is impossible to choose $[\![A]\!]$ finite-dimensional such that all binary integers are linearly independent in $[\![\mathbf{bint}_A]\!]$, which stands in contrast to the situation with integers (Proposition 3.8).

For a binary integer T, one can write the value of $[\![\underline{T}_A]\!]$ on arbitrary kets as a sum of its values on vacuum vectors. This will simplify the task of checking whether binary integers have linearly dependent denotations; at least in the case where we have a fixed number of zeroes and ones, we only need to check linear dependence after evaluating on vacuum vectors. From this point onward let A be a fixed type and write $[\![\underline{T}]\!]$ for $[\![\underline{T}_A]\!]$.

Proposition 3.18. Let $m, n \geq 0$, and let $T \in \{0,1\}^*$ contain exactly m zeroes and n ones. Then for $\alpha_i, \beta_i, \gamma, \delta \in [\![A \multimap A]\!]$ we have:

$$[\![\underline{T}]\!](|\alpha_1, ..., \alpha_m\rangle_\gamma, |\beta_1, ..., \beta_n\rangle_\delta) = \sum_{I\subseteq[m]} \sum_{J\subseteq[n]} (-1)^{m-|I|}(-1)^{n-|J|}[\![\underline{T}]\!] \left(|\emptyset\rangle_{\sum_{i\in I} \alpha_i}, |\emptyset\rangle_{\sum_{j\in J} \beta_j} \right). \quad (*)$$

Note that the right hand side does not depend on γ and δ.

Proof. A single term of the double sum on the right hand side of $(*)$ corresponds to replacing (in the reversal of T) each 0 with $\sum_i \alpha_i$ and each 1 with $\sum_j \beta_j$. After expanding these sums, consider the coefficient of a particular noncommutative monomial p which contains only the variables $\alpha_{i_1}, ..., \alpha_{i_k}, \beta_{j_1}, ..., \beta_{j_l}$, where $\{i_1, ..., i_k\} \subseteq [m]$ and $\{j_1, ..., j_l\} \subseteq [n]$. The terms in the right hand side of $(*)$ which contribute to this coefficient are precisely those for which the set I contains each of $i_1, ..., i_k$, and J contains each of $j_1, ..., j_l$. Hence the coefficient of p is

$$\begin{aligned}\sum_{\substack{\{i_1,...,i_k\}\subseteq I\subseteq[m]\\ \{j_1,...,j_l\}\subseteq J\subseteq[n]}} (-1)^{m-|I|}(-1)^{n-|J|} &= \sum_{i=k}^{m}\sum_{j=l}^{n}(-1)^{m-i}(-1)^{n-j}\binom{m-k}{i-k}\binom{n-l}{j-l} \\ &= \sum_{i=k}^{m}(-1)^{m-i}\binom{m-k}{i-k}\sum_{j=l}^{n}(-1)^{n-j}\binom{n-l}{j-l} \\ &= \begin{cases} 1 & m=k \text{ and } n=l \\ 0 & \text{otherwise.}\end{cases}\end{aligned}$$

So the only monomials p with non-zero coefficient are those which contain each α_i and each β_j exactly once, which agrees with the left hand side of $(*)$ by Proposition 3.15. □

In order to make the content of the above proposition clear, we give a concrete example involving a specific binary sequence.

Example 3.19. Let $T = 0010$. The left hand side of $(*)$ is therefore

$$\sum_{\sigma\in S_3} \alpha_{\sigma(1)}\beta_1\alpha_{\sigma(2)}\alpha_{\sigma(3)}$$

while the right hand side is

$$\begin{aligned}&(\alpha_1+\alpha_2+\alpha_3)\beta_1(\alpha_1+\alpha_2+\alpha_3)(\alpha_1+\alpha_2+\alpha_3) - (\alpha_1+\alpha_2)\beta_1(\alpha_1+\alpha_2)(\alpha_1+\alpha_2) - \\ &(\alpha_1+\alpha_3)\beta_1(\alpha_1+\alpha_3)(\alpha_1+\alpha_3) - (\alpha_2+\alpha_3)\beta_1(\alpha_2+\alpha_3)(\alpha_2+\alpha_3) + \\ &\alpha_1\beta_1\alpha_1\alpha_1 + \alpha_2\beta_1\alpha_2\alpha_2 + \alpha_3\beta_1\alpha_3\alpha_3.\end{aligned}$$

After expanding the above, consider for example the monomial $\alpha_1\beta_1\alpha_1\alpha_1$:

$$\begin{aligned}&(\alpha_1+\alpha_2+\alpha_3)\beta_1(\alpha_1+\alpha_2+\alpha_3)(\alpha_1+\alpha_2+\alpha_3) - (\alpha_1+\alpha_2)\beta_1(\alpha_1+\alpha_2)(\alpha_1+\alpha_2) - \\ &(\alpha_1+\alpha_3)\beta_1(\alpha_1+\alpha_3)(\alpha_1+\alpha_3) - (\alpha_2+\alpha_3)\beta_1(\alpha_2+\alpha_3)(\alpha_2+\alpha_3) + \\ &\alpha_1\beta_1\alpha_1\alpha_1 + \alpha_2\beta_1\alpha_2\alpha_2 + \alpha_3\beta_1\alpha_3\alpha_3.\end{aligned}$$

As expected, the coefficient is $\binom{2}{2} - \binom{2}{1} + \binom{2}{0} = 1 - 2 + 1 = 0$.

Corollary 3.20. Let $m, n \geq 0$, let $T \in \{0,1\}^*$ contain exactly m zeroes and n ones, and let $0 \leq k \leq m$ and $0 \leq l \leq n$. Then

$$[\![\underline{T}]\!](|\alpha_1, ..., \alpha_k\rangle_\gamma, |\beta_1, ..., \beta_l\rangle_\delta) = \sum_{I\subseteq[m]}\sum_{J\subseteq[n]} \frac{(-1)^{m-|I|}(-1)^{n-|J|}}{(m-k)!(n-l)!}[\![\underline{T}]\!]\left(|\emptyset\rangle_{\sum_{i\in I}\alpha_i}, |\emptyset\rangle_{\sum_{j\in J}\beta_j}\right),$$

where on the right hand side we define $\alpha_i = \gamma$ for $i > k$ and $\beta_j = \delta$ for $j > l$.

Proof. Note that by Proposition 3.15

$$[\![\underline{T}]\!](|\alpha_1, ..., \alpha_k\rangle_\gamma, |\beta_1, ..., \beta_l\rangle_\delta) = \frac{[\![\underline{T}]\!](|\alpha_1, ..., \alpha_k, \gamma, ..., \gamma\rangle_\gamma, |\beta_1, ..., \beta_l, \delta, ..., \delta\rangle_\delta)}{(m-k)!(n-l)!},$$

where the kets on the right have length m and n respectively. Then apply the previous proposition. □

Lemma 3.21. Let $T_1, ..., T_r \in \{0,1\}^*$ each contain exactly m zeroes and n ones, and let $c_1, ..., c_r \in k\backslash\{0\}$. Then $\sum_s c_s[\![\underline{T_s}]\!] = 0$ in $[\![\mathbf{bint}_A]\!]$ if and only if $\sum_s c_s[\![\underline{T_s}]\!](|\emptyset\rangle_\alpha, |\emptyset\rangle_\beta) = 0$ in $[\![A \multimap A]\!]$ for all $\alpha, \beta \in [\![A \multimap A]\!]$.

Proof. ($\Rightarrow$) is immediate. For ($\Leftarrow$), suppose that $\sum_{s=1}^r c_s[\![\underline{T_s}]\!](|\emptyset\rangle_\alpha, |\emptyset\rangle_\beta) = 0$ for all α, β. It follows from Corollary 3.20 that

$$\begin{aligned}
&\sum_{s=1}^r c_s[\![\underline{T_s}]\!](|\alpha_1, ..., \alpha_k\rangle_\gamma, |\beta_1, ..., \beta_l\rangle_\delta) \\
&= \sum_{I\subseteq[m]} \sum_{J\subseteq[n]} \frac{(-1)^{m-|I|}(-1)^{n-|J|}}{(m-k)!(n-l)!} \sum_{s=1}^r c_s[\![\underline{T_s}]\!] \left(|\emptyset\rangle_{\sum_{i\in I} \alpha_i}, |\emptyset\rangle_{\sum_{j\in J} \beta_j} \right) \\
&= \sum_{I\subseteq[m]} \sum_{J\subseteq[n]} \frac{(-1)^{m-|I|}(-1)^{n-|J|}}{(m-k)!(n-l)!} \cdot 0 \\
&= 0,
\end{aligned}$$

where we again define $\alpha_i = \gamma$ for $i > k$ and $\beta_j = \delta$ for $j > l$. □

Suppose that A is a formula with $\dim[\![A]\!] = n < \infty$. By the above lemma, the existence of distinct binary integers with linearly dependent denotations reduces to the task of finding a non-zero noncommutative polynomial $t_n(x, y)$ such that $t_n(\alpha, \beta) = 0$ for all $n \times n$ matrices $\alpha, \beta \in [\![A \multimap A]\!]$. To describe such a polynomial, we will require the following theorem.

Theorem 3.22. (Amitsur-Levitzki Theorem.) For $n \in \mathbb{N}$, let $k\langle x_1, ..., x_n\rangle$ denote the ring of noncommutative polynomials in n variables, and let $s_n \in k\langle x_1, ..., x_n\rangle$ be the polynomial

$$s_n = \sum_{\sigma\in S_n} \mathrm{sgn}(\sigma) x_{\sigma(1)} \cdots x_{\sigma(n)}.$$

Then for all $\alpha_1, ..., \alpha_{2n} \in M_n(k)$, we have $s_{2n}(\alpha_1, ..., \alpha_{2n}) = 0$. Furthermore, $M_n(k)$ does not satisfy any polynomial identity of degree less than $2n$.

Proof. See [7, Theorem 3.1.4]. □

Corollary 3.23. For $n \in \mathbb{N}$, there exists a non-zero polynomial $t_n \in k\langle x, y\rangle$ such that for all $\alpha, \beta \in M_n(k)$ we have $t_n(\alpha, \beta) = 0$.

Proof. The polynomial $t_n(x,y) = s_{2n}(x, xy, ..., xy^{2n-1})$ is non-zero and satisfies the desired property. □

Proposition 3.24. For any formula A such that $\dim[\![A]\!] < \infty$, there exist distinct binary integers $T_1, ..., T_r \in \{0,1\}^*$ such that $[\![\underline{T_1}]\!], ..., [\![\underline{T_r}]\!]$ are linearly dependent in $[\![\mathbf{bint}_A]\!]$.

Proof. Let $n = \dim[\![A]\!]$, so that $[\![A \multimap A]\!] \cong M_n(k)$. For $1 \leq i \leq 2n$, let R_i be the binary integer $1^{i-1}0$. Note that for all $\alpha, \beta \in M_n(k)$ we have

$$\sum_{\sigma \in S_{2n}} \mathrm{sgn}(\sigma)[\![\underline{R_{\sigma(2n)} \cdots R_{\sigma(1)}}]\!](|\emptyset\rangle_\alpha, |\emptyset\rangle_\beta) = t_n(\alpha, \beta) = 0.$$

Hence $\sum_{\sigma \in S_{2n}} \mathrm{sgn}(\sigma)[\![\underline{R_{\sigma(2n)} \cdots R_{\sigma(1)}}]\!] = 0$ by Lemma 3.21. □

Remark 3.25. We will see later in Remark 3.31 that the hypothesis that $[\![A]\!]$ is finite-dimensional cannot be dropped.

Note that despite the above proposition, if we have a *particular* finite collection of binary integers $T_1, ..., T_r$ in mind it is always possible for A atomic to choose $[\![A]\!]$ such that $[\![\underline{T_1}]\!], ..., [\![\underline{T_r}]\!]$ are linearly independent in $[\![\mathbf{bint}_A]\!]$. To see this, let d denote the maximum length of the T_s, and note that a linear dependence relation between the $[\![\underline{T_s}]\!]$ gives rise to a polynomial identity for $M_n(k)$ of degree d, where $n = \dim[\![A]\!]$. By Amitsur-Levitzki we must therefore have $d \geq 2n$, so if we choose $\dim[\![A]\!] > d/2$ then $[\![\underline{T_1}]\!], ..., [\![\underline{T_r}]\!]$ must be linearly independent.

In addition, while linear independence does not always hold for an arbitrary collection of binary integers, it turns out that we do have linear independence for any *pair* of distinct binary integers, so long as $\dim[\![A]\!]$ is at least 2.

Proposition 3.26. Let A be a formula with $\dim[\![A]\!] \geq 2$. The function $\{0,1\}^* \to [\![\mathbf{bint}_A]\!]$ which maps S to $[\![\underline{S}]\!]$ is injective.

Proof. Let $n = \dim[\![A]\!]$. For simplicity of notation we suppose that n is finite, as the case where n is infinite is similar. Consider the subgroup G of $GL_n(k)$ generated by

$$\alpha = \begin{bmatrix} 1 & 2 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix} \text{ and } \beta = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 2 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix}.$$

It is well known that G is freely generated by α and β; see [5, §II.B]. Suppose that $[\![\underline{S}]\!] = [\![\underline{T}]\!]$, so that in particular we have $[\![\underline{S}]\!](|\emptyset\rangle_\alpha, |\emptyset\rangle_\beta) = [\![T]\!](|\emptyset\rangle_\alpha, |\emptyset\rangle_\beta)$. In other words, the composite obtained by substituting α for zero and β for one into the digits of S is equal to the corresponding composite for T. Since α and β generate a free group, it follows that $S = T$. □

Proposition 3.27. Let A be a formula with $\dim[\![A]\!] \geq 2$, and let $S, T \in \{0,1\}^*$ with $S \neq T$. The denotations $[\![\underline{S}]\!], [\![\underline{T}]\!]$ are linearly independent in $[\![\mathbf{bint}_A]\!]$.

Proof. Suppose we are given $S, T \in \{0,1\}^*$ such that $a[\![\underline{S}]\!] + b[\![\underline{T}]\!] = 0$ for some $a, b \neq 0$. With α, β as above, it follows that

$$[\![\underline{S}]\!](|\emptyset\rangle_\alpha, |\emptyset\rangle_\beta) \circ [\![\underline{T}]\!](|\emptyset\rangle_\alpha, |\emptyset\rangle_\beta)^{-1} = -\frac{b}{a} I$$

So $[\![\underline{S}]\!](|\emptyset\rangle_\alpha, |\emptyset\rangle_\beta) \circ [\![\underline{T}]\!](|\emptyset\rangle_\alpha, |\emptyset\rangle_\beta)^{-1}$ is in the center of G, which is trivial since G is free of rank 2, and hence $a = -b$. It follows that $[\![\underline{S}]\!] = [\![\underline{T}]\!]$ and therefore $S = T$ by the previous proposition. □

3.4 Iteration and copying

The ability to encode nontrivial functions into linear logic depends crucially on iteration. Given that the integer n is encoded in linear logic as the function $f \mapsto f^n$, there is an obvious way to achieve this.

Definition 3.28. Let π be a proof of $A \vdash A$, and ρ a proof of $\vdash A$. Define the proof $\underline{\mathrm{iter}}(\pi, \rho)$ [13, §7] as the following:

$$\dfrac{\dfrac{\dfrac{\dfrac{\pi}{\vdots}}{A \vdash A}}{\vdash A \multimap A}\multimap R}{!(A \multimap A)}\,\mathrm{prom} \qquad \dfrac{\dfrac{\dfrac{\rho}{\vdots}}{\vdash A} \qquad \dfrac{}{A \vdash A}}{A \multimap A \vdash A}\multimap L}{\mathbf{int}_A \vdash A}\multimap L$$

When cut against a Church numeral $\underline{n}_A$, the result is equivalent under cut elimination to n copies of π cut against each other and then against ρ.

Example 3.29. Recall the proof $\underline{\mathrm{mult}}_A$ from Example 3.11 which multiplies two integers. Cutting against the promotion a Church numeral $\underline{n}_A$, we obtain a proof $\underline{\mathrm{mult}}(n, -)$ of $\mathbf{int}_A \vdash \mathbf{int}_A$. By the above construction, we can easily encode exponentiation $(t \mapsto n^t)$ as the proof $\underline{\mathrm{iter}}(\underline{\mathrm{mult}}_A(n, -), \underline{1}_A)$; that is:

$$\dfrac{\dfrac{\dfrac{\dfrac{\underline{\mathrm{mult}}_A}{\vdots}}{\mathbf{int}_A \vdash \mathbf{int}_A}}{\vdash \mathbf{int}_A \multimap \mathbf{int}_A}\multimap R}{!(\mathbf{int}_A \multimap \mathbf{int}_A)}\,\mathrm{prom} \qquad \dfrac{\dfrac{\dfrac{\underline{1}_A}{\vdots}}{\vdash \mathbf{int}_A} \qquad \dfrac{}{\mathbf{int}_A \vdash \mathbf{int}_A}}{\mathbf{int}_A \multimap \mathbf{int}_A \vdash \mathbf{int}_A}\multimap L}{\mathbf{int}_{\mathbf{int}_A} \vdash \mathbf{int}_A}\multimap L$$

With the ability to iterate endomorphisms, the variety of functions one can encode in linear logic is greatly increased. We close this section with a somewhat pathological example, from [8, §5.3.2].

Example 3.30. There is a proof $\underline{\text{intcopy}}_A$ which allows one to copy an integer arbitrarily many times:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\underline{\text{add}}_A(-,1)}{\vdots}}{\mathbf{int}_A \vdash \mathbf{int}_A}\,\text{der}}{!\mathbf{int}_A \vdash \mathbf{int}_A}\,\text{prom}}{!\mathbf{int}_A \vdash !\mathbf{int}_A}}{\vdash !\mathbf{int}_A \multimap !\mathbf{int}_A}\multimap R}{\vdash !(!\mathbf{int}_A \multimap !\mathbf{int}_A)}\,\text{prom} \qquad \dfrac{\dfrac{\dfrac{\underline{0}_A}{\vdots}}{\vdash \mathbf{int}_A}}{\vdash !\mathbf{int}_A}\,\text{prom} \qquad \dfrac{}{!\mathbf{int}_A \vdash !\mathbf{int}_A}}{!\mathbf{int}_A \multimap !\mathbf{int}_A \vdash !\mathbf{int}_A}\multimap L}{\mathbf{int}_{!\mathbf{int}_A} \vdash !\mathbf{int}_A}\multimap L$$

Note that $\underline{\text{intcopy}}_A$ is the proof

$$\underline{\text{iter}}(!\underline{\text{add}}_A(-,1), !\underline{0}_A).$$

When cut against a Church numeral $\underline{n}_{!\mathbf{int}_A}$, the effect is to apply $!\underline{\text{add}}_A(-,1)$ a total of n times to $!\underline{0}_A$, yielding $!\underline{n}_A$: an infinite supply of the Church numeral $\underline{n}_A$.

Using a similar trick we can also copy binary integers, resulting in a proof $\underline{\text{bintcopy}}_A$ of $\mathbf{bint}_{!\mathbf{bint}_A} \vdash !\mathbf{bint}_A$. Writing B for $\mathbf{bint}_A$, this proof is:

$$\dfrac{\dfrac{\underline{\text{concat}}_A(-,0)}{\vdots}}{\dfrac{\dfrac{\dfrac{B \vdash B}{!B \vdash !B}\,\text{der, prom}}{\vdash !B \multimap !B}\multimap R}{\vdash !(!B \multimap !B)}\,\text{prom}} \qquad \dfrac{\dfrac{\dfrac{\dfrac{\underline{\text{concat}}_A(-,1)}{\vdots}}{\dfrac{B \vdash B}{!B \vdash !B}\,\text{der, prom}}\multimap R}{\vdash !(!B \multimap !B)}\,\text{prom} \qquad \dfrac{\dfrac{\dfrac{\underline{\emptyset}_A}{\vdots}}{\vdash B}}{\vdash !B}\,\text{prom} \qquad \dfrac{}{!B \vdash !B}}{!B \multimap !B \vdash !B}\multimap L}{\mathbf{int}_{!B} \vdash !B}\multimap L}{\mathbf{bint}_{!B} \vdash !B}\multimap L$$

If S is a binary integer, then $[\![\underline{S}_{!B}]\!]$ is a map which given inputs $\alpha, \beta : ![\![B]\!] \to ![\![B]\!]$, returns some composite of α and β. When cut against $\underline{S}_{!B}$, the map $[\![\underline{\text{bintcopy}}_A]\!]$ essentially substitutes $[\![!\underline{\text{concat}}_A(-,0)]\!]$ for α and $[\![!\underline{\text{concat}}_A(-,1)]\!]$ for β, and applies these to the empty list on A. The overall result is rebuilding the binary integer S starting from the empty list.

Remark 3.31. The proof $\underline{\text{bintcopy}}_A$ also provides us with a counter example for the infinite-dimensional analogue of Proposition 3.24. For any binary integer S, we have

$$[\![\underline{\text{bintcopy}}_A]\!]([\![\underline{S}_{!\mathbf{bint}_A}]\!]) = |\emptyset\rangle_{[\![\underline{S}_A]\!]} \in ![\![\mathbf{bint}_A]\!].$$

Since any collection of distinct vacuums is linearly independent and $[\![\underline{\text{bintcopy}}_A]\!]$ is linear, it follows that the set of all $[\![\underline{S}_{!\mathbf{bint}_A}]\!]$ is linearly independent in $[\![\mathbf{bint}_{!\mathbf{bint}_A}]\!]$.

The existence of proofs which modify the base type of integers motivates the idea of second order linear logic, which gives a method to quantify over *all* possible base types [8]. We gain an additional method of creating formulas: if A is a formula and x an atomic formula, then $\forall x.A$ is a formula. The following deduction rules are added [8, Definition 1.24]:

$$\frac{\Gamma \vdash A}{\Gamma \vdash \forall x.A}\,\forall R \qquad \frac{\Delta, A[B/x] \vdash C}{\Delta, \forall x.A \vdash C}\,\forall L$$

where x is not free in Γ, and $A[B/x]$ means the formula A with all free instances of x replaced by B. In this system, the type of integers is

$$\mathbf{int} = \forall x.\mathbf{int}_x = \forall x.!(x \multimap x) \multimap (x \multimap x).$$

With this viewpoint, Example 3.30 shows that the sequent $\mathbf{int} \vdash !\mathbf{int}$ is provable in second order linear logic. We view this as perhaps the strongest argument *against* the introduction of second order. Linear logic is often characterised as a 'resource sensitive' logic, in which the copying of formulas is made explicit by use of the exponential. The provability of $\mathbf{int} \vdash !\mathbf{int}$ undermines this characterisation, allowing one to hide the fact that a proof is copying by making the types in the subscript invisible. One can no longer tell if a proof is using its inputs in a linear or nonlinear way simply by inspecting its type.

From the point of view of the Sweedler semantics, there is also a sense in which the copying performed by $\underline{\text{intcopy}}_A$ is fake. The denotation of $\underline{\text{intcopy}}_A$ sends a Church numeral $[\![\underline{n}]\!]$ to the vacuum vector $|\emptyset\rangle_{[\![\underline{n}]\!]}$, and so we have

$$[\![\underline{\text{intcopy}}_A]\!](c[\![\underline{m}]\!] + d[\![\underline{n}]\!]) = c|\emptyset\rangle_{[\![\underline{m}]\!]} + d|\emptyset\rangle_{[\![\underline{n}]\!]} \neq |\emptyset\rangle_{c[\![\underline{m}]\!]+d[\![\underline{n}]\!]}.$$

This is similar to how if one chooses a basis $\{v_i\}_i$ of a vector space V, there is of course a linear map $\varphi : V \to V \otimes V$ which naively 'copies', sending v_i to $v_i \otimes v_i$ for all i. But this is superficial; while φ certainly copies the specific vectors v_i, it fails to copy linear combinations, as in general:

$$\varphi(cv_i + dv_j) = cv_i \otimes v_i + dv_j \otimes v_j \neq (cv_i + dv_j) \otimes (cv_i + dv_j).$$

If one is interested in differentiating proofs, then we have an additional intrinsic reason to oppose the use of second order and proofs like $\underline{\text{bintcopy}}_A$. Considering for example the proof $\underline{\text{repeat}}_A$ of $!\mathbf{bint}_A \vdash \mathbf{bint}_A$, one can cut against $\underline{\text{bintcopy}}_A$ to obtain a proof of $\mathbf{bint}_{!\mathbf{bint}_A} \vdash \mathbf{bint}_A$, which of course cannot be differentiated; the premise no longer possesses an exponential modality. Allowing the use of second order therefore causes any connection to differentiation to be missed. In particular, the encodings of Turing machines in [8, 19] are incompatible with differential linear logic due to the presence of second order at many intermediate steps, and it is this which motivates our development of a new encoding in Section 5.

4 Syntax of differential linear logic

So far our analysis of the differentiable structure of linear logic has been from a purely semantic perspective. It is natural to ask whether this structure can be understood *syntactically* by the addition of new deduction rules. Stated another way, the parts of the Sweedler semantics which are directly reflected by the syntax of linear logic are $\otimes, \oplus, \mathrm{Hom}$ and the coalgebraic structure of $!V$. But there is additional structure on $!V$ which is invisible from the point of view of the syntax. In this section we will define the syntax of differentiable linear logic, and show that it additionally reflects both *bialgebraic* structure of $!V$ and the interactions with $(k[t]/t^2)^*$.

Definition 4.1. The sequent calculus of **differential linear logic** [6, 23] consists of the usual deduction rules, together with three new rules, called coweakening, cocontraction and codereliction respectively.

$$\frac{\Gamma, !A \vdash B}{\Gamma, A \vdash B}\ \text{coder} \qquad \frac{\Gamma, !A \vdash B}{\Gamma \vdash B}\ \text{coweak} \qquad \frac{\Gamma, !A \vdash B}{\Gamma, !A, !A \vdash B}\ \text{coctr}$$

There are new cut elimination rules, described[3] in [6].

There are a number of equivalent formulations of the new rules; see [23] for another example. It is easily verified that the rules given there are equivalent to the rules of Definition 4.1. We have chosen this presentation since it makes clear the dual relationship between the rules of dereliction, weakening and contraction, and their corresponding 'co'-rules.

Remark 4.2. The new rules of Definition 4.1 may appear incongruous with our remarks on copying at the end of the previous section, since it is not difficult to devise a proof in differential linear logic of $A \vdash !A$. From a logician's perspective the new rules may seem particularly offensive, given that in differential linear logic, any sequent is provable!

$$\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{}{A \vdash A}}{!\Gamma, A \vdash A}\ \text{weak}}{\Gamma, A \vdash A}\ \text{coder}}{\Gamma, !A \vdash A}\ \text{der}}{\Gamma \vdash A}\ \text{coweak}$$

However from the perspective of computer science, the new rules are not so strange, as we are not primarily interested in the notion of *provability*. Rather, our interest lies in the computational content of the proof: its behaviour under cut elimination. In this light, a proof of $A \vdash !A$ using codereliction should not be thought of as copying A.

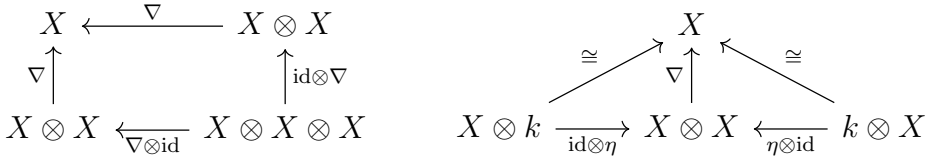
[3] The cut elimination procedure is actually defined in [6] in the language of *proof nets*, but there are obvious analogues of the reductions for the sequent calculus presentation.

Rather, it allows us to produce instances of $!A$ which have a *linear* dependence on A. This is perfectly reasonable, since the map

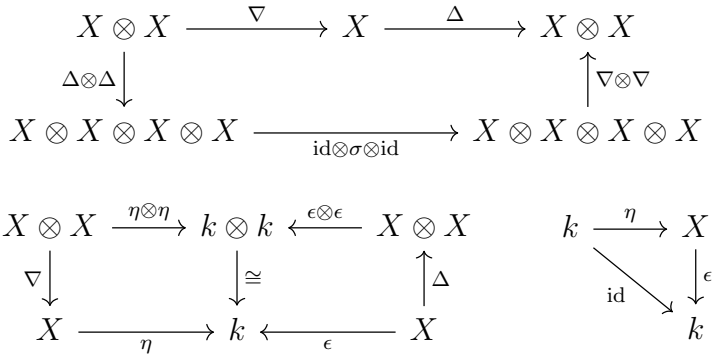
$$\alpha \mapsto |\alpha\rangle_0$$

produces an element of $!\llbracket A \rrbracket$ in a linear way from $\llbracket A \rrbracket$.

In order to understand these new rules semantically, it will be useful to know that $!V$ is not only a coalgebra, but a *bialgebra*. In brief, an (associative unital) **algebra** X is a k-vector space equipped with linear maps $\nabla : X \otimes X \to X$ and $\eta : k \to X$ called the **product** and **unit** respectively and such that



commute[4]. An algebra X which is also a coalgebra is called a **bialgebra** [22, §3.1] if the two structures are compatible, in the sense that the following diagrams also commute. Here σ denotes the map $a \otimes b \mapsto b \otimes a$.



Lemma 4.3. For a vector space V, the cofree coalgebra $!V$ is a bialgebra.

Proof. This is outlined in [22, §6.4]. Define the product $\nabla : !V \otimes !V \to !V$ as

$$\nabla(|\alpha_1, ..., \alpha_m\rangle_\gamma \otimes |\beta_1, ..., \beta_n\rangle_\delta) = |\alpha_1, ..., \alpha_m, \beta_1, ..., \beta_n\rangle_{\gamma+\delta},$$

and the unit $\eta : k \to !V$ as $\epsilon(1) = |\emptyset\rangle_0$. It is easily verified that these definitions together with the definitions of Section 2 equip $!V$ with the structure of a bialgebra. □

Given that the rules of contraction and weakening correspond semantically to the coproduct and counit of $!\llbracket A \rrbracket$, it is therefore natural that one should interpret cocontraction and coweakening as the product and unit respectively.

[4]This is the categorical dual of Definition 2.3.

$\dfrac{\begin{array}{c}\pi_1\\ \vdots\\ \Gamma, !A \vdash B\end{array}}{\Gamma, A \vdash B}\ \text{coder}$	$[\![\pi]\!](\gamma \otimes a) = [\![\pi_1]\!](\gamma \otimes \lvert a\rangle_0)$
$\dfrac{\begin{array}{c}\pi_1\\ \vdots\\ \Gamma, !A \vdash B\end{array}}{\Gamma \vdash B}\ \text{coweak}$	$[\![\pi]\!](\gamma) = [\![\pi_1]\!](\gamma \otimes \lvert\emptyset\rangle_0)$
$\dfrac{\begin{array}{c}\pi_1\\ \vdots\\ \Gamma, !A \vdash B\end{array}}{\Gamma, !A, !A \vdash B}\ \text{coctr}$	$[\![\pi]\!](\gamma \otimes \overline{a_1} \otimes \overline{a_1}) = [\![\pi_1]\!](\gamma \otimes \nabla(\overline{a_1} \otimes \overline{a_2}))$

Table 4.1: Denotations of the codereliction, coweakening, and cocontraction rules.

Remark 4.4. Note that the cocontraction rule formalises the notion of linear combinations of proofs in the syntax. Consider the following proof:

$$\dfrac{\dfrac{\dfrac{}{A \vdash A}}{!A \vdash A}\ \text{der}}{!A, !A \vdash A}\ \text{coctr}$$

which acts on vacuums by $\lvert\emptyset\rangle_\gamma \otimes \lvert\emptyset\rangle_\delta \mapsto \gamma + \delta$. If π_1, π_2 are proofs of $\Gamma \vdash A$, then we can form the sum of π_1 and π_2 via the following proof:

$$\dfrac{\dfrac{\dfrac{\begin{array}{c}\pi_1\\ \vdots\\ \Gamma \vdash A\end{array}}{!\Gamma \vdash !A}\ \text{der, prom} \qquad \dfrac{\begin{array}{c}\pi_2\\ \vdots\\ \Gamma \vdash A\end{array}}{!\Gamma \vdash !A}\ \text{der, prom}}{!\Gamma \vdash !A \otimes !A}\otimes R \qquad \dfrac{\dfrac{\dfrac{}{A \vdash A}}{!A \vdash A}\ \text{der}}{!A, !A \vdash A}\ \text{coctr}}{!A \otimes !A \vdash A}\otimes L}{!\Gamma \vdash A}\ \text{cut}$$

Linear combinations of proofs have a natural interpretation in terms of nondeterminism. For proofs $\pi_1, ..., \pi_n$ of $\vdash A$, the sum $[\![\pi_1]\!] + ... + [\![\pi_n]\!]$ should be thought of as a superposition of the given proofs from which one can nondeterministically 'choose' [17]. The connection between formal sums of proofs and nondeterminism is well studied; often this is done by explicitly adding a 'sum' rule [11]. We will further elaborate on the connection between cocontraction and nondeterminism in Section 5.3.

The reasoning behind the denotation of the codereliction rule is provided by the following definition, which reifies Definition 2.15 into the level of syntax.

Definition 4.5. Let π be a proof of $!A \vdash B$. Define the **derivative** of π, written $\partial\pi$, as the following proof:

$$\dfrac{\dfrac{\begin{array}{c}\pi\\ \vdots\\ !A \vdash B\end{array}}{!A, !A \vdash B}\,\mathrm{coctr}}{!A, A \vdash B}\,\mathrm{coder}$$

which has denotation

$$[\![\partial\pi]\!](|\alpha_1, ..., \alpha_n\rangle_\gamma \otimes a) = [\![\pi]\!] \circ \nabla(|\alpha_1, ..., \alpha_n\rangle_\gamma \otimes |a\rangle_0) = [\![\pi]\!]|a, \alpha_1, ..., \alpha_n\rangle_\gamma.$$

In particular, this means that

$$[\![\partial\pi]\!]_{\mathrm{nl}} = \partial[\![\pi]\!] : [\![A]\!] \otimes [\![A]\!] \to [\![B]\!]$$

in the sense of Definition 2.15.

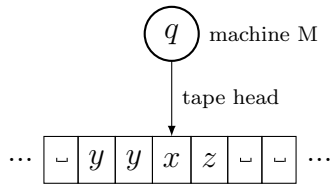


Figure 5.1: A Turing machine. The internal state is q, and the machine is currently reading the square marked x.

5 Turing machines

The examples of proofs we have given thus far correspond to relatively elementary functions such as addition and multiplication, so one could possibly form the impression that the computational power of linear logic is somewhat limited. Our next object of investigation aims to convince the reader that this is not the case.

We briefly recall the definition of a Turing machine; for a more comprehensive treatment, see [1, 21]. Informally speaking, a Turing machine is a computer which possesses a finite number of internal states, and a one dimensional 'tape' as memory. We adopt the convention that the tape is unbounded in both directions. The tape is divided into individual squares each of which contains some symbol from a fixed alphabet; at any instant only one square is being read by the 'tape head'. Depending on the symbol on this square and the current internal state, the machine will write a symbol to the square under the tape head, possibly change the internal state, and then move the tape head either left or right. Formally,

Definition 5.1. A **Turing machine** $M = (\Sigma, Q, \delta)$ is a tuple where Q is a finite set of states, Σ is a finite set of symbols called the **tape alphabet**, and

$$\delta : \Sigma \times Q \to \Sigma \times Q \times \{\text{left, right}\}$$

is a function, called the **transition function**.

The set Σ is assumed to contain some designated blank symbol '$\sqcup$' which is the only symbol which is allowed to occur infinitely often on the tape. Often one also designates a starting state, as well as a special accept state which terminates computation if reached.

If M is a Turing machine, a **Turing configuration** of M is a tuple $\langle S, T, q\rangle$, where $S, T \in \Sigma^*$ and $q \in Q$. This is interpreted as the instantaneous configuration of the Turing machine in the following way. The string S corresponds to the non-blank contents of the tape to the left of the tape head, including the symbol currently being scanned. The string T corresponds to a *reversed* copy of the contents of the tape to the right of the tape head, and q stores the current state of the machine. The reason for T being reversed is a matter of convenience, as we will see in the next section.

Despite the relative simplicity of the Turing machine model, it is generally accepted [21] that any function $f : \Sigma^* \to \Sigma^*$ which can naturally be regarded as computable is

able to be computed by a Turing machine, in the sense that there exists a machine M which if run on a tape which initially contains $x \in \Sigma^*$, then after sufficient time M will halt with just $f(w)$ printed on its tape.

The eventual goal of this section will be to present a method of encoding of Turing machines in linear logic. This is based on work by Girard in [9], which encodes Turing configurations via a variant of second order linear logic called *light* linear logic. The encoding does not use light linear logic in a crucial way, but requires second order in many intermediate steps, making it incompatible with differentiation. Our main contribution will be to develop an encoding which is able to be differentiated.

Remark 5.2. It is important to clarify that we are not claiming that linear logic is *Turing complete* - that is, able to compute the same class of functions as Turing machines. Our encoding provides us with a proof which simulates a Turing machine for any fixed number of computation steps, or if one is willing to use second order, a proof which can simulate a Turing machine for a variable number of steps. But this is not the same as computing *indefinitely* until a certain state is reached. Indeed, since linear logic is strongly normalising, it is necessarily unable to simulate any computation which fails to halt.

Definition 5.3. Fix a finite set of states $Q = \{0, ..., n-1\}$, and a tape alphabet[5] $\Sigma = \{0, 1\}$, with 0 being the blank symbol. The type of **Turing configurations** on A is:

$$\mathbf{Tur}_A = !\mathbf{bint}_A \otimes !\mathbf{bint}_A \otimes !_n\mathbf{bool}_A.$$

The configuration $\langle S, T, q\rangle$ is represented by the element

$$[\![\langle S, T, q\rangle]\!] = |\emptyset\rangle_{[\![\underline{S}_A]\!]} \otimes |\emptyset\rangle_{[\![\underline{T}_A]\!]} \otimes |\emptyset\rangle_{[\![\underline{q}_A]\!]} \in [\![\mathbf{Tur}_A]\!].$$

Our aim is to simulate a single transition step of a given Turing machine M as a proof ${}_\delta\underline{\mathrm{step}}_A$ of $\mathbf{Tur}_B \vdash \mathbf{Tur}_A$ for some formula B which depends on A, in the sense that if said proof is cut against a Turing configuration of M at time t, the result will be equivalent under cut elimination to the Turing configuration of M at time step $t+1$. This will be achieved in Theorem 5.12. Inspired by [9] and [24], our strategy will be as follows. Let $\langle S\sigma, T\tau, q\rangle$ be the (initial) configuration of the given Turing machine.

1. Decompose the binary integers $S\sigma$ and $T\tau$ to extract their final digits, giving S, T, σ and τ. Note that σ is the symbol currently under the head, and τ is the symbol immediately to its right.

2. Using the symbol σ together with the current state $q \in Q$, compute the new symbol σ', the new state q', and the direction to move d.

[5]It is straightforward to modify what follows to allow larger tape alphabets. At any rate, any Turing machine can be simulated by one whose tape alphabet is $\{0, 1\}$, so this really isn't as restrictive as it might seem [18].

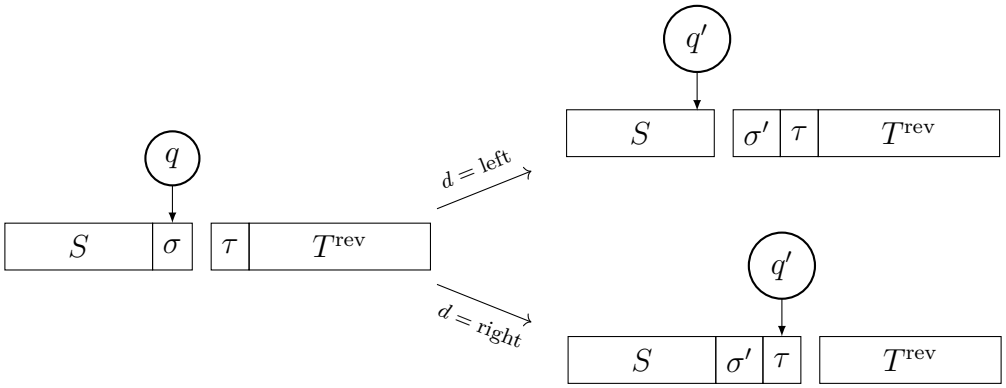


Figure 5.2: A single transition step of a Turing machine.

3. If $d = \text{right}$, append $\sigma'\tau$ to S. Otherwise, append $\tau\sigma'$ to T; remember that the binary integer T is the *reversal* of the contents of the tape to the right of the tape head. This is summarised in Figure 5.2.

5.1 The encoding

In order to feed the current symbol into the transition function, it is necessary to extract this digit from the binary integer which represents the tape. To do this we must decompose a binary integer $S' = S\sigma$ of length $l \geq 1$ into two parts S and σ, the former being a **bint** consisting of the first $l - 1$ digits (the *tail*) and the latter being a **bool** corresponding to the final digit (the *head*).

Throughout, let $A^n = A \,\&\, ... \,\&\, A$, where there are n copies of A.

Proposition 5.4. There exists a proof $\underline{\text{head}}_A$ of $\mathbf{bint}_{A^3} \vdash \mathbf{bool}_A$ which encodes the function $[\![\underline{S\sigma}_{A^3}]\!] \mapsto [\![\underline{\sigma}_A]\!]$.

Proof. The construction we will use is similar to that in [9, §2.5.3]. Let π_0, π_1 be the (easily constructed) proofs of $A^3 \vdash A^3$ whose denotations are $[\![\pi_0]\!](x, y, z) = (x, y, x)$ and $[\![\pi_1]\!](x, y, z) = (x, y, y)$ respectively. Similarly let ρ be the proof of $A^2 \vdash A^3$ with denotation $[\![\rho]\!](x, y) = (x, y, x)$. Define by $\underline{\text{head}}_A$ the following proof:

$$\dfrac{\dfrac{\dfrac{\dfrac{\pi_0}{A^3 \vdash A^3}}{\vdash A^3 \multimap A^3}\multimap R}{\vdash\, !(A^3 \multimap A^3)}\text{prom} \qquad \dfrac{\dfrac{\dfrac{\dfrac{\pi_1}{A^3 \vdash A^3}}{\vdash A^3 \multimap A^3}\multimap R}{\vdash\, !(A^3 \multimap A^3)}\text{prom} \qquad \dfrac{\dfrac{\dfrac{\rho}{A^2 \vdash A^3} \qquad \dfrac{\dfrac{}{A \vdash A}}{A^3 \vdash A}\&L_2}{A^3 \multimap A^3, A^2 \vdash A}\multimap L}{A^3 \multimap A^3 \vdash \mathbf{bool}_A}\multimap R}{\mathbf{int}_{A^3} \vdash \mathbf{bool}_A}\multimap L}{\mathbf{bint}_{A^3} \vdash \mathbf{bool}_A}\multimap L$$

where the rule $\&L_2$ introduces two new copies of A on the left, such that the original copy is at position 2 (that is, the third element of the triple).

We now show that $[\![\underline{\mathrm{head}}_A]\!]([\![\underline{S\sigma}_{A^3}]\!]) = [\![\underline{\sigma}_A]\!]$ as claimed. Recall that the denotation $[\![\underline{S\sigma}_{A^3}]\!]$ of a binary integer is a function which, given inputs α and β of type $A^3 \multimap A^3$ corresponding to the digits zero and one, returns some composite of α and β. The effect of the two leftmost branches of $\underline{\mathrm{head}}_A$ is to substitute $[\![\pi_0]\!]$ for α and $[\![\pi_1]\!]$ for β in this composite, giving a linear map $\varphi : [\![A^3]\!] \to [\![A^3]\!]$. The rightmost branch then computes $\mathrm{proj}_2 \circ \varphi \circ [\![\rho]\!] : [\![A^2]\!] \to [\![A]\!]$, giving a boolean.

In other words, $[\![\underline{\mathrm{head}}_A]\!]([\![\underline{S\sigma}_{A^3}]\!])$ is the element of $[\![\mathbf{bool}_A]\!]$ given by:

$$(a_0, a_1) \mapsto \mathrm{proj}_2 \circ \varphi \circ [\![\rho]\!](a_0, a_1) = \mathrm{proj}_2 \circ \varphi(a_0, a_1, a_0),$$

where φ is the composite of $[\![\pi_0]\!]$ and $[\![\pi_1]\!]$ as above. Note however that repeated applications of the functions $[\![\pi_i]\!]$ only serve to update the final digit of the triple, and thus only the final copy of $[\![\pi_i]\!]$ determines the output value. Hence the above simplifies to

$$\mathrm{proj}_2 \circ \varphi(a_0, a_1, a_0) = \mathrm{proj}_2 \circ [\![\pi_\sigma]\!](a_0, a_1, a_0) = \mathrm{proj}_2(a_0, a_1, a_\sigma) = a_\sigma.$$

Thus $[\![\underline{\mathrm{head}}_A]\!]([\![\underline{S\sigma}_{A^3}]\!]) = \mathrm{proj}_\sigma$, which is indeed the boolean corresponding to σ.

Lastly, we consider the special case when $S\sigma$ is the empty list. In this case, the computation gives:

$$\mathrm{proj}_2 [\![\rho]\!](a_0, a_1) = \mathrm{proj}_2(a_0, a_1, a_0) = a_0$$

which captures the fact that any symbols outside the working section of the tape are assumed to be the blank symbol, 0. □

Proposition 5.5. There exists a proof $\underline{\mathrm{tail}}_A$ of $\mathbf{bint}_{A^3} \vdash \mathbf{bint}_A$ which encodes the function $[\![\underline{S\sigma}_{A^3}]\!] \mapsto [\![\underline{S}_A]\!]$.

Remark. This could also be encoded as a proof of $\mathbf{bint}_{A^2} \vdash \mathbf{bint}_A$. However it will be much more convenient later if the sequents proven by $\underline{\mathrm{head}}_A$ and $\underline{\mathrm{tail}}_A$ have the same premise, since we will need to apply them both to two copies of the same binary integer.

Proof. This is largely based on the predecessor for $\mathbf{int}_A$; see Example 3.12. Define π to be the following proof:

$$\dfrac{\dfrac{\overline{A \vdash A} \quad \overline{A \vdash A} \quad \overline{A \vdash A}}{A \vdash A^3}\,\&R \qquad \dfrac{\dfrac{\overline{A \vdash A}}{A^3 \vdash A}\,\&L_0}{A, A^3 \multimap A^3 \vdash A}\,\multimap L}{A^3 \multimap A^3 \vdash A \multimap A}\,\multimap R$$

which has denotation:

$$[\![\pi]\!](\varphi)(a) = \mathrm{proj}_0(\varphi(a, a, a)).$$

Define ρ to be the following proof:

$$\dfrac{\dfrac{\dfrac{\overline{A \vdash A}}{A^3 \vdash A}\,\&L_2}{A^3, !(A \multimap A) \vdash A}\,\mathrm{weak} \qquad \dfrac{\dfrac{\overline{A \vdash A}}{A^3 \vdash A}\,\&L_2}{A^3, !(A \multimap A) \vdash A}\,\mathrm{weak} \qquad \dfrac{\dfrac{\dfrac{\overline{A \vdash A} \quad \overline{A \vdash A}}{A, A \multimap A \vdash A}\,\multimap L}{A^3, A \multimap A \vdash A}\,\&L_2}{A^3, !(A \multimap A) \vdash A}\,\mathrm{der}}{A^3, !(A \multimap A) \vdash A^3}\,\&R$$
$$\dfrac{}{!(A \multimap A) \vdash A^3 \multimap A^3}\,\multimap R$$

The denotation $[\![\rho]\!]$ is

$$[\![\rho]\!](|\alpha_1, ..., \alpha_s\rangle_\gamma)(a_0, a_1, a_2) = \begin{cases} (a_2, a_2, \gamma a_2) & s = 0 \\ (a_2, a_2, \alpha_1 a_2) & s = 1 \\ (a_2, a_2, 0) & s > 1. \end{cases}$$

Finally, define $\underline{\text{tail}}_A$ to be the following proof:

$$\dfrac{\dfrac{\dfrac{\dfrac{\rho}{\vdots}}{!(A \multimap A) \vdash A^3 \multimap A^3}}{!(A \multimap A) \vdash !(A^3 \multimap A^3)}\,\text{prom} \quad \dfrac{\dfrac{\dfrac{\dfrac{\rho}{\vdots}}{!(A \multimap A) \vdash A^3 \multimap A^3}}{!(A \multimap A) \vdash !(A^3 \multimap A^3)}\,\text{prom} \quad \dfrac{\dfrac{\pi}{\vdots}}{A^3 \multimap A^3 \vdash A \multimap A}}{!(A \multimap A), \mathbf{int}_{A^3} \vdash A \multimap A}\,\multimap L}{!(A \multimap A), !(A \multimap A), \mathbf{bint}_{A^3} \vdash A \multimap A}\,\multimap L}{\mathbf{bint}_{A^3} \vdash \mathbf{bint}_A}\,2\times\,\multimap R$$

Evaluated on the binary integer S, this gives a binary integer T which if fed two vacuum vectors $|\emptyset\rangle_\gamma$ and $|\emptyset\rangle_\delta$ (corresponding to the digits 0, 1) will return the composite $[\![A]\!] \to [\![A]\!]$ obtained by substituting $[\![\rho]\!]|\emptyset\rangle_\gamma$ and $[\![\rho]\!]|\emptyset\rangle_\delta$ for each copy of the digits 0 and 1 respectively in S, and then finally keeping the 0th projection by the left introduction of π.

As an example, suppose that the binary integer S is 0010. Then the corresponding linear map $[\![A]\!] \to [\![A]\!]$ is

$$a \mapsto \text{proj}_0(\tilde{\gamma} \circ \tilde{\delta} \circ \tilde{\gamma} \circ \tilde{\gamma}(a, a, a))$$

where $\tilde{\gamma} = [\![\rho]\!]|\emptyset\rangle_\gamma$, which is the morphism $(a_0, a_1, a_2) \mapsto (a_2, a_2, \gamma a_2)$, and similarly for $\tilde{\delta}$. Thus, we have:

$$\begin{aligned} \text{proj}_0(\tilde{\gamma} \circ \tilde{\delta} \circ \tilde{\gamma} \circ \tilde{\gamma}(a, a, a)) &= \text{proj}_0(\tilde{\gamma} \circ \tilde{\delta} \circ \tilde{\gamma}(a, a, \gamma(a))) \\ &= \text{proj}_0(\tilde{\gamma} \circ \tilde{\delta}(\gamma(a), \gamma(a), \gamma\gamma(a))) \\ &= \text{proj}_0(\tilde{\gamma}(\gamma\gamma(a), \gamma\gamma(a), \delta\gamma\gamma(a))) \\ &= \text{proj}_0(\delta\gamma\gamma(a), \delta\gamma\gamma(a), \gamma\delta\gamma\gamma(a)) \\ &= \delta\gamma\gamma(a). \end{aligned}$$

□

When fed through the decomposition steps, the base type of the binary integers changes from A^3 to A. We therefore also need to modify the base type of the n-boolean representing the state, in order to keep the base types compatible.

Lemma 5.6. There exists a proof ${}_n\underline{\text{booltype}}_A$ of ${}_n\mathbf{bool}_{A^3} \vdash {}_n\mathbf{bool}_A$ which converts an n-boolean on A^3 to the equivalent n-boolean on A; that is, it encodes $[\![\underline{i}_{A^3}]\!] \mapsto [\![\underline{i}_A]\!]$.

Proof. For $i \in \{0, ..., n-1\}$, let π_i be the proof of $A^n \vdash A^3$ whose denotation is $(a_0, ..., a_{n-1}) \mapsto (a_i, a_i, a_i)$. Define ${}_n\underline{\text{booltype}}_A$ as the proof:

$$\dfrac{\dfrac{\dfrac{\begin{matrix}\pi_0\\ \vdots\\ A^n \vdash A^3\end{matrix} \quad \cdots \quad \begin{matrix}\pi_{n-1}\\ \vdots\\ A^n \vdash A^3\end{matrix}}{A^n \vdash (A^3)^n}\,\&R \qquad \dfrac{\overline{A \vdash A}}{A^3 \vdash A}\,\&L_0}{A^n, {}_n\mathbf{bool}_{A^3} \vdash A}\,\multimap L}{{}_n\mathbf{bool}_{A^3} \vdash {}_n\mathbf{bool}_A}\,\multimap R$$

The denotation of ${}_n\underline{\text{booltype}}_A$ is the function

$$[\![{}_n\underline{\text{booltype}}_A]\!](\varphi)(a_0, ..., a_{n-1}) = \text{proj}_0 \circ \varphi((a_0, a_0, a_0), ..., (a_{n-1}, a_{n-1}, a_{n-1})),$$

and hence $[\![{}_n\underline{\text{booltype}}_A]\!]([\![\underline{i}_{A^3}]\!]) = [\![\underline{i}_A]\!]$. □

We now move to the task of encoding the transition function $\delta : \Sigma \times Q \to \Sigma \times Q \times \{\text{left, right}\}$ of a given Turing machine.

Lemma 5.7. Given any function $f : \{0, ..., n-1\} \to \{0, ..., m-1\}$, there exists a proof F of ${}_n\mathbf{bool}_A \vdash {}_m\mathbf{bool}_A$ which encodes f.

Proof. Let F be the following proof

$$\dfrac{\dfrac{\dfrac{\dfrac{\overline{A \vdash A}}{A^m \vdash A}\,\&L_{f(1)} \quad ... \quad \dfrac{\overline{A \vdash A}}{A^m \vdash A}\,\&L_{f(n)}}{A^m \vdash A^n}\,\&R \qquad \overline{A \vdash A}}{A^m, {}_n\mathbf{bool}_A \vdash A}\,\multimap L}{{}_n\mathbf{bool}_A \vdash {}_m\mathbf{bool}_A}\,\multimap R$$

The denotation of F is, for $\varphi \in [\![{}_n\mathbf{bool}_A]\!]$ and $(a_0, ..., a_{m-1}) \in [\![A^m]\!]$:

$$[\![F]\!](\varphi)(a_0, ..., a_{m-1}) = \varphi(a_{f(0)}, ..., a_{f(n-1)}).$$

In particular, this means that $[\![F]\!]([\![\underline{i}_A]\!])(a_0, ..., a_{m-1}) = \text{proj}_i(a_{f(0)}, ..., a_{f(n-1)}) = a_{f(i)}$, and hence $[\![F]\!]([\![\underline{i}_A]\!]) = \text{proj}_{f(i)} = [\![\underline{f(i)}_A]\!]$ as desired. □

Proposition 5.8. Given a transition function $\delta : \Sigma \times Q \to \Sigma \times Q \times \{\text{left, right}\}$, write δ_i for the component $\text{proj}_i \circ \delta$ ($i \in \{0, 1, 2\}$). Then there exists proofs

$$\begin{aligned}
&{}^0_\delta\underline{\text{trans}}_A : \mathbf{bool}_A, {}_n\mathbf{bool}_A \vdash \mathbf{bool}_A\\
&{}^1_\delta\underline{\text{trans}}_A : \mathbf{bool}_A, {}_n\mathbf{bool}_A \vdash {}_n\mathbf{bool}_A\\
&{}^2_\delta\underline{\text{trans}}_A : \mathbf{bool}_A, {}_n\mathbf{bool}_A \vdash \mathbf{bool}_A
\end{aligned}$$

which encode δ_i, for $i = 0, 1, 2$. For δ_2, we are using the convention that left $= 0$ and right $= 1$.

Proof. Given $i \in \{0, 1, 2\}$ and and $j \in \Sigma = \{0, 1\}$, let $\Delta_{i,j}$ be the proof obtained from Lemma 5.7 corresponding to the function $\delta_i(j, -)$, omitting the final $\multimap R$ rule. Define ${}^i_\delta\underline{\text{trans}}_A$ as the following proof, where $m = n$ if $i = 1$ and $m = 2$ otherwise:

$$\dfrac{\dfrac{\dfrac{\begin{matrix}\Delta_{i,0}\\ \vdots\end{matrix}}{A^m, {}_n\mathbf{bool}_A \vdash A} \quad \dfrac{\begin{matrix}\Delta_{i,1}\\ \vdots\end{matrix}}{A^m, {}_n\mathbf{bool}_A \vdash A}}{A^m, {}_n\mathbf{bool}_A \vdash A^2}\,\&R \qquad \dfrac{}{A \vdash A}}{A^m, \mathbf{bool}_A, {}_n\mathbf{bool}_A \vdash A}\,\multimap L}{\mathbf{bool}_A, {}_n\mathbf{bool}_A \vdash {}_m\mathbf{bool}_A}\,\multimap L$$

Then $[\![{}^i_\delta \underline{\mathrm{trans}}_A]\!]$ is the function

$$[\![{}^i_\delta \underline{\mathrm{trans}}_A]\!](\psi \otimes \varphi)(a_0, ..., a_{m-1}) = \psi(\varphi(a_{\delta_i(0,0)}, ..., a_{\delta_i(0,n-1)}), \varphi(a_{\delta_i(1,0)}, ..., a_{\delta_i(1,n-1)})),$$

and thus we have $[\![{}^i_\delta \underline{\mathrm{trans}}_A]\!]([\![\underline{\sigma}]\!] \otimes [\![\underline{q}]\!]) = \mathrm{proj}_{\delta_i(\sigma,q)} = [\![\underline{\delta_i(\sigma, q)}]\!]$. □

Once the new state, symbol and direction have been computed, our remaining task is to recombine the symbols with the binary integers representing the tape.

Lemma 5.9. Let $W_{00}, W_{01}, W_{10}, W_{11}$ be fixed binary integers, possibly empty. There exists a proof $\pi(W_{00}, W_{01}, W_{10}, W_{11})$ of $\mathbf{bint}_A, \mathbf{bool}_A, \mathbf{bool}_A \vdash \mathbf{bint}_A$ which encodes the function

$$(S, \sigma, \tau) \mapsto SW_{\sigma\tau}.$$

Proof. Let $E = A \multimap A$. We give a proof corresponding to the simpler function $(S, \sigma) \mapsto SW_\sigma$, where W_0, W_1 are fixed binary sequences:

$$\dfrac{\dfrac{\dfrac{\begin{matrix}\underline{\mathrm{concat}}_A(-, W_0)\\ \vdots\end{matrix}}{\mathbf{bint}_A, !E, !E, A \vdash A} \quad \dfrac{\begin{matrix}\underline{\mathrm{concat}}_A(-, W_1)\\ \vdots\end{matrix}}{\mathbf{bint}_A, !E, !E, A \vdash A}}{\mathbf{bint}_A, !E, !E, A \vdash A\&A}\,\&R \qquad \dfrac{}{A \vdash A}}{\mathbf{bint}_A, \mathbf{bool}_A, !E, !E, A \vdash A}\,\multimap L}{\mathbf{bint}_A, \mathbf{bool}_A \vdash \mathbf{bint}_A}\,3\times \multimap R$$

where $\underline{\mathrm{concat}}_A$ is as given in Example 3.16. The required proof $\pi(W_{00}, W_{01}, W_{10}, W_{11})$ is an easy extension of this, involving two instances of the $\&R$ and $\multimap L$ rules rather than one. □

Proposition 5.10. There exist proofs ${}^0\underline{\mathrm{recomb}}_A$ and ${}^1\underline{\mathrm{recomb}}_A$ of

$$\mathbf{bint}_A, 3\,\mathbf{bool}_A \vdash \mathbf{bint}_A$$

which encode the functions

$$(S, \tau, \sigma, d) \mapsto \begin{cases} S & \text{if } d = 0 \text{ (left)} \\ S\sigma\tau & \text{if } d = 1 \text{ (right)} \end{cases} \quad \text{and} \quad (T, \tau, \sigma, d) \mapsto \begin{cases} T\tau\sigma & \text{if } d = 0 \text{ (left)} \\ T & \text{if } d = 1 \text{ (right)} \end{cases}$$

respectively.

Proof. Define $\pi(-, -, -, -)$ as described in Lemma 5.9, omitting the final $\multimap R$ rules. The desired proof ${}^0\underline{\mathrm{recomb}}_A$ is:

$$\dfrac{\dfrac{\dfrac{\pi(\emptyset,\emptyset,\emptyset,\emptyset)}{\vdots}\quad\dfrac{\pi(00,10,01,11)}{\vdots}}{\dfrac{\mathbf{bint}_A, 2\,\mathbf{bool}_A, !E, !E, A \vdash A \qquad \mathbf{bint}_A, 2\,\mathbf{bool}_A, !E, !E, A \vdash A}{\mathbf{bint}_A, 2\,\mathbf{bool}_A, !E, !E, A \vdash A \& A}\,\&R}\qquad \dfrac{}{A \vdash A}}{\dfrac{\mathbf{bint}_A, 3\,\mathbf{bool}_A, !E, !E, A \vdash A}{\mathbf{bint}_A, 3\,\mathbf{bool}_A \vdash \mathbf{bint}_A}\,3\times \multimap R}\,\multimap L$$

and ${}^1\underline{\mathrm{recomb}}_A$ is the same, with the leftmost branch replaced by $\pi(00,01,10,11)$ and the second branch replaced by $\pi(\emptyset,\emptyset,\emptyset,\emptyset)$. □

Proposition 5.11. There exist proofs

$$\begin{aligned}
{}_\delta\underline{\mathrm{left}}_A &: 3\,\mathbf{bint}_{A^3},\quad \mathbf{bint}_{A^3}, 2\,{}_n\mathbf{bool}_{A^3} \vdash \mathbf{bint}_A\\
{}_\delta\underline{\mathrm{right}}_A &: 2\,\mathbf{bint}_{A^3}, 2\,\mathbf{bint}_{A^3}, 2\,{}_n\mathbf{bool}_{A^3} \vdash \mathbf{bint}_A\\
{}_\delta\underline{\mathrm{state}}_A &:\quad \mathbf{bint}_{A^3},\qquad\qquad {}_n\mathbf{bool}_{A^3} \vdash {}_n\mathbf{bool}_A
\end{aligned}$$

which, if fed the indicated number of copies of S, T and q corresponding to a Turing configuration, update the left part of the tape, the right part of the tape, and the state respectively.

Proof. We simply compose (using cuts) the proofs from Propositions 5.4 through 5.10; the exact sequence of cuts is given in Figures 5.3 - 5.5. The verification that the proofs perform the desired tasks is made clear through the following informal computations. Here $\langle S\sigma, T\tau, q\rangle$ is the configuration of the Turing machine at time t, and $\langle S', T', q'\rangle$ is its configuration at time $t+1$. In other words, we have $\delta(\sigma, q) = (\sigma', q', d)$, and

$$(S', T') = \begin{cases} (S, T\tau\sigma') & d = 0 \text{ (left)}\\ (S\sigma'\tau, T) & d = 1 \text{ (right)}.\end{cases}$$

$$\begin{aligned}
{}_\delta\underline{\mathrm{left}}_A \text{ is :}\quad & (S\sigma)^{\otimes 3} \otimes (T\tau) \otimes q^{\otimes 2} & \\
\longmapsto\ & S \otimes \sigma^{\otimes 2} \otimes \tau \otimes q^{\otimes 2} & (\underline{\mathrm{tail}}_A \otimes \underline{\mathrm{head}}_A^{\otimes 3} \otimes {}_n\underline{\mathrm{booltype}}_A^{\otimes 2})\\
\longmapsto\ & S \otimes \tau \otimes (\sigma \otimes q)^{\otimes 2} & (\text{exchange})\\
\longmapsto\ & S \otimes \tau \otimes \sigma' \otimes d & (\mathrm{id}^{\otimes 2} \otimes {}^0_\delta\underline{\mathrm{trans}}_A \otimes {}^2_\delta\underline{\mathrm{trans}}_A)\\
\longmapsto\ & S' & ({}^0\underline{\mathrm{recomb}}_A)
\end{aligned}$$

$$
\begin{aligned}
{}_\delta\underline{\mathrm{right}}_A \text{ is : } \quad & (S\sigma)^{\otimes 2} \otimes (T\tau)^{\otimes 2} \otimes q^{\otimes 2} && \\
\longmapsto\ & \sigma^{\otimes 2} \otimes \tau \otimes T \otimes q^{\otimes 2} && (\underline{\mathrm{head}}_A^{\otimes 3} \otimes \underline{\mathrm{tail}}_A \otimes {}_n\underline{\mathrm{booltype}}_A^{\otimes 2}) \\
\longmapsto\ & T \otimes \tau \otimes (\sigma \otimes q)^{\otimes 2} && (\text{exchange}) \\
\longmapsto\ & T \otimes \tau \otimes \sigma' \otimes d && (\mathrm{id}^{\otimes 2} \otimes {}^0_\delta\underline{\mathrm{trans}}_A \otimes {}^2_\delta\underline{\mathrm{trans}}_A) \\
\longmapsto\ & T' && ({}^1\underline{\mathrm{recomb}}_A)
\end{aligned}
$$

$$
\begin{aligned}
{}_\delta\underline{\mathrm{state}}_A \text{ is : } \quad & (S\sigma) \otimes q && \\
\longmapsto\ & \sigma \otimes q && (\underline{\mathrm{head}}_A \otimes {}_n\underline{\mathrm{booltype}}_A) \\
\longmapsto\ & q'. && ({}^1_\delta\underline{\mathrm{trans}}_A)
\end{aligned}
$$

□

Theorem 5.12. There exists a proof ${}_\delta\underline{\mathrm{step}}_A$ of $\mathbf{Tur}_{A^3} \vdash \mathbf{Tur}_A$ which encodes a single transition step of a given Turing machine.

Proof. The desired proof ${}_\delta\underline{\mathrm{step}}_A$ is given in Figure 5.6. □

By cutting the above construction against itself, we obtain:

Corollary 5.13. For each $p \geq 1$, there exists a proof ${}^p_\delta\underline{\mathrm{step}}_A$ of $\mathbf{Tur}_{A^{3^p}} \vdash \mathbf{Tur}_A$ which encodes p transition steps of a given Turing machine.

Note that this iteration must be performed 'by hand' for each p; we cannot iterate for a variable number of steps. By this we mean that it is not possible to devise a proof of $\mathbf{int}_B, \mathbf{Tur}_C \vdash \mathbf{Tur}_A$ (for suitable types B, C) which simulates a given Turing machine for n steps when cut against the Church numeral $\underline{n}_B$. The fundamental problem is that iteration as in Section 3.4 only allows iteration of *endomorphisms* $B \multimap B$, and so the fact that our base type changes in each iteration of ${}_\delta\underline{\mathrm{step}}_A$ makes this impossible.

If one is willing to use second order, then iteration in the above sense becomes possible via the following proof, where $\mathbf{Tur} = \forall A.\mathbf{Tur}_A$:

$$
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{{}_\delta\underline{\mathrm{step}}_A}{\vdots}}{\mathbf{Tur}_{A^3} \vdash \mathbf{Tur}_A}\,\forall L}{\mathbf{Tur} \vdash \mathbf{Tur}_A}\,\forall R}{\mathbf{Tur} \vdash \mathbf{Tur}}\,\multimap R}{\vdash \mathbf{Tur} \multimap \mathbf{Tur}}\,\mathrm{prom}}{\vdash\, !(\mathbf{Tur} \multimap \mathbf{Tur})} \qquad \dfrac{\dfrac{}{\mathbf{Tur} \vdash \mathbf{Tur}} \quad \dfrac{}{\mathbf{Tur} \vdash \mathbf{Tur}}}{\mathbf{Tur} \multimap \mathbf{Tur}, \mathbf{Tur} \vdash \mathbf{Tur}}\,\multimap L}{\mathbf{int}_{\mathbf{Tur}}, \mathbf{Tur} \vdash \mathbf{Tur}}\,\mathrm{prom}
$$

$$
\dfrac{\mathbf{int}_{\mathbf{Tur}}, \mathbf{Tur} \vdash \mathbf{Tur}}{\mathbf{int}, \mathbf{Tur} \vdash \mathbf{Tur}}\,\forall L
$$

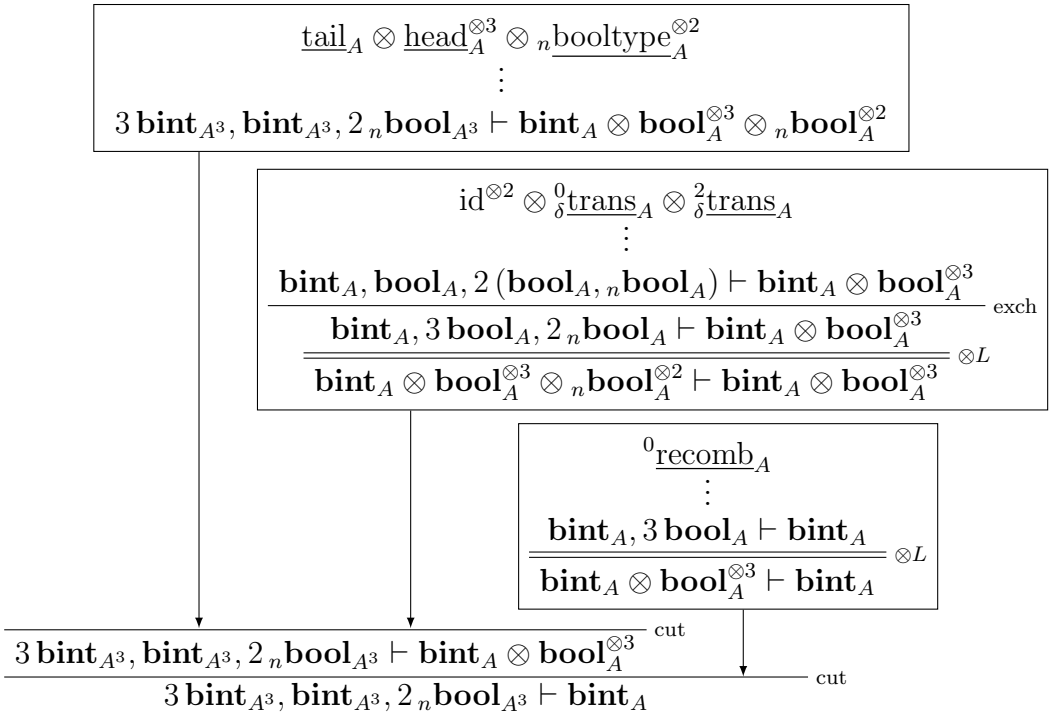


Figure 5.3: The proof ${}_\delta\underline{\text{left}}_A$.

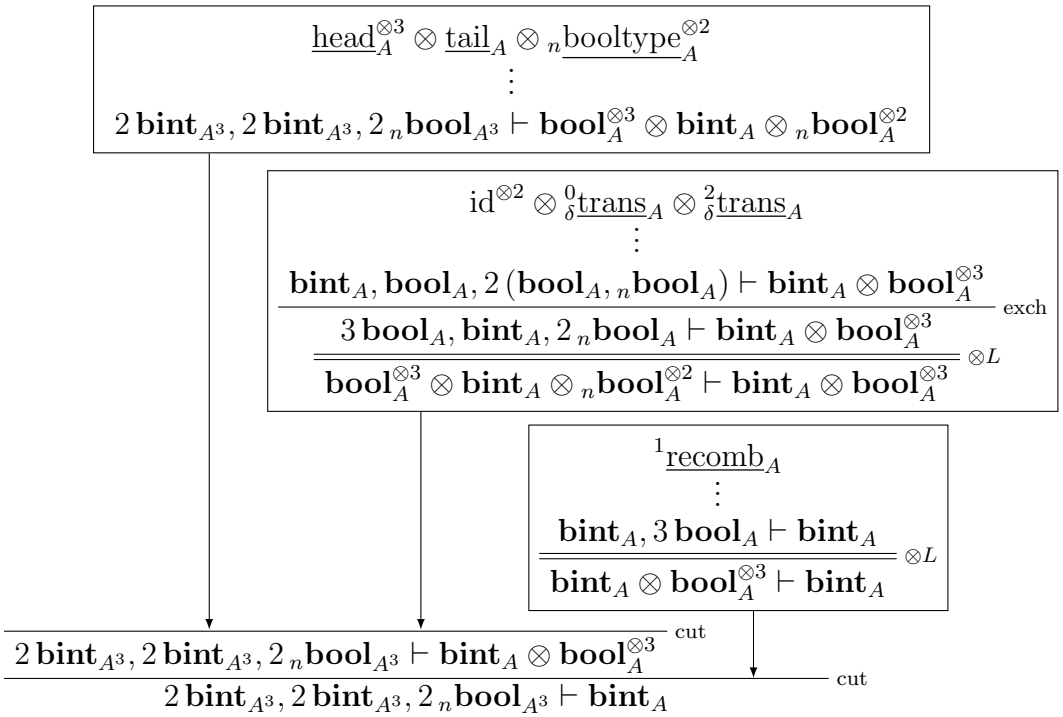


Figure 5.4: The proof ${}_\delta\underline{\text{right}}_A$.

$$\dfrac{\dfrac{\underline{\mathrm{tail}}_A \otimes {}_n\underline{\mathrm{booltype}}_A}{\vdots}\quad \dfrac{\dfrac{{}^1_\delta\underline{\mathrm{trans}}_A}{\vdots}}{\mathbf{bool}_A, {}_n\mathbf{bool}_A \vdash {}_n\mathbf{bool}_A}}{\mathbf{bint}_{A^3}, {}_n\mathbf{bool}_{A^3} \vdash \mathbf{bool}_A \otimes {}_n\mathbf{bool}_A \qquad \mathbf{bool}_A \otimes {}_n\mathbf{bool}_A \vdash {}_n\mathbf{bool}_A}\otimes L}{\mathbf{bint}_{A^3}, {}_n\mathbf{bool}_{A^3} \vdash {}_n\mathbf{bool}_A}\,\mathrm{cut}$$

Figure 5.5: The proof ${}_\delta\underline{\mathrm{state}}_A$.

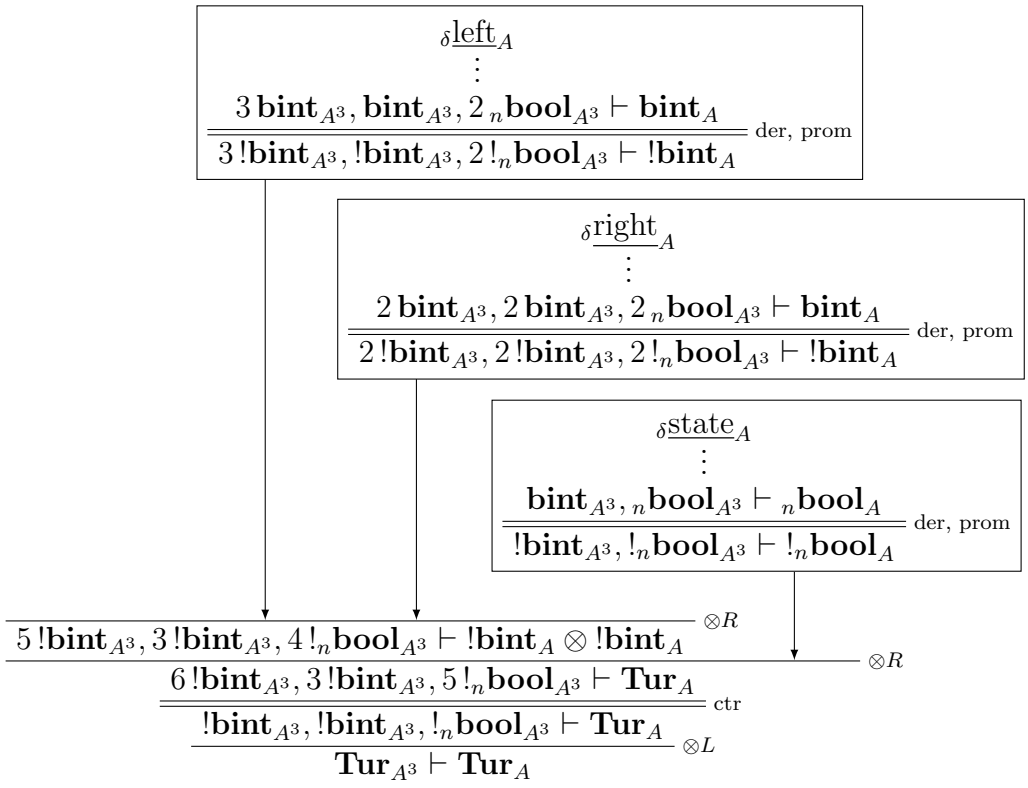


Figure 5.6: The proof ${}_\delta\underline{\mathrm{step}}_A$.

5.2 Copying

One aspect in which our approach differs from that given by Girard in [9] is that he presents the type of Turing configurations without exponential modalities, as $\mathbf{bint} \otimes \mathbf{bint} \otimes {}_n\mathbf{bool}$.[6] Our choice is made in accordance with the comments made at the end of Section 3.4; the encoding requires copying some inputs, and our understanding of the philosophy of linear logic is that any copying should be made explicit. Nevertheless, if one does not mind the use of proofs such as $\underline{\text{bintcopy}}_A$, it is possible to remove the exponentials in the definition of $\mathbf{Tur}_A$. We will now elaborate on this construction.

Similarly to $\underline{\text{bintcopy}}_A$, one can easily devise a proof ${}_n\underline{\text{boolcopy}}_A$ of

$$ {}_n\mathbf{bool}_{!_n\mathbf{bool}_A} \vdash !_n\mathbf{bool}_A $$

which copies n-booleans. One would therefore like to precompose ${}^{p}_{\delta}\underline{\text{step}}_A$ with the proof

$$ \underline{\text{bintcopy}}_{A^{3^p}} \otimes \underline{\text{bintcopy}}_{A^{3^p}} \otimes {}_n\underline{\text{boolcopy}}_{A^{3^p}}, $$

but a problem immediately arises: the base types of the binary integers would no longer be the same as that of the n-boolean. Fortunately, the fix of this is fairly straightforward. Let B be the type $!\mathbf{bint}_A \,\&\, !_n\mathbf{bool}_A$. For any binary integer S and any n-boolean i, there is a proof $\pi_{S,i}$ of $\vdash B$ which prepares the tuple $(!\llbracket \underline{S}_A \rrbracket, !\llbracket \underline{i}_A \rrbracket)$:

$$ \dfrac{\dfrac{\dfrac{\underline{S}_A}{\vdots}}{\dfrac{\vdash \mathbf{bint}_A}{\vdash !\mathbf{bint}_A}\,\text{prom}} \qquad \dfrac{\dfrac{\underline{i}_A}{\vdots}}{\dfrac{\vdash {}_n\mathbf{bool}_A}{\vdash !_n\mathbf{bool}_A}\,\text{prom}}}{\vdash B}\,\&R $$

Informally speaking, the pair $(!\llbracket \underline{S}_A \rrbracket, !\llbracket \underline{i}_A \rrbracket)$ should be thought of as an infinite supply of copies of both the binary integer S and the n-boolean i. The idea we will use is to design variants of ${}_n\underline{\text{boolcopy}}_A$ and $\underline{\text{bintcopy}}_A$ using this tuple, and then discard the other element by an appropriate $\&L$ introduction rule.

Lemma 5.14. There exists a proof ${}_n\underline{\text{boolcopy}}^*_A$ of ${}_n\mathbf{bool}_B \vdash !_n\mathbf{bool}_A$ which allows an n-boolean to be copied arbitrarily many times, where $B = !\mathbf{bint}_A \,\&\, !_n\mathbf{bool}_A$.

Proof. Define ${}_n\underline{\text{boolcopy}}^*_A$ as the proof

$$ \dfrac{\dfrac{\dfrac{\dfrac{\pi_{\emptyset,0}}{\vdots}}{\vdash B} \quad \ldots \quad \dfrac{\dfrac{\pi_{\emptyset,n-1}}{\vdots}}{\vdash B}\,\&R}{\vdash B^n} \qquad \dfrac{\dfrac{}{!_n\mathbf{bool}_A \vdash !_n\mathbf{bool}_A}\,\&L_1}{B \vdash !_n\mathbf{bool}_A}}{{}_n\mathbf{bool}_B \vdash !_n\mathbf{bool}_A}\,\multimap L $$

The value of $\llbracket {}_n\underline{\text{boolcopy}}^*_A \rrbracket(\llbracket \underline{i}_B \rrbracket)$ corresponds to projecting onto the ith component of B^n, which is $(!\llbracket \underline{\emptyset}_A \rrbracket, !\llbracket \underline{i}_A \rrbracket)$. The $\&L_1$ rule on the rightmost branch then discards the $\llbracket !\underline{\emptyset}_A \rrbracket$, and hence we obtain an unlimited supply of copies of $\llbracket \underline{i}_A \rrbracket$ as desired. □

[6] Here **bint** is the second-order formula $\mathbf{bint} = \forall x.\mathbf{bint}_x$, and similarly for ${}_n\mathbf{bool}$.

Lemma 5.15. There exists a proof $\underline{\text{bintcopy}}^*_A$ of $\mathbf{bint}_B \vdash !\mathbf{bint}_A$ which copies a binary integer arbitrarily many times, where $B = !\mathbf{bint}_A \,\&\, !_n\mathbf{bool}_A$.

Proof. For $\sigma \in \{0,1\}$, let ρ_σ denote the following proof:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\underline{\text{concat}}_A(-,\sigma)}{\vdots}}{\mathbf{bint}_A \vdash \mathbf{bint}_A}\,\text{der}}{!\mathbf{bint}_A \vdash \mathbf{bint}_A}\,\text{prom}}{!\mathbf{bint}_A \vdash !\mathbf{bint}_A}\,\&L_0 \qquad \dfrac{\dfrac{}{!_n\mathbf{bool}_A \vdash !_n\mathbf{bool}_A}}{B \vdash !_n\mathbf{bool}_A}\,\&L_1}{B \vdash !\mathbf{bint}_A \,\&\, !_n\mathbf{bool}_A}\,\&R}{\dfrac{\dfrac{B \vdash B}{\vdash B \multimap B}\multimap R}{\vdash !(B \multimap B)}\,\text{prom}}$$

Then $[\![\rho_\sigma]\!]$ corresponds to an infinite supply of the function $[\![B]\!] \to [\![B]\!]$ given by

$$(![\![\underline{S}_A]\!], ![\![\underline{q}_A]\!]) \mapsto (![\![\underline{S\sigma}_A]\!], ![\![\underline{q}_A]\!]).$$

We define $\underline{\text{bintcopy}}^*_A$ as the following proof.

$$\dfrac{\dfrac{\rho_0}{\vdots}}{\vdash !(B \multimap B)} \qquad \dfrac{\dfrac{\dfrac{\rho_1}{\vdots}}{\vdash !(B \multimap B)} \qquad \dfrac{\dfrac{\dfrac{\pi_{\emptyset,0}}{\vdots}}{\vdash B} \qquad \dfrac{\dfrac{}{!\mathbf{bint}_{A^3} \vdash !\mathbf{bint}_{A^3}}}{B \vdash !\mathbf{bint}_{A^3}}\,\&L_0}{B \multimap B \vdash \mathbf{bint}_{A^3}}\multimap L}{\mathbf{int}_B \vdash !\mathbf{bint}_{A^3}}\multimap L}{\mathbf{bint}_B \vdash !\mathbf{bint}_{A^3}}\multimap L$$

We give a sketch of why this construction does in fact copy the input. Let S be some binary integer of type $\mathbf{bint}_B$. Then $[\![\underline{\text{bintcopy}}^*_A]\!]$ substitutes $[\![\rho_0]\!]$ and $[\![\rho_1]\!]$ for the digits 0 and 1 in S respectively, and applies these to the pair $(![\![\underline{\emptyset}_A]\!], ![\![\underline{0}_A]\!])$. This rebuilds S starting from the empty list, giving $(![\![\underline{S}_A]\!], ![\![\underline{0}_A]\!])$. Finally, as in Lemma 5.14 we project onto the desired component. □

By precomposing the proofs from Section 5.1 with the above proofs we obtain a proof which encodes a single step transition without exponentials:

$$\dfrac{\underline{\text{bintcopy}}^*_{A^3} \otimes \underline{\text{bintcopy}}^*_{A^3} \otimes {}_n\underline{\text{boolcopy}}^*_{A^3}}{\vdots}}{\mathbf{Tur}^*_C \vdash \mathbf{Tur}_{A^3}} \qquad \dfrac{\dfrac{{}_\delta\underline{\text{left}}_A \otimes {}_\delta\underline{\text{right}}_A \otimes {}_\delta\underline{\text{state}}_A}{\vdots}}{\dfrac{6\,\mathbf{bint}_{A^3}, 3\,\mathbf{bint}_{A^3}, 5\,{}_n\mathbf{bool}_{A^3} \vdash \mathbf{Tur}^*_A}{!\mathbf{bint}_{A^3}, !\mathbf{bint}_{A^3}, !_n\mathbf{bool}_{A^3} \vdash \mathbf{Tur}^*_A}\,\text{der, ctr}}{\mathbf{Tur}_{A^3} \vdash \mathbf{Tur}^*_A}\otimes L}\,\text{cut}$$

where $C = !\mathbf{bint}_{A^3} \,\&\, !_n\mathbf{bool}_{A^3}$ and $\mathbf{Tur}^*_A = \mathbf{bint}_A \otimes \mathbf{bint}_A \otimes {}_n\mathbf{bool}_A$.

5.3 Nondeterministic Turing machines

One variant of the standard Turing machine incorporates nondeterminism, allowing the machine to make choices at each time step [1, §2.1.2].

Definition 5.16. A **nondeterministic Turing machine** $M = (\Sigma, Q, \delta_0, \delta_1)$ is a variant of a Turing machine which has two transition functions

$$\delta_i : \Sigma \times Q \to \Sigma \times Q \times \{\text{left, right}\}$$

rather than only one.

At any step of the computation, M will choose to use either δ_0 or δ_1. The runtime behaviour of M can therefore be described by a binary integer, which specifies which δ_i was used at each time step. If q_0 is the starting state and q_a the accept state for M, then we say that M **accepts** a given input string S if there exists some sequence of choices at each time step which causes M to reach an accept state starting from $\langle S, \emptyset, q_0 \rangle$.

Given our discussion of nondeterminism in Section 4, it should come as no surprise that one can devise an encoding of nondeterministic Turing machines in differential linear logic. A configuration of a nondeterministic Turing machine can be thought of as a superposition of configurations, which can be represented as the following sum:

$$[\![\langle S_1, T_1, q_1 \rangle]\!] + ... + [\![\langle S_l, T_l, q_l \rangle]\!] \in [\![\mathbf{Tur}_A]\!]. \tag{$*$}$$

Theorem 5.17. There exists a proof ${}_\delta\underline{\text{step}}^{\text{ND}}_A$ of $!\mathbf{Tur}_{A^3} \vdash !\mathbf{Tur}_A$ which encodes a single transition step of a given non-deterministic Turing machine.

Proof. Let ${}_\delta\underline{\text{step}}^0_A, {}_\delta\underline{\text{step}}^1_A$ denote the proofs of $\mathbf{Tur}_{A^3} \vdash \mathbf{Tur}$ obtained from Theorem 5.12 using the transition functions δ_0, δ_1 respectively. Define ${}_\delta\underline{\text{step}}^{\text{ND}}_A$ as the proof

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{{}_\delta\underline{\text{step}}^0_A}{\vdots}}{\mathbf{Tur}_{A^3} \vdash \mathbf{Tur}_A}}{!\mathbf{Tur}_{A^3} \vdash !\mathbf{Tur}_A}\,\text{der, prom} \quad \dfrac{\dfrac{\dfrac{{}_\delta\underline{\text{step}}^1_A}{\vdots}}{\mathbf{Tur}_{A^3} \vdash \mathbf{Tur}_A}}{!\mathbf{Tur}_{A^3} \vdash !\mathbf{Tur}_A}\,\text{der, prom}}{!\mathbf{Tur}_{A^3}, !\mathbf{Tur}_{A^3} \vdash !\mathbf{Tur}_A \otimes !\mathbf{Tur}_A}\otimes R}{!\mathbf{Tur}_{A^3} \vdash !\mathbf{Tur}_A \otimes !\mathbf{Tur}_A}\,\text{ctr} \quad \dfrac{\dfrac{\dfrac{}{!\mathbf{Tur}_A \vdash !\mathbf{Tur}_A}}{!\mathbf{Tur}_A, !\mathbf{Tur}_A \vdash !\mathbf{Tur}_A}\,\text{coctr}}{!\mathbf{Tur}_A \otimes !\mathbf{Tur}_A \vdash !\mathbf{Tur}_A}\otimes L}{!\mathbf{Tur}_{A^3} \vdash !\mathbf{Tur}_A}\,\text{cut}$$

which has denotation

$$[\![{}_\delta\underline{\text{step}}^{\text{ND}}_A]\!] = \nabla \circ (![\![{}_\delta\underline{\text{step}}^0_A]\!] \otimes ![\![{}_\delta\underline{\text{step}}^1_A]\!]) \circ \Delta.$$

Let $\langle S_1, T_1, q_1 \rangle, ..., \langle S_l, T_l, q_l \rangle$ be Turing configurations. Using the formula for the cocontraction rule in Table 4.1 (see also Lemma 4.3) we compute that the image of the

vacuum at the superposition $(*)$ under $[\![_\delta \underline{\mathrm{step}}^{\mathrm{ND}}_A]\!]$ is:

$$\begin{aligned}
&[\![_\delta \underline{\mathrm{step}}^{\mathrm{ND}}_A]\!]|\emptyset\rangle_{\sum_{i=1}^{l}[\![\langle S_i,T_i,q_i\rangle]\!]} \\
&= \nabla \circ (![\![_\delta \underline{\mathrm{step}}^{0}_A]\!] \otimes ![\![_\delta \underline{\mathrm{step}}^{1}_A]\!]) \left(|\emptyset\rangle_{\sum_{i=1}^{l}[\![\langle S_i,T_i,q_i\rangle]\!]} \otimes |\emptyset\rangle_{\sum_{i=1}^{l}[\![\langle S_i,T_i,q_i\rangle]\!]} \right) \\
&= \nabla \left(|\emptyset\rangle_{[\![_\delta \underline{\mathrm{step}}^{0}_A]\!]\left(\sum_{i=1}^{l}[\![\langle S_i,T_i,q_i\rangle]\!]\right)} \otimes |\emptyset\rangle_{[\![_\delta \underline{\mathrm{step}}^{1}_A]\!]\left(\sum_{i=1}^{l}[\![\langle S_i,T_i,q_i\rangle]\!]\right)} \right) \\
&= \nabla \left(|\emptyset\rangle_{\sum_{i=1}^{l}[\![_\delta \underline{\mathrm{step}}^{0}_A]\!][\![\langle S_i,T_i,q_i\rangle]\!]} \otimes |\emptyset\rangle_{\sum_{i=1}^{l}[\![_\delta \underline{\mathrm{step}}^{1}_A]\!][\![\langle S_i,T_i,q_i\rangle]\!]} \right) \\
&= |\emptyset\rangle_{\sum_{i=1}^{l}[\![_\delta \underline{\mathrm{step}}^{0}_A]\!][\![\langle S_i,T_i,q_i\rangle]\!]+[\![_\delta \underline{\mathrm{step}}^{1}_A]\!][\![\langle S_i,T_i,q_i\rangle]\!]}
\end{aligned}$$

□

Remark 5.18. If X is an algebra with product ∇ and C a coalgebra with coproduct Δ, then $\mathrm{Hom}_k(C, A)$ can be equipped with the structure of an algebra via the *convolution product* [22, §4]. For $f, g \in \mathrm{Hom}_k(C, A)$, their product is defined as the composite

$$C \xrightarrow{\quad\Delta\quad} C \otimes C \xrightarrow{\quad f \otimes g\quad} A \otimes A \xrightarrow{\quad\nabla\quad} A.$$

The above theorem therefore realises $[\![_\delta \underline{\mathrm{step}}^{\mathrm{ND}}_A]\!]$ as the convolution product of $![\![_\delta \underline{\mathrm{step}}^{0}_A]\!]$ and $![\![_\delta \underline{\mathrm{step}}^{1}_A]\!]$, which makes sense since $[\![!\mathbf{Tur}_A]\!]$ is a bialgebra.

6 Polynomiality and Taylor series

One of the main uses of the exponential modality for the proofs we have studied has been to produce a number of copies of a formula A via contraction and then dereliction, as in the following.

$$\dfrac{\dfrac{\begin{array}{c}\pi\\ \vdots \\ n\,A \vdash B\end{array}}{n\,!A \vdash B}\; n\times \text{ der}}{!A \vdash B}\; n-1\times \text{ ctr} \qquad (\dagger)$$

Many of the proofs we have considered so far have copied their inputs in this way, including the Church numerals, binary integers, and proofs such as $\underline{\text{repeat}}_A$, and ${}_\delta\underline{\text{step}}_A$. In this section, we give a general discussion on the types of calculus one can do on such proofs. In particular, we will show that proofs of the form (†) are polynomial in a suitable sense, and that their coalgebraic derivatives agree with the usual notion of a derivative. As a consequence, proofs of this form admit a kind of Taylor expansion [17] in terms of their coalgebraic derivatives. We will then apply this to the specific example of the single step Turing transition function.

The motivating example is the proof $\underline{\text{repeat}}_A$ of Example 3.17:

$$\dfrac{\dfrac{\begin{array}{c}\underline{\text{concat}}_A\\ \vdots \\ \mathbf{bint}_A, \mathbf{bint}_A \vdash \mathbf{bint}_A\end{array}}{!\mathbf{bint}_A, !\mathbf{bint}_A \vdash \mathbf{bint}_A}\; 2\times \text{ der}}{!\mathbf{bint}_A \vdash \mathbf{bint}_A}\; \text{ctr}$$

If S, T are fixed binary sequences, we compute:

$$\begin{aligned}[\![\underline{\text{repeat}}_A]\!]_{\text{nl}}(a[\![\underline{S}]\!] + b[\![\underline{T}]\!]) &= [\![\underline{\text{concat}}_A]\!]((a[\![\underline{S}]\!] + b[\![\underline{T}]\!]) \otimes (a[\![\underline{S}]\!] + b[\![\underline{T}]\!])) \\ &= a^2[\![\underline{SS}]\!] + ab([\![\underline{ST}]\!] + [\![\underline{TS}]\!]) + b^2[\![\underline{TT}]\!]\end{aligned}$$

The coefficients of the binary sequences in the output are polynomial functions of a and b. If we compute the coalgebraic derivative (as in Definition 2.15) of $[\![\underline{\text{repeat}}_A]\!]$ at $a[\![\underline{S}]\!] + b[\![\underline{T}]\!]$ in the direction of $[\![\underline{S}]\!]$, we find that it agrees with the usual derivative applied to the coefficients of $[\![\underline{SS}]\!], [\![\underline{ST}]\!], [\![\underline{TS}]\!]$ and $[\![\underline{TT}]\!]$:

$$\begin{aligned}[\![\underline{\text{repeat}}_A]\!]\,|\,[\![\underline{S}]\!]\rangle_{a[\![\underline{S}]\!]+b[\![\underline{T}]\!]} &= [\![\underline{\text{concat}}_A]\!]((a[\![\underline{S}]\!] + b[\![\underline{T}]\!]) \otimes [\![\underline{S}]\!] + [\![\underline{S}]\!] \otimes (a[\![\underline{S}]\!] + b[\![\underline{T}]\!])) \\ &= 2a[\![\underline{SS}]\!] + b([\![\underline{ST}]\!] + [\![\underline{TS}]\!]) \\ &= \frac{\partial}{\partial a}[\![\underline{\text{repeat}}_A]\!]_{\text{nl}}(a[\![\underline{S}]\!] + b[\![\underline{T}]\!]).\end{aligned}$$

We wish to generalise this construction. To this end, let A and B be formulas, $n \geq 1$ and suppose that π is a proof of $n\,A \vdash B$. We write $\pi^\dagger$ for the proof (†) above.

Definition 6.1. For $\alpha = (\alpha_0, \alpha_1) \in [\![A]\!]^2$, define a function $F^\pi_\alpha : k \times k \to [\![B]\!]$ by:

$$F^\pi_\alpha(a, b) = [\![\pi^\dagger]\!]_{\mathrm{nl}}(a\alpha_0 + b\alpha_1).$$

If $S = s_1 s_2 \dots s_n \in \{0,1\}^n$ is a binary integer, write α_S for the tensor

$$\alpha_{s_1} \otimes \cdots \otimes \alpha_{s_n} \in [\![A]\!]^{\otimes n}.$$

With this notation, one can write the value of F^π_α as

$$\begin{aligned} F^\pi_\alpha(a, b) &= [\![\pi]\!] \circ d^{\otimes n} \circ \Delta^{n-1} |\emptyset\rangle_{a\alpha_0 + b\alpha_1} \\ &= [\![\pi]\!]((a\alpha_0 + b\alpha_1)^{\otimes n}) \\ &= \sum_{S \in \{0,1\}^n} a^{|S_0|} b^{|S_1|} [\![\pi]\!](\alpha_S), \end{aligned}$$

where $S_0 = \{i \mid s_i = 0\}$ and $S_1 = \{i \mid s_i = 1\}$.

Choose a basis $\{u_1, ..., u_r\}$ of the subspace $V^\pi_\alpha = \mathrm{span}\,\{[\![\pi]\!](\alpha_S) \mid S \in \{0,1\}^n\} \subseteq [\![B]\!]$. By construction, for all $a, b \in k$ we can write the value of $F^\pi_\alpha(a, b)$ in terms of this basis:

$$F^\pi_\alpha(a, b) = \sum_{k=1}^{r} P_k(a, b) u_k,$$

where the $P_k(a, b)$ are functions of a and b.

Lemma 6.2. The $P_k(a, b)$ are polynomials, and if one writes

$$\frac{\partial^i}{\partial a^i} \frac{\partial^j}{\partial b^j} F^\pi_\alpha(a, b) = \sum_{k=1}^{r} \frac{\partial^i}{\partial a^i} \frac{\partial^j}{\partial b^j} [P_k(a, b)] u_k,$$

then this expression is independent as a vector in $[\![B]\!]$ of the choice of basis $\{u_1, ..., u_r\}$.

Proof. First assume that $\{[\![\pi]\!](\alpha_S) \mid S \in \{0,1\}^n\}$ is linearly independent, and hence is a basis of V^π_α. The coefficients P_S corresponding to this basis are $P_S(a, b) = a^{|S_0|} b^{|S_1|}$, which are polynomials. If not all of the $[\![\pi]\!](\alpha_S)$ are linearly independent, then the $P_S(a, b)$ will instead be sums of the monomials $a^{|S_0|} b^{|S_1|}$, hence a polynomial. The result then follows for a general basis $\{u_1, ..., u_r\}$ since changing basis is a linear operation.

Now, let $\{v_1, ..., v_r\}$ be another basis of V^π_α, and let Λ be the transition matrix between these bases, so that

$$u_k = \Lambda_{k,1} v_1 + ... + \Lambda_{k,r} v_r$$

for $k = 1, ..., r$. We therefore have:

$$F^\pi_\alpha(a, b) = \sum_{k=1}^{r} P_k(a, b) u_k = \sum_{k=1}^{r} P_k(a, b) \left(\sum_{l=1}^{r} \Lambda_{k,l} v_l \right) = \sum_{l=1}^{r} \left(\sum_{k=1}^{r} \Lambda_{k,l} P_k(a, b) \right) v_l.$$

Since differentiation is linear, it follows that

$$\sum_{l=1}^{r} \frac{\partial^i}{\partial a^i}\frac{\partial^j}{\partial b^j}\left[\sum_{k=1}^{r} \Lambda_{k,l} P_k(a,b)\right] v_l = \sum_{k=1}^{r} \frac{\partial^i}{\partial a^i}\frac{\partial^j}{\partial b^j}[P_k(a,b)]\left(\sum_{l=1}^{r} \Lambda_{k,l} v_l\right)$$
$$= \sum_{k=1}^{r} \frac{\partial^i}{\partial a^i}\frac{\partial^j}{\partial b^j}[P_k(a,b)]u_k.$$

□

Proposition 6.3. Let $\alpha = (\alpha_0, \alpha_1) \in [\![A]\!]^2$ and $i, j \geq 0$. Then

$$\frac{\partial^i}{\partial a^i}\frac{\partial^j}{\partial b^j} F^{\pi}_{\alpha}(a,b) = [\![\pi^\dagger]\!]|\alpha_0^{\otimes i}, \alpha_1^{\otimes j}\rangle_{a\alpha_0 + b\alpha_1}. \tag{$*$}$$

Proof. Let $\mathrm{Inj}(X, Y)$ denote the set of injective functions $X \to Y$. By a similar computation to Proposition 3.5, the right hand side of $(*)$ is:

$$[\![\pi^\dagger]\!]|\alpha_0^{\otimes i}, \alpha_1^{\otimes j}\rangle_{a\alpha_0+b\alpha_1} = [\![\pi]\!] \circ d^{\otimes n} \circ \Delta^{n-1}(|\alpha_0^{\otimes i}, \alpha_1^{\otimes j}\rangle_{a\alpha_0+b\alpha_1})$$
$$= \sum_{f,g} [\![\pi]\!](\Gamma_1^{f,g} \otimes \cdots \otimes \Gamma_n^{f,g}),$$

where the sum is over all $f \in \mathrm{Inj}([i],[n]), g \in \mathrm{Inj}([j],[n])$ such that $\operatorname{im} f \cap \operatorname{im} g = \emptyset$ and

$$\Gamma_s^{f,g} = \begin{cases} \alpha_0 & s \in \operatorname{im} f \\ \alpha_1 & s \in \operatorname{im} g \\ a\alpha_0 + b\beta_1 & \text{otherwise.} \end{cases}$$

For a binary sequence $S \in \{0,1\}^n$, the only terms on the right hand side of $(*)$ which contribute to the coefficient of $[\![\pi]\!](\alpha_S)$ are those for which $\operatorname{im} f \subseteq S_0$ and $\operatorname{im} g \subseteq S_1$. The coefficient of $[\![\pi]\!](\alpha_S)$ is therefore:

$$|\operatorname{Inj}([i], S_0)||\operatorname{Inj}([j], S_1)|\, a^{|S_0|-i} b^{|S_1|-j} = \frac{|S_0|!\,|S_1|!\, a^{|S_0|-i} b^{|S_1|-j}}{(|S_0|-i)!(|S_1|-j)!},$$

and hence

$$\frac{\partial^i}{\partial a^i}\frac{\partial^j}{\partial b^j} F^{\pi}_{\alpha}(a,b) = \sum_{S \in \{0,1\}^n} \frac{|S_0|!|S_1|!\, a^{|S_0|-i} b^{|S_1|-j}}{(|S_0|-i)!(|S_1|-j)!} [\![\pi]\!](\alpha_S) = [\![\pi^\dagger]\!]|\alpha_0^{\otimes i}, \alpha_1^{\otimes j}\rangle_{a\alpha_0+b\alpha_1}.$$

□

As a consequence of Proposition 6.3, the proof $\pi^\dagger$ admits a Taylor expansion in the following sense.

Proposition 6.4. For all $a, b, x, y \in k$, we have

$$[\![\pi^\dagger]\!]|\emptyset\rangle_{x\alpha_0+y\alpha_1} = \sum_{i,j\geq 0} \frac{1}{i!j!}[\![\pi^\dagger]\!]|\alpha_0^{\otimes i}, \alpha_1^{\otimes j}\rangle_{a\alpha_0+b\alpha_1}(x-a)^i(y-b)^j.$$

Proof. Let $P_S(a,b) = a^{|S_0|}b^{|S_1|}$, which is the coefficient of $[\![\pi]\!](\alpha_S)$ in $F^\pi_\alpha(a,b)$. This is a polynomial function of a and b, and thus P_S admits a Taylor expansion around each point $(a,b) \in k^2$, as a formal identity in $k[a,b,x,y]$:

$$P_S(x,y) = \sum_{i,j\geq 0} \frac{1}{i!j!}\frac{\partial^i}{\partial a^i}\frac{\partial^j}{\partial b^j}[P_S(a,b)](x-a)^i(y-b)^j.$$

Hence we have

$$\begin{aligned}
[\![\pi^\dagger]\!]|\emptyset\rangle_{x\alpha_0+y\alpha_1} &= \sum_{S\in\{0,1\}^n} P_S(x,y)[\![\pi]\!](\alpha_S) \\
&= \sum_{S\in\{0,1\}^n}\sum_{i,j\geq 0} \frac{1}{i!j!}\frac{\partial^i}{\partial a^i}\frac{\partial^j}{\partial b^j}[P_S(a,b)](x-a)^i(y-b)^j[\![\pi]\!](\alpha_S) \\
&= \sum_{i,j\geq 0}\sum_{S\in\{0,1\}^n} \frac{1}{i!j!}\frac{\partial^i}{\partial a^i}\frac{\partial^j}{\partial b^j}[P_S(a,b)](x-a)^i(y-b)^j[\![\pi]\!](\alpha_S) \\
&= \sum_{i,j\geq 0}\sum_{S\in\{0,1\}^n} \frac{1}{i!j!}[\![\pi^\dagger]\!]|\alpha_0^{\otimes i}, \alpha_1^{\otimes j}\rangle_{a\alpha_0+b\alpha_1}(x-a)^i(y-b)^j
\end{aligned}$$

by Proposition 6.3. □

In the particular case where $x = b = 0$ and $y = a = 1$, this simplifies to

$$[\![\pi^\dagger]\!]|\emptyset\rangle_{\alpha_0} = \sum_{i,j\geq 0} \frac{(-1)^j}{i!j!}[\![\pi^\dagger]\!]|\alpha_0^{\otimes i}, \alpha_1^{\otimes j}\rangle_{\alpha_1}.$$

A remarkable special case is where $\alpha_1 = 0$, which gives

$$[\![\pi^\dagger]\!]|\emptyset\rangle_{\alpha_0} = \sum_{i\geq 0} \frac{1}{i!}[\![\pi^\dagger]\!]|\alpha_0^{\otimes i}\rangle_0.$$

Thus, for proofs of the form $\pi^\dagger$, the value of $[\![\pi^\dagger]\!]_{\mathrm{nl}}$ is determined by the values of $[\![\pi^\dagger]\!]$ on kets based at zero.

Example 6.5. Consider $\pi^\dagger = \underline{\text{repeat}}_A$ and $\alpha_0 = [\![\underline{S}]\!], \alpha_1 = [\![\underline{T}]\!]$. We compute:

$$\begin{aligned}
&\sum_{i,j\geq 0} \frac{(-1)^i}{i!j!}[\![\underline{\text{repeat}}_A]\!]|[\![\underline{S}]\!]^{\otimes i}, [\![\underline{T}]\!]^{\otimes j}\rangle_{[\![\underline{T}]\!]} \\
&= [\![\underline{\text{repeat}}_A]\!](|\emptyset\rangle_{[\![\underline{T}]\!]} - |[\![\underline{T}]\!]\rangle_{[\![\underline{T}]\!]} + |[\![\underline{T}]\!], [\![\underline{T}]\!]\rangle_{[\![\underline{T}]\!]} \\
&\qquad\qquad + |[\![\underline{S}]\!]\rangle_{[\![\underline{T}]\!]} - |[\![\underline{S}]\!], [\![\underline{T}]\!]\rangle_{[\![\underline{T}]\!]} + |[\![\underline{S}]\!], [\![\underline{S}]\!]\rangle_{[\![\underline{T}]\!]}) \\
&= [\![\underline{TT}]\!] - 2[\![\underline{TT}]\!] + [\![\underline{TT}]\!] + ([\![\underline{ST}]\!] + [\![\underline{TS}]\!]) - ([\![\underline{ST}]\!] + [\![\underline{TS}]\!]) + [\![\underline{SS}]\!] \\
&= [\![\underline{\text{repeat}}_A]\!]|\emptyset\rangle_{[\![\underline{S}]\!]}.
\end{aligned}$$

A generalisation of the limit calculation of Example 3.17 also applies to $\pi^\dagger$.

Proposition 6.6. When $k = \mathbb{C}$, we have

$$[\![\pi^\dagger]\!]|\alpha_1\rangle_{\alpha_0} = \lim_{h\to 0} \frac{[\![\pi^\dagger]\!]|\emptyset\rangle_{\alpha_0+h\alpha_1} - [\![\pi^\dagger]\!]|\emptyset\rangle_{\alpha_0}}{h},$$

where the limit is taken in the finite-dimensional span of all $[\![\pi]\!](\alpha_S)$, $S \in \{0,1\}^*$.

Proof. We compute:

$$\begin{aligned}\lim_{h\to 0} \frac{1}{h}\left([\![\pi^\dagger]\!]|\emptyset\rangle_{\alpha_0+h\alpha_1} - [\![\pi^\dagger]\!]|\emptyset\rangle_{\alpha_0}\right) &= \lim_{h\to 0}\frac{1}{h}\sum_{\substack{S\in\{0,1\}^n\\ S\neq 0^n}} h^{|S_1|}[\![\pi]\!](\alpha_S)\\ &= \sum_{i=1}^{n}[\![\pi]\!](\alpha_0^{\otimes i-1}\otimes\alpha_1\otimes\alpha_0^{\otimes n-i})\\ &= [\![\pi^\dagger]\!]|\alpha_1\rangle_{\alpha_0}.\end{aligned}$$

□

Remark 6.7. Note that we know of at least one example of a proof which is *not* of the form $\pi^\dagger$ for which a similar calculation applies, namely $\underline{\text{mult}}_A$ from Example 3.11. From the example, one can see that it is really the fact that $\underline{\text{mult}}_A$ depends in a polynomial way on its inputs that makes such limiting calculations possible. It is our belief that in fact all proofs are polynomial in an appropriate sense, but we have been unable to prove this assertion.

We devote the remainder of this section towards a discussion of the derivative of the single step transition function of a Turing machine. Our purpose is to give a simple example where the derivative of a proof has a clear computational meaning. With $\Sigma = \{0,1\}$ and $Q = \{0,...,n-1\}$, let $\delta : \Sigma \times Q \to \Sigma \times Q \times \{0,1\}$ be a fixed transition function. As above, write ${}_\delta\underline{\text{left}}^\dagger_A$ for the proof:

$$\dfrac{\dfrac{\dfrac{\begin{array}{c}{}_\delta\underline{\text{left}}_A\\ \vdots\\ 3\,\mathbf{bint}_{A^3}, \mathbf{bint}_{A^3}, 2\,{}_n\mathbf{bool}_{A^3} \vdash \mathbf{bint}_A\end{array}}{3\,!\mathbf{bint}_{A^3}, !\mathbf{bint}_{A^3}, 2\,!_n\mathbf{bool}_{A^3} \vdash \mathbf{bint}_A}\,\text{der}}{!\mathbf{bint}_{A^3}, !\mathbf{bint}_{A^3}, !_n\mathbf{bool}_{A^3} \vdash \mathbf{bint}_A}\,\text{ctr}}$$

and similarly for ${}_\delta\underline{\text{right}}^\dagger_A$ and ${}_\delta\underline{\text{state}}^\dagger_A$. By the above, one can write the single step transition of a Turing machines in terms of vacuums at polynomial functions of its inputs. Suppose that $S_0, S_1, T \in \Sigma^*$, $\sigma_0, \sigma_1, \tau \in \Sigma$, $q \in Q$, $a, b \in k$, and write $(\sigma_i', q_i', d_i) = \delta(\sigma_i, q)$ for $i = 0, 1$.

Proposition 6.8. The single step transition evaluates as:

$$[\![_\delta \underline{\text{step}}_A]\!](|\emptyset\rangle_{a[\![\underline{S_0\sigma_0}]\!]+b[\![\underline{S_1\sigma_1}]\!]} \otimes |\emptyset\rangle_{[\![\underline{T\tau}]\!]} \otimes |\emptyset\rangle_{[\![\underline{q}]\!]}) = |\emptyset\rangle_\alpha \otimes |\emptyset\rangle_\beta \otimes |\emptyset\rangle_\gamma,$$

with

$$\begin{aligned}
\alpha &= (a+b)(a\delta_{d_0=0} + b\delta_{d_1=0})(a[\![\underline{S_0}]\!] + b[\![\underline{S_1}]\!]) \\
&\quad + (a\delta_{d_0=1} + b\delta_{d_1=1})(a^2[\![\underline{S_0\sigma_0'\tau}]\!] + ab[\![\underline{S_0\sigma_1'\tau}]\!] + ab[\![\underline{S_1\sigma_0'\tau}]\!] + b^2[\![\underline{S_1\sigma_1'\tau}]\!]). \\
\beta &= (a+b)(a\delta_{d_0=1} + b\delta_{d_1=1})[\![\underline{T}]\!] + (a\delta_{d_0=0} + b\delta_{d_1=0})(a[\![\underline{T\tau\sigma_0'}]\!] + b[\![\underline{T\tau\sigma_1'}]\!]) \\
\gamma &= a[\![\underline{q_0'}]\!] + b[\![\underline{q_1'}]\!],
\end{aligned}$$

where δ is the Kronecker delta.

Proof. See Appendix A. □

Remark 6.9. While one is tempted to say that the formulas for α, β, γ are simply a nondeterministic sum, the full picture is more complicated. Many of the terms appear to have no obvious classical meaning. For instance

$$ab\delta_{d_1=0}[\![\underline{T\tau\sigma_0'}]\!]$$

has used σ_0 to produce the new state, but σ_1 to produce the direction to move!

It is natural to ask whether this variant of nondeterminism has any intrinsic computational meaning. Assume that $0 \leq a_i \leq 1$ and $\sum_i a_i = 1$, and interpret $a_1, ..., a_s$ as being a probability distribution over states $\alpha_1, ..., \alpha_s$. The essential difference between classical nondeterminism and the above seems to be that classical nondeterminism has its roots in copying *values*

$$\sum_i a_i |\emptyset\rangle_{\alpha_i} \xmapsto{\Delta} \sum_i a_i |\emptyset\rangle_{\alpha_i} \otimes |\emptyset\rangle_{\alpha_i},$$

in which the superposition of values simply refers to our lack of knowledge about the 'true' computational state. In contrast, the variant of nondeterminism in Proposition 6.8 corresponds to copying the distribution itself

$$|\emptyset\rangle_{\sum_i a_i\alpha_i} \xmapsto{\Delta} |\emptyset\rangle_{\sum_i a_i\alpha_i} \otimes |\emptyset\rangle_{\sum_i a_i\alpha_i},$$

in which the state of the machine is genuinely undetermined until the distribution is sampled from by a dereliction. We believe this form of nondeterminism deserves further study.

Let us now examine the computational content of Proposition 6.8 in the special case

where $\sigma = \sigma_0 = \sigma_1$. The polynomial for the right hand part of the tape is

$$\begin{aligned}
&[\![_\delta \underline{\mathrm{right}}^\dagger_A]\!](|\emptyset\rangle_{a[\![\underline{S_0\sigma}]\!]+b[\![\underline{S_1\sigma}]\!]} \otimes |\emptyset\rangle_{[\![\underline{T\tau}]\!]} \otimes |\emptyset\rangle_{[\![\underline{q}]\!]}) \\
&= [\![_\delta \underline{\mathrm{right}}_A]\!]((a[\![\underline{S_0\sigma}]\!] + b[\![\underline{S_1\sigma}]\!])^{\otimes 2} \otimes [\![\underline{T\tau}]\!]^{\otimes 2} \otimes [\![\underline{q}]\!]^{\otimes 2}) \\
&= a^2[\![_\delta \underline{\mathrm{right}}_A]\!]([\![\underline{S_0\sigma}]\!] \otimes [\![\underline{S_0\sigma}]\!] \otimes [\![\underline{T\tau}]\!]^{\otimes 2} \otimes [\![\underline{q}]\!]^{\otimes 2}) + \\
&\quad ab[\![_\delta \underline{\mathrm{right}}_A]\!]([\![\underline{S_0\sigma}]\!] \otimes [\![\underline{S_1\sigma}]\!] \otimes [\![\underline{T\tau}]\!]^{\otimes 2} \otimes [\![\underline{q}]\!]^{\otimes 2}) + \\
&\quad ba[\![_\delta \underline{\mathrm{right}}_A]\!]([\![\underline{S_1\sigma}]\!] \otimes [\![\underline{S_0\sigma}]\!] \otimes [\![\underline{T\tau}]\!]^{\otimes 2} \otimes [\![\underline{q}]\!]^{\otimes 2}) + \\
&\quad b^2[\![_\delta \underline{\mathrm{right}}_A]\!]([\![\underline{S_1\sigma}]\!] \otimes [\![\underline{S_1\sigma}]\!] \otimes [\![\underline{T\tau}]\!]^{\otimes 2} \otimes [\![\underline{q}]\!]^{\otimes 2}) \\
&= (a+b)^2(\delta_{d=0}[\![\underline{T\tau\sigma'}]\!] + \delta_{d=1}[\![\underline{T}]\!]).
\end{aligned}$$

Observe that the polynomial coefficient $(a+b)^2$ is symmetric in a and b. This is no accident, and in fact one can restate this fact in terms of the vanishing of a derivative as follows. Note that we have a canonical isomorphism of k-algebras

$$k[a,b] \xrightarrow{\cong} k[x,y]$$

via $x = a+b, y = a-b$, and moreover for $f \in k[a,b]$ we have $\frac{\partial}{\partial y} f = 0$ if and only if f is a polynomial in x. Note that

$$\frac{\partial f}{\partial y} = \frac{\partial f}{\partial a}\frac{\partial a}{\partial y} + \frac{\partial f}{\partial b}\frac{\partial b}{\partial y} = \frac{1}{2}\left(\frac{\partial f}{\partial x} - \frac{\partial f}{\partial y}\right).$$

So the fact that evaluation by ${}_\delta \underline{\mathrm{right}}_A$ gives a polynomial in $a+b$ is exactly captured by the vanishing of the following coalgebraic derivative.

Proposition 6.10. $[\![_\delta \underline{\mathrm{right}}^\dagger_A]\!](|[\![\underline{S_0\sigma}]\!] - [\![\underline{S_1\sigma}]\!]\rangle_{a[\![\underline{S_0\sigma}]\!]+b[\![\underline{S_1\sigma}]\!]} \otimes |\emptyset\rangle_{[\![\underline{T\tau}]\!]} \otimes |\emptyset\rangle_{[\![\underline{q}]\!]}) = 0.$

Proof. Simply note that applying $\underline{\mathrm{head}}_A$ to $[\![\underline{S_0\sigma}]\!] - [\![\underline{S_1\sigma}]\!]$ yields $[\![\underline{\sigma}]\!] - [\![\underline{\sigma}]\!] = 0$. □

This is precisely the type of computational content one would hope to be visible in the derivatives, namely the fact that the right part of the tape does not depend on S_0 versus S_1 after one time step. Moreover, this fact appears to be independent of the minor details of the encoding.

A Polynomiality of Turing machines

In this appendix, we compute the value of $[\![_\delta \underline{\mathrm{step}}_A]\!]$ on

$$\Psi = |\emptyset\rangle_{a[\![\underline{S_0\sigma_0}]\!]+b[\![\underline{S_1\sigma_1}]\!]} \otimes |\emptyset\rangle_{[\![\underline{T\tau}]\!]} \otimes |\emptyset\rangle_{[\![\underline{q}]\!]}.$$

Note that

$$[\![_\delta \underline{\mathrm{step}}_A]\!](\Psi) = |\emptyset\rangle_{[\![_\delta \underline{\mathrm{left}}^\dagger_A]\!](\Psi)} \otimes |\emptyset\rangle_{[\![_\delta \underline{\mathrm{right}}^\dagger_A]\!](\Psi)} \otimes |\emptyset\rangle_{[\![_\delta \underline{\mathrm{state}}^\dagger_A]\!](\Psi)},$$

and so the problem reduces to computing the values of each of $[\![_\delta \underline{\mathrm{left}}^\dagger_A]\!]$, $[\![_\delta \underline{\mathrm{right}}^\dagger_A]\!]$, and $[\![_\delta \underline{\mathrm{state}}^\dagger_A]\!]$ on Ψ. By similar computations to those in Proposition 5.11, we find that for all $i, j, k \in \{0, 1\}$ we have

- $[\![_\delta \underline{\mathrm{left}}_A]\!]([\![\underline{S_i\sigma_i}]\!] \otimes [\![\underline{S_j\sigma_j}]\!] \otimes [\![\underline{S_k\sigma_k}]\!] \otimes [\![\underline{T\tau}]\!] \otimes [\![\underline{q}]\!]^{\otimes 2}) = \delta_{d_k=0}[\![\underline{S_i}]\!] + \delta_{d_k=1}[\![\underline{S_i\sigma'_j\tau}]\!]$,
- $[\![_\delta \underline{\mathrm{right}}_A]\!]([\![\underline{S_j\sigma_j}]\!] \otimes [\![\underline{S_k\sigma_k}]\!] \otimes [\![\underline{T\tau}]\!]^{\otimes 2} \otimes [\![\underline{q}]\!]^{\otimes 2}) = \delta_{d_k=0}[\![\underline{T\tau\sigma'_j}]\!] + \delta_{d_k=1}[\![\underline{T}]\!]$,
- $[\![_\delta \underline{\mathrm{state}}_A]\!]([\![\underline{S_i\sigma_i}]\!][\![\underline{q}]\!]) = [\![\underline{q'_i}]\!]$,

where δ on the right is the Kronecker delta. Using this, we compute

$$\begin{aligned}
[\![_\delta \underline{\mathrm{left}}^\dagger_A]\!](\Psi) = \; & a^3[\![_\delta \underline{\mathrm{left}}_A]\!]([\![\underline{S_0\sigma_0}]\!] \otimes [\![\underline{S_0\sigma_0}]\!] \otimes [\![\underline{S_0\sigma_0}]\!] \otimes [\![\underline{T\tau}]\!] \otimes [\![\underline{q}]\!]^{\otimes 2}) \\
& + a^2b([\![_\delta \underline{\mathrm{left}}_A]\!]([\![\underline{S_0\sigma_0}]\!] \otimes [\![\underline{S_0\sigma_0}]\!] \otimes [\![\underline{S_1\sigma_1}]\!] \otimes [\![\underline{T\tau}]\!] \otimes [\![\underline{q}]\!]^{\otimes 2}) \\
& \quad + [\![_\delta \underline{\mathrm{left}}_A]\!]([\![\underline{S_0\sigma_0}]\!] \otimes [\![\underline{S_1\sigma_1}]\!] \otimes [\![\underline{S_0\sigma_0}]\!] \otimes [\![\underline{T\tau}]\!] \otimes [\![\underline{q}]\!]^{\otimes 2}) \\
& \quad + [\![_\delta \underline{\mathrm{left}}_A]\!]([\![\underline{S_1\sigma_1}]\!] \otimes [\![\underline{S_0\sigma_0}]\!] \otimes [\![\underline{S_0\sigma_0}]\!] \otimes [\![\underline{T\tau}]\!] \otimes [\![\underline{q}]\!]^{\otimes 2})) \\
& + ab^2([\![_\delta \underline{\mathrm{left}}_A]\!]([\![\underline{S_0\sigma_0}]\!] \otimes [\![\underline{S_1\sigma_1}]\!] \otimes [\![\underline{S_1\sigma_1}]\!] \otimes [\![\underline{T\tau}]\!] \otimes [\![\underline{q}]\!]^{\otimes 2}) \\
& \quad + [\![_\delta \underline{\mathrm{left}}_A]\!]([\![\underline{S_1\sigma_1}]\!] \otimes [\![\underline{S_1\sigma_1}]\!] \otimes [\![\underline{S_0\sigma_0}]\!] \otimes [\![\underline{T\tau}]\!] \otimes [\![\underline{q}]\!]^{\otimes 2}) \\
& \quad + [\![_\delta \underline{\mathrm{left}}_A]\!]([\![\underline{S_1\sigma_1}]\!] \otimes [\![\underline{S_0\sigma_0}]\!] \otimes [\![\underline{S_1\sigma_1}]\!] \otimes [\![\underline{T\tau}]\!] \otimes [\![\underline{q}]\!]^{\otimes 2})) \\
& + b^3[\![_\delta \underline{\mathrm{left}}_A]\!]([\![\underline{S_1\sigma_1}]\!] \otimes [\![\underline{S_1\sigma_1}]\!] \otimes [\![\underline{S_1\sigma_1}]\!] \otimes [\![\underline{T\tau}]\!] \otimes [\![\underline{q}]\!]^{\otimes 2}) \\
= \; & a^3(\delta_{d_0=0}[\![\underline{S_0}]\!] + \delta_{d_0=1}[\![\underline{S_0\sigma'_0\tau}]\!]) + a^2b(\delta_{d_1=0}[\![\underline{S_0}]\!] + \delta_{d_1=1}[\![\underline{S_0\sigma'_0\tau}]\!] \\
& + \delta_{d_0=0}[\![\underline{S_0}]\!] + \delta_{d_0=1}[\![\underline{S_0\sigma'_1\tau}]\!] + \delta_{d_0=0}[\![\underline{S_1}]\!] + \delta_{d_0=1}[\![\underline{S_1\sigma'_0\tau}]\!]) \\
& + ab^2(\delta_{d_1=0}[\![\underline{S_0}]\!] + \delta_{d_1=1}[\![\underline{S_0\sigma'_1\tau}]\!] + \delta_{d_0=0}[\![\underline{S_1}]\!] + \delta_{d_0=1}[\![\underline{S_1\sigma'_1\tau}]\!] \\
& + \delta_{d_1=0}[\![\underline{S_1}]\!] + \delta_{d_1=1}[\![\underline{S_1\sigma'_0\tau}]\!]) + b^3(\delta_{d_1=0}[\![\underline{S_1}]\!] + \delta_{d_1=1}[\![\underline{S_1\sigma'_1\tau}]\!]) \\
= \; & (a+b)(a\delta_{d_0=0} + b\delta_{d_1=0})(a[\![\underline{S_0}]\!] + b[\![\underline{S_1}]\!]) \\
& + (a\delta_{d_0=1} + b\delta_{d_1=1})(a^2[\![\underline{S_0\sigma'_0\tau}]\!] + ab[\![\underline{S_0\sigma'_1\tau}]\!] + ab[\![\underline{S_1\sigma'_0\tau}]\!] + b^2[\![\underline{S_1\sigma'_1\tau}]\!]).
\end{aligned}$$

$$
\begin{aligned}
[\![_\delta \underline{\mathrm{right}}^\dagger_A]\!](\Psi) = \;& a^2 [\![_\delta \underline{\mathrm{right}}_A]\!]([\![\underline{S_0\sigma_0}]\!] \otimes [\![\underline{S_0\sigma_0}]\!] \otimes [\![\underline{T\tau}]\!]^{\otimes 2} \otimes [\![\underline{q}]\!]^{\otimes 2}) \\
& + ab([\![_\delta \underline{\mathrm{right}}_A]\!]([\![\underline{S_0\sigma_0}]\!] \otimes [\![\underline{S_1\sigma_1}]\!] \otimes [\![\underline{T\tau}]\!]^{\otimes 2} \otimes [\![\underline{q}]\!]^{\otimes 2}) \\
& \quad + [\![_\delta \underline{\mathrm{right}}_A]\!]([\![\underline{S_1\sigma_1}]\!] \otimes [\![\underline{S_0\sigma_0}]\!] \otimes [\![\underline{T\tau}]\!]^{\otimes 2} \otimes [\![\underline{q}]\!]^{\otimes 2})) \\
& + b^2 [\![_\delta \underline{\mathrm{right}}_A]\!]([\![\underline{S_1\sigma_1}]\!] \otimes [\![\underline{S_1\sigma_1}]\!] \otimes [\![\underline{T\tau}]\!]^{\otimes 2} \otimes [\![\underline{q}]\!]^{\otimes 2}) \\
= \;& a^2(\delta_{d_0=0}[\![\underline{T\tau\sigma'_0}]\!] + \delta_{d_0=1}[\![\underline{T}]\!]) \\
& + ab(\delta_{d_1=0}[\![\underline{T\tau\sigma'_0}]\!] + \delta_{d_1=1}[\![\underline{T}]\!] + \delta_{d_0=0}[\![\underline{T\tau\sigma'_1}]\!] + \delta_{d_0=1}[\![\underline{T}]\!]) \\
& + b^2(\delta_{d_1=0}[\![\underline{T\tau\sigma'_1}]\!] + \delta_{d_1=1}[\![\underline{T}]\!]) \\
= \;& (a+b)(a\delta_{d_0=1} + b\delta_{d_1=1})[\![\underline{T}]\!] + (a\delta_{d_0=0} + b\delta_{d_1=0})(a[\![\underline{T\tau\sigma'_0}]\!] + b[\![\underline{T\tau\sigma'_1}]\!]). \\
[\![_\delta \underline{\mathrm{state}}^\dagger_A]\!](\Psi) = \;& a[\![_\delta \underline{\mathrm{state}}_A]\!]([\![\underline{S_0\sigma_0}]\!] \otimes [\![\underline{q}]\!]) + b[\![_\delta \underline{\mathrm{state}}_A]\!]([\![\underline{S_1\sigma_1}]\!] \otimes [\![\underline{q}]\!]) \\
= \;& a[\![\underline{q'_0}]\!] + b[\![\underline{q'_1}]\!].
\end{aligned}
$$

References

[1] S. Arora, B. Barak, *Computational complexity: a modern approach*, Cambridge University Press, New York, 2009.

[2] M. Barr, *Coalgebras over a commutative ring*, Journal of Algebra **32** (1974), 600–610.

[3] ———, **-Autonomous categories and linear logic*, Mathematical Structures in Computer Science **1** (1991), 159–178.

[4] A. Blass, *A game semantics for linear logic*, Annals of Pure and Applied Logic **56** (1992), 183–220.

[5] P. de la Harpe, *Topics in geometric group theory*, The University of Chicago Press, 2000.

[6] T. Ehrhard, *An introduction to differential linear logic: proof-nets, models and antiderivatives*, arXiv preprint [arXiv: 1606.01642] (2016).

[7] V. Drensky, E. Formanek, *Polynomial identity rings*, Birkhäuser Basel, 2004.

[8] J.-Y. Girard, *Linear logic*, Theoretical Computer Science **50** (1987), 1–101.

[9] ———, *Light linear logic*, Information and Computation **143** (1998), 175–204.

[10] D. Eisenbud, J. Harris, *The geometry of schemes*, Springer-Verlag, New York, 2000.

[11] F. Maurel, *Nondeterministic light logics and NP-time*, vol. 2701, 2003, pp. 241–255.

[12] P.A. Melliès, *Categorical semantics of linear logic*, in Interactive models of computation and program behaviour, vol. 27, 2009.

[13] D. Murfet, *Logic and linear algebra: an introduction*, arXiv preprint [arXiv: 1701.01285] (2014).

[14] ———, *On Sweedler's cofree cocommutative coalgebra*, J. Pure and Applied Algebra [arXiv: 1701.01285] **219** (2015), 5289–5304.

[15] J. Clift, D. Murfet, *Cofree coalgebras and differentiable linear logic*, arXiv preprint [arXiv: 1701.01285] (2017).

[16] P. Etingof, S. Gelaki, D. Nikshych, V. Ostrik, *Tensor categories*, vol. 205, American Mathematical Society, Providence, Rhode Island, 2015.

[17] T. Ehrhard, L. Regnier, *The differential lambda-calculus*, Theoretical Computer Science **309** (2003), 1–41.

[18] Y. Rogozhin, *Small universal Turing machines*, Theoretical Computer Science **168** (1996), 215–240.

[19] L. Roversi, *A P-time completeness proof for light logics*, in Proceedings of CSL, 1999, pp. 469–483.

[20] R. Blute, J. Cockett, R. Seely, *Differential categories*, Mathematical Structures in Computer Science **16** (2006), 1049–1083.

[21] M. Sipser, *Introduction to the theory of computation*, Thomson Course Technology, Boston, 2006.

[22] M. Sweedler, *Hopf algebras*, W.A. Benjamin, New York, 1969.

[23] R. Blute, T. Ehrhard, C. Tasson, *A convenient differential category*, arXiV preprint [arXiv:1006.3140] (2010).

[24] H. Mairson, K. Terui, *On the computational complexity of cut-elimination in linear logic*, in Proceedings of ICTCS 2003, vol. 2841, Springer, 2003, pp. 23–36.

[25] K. Terui, *Light logic and polynomial time computation*, Ph.D. thesis, Keio University, 2002.

Notation

Σ^*	The set of (finite) words over an alphabet Σ
$[n]$	The set $\{1, ..., n\}$
Δ	Coproduct of a coalgebra, $C \to C \otimes C$, 2.3
ϵ	Counit of a coalgebra, $C \to k$, 2.3
$G(C)$	Set of grouplike elements of a coalgebra C, 2.5
$P_\gamma(C)$	Set of primitive elements over γ of a coalgebra C, 2.5
$!V$	Cofree cocommutative coalgebra generated by V, 2.8
$\lvert\alpha_1, ..., \alpha_n\rangle_\gamma$	Element of $!V$: the equivalence class of $\alpha_1 \otimes ... \otimes \alpha_n$ in the copy of $\mathrm{Sym}(V)$ associated to γ, 2.10
$\lvert\emptyset\rangle_\gamma$	Vacuum vector (empty ket) at γ, 2.10
d	Dereliction map, $!V \to V$, 2.8
$\mathrm{prom}\,\varphi$	Promotion of $\varphi : !V \to W$ to a coalgebra morphism $\Phi : !V \to !W$, 2.8
$[\![\cdot]\!]$	Denotation of a formula or proof, 2.11, 2.12
$[\![\pi]\!]_{\mathrm{nl}}$	Nonlinear map induced by a proof π of $!A \vdash B$, 2.13
$\partial_\gamma [\![\pi]\!]$	Derivative of $[\![\pi]\!]$ at γ, 2.15
$\partial\pi$	Derivative of a proof π of $!A \vdash B$, 4.5
∇	Product of an algebra, $C \to C \otimes C$, 4.1
η	Unit of an algebra, $C \to k$, 4.1
$\langle S, T, q\rangle$	Configuration of a Turing machine, 5.1
δ	Transition function of a Turing machine, 5.1
$\pi^\dagger$	The proof of $!A \vdash B$ obtained from $\pi : n\,A \vdash B$ by appending n derelictions then $n-1$ contractions, 6.1
F^π_α	The nonlinear function $F^\pi_\alpha(a, b) = [\![\pi^\dagger]\!]_{\mathrm{nl}}(a\alpha_0 + b\alpha_1)$, $k^2 \to [\![B]\!]$, 6.1
α_S	For $S \in \{0,1\}^n$, the tensor $\alpha_{s_1} \otimes ... \otimes \alpha_{s_n}$, 6.1
$n\,A$	n copies of a formula A

E	The type $A \multimap A$
$\mathbf{bool}_A$	Type of booleans, 3.1 $A^2 \multimap A$
${}_n\mathbf{bool}_A$	Type of n-booleans, 3.2 $A^n \multimap A$, $n \geq 2$
$\mathbf{int}_A$	Type of integers, 3.3 $!(A \multimap A) \multimap (A \multimap A)$
$\mathbf{bint}_A$	Type of binary integers, 3.13 $!(A \multimap A) \multimap (!(A \multimap A) \multimap (A \multimap A))$
$\mathbf{Tur}_A$	Type of Turing configurations, 5.3 $!\mathbf{bint}_A \otimes !\mathbf{bint}_A \otimes !{}_n\mathbf{bool}_A$
$\underline{i}_A$	Church numeral or n-boolean corresponding to $i \in \mathbb{N}$, 3.3, 3.2 $\vdash \mathbf{int}$ or $\vdash {}_n\mathbf{bool}$
$\underline{S}_A$	Proof corresponding to a binary integer $S \in \{0,1\}^*$, 3.13 $\vdash \mathbf{bint}$
$\underline{\mathrm{comp}}_A$	Composition of two endomorphisms, 3.4 $A, A \multimap A, A \multimap A \vdash A$
$\underline{\mathrm{add}}_A$	Addition of two integers, 3.10 $\mathbf{int}_A, \mathbf{int}_A \vdash \mathbf{int}_A$
$\underline{\mathrm{mult}}_A$	Multiplication of two integers, 3.11 $!\mathbf{int}_A, \mathbf{int}_A \vdash \mathbf{int}_A$
$\underline{\mathrm{pred}}_A$	Predecessor of an integer, 3.12 $\mathbf{int}_{A^2} \vdash \mathbf{int}_A$
$\underline{\mathrm{concat}}_A$	Concatenation of binary integers, 3.16 $\mathbf{bint}, \mathbf{bint} \vdash \mathbf{bint}_A$
$\underline{\mathrm{repeat}}_A$	Repetition of binary integers, 3.17 $!\mathbf{bint} \vdash \mathbf{bint}_A$
$\underline{\mathrm{iter}}(\pi, \rho)$	Iteration of the proof π evaluated on ρ, 3.28 $\mathbf{int}_A \vdash A$
$\underline{\mathrm{intcopy}}_A$	Copies an integer arbitrarily many times, 3.30 $\mathbf{int}_{!\mathbf{int}} \vdash !\mathbf{int}_A$
$\underline{\mathrm{head}}_A$	Returns the final digit of a binary integer, 5.4 $\mathbf{bint}_{A^3} \vdash \mathbf{bool}_A$

$\underline{\text{tail}}_A$	Deletes the final digit of a binary integer, 5.5 $\mathbf{bint}_{A^3} \vdash \mathbf{bint}_A$
$\underline{\text{booltype}}_A$	Converts the base type of an n-boolean, 5.6 ${}_n\mathbf{bool}_{A^3} \vdash {}_n\mathbf{bool}_A$
${}^i_\delta\underline{\text{trans}}_A$	Computes the transition function of a Turing machine, 5.8 $\mathbf{bint}_A, {}_n\mathbf{bool}_A \vdash {}_m\mathbf{bool}_A$
${}^i\underline{\text{recomb}}_A$	Recombines the tape binary integers, 5.10 $\mathbf{bint}_A, \mathbf{bool}_A, \mathbf{bool}_A \vdash \mathbf{bint}_A$
${}_\delta\underline{\text{left}}_A$	Updates the left part of the tape of a Turing machine, 5.11 $3\,\mathbf{bint}_{A^3}, \mathbf{bint}_{A^3}, 2\,{}_n\mathbf{bool}_{A^3} \vdash \mathbf{bint}_A$
${}_\delta\underline{\text{right}}_A$	Updates the right part of the tape of a Turing machine, 5.11 $2\,\mathbf{bint}_{A^3}, 2\,\mathbf{bint}_{A^3}, 2\,{}_n\mathbf{bool}_{A^3} \vdash \mathbf{bint}_A$
${}_\delta\underline{\text{state}}_A$	Updates the state of a Turing machine, 5.11 $\mathbf{bint}_{A^3}, {}_n\mathbf{bool}_{A^3} \vdash {}_n\mathbf{bool}_A$
${}_\delta\underline{\text{step}}_A$	Simulates a transition step of a Turing machine, 5.12 $\mathbf{Tur}_{A^3} \vdash \mathbf{Tur}_A$
${}_\delta\underline{\text{step}}^{\text{ND}}_A$	Simulates a transition step of a nondeterministic Turing machine, 5.17 $!\mathbf{Tur}_{A^3} \vdash !\mathbf{Tur}_A$