Models of computation and deductive systems

The $\lambda\mu$–calculus and Classical Logic

Abstract

In this note, we present the $\lambda\mu$–calculus of Parigot [Par92a], we follow the original paper of Parigot, that we seek to decompose in more explicit substeps. The first step we take is analyzing the classical sequent calculus **LK** in order to exhibit a minimal proof system that we call **LKm** equivalent to **LK**. In a second time, we show how the proofs in **LKm** can be mapped to the proofs in the deductive system **NK**. **NK** is the classical natural deduction system and was also introduced by Parigot, based on its earlier work on free deduction [Par92b]. To obtain a calculus we must then introduce a notion of named sequents obtaining the sequent calculus $\mathbf{NK}_\mu$ equivalent to **NK** (and so to **LK**), such that each proofs correspond to a term of the $\lambda\mu$–calculus. To exhibit the "Curry–Howard correspondence" we must then construct an invertible map from $\mathbf{NK}_\mu$ proofs to simply typable $\lambda\mu$–terms (where variables are typed 'à la Church'), such that the reductions are preserved by the map. To show this we prove that the notion of substitution in the $\lambda\mu$–calculus and $\mathbf{NK}_\mu$ proofs are compatible with the constructed invertible map, this is the key that allow to conclude the correspondence, this is heavily related to the definition of substitution, the proof is easy but require many cases for the induction.

Contents

Introduction

The notion of *proof* is central to mathematics and hence to logic. In the beginning of the 20th century, mathematicians have been proposing various notions of *formal proofs*, thus making the notion of proof a mathematical concept that could be studied by mathematicians. Chronologically, the three notion of formal proofs that have been standing out are the *Hilbert style deduction system* due to David Hilbert, the *natural deduction* and the *sequent calculus* both due to Gerhard Gentzen (introduced in the early 1930's).

In parallel to the discoveries of these logicians, still in the 1930's, Alonzo Church introduces the λ–calculus a formal system able to effectively describe the calculable functions. While in 1936, Alan Turing

invents the notion that is today known as *Turing Machine's*. These events can be seen as foundational works leading to the birth of computer science, and research in *computation* will continue from there.

At this point the links between the notion of formal proof and the one of program is not obvious. Yet years later, in 1958, Curry establishes a correspondence between the notion of proofs from a Hilbert style deduction systems and terms (i.e. programs) from combinational logic. A decade later in 1969, Howard also notices a correspondence, similar to the one exhibited by Curry: the proofs of intuitionistic natural deduction correspond to the terms of the simply typed λ–calculus. These two results are known as the *Curry–Howard correspondence* and state that the notion of formal proof and program are the same (in these specific contest). Since, mathematicians and informaticians have been looking at other correspondence of the sort. For instance Girard and Reynolds introduced indenpendantly the system F in 1972 and 1974, and it has been shown that the System F corresponds to the fragment of second-order intuitionistic minimal logic. A natural question though that was still unanswered was wether such a correspondence existed for *classical logic*. This remain unanswered until the early nineties, but the answer was positive, such a correspondence exists and is obtained by adding to the regular λ–calculus a notion of control operator (similar to the call/cc scheme) corresponding to double negation elimination [Par92a; Gri89].

The goal of this note is to expose this correspondence, between the proofs from classical logic and the $\lambda\mu$–calculus. We follow the approach proposed by Parigot [Par92a], as it is the most widely used, but beware that there exists others [Gri89; FH92]. The first step to take is to construct a natural deduction for classical logic, this is the key to obtain the correspondence as it is known that the proof–program correspondence behaves better with natural deduction than with sequent calculus. To do so we exhibit a fragment of the classical sequent calculus able to simulate the whole fragment, we call it the minimal fragment, then we define and establish the bisimilarity of this fragment with the classical natural deduction. Once we have obtained this classical natural deduction, we modify it with a notion of *names* that allows us to simulate the *structural rules* on the right side of the sequent. It is key since we know that in intuitionistic logic the structural rules cannot happen on the right side of the sequent (although they are permitted on the left), also, the falsehood constant $\bot$ may not be introduced on the right side of a sequent in intuitionistic logic. From this classical natural deduction with names we construct the $\lambda\mu$–calculus. Finally, we show that the $\lambda\mu$–terms i.e. programs can be mapped to the proofs of the deductive system we have constructed, and that this can be done by an invertible map. But also, we show that the notions of reduction in the $\lambda\mu$–calculus and in classical natural deduction coincide and are preserved by the map. Thus, we exhibit a *Curry–Howard correspondence*.

The goal of this note is to explicit the technical details needed to construct the correspondence, we follow mainly the work of Parigot [Par92a]. Although we exhibit the equivalences between the proofs system, and construct the correspondence using a typing 'a la Church' so that our map is invertible (as does J.Y. Girard in Proofs and Types [GTL89]). Also we consider special symbol to indicate μ–abstraction that bind a name that does not occur in a term (i.e. correspond to a weakening), this is needed for us to construct an invertible map and thus having a so–called isomorphism.

In the first subsection we recall notions from first order logic, and present some simple lemma's that will be our tools to show that two sequent calculus are equivalent.

1 Natural Deduction for Classical logic

1.1 First order classical logic, deductive systems and their equivalences

The goal of this section is to explicit a subfragment of **LK**, that we will denote **LKm**, which is equivalent to **LK**. We recall basic notions from first order classical logic.

Definition 1.1 (Signature)**.** A first order *signature* is a tuple $\Sigma = (\Sigma_V, \Sigma_C, (\Sigma_F^i)_{i\in\mathbb{N}}, (\Sigma_R^i)_{i\in\mathbb{N}})$ such that:

- Σ_V is a set of elements which are called *variables*. We will make variables range over $x, y, z, \ldots$.
- Σ_C is a set of elements called *constants*. We make constants range over $c, d, e, \ldots$.
- Given an integer i, Σ_F^i is a set of symbols which are called *function of arity i*.
- Given an integer i, Σ_R^i is a set of symbols which are called *relation of arity i*.

Definition 1.2 (Terms over a signature)**.** Given a signature Σ we define the Σ–terms, or terms over Σ by the following grammar:

$$t := x \in \Sigma_V \quad | \quad c \in \Sigma_C \quad | \quad f(t_1, ..., t_n) \quad (f \in \Sigma_F^n)$$

Along this section we know assume we are given a set of propositional variables $\mathcal{P}$ and a first order signature $\Sigma = (\Sigma_V, \Sigma_C, (\Sigma_F^i)_{i\in\mathbb{N}}, (\Sigma_R^i)_{i\in\mathbb{N}})$.

Definition 1.3 (Formula)**.** The formula's of classical logic are defined inductively by the following grammar, We denote the set of formula's as $\mathcal{F}$.

$$F, G := X \in \mathcal{P} \quad | \quad R(t_1, ..., t_n) \quad | \quad F \vee G \quad | \quad F \wedge G \quad | \quad F \to G \quad | \quad \neg F \quad | \quad \forall x.F \quad | \quad \exists x.F$$

Where $t_1, ..., t_n$ are Σ–terms.

Definition 1.4 (Substitutions, in terms and formula's)**.** Inductively we define the *substitution operation* in Σ–term t, the substitution replaces a variable $x \in \Sigma_C$ by some Σ–term. So given two terms t and u and a variable x, $t[x/u]$ is defined inductively as follow:

- If t is a variable, $x[x/u] = u$ and $y[x/u] = y$.
- If t is a constant $c[x/u] = c$.
- The inductive step is $f(t_1, ..., t_n)[x/u] = f(t_1[x/u], ..., t_n[x/u])$

Similarly given a formula A, we define inductively the substitution $A[t/x]$:

- In the base cases $X[t/x] = X$ and $R(t_1, ..., t_n)[t/x] = R(t_1[t/x], ..., t_n[t/x])$
- In the inductive cases we have $(F \vee G)[t/x] = F[t/x] \vee G[t/x]$, $(F \wedge G)[t/x] = F[t/x] \wedge G[t/x]$, $(\neg F)[t/x] = \neg F[t/x]$, $(\forall x.F)[t/x] = \forall x.F[t/x]$, $(\exists x.F)[t/x] = \exists x.F[t/x]$

Notation. Given a set X, we denote by $\mathcal{P}(X)$ the set of its parts.

Definition 1.5 (Relation, equivalence, compatibility and closures)**.** A *binary relation* $\mathcal{R}$ over a set X (or merely relation if there is no ambiguity) is a subset of the cartesian product X^2. Then given two points x, y of X, we denote $x\mathcal{R}y$ for $(x, y) \in \mathcal{R}$.

Given a relation $\mathcal{R}$ over a set X

- The relation $\mathcal{R}$ is *reflexive* if for any $x \in X$, $x\mathcal{R}x$.
- $\mathcal{R}$ is *symmetric* if for any $x, y \in X$, $x\mathcal{R}y$ implies $y\mathcal{R}x$.
- $\mathcal{R}$ is transitive is for any x, y, z in X, if $x\mathcal{R}y$ while $y\mathcal{R}z$ then $x\mathcal{R}z$.
- A relation $\mathcal{R}$ is said to be an *equivalence* relation, if it is reflexive, transitive and symmetric.

Definition 1.6 (Multiset, support, size, representation, sequents)**.** A *multiset* of a set X is a function $\Gamma : X \to \mathbb{N}$. Given a sequence $s = (x_1, ..., x_n)$ of elements of X the *associated multiset* of s is the function $[s] : X \to \mathbb{N}$ defined as the map $x \mapsto card\{i \in \mathbb{N} | x_i = x\}$.

Given a multiset Γ on X,

- The *support* of Γ is the set $Supp(\Gamma) = \{x \in X | \Gamma(x) \neq 0\}$.
- The *size* of Γ is the cardinal of its support $Supp(\Gamma)$.
- We call *representation* of Γ any sequence s of elements of X, such that its associated multiset is Γ.
- A *sequent* is a pair of multisets (Γ, Δ), written as $\Gamma \vdash \Delta$.
- We will denote the set of multisets over X by $\mathbb{N}^X$, and the set of sequents over X by $(\mathbb{N}^X)^2$.

We may denote a sequent Γ as one of its representation $A_1, ..., A_n$, this is an abuse of notation but in the case of classical logic it does not have too much consequences since this logic is *commutative*.

Definition 1.7 (deduction rule, axiom, compatibility with an equivalence, sequent calculus)**.** A *deduction rule* or simply *rule* of arity $k \in \mathbb{N}$ is a *partial* function that maps k sequents to one sequent.

- A *rule* of arity 0 is called an axiom.
- A sequent calculus is a finite set of *deduction rules*.

Definition 1.8 (Deduction and proofs)**.** Given a sequent calculus **SC** A *deduction*in the calculus **SC** is a tree (i.e. a partially ordered set , but more precisely is a labelled tree) $T = (V, \leq, \ell)$ such that,

- The vertices are labelled by sequent, i.e. $\ell : V \to (\mathbb{N}^X)^2$.
- The relation $\leq$ is defined as follow: two sequents S_0 and S_1 are such that $S_0 \leq S_1$, if there exists $S_2, ..., S_k$ sequent and a rule of arity k, such that; r maps $S_1, ..., S_k$ to S_0, and, the only successors of S are the sequents $S_1, ..., S_k$ i.e. $succ(S_0) = \{S_1, ..., S_k\}$.

The *conclusion* of a deduction is the sequent labelling the root of the tree.

A *proof* is a deduction such that for each leaf of the tree n there exists an axiom mapping $\emptyset$ to the label of the leaf i.e. $\ell(n)$.

Definition 1.9 (compatibility and equivalent multisets, sequents, and deductions)**.** Given an equivalence $\mathcal{R}$ on formula's. Two representations of two multisets Γ and Δ are $\mathcal{R}$*–equivalent* if, Γ and Δ are of the same size k and for each $1 \leq i \leq k$ we have $\Gamma(i)\mathcal{R}\Delta(i)$.

We lift the definition of $\mathcal{R}$–equivalent multisets to sequents and proofs, and define the notion of compatibility.

- Two sequents $\Gamma \vdash \Delta$ and $\Sigma \vdash \Pi$ are $\mathcal{R}$ equivalent if,
  - Γ and Σ are $\mathcal{R}$–equivalent while
  - Δ and Π are $\mathcal{R}$–equivalent.
- Two deductions are $\mathcal{R}$–equivalent if
  - Their conclusions (they are sequents) are $\mathcal{R}$–equivalent.
  - their proper leaves (leaves that are not simultaneously roots) are in bijection by means of a map commuting with the label function. i.e. there is a bijection $\phi : leaf(\mathcal{D}) \to leaf(\mathcal{D}')$ such that $\phi(\ell(n)) = \ell(\phi(n))$ for any $n \in leaf(\mathcal{D})$.
- A deduction rule is *compatible* with an equivalence relation $\mathcal{R}$ on formula's if, given some sequents $(S_1, ..., S_k)$ and some $\mathcal{R}$–equivalent sequence of sequents $(S'_1, ..., S'_k)$, if r maps $(S_1, ..., S_k)$ to a sequent S then r also maps $(S'_1, ..., S'_k)$ to a sequent S' such that $S\mathcal{R}S'$.
- A sequent calculus is *compatible* with an equivalence relation $\mathcal{R}$ if all of its rules are.

Definition 1.10 (Simulation and equivalence of sequent calculus)**.** Given two sequent calculus S_1 and S_2, and an equivalence relation $\mathcal{R}$ on formula's.

- The sequent calculus S_2 *simulates* the calculus S_1 modulo the relation $\mathcal{R}$, if for each deduction π_1 in S_1 of a sequent S there exists a $\mathcal{R}$–equivalent deduction π_2 in S_2. In this case we write $\mathsf{S}_1 \lesssim_{\mathcal{R}} \mathsf{S}_2$.
- If two calculus S_1 and S_2 simulate one another modulo $\mathcal{R}$, we say they are $\mathcal{R}$*–equivalent*, which we denote by $\mathsf{S}_1 \sim_{\mathcal{R}} \mathsf{S}_2$.

- A sequent calculus S_1 simulates a sequent calculus S_2 *in the presence of* a set of rules S w.r.t. an equivalence $\mathcal{R}$, if $\mathsf{S}_1 \cup \mathsf{S}$ simulates $\mathsf{S}_2 \cup \mathsf{S}$ w.r.t. $\mathcal{R}$.

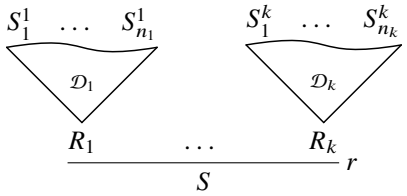
Lemma 1.1. *Given a sequent calculus* S *compatible with an equivalence* $\mathcal{R}$ *on formula's, and a derivation* $\mathcal{D}$ *from* k *sequents* $(S_1, ..., S_k)$ *to a sequent* S.

For any sequents $(S'_1, ..., S'_k)\mathcal{R}(S_1, ..., S_k)$ $\mathcal{D}$ *is a derivation from* $(S'_1, ..., S'_k)$ *to a sequent* S'*, such that* $S\mathcal{R}S'$.

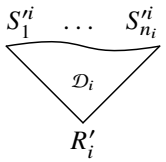
Proof. We do so by induction on $\mathcal{D}$. Assume the derivation is of length one, then we have two cases;

- $\mathcal{D}$ is an identity from S to S, then it is also an identity from S' to S' with $S\mathcal{R}S'$.
- $\mathcal{D}$ is the result of a rule of arity 0, then r maps to S, but since S is compatible with $\mathcal{R}$ r also maps to S' with $S'\mathcal{R}S$.

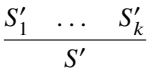
In the induction case we have $\mathcal{D}$ of the following form:



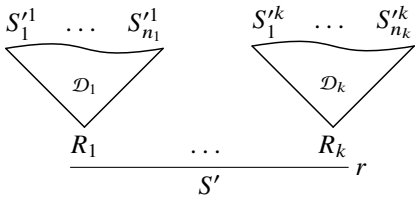
Now applying the induction hypothesis on the sub–derivations $\mathcal{D}_i$ for each i we show that $\mathcal{D}$ is a derivation of $R'_i\mathcal{R}R_i$ for $\mathcal{R}$ equivalent hypothesis.



From there noticing that the sequent calculus is $\mathcal{R}$ compatible, and that for each i we have $R_i\mathcal{R}R'_i$, we conclude that the sequent S' obtained by the application of the rule r on $S'_1, ..., S'_k$ – i.e. obtained by the following rule – must be $\mathcal{R}$ equivalent to S.



From there we can obtain the derivation.



□

Lemma 1.2 (Simulation lemma)**.** *Given two sequent calculus* S_1 *and* S_2 *compatible with an equivalence* $\mathcal{R}$.

S_2 *simulates* S_1*, if and only if, for any rule* r *of arity* k *in* S_1 *mapping* $(S_1, ..., S_k)$ *to* S *there exists in* S_2 *a deduction* $\mathcal{D}_r$ *from* $(S_1, ..., S_k)$ *to* S'*, such that* $S\mathcal{R}S'$.

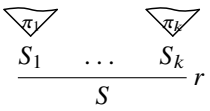
Proof. **1** $\Rightarrow$ **2**. It is clear that if S_2 simulates S_1 since rules are particular deductions, each rules of S_1 can be simulated by some deduction of S_2 with the same hypothesis.

2 $\Rightarrow$ **1**. On the other hand assume each rule of S_1 can be simulated by a deduction in S_2. Consider then π a deduction in S_1. We show the simulation of π by induction.

If π is made of only one root. Two cases may occur:

- π is a deduction from a sequent S to itself. In that case the same deduction is a deduction from S to itself in S_2.
- π is a proof of a sequent S by applying an axiom. In that case we apply the hypothesis on the rule of arity 0, obtaining a proof $\mathcal{D}$ of $S'\mathcal{R}S$ in S_2.

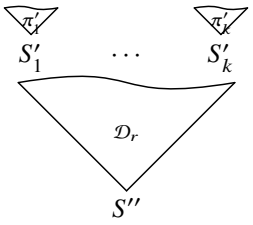
Then by induction π is of the following form:



We can apply the induction hypothesis on each sub–derivation $\pi_1, ..., \pi_k$, obtain derivations $\pi'_1, ..., \pi'_k$ of conclusions $(S'_1, ..., S'_k)\mathcal{R}(S_1, ..., S_k)$.

Then applying the hypothesis of simulation for the rule r we are given a derivation $\mathcal{D}_r$ mapping $S_1, ..., S_k$ to a sequent $S'\mathcal{R}S$.

But now we call the previous lemma on the sequent calculus $\mathcal{S}_2$, and deduce that $\mathcal{D}_r$ is also a derivation from $(S'_1, ..., S'_k)$ to $S''\mathcal{R}S'$ (and so $S''\mathcal{R}S$). Therefore we construct the wanted derivation.



Noticing that π' has an equivalent conclusion as π, and the same hypotheses, we have shown by induction that each derivation in S_1 can be simulated in S_2.

□

Definition 1.11 (Equivalence and compatible closure)**.** A relation is *compatible* with the grammar of formula's if and only if, given two formula's A and B such that $A\mathcal{R}B$ we have

- $\forall x.A\mathcal{R}\forall x.B$, $\exists x.A\mathcal{R}\exists x.B$ and $\neg A\mathcal{R}\neg B$
- Given a formula C, $A \vee C\mathcal{R}B \vee C$, $A \wedge C\mathcal{R}B \wedge C$ and $A \rightarrow C\mathcal{R}B \rightarrow C$

We have also notions of closure

- The equivalence closure of a relation $\mathcal{R}$ is the smallest [1] equivalence relation containing $\mathcal{R}$.
- The compatible closure of a relation $\mathcal{R}$ on formula's is the smallest compatible relation containing $\mathcal{R}$.

NB. The notion of compatible relation can be generalized to any grammar, and so a notion of compatible relation with a grammar exists. This is also true for the notion of compatible closure then.

[1] in the sense of the subset, i.e. in the ordered structured $(\mathcal{P}(X^2), \subset)$, since a relation is an element of $\mathcal{P}(X^2)$

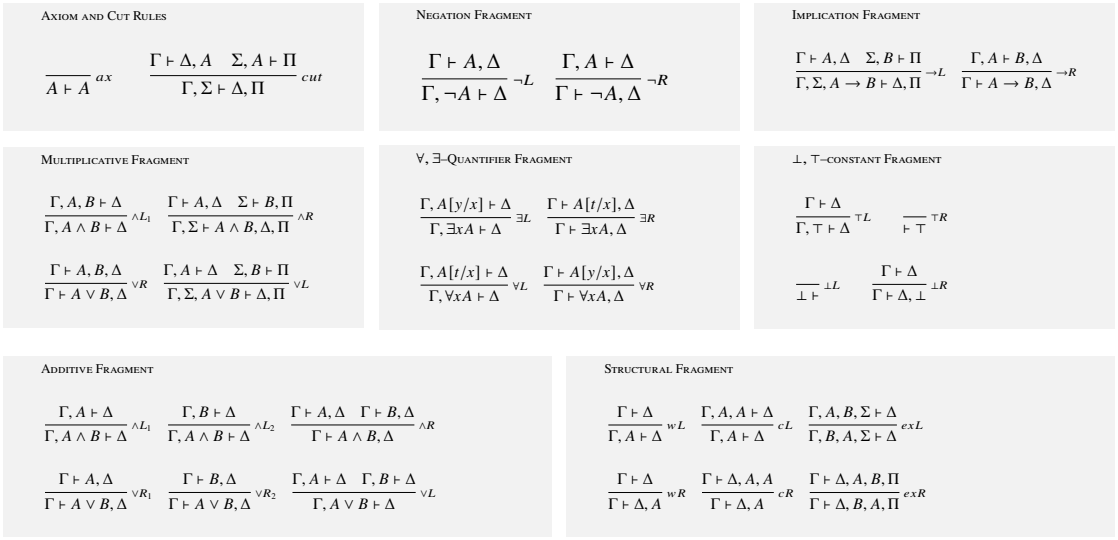


Figure 1: Classical sequent calculus **LK**

1.2 A minimal proof system for classical logic

The goal of this section is to exhibit a minimal proof system for classical logic. We give the full sequent calculus for classical logic in figure 1. We will use the results of the previous section to show the equivalence between fragments of classical logic. Our relation of equivalence will be usual translations like the one of de Morgan. A preliminary result we need is to show that **LK** is compatible with our equivalence relations, since each of the equivalence considered are compatible closures and that the sequent calculus **LK** has the sub–formula property, this ensure that **LK** is compatible with the equivalences with consider.

Definition 1.12 (The De Morgan translation)**.** We define the De Morgan translation as the compatible and equivalence closure of the following relation:

- $A \vee B \equiv \neg(\neg A \wedge \neg B)$
- $A \wedge B \equiv \neg(\neg A \vee \neg B)$
- $\exists x A \equiv \neg \forall x \neg A$
- $\forall x A \equiv \neg \exists x \neg A$
- $\bot \equiv \neg \top$
- $\top \equiv \neg \bot$

First we want to point out that some rules of **LK** are obviously redundant: this is the case for the constants rules $\bot R$ and $\top L$, both of these rules can be simulated using respectively the right and left weakening rules.

We define as the fragment of **LK** consisting of all its rules expect the left structural rules. We define **LKm** as the rules of **LK** such that the logical rules are restricted to the constants, negation, and implication rules and the structural rules are restricted to the right side. And $\mathbf{LKm}^*$ as the rules of **LK** restricted to constants, implication rules, right structural rules, quantifiers rules, axiom and cut. i.e. We define $\mathbf{LKm} = \{ax, cut \to R, \to L, \neg R, \neg R, \forall R, \forall L, \bot R, \bot L, wR, cR, exR\}$ and $\mathbf{LKm}^* = \{ax, cut \to R, \to L, \forall R, \forall L, \bot R, \bot L, wR, cR, exR\}$

Proposition 1.1 (Bisimilar fragments up to de Morgan translation in the presence of the negation)**.** *We have the following relations between fragments of* ***LK****.*

1. *$\{wR, cR, exR, \neg L, \neg R\}$ and $\{wL, cL, exL, \neg L, \neg R\}$ are bisimilar.*
2. *$\{\wedge R, \wedge L1, \wedge L2, \neg L, \neg R\}$ and $\{\vee L, \vee R1, \vee R2, \neg L, \neg R\}$ are bisimilar up to the de Morgan translation.*
3. *$\{\forall L, \forall R, \neg R\}$ and $\{\exists R, \exists L, \neg L, \neg R\}$ are bisimilar up to de Morgan translation.*
4. *$\{\top R, \top L, \neg L, \neg R\}$ and $\{\bot R, \bot L, \neg L, \neg R\}$ are bisimilar up to de Morgan translation.*

Proof. **1.** Note that the left (resp. right) structural rules can simulate the right (resp. left) structural rules. This is possible using the left and right negation rules. Here we show only how to simulate the right structural fragment with left one, but the other direction is perfectly symmetric.

$$\frac{\Gamma \vdash \Delta}{\Gamma \vdash \Delta, A}\, wR \quad \mapsto \quad \frac{\dfrac{\Gamma \vdash \Delta}{\Gamma, \neg A \vdash \Delta}\, wL}{\Gamma \vdash \Delta, A}\, \neg R \qquad \frac{\Gamma \vdash \Delta, A, A}{\Gamma \vdash \Delta, A}\, cR \quad \mapsto \quad \frac{\dfrac{\Gamma \vdash \Delta, A, A}{\Gamma, \neg A, \neg A \vdash \Delta}\, \neg L\times 2}{\Gamma, \neg A \vdash \Delta}\, cL}{\Gamma \vdash \Delta, A}\, \neg R \qquad \frac{\Gamma \vdash \Delta, A, B, \Pi}{\Gamma \vdash \Delta, B, A, \Pi}\, exR \quad \mapsto \quad \frac{\dfrac{\Gamma \vdash \Delta, A, B, \Pi}{\Gamma, \neg\Pi, \neg B, \neg A \vdash \Delta}\, \neg L\times n}{\Gamma, \neg\Pi, \neg A, \neg B \vdash \Delta}\, exL}{\Gamma \vdash \Delta, B, A, \Pi}\, \neg R\times n$$

2. We show how to simulate (up to de Morgan equivalence) the $\wedge$–fragment with the $\vee$–fragment in the presence of the negation. Like the previous point we only show one direction, as the other direction is perfectly symmetric.

$$\frac{\Gamma, A \vdash \Delta}{\Gamma, A \wedge B \vdash \Delta}\, \wedge L1 \quad \mapsto \quad \frac{\dfrac{\dfrac{\Gamma, A \vdash \Delta}{\Gamma \vdash \Delta, \neg A}\, \neg R}{\Gamma \vdash \Delta, \neg A \vee \neg B}\, \vee L1}{\Gamma, \neg(\neg A \vee \neg B) \vdash \Delta}\, \neg L \qquad \frac{\Gamma, B \vdash \Delta}{\Gamma, A \wedge B \vdash \Delta}\, \wedge L2 \quad \mapsto \quad \frac{\dfrac{\dfrac{\Gamma, B \vdash \Delta}{\Gamma \vdash \Delta, \neg B}\, \neg R}{\Gamma \vdash \Delta, \neg A \vee \neg B}\, \vee L2}{\Gamma, \neg(\neg A \vee \neg B) \vdash \Delta}\, \neg L$$

$$\frac{\Gamma \vdash \Delta, A \quad \Sigma \vdash \Pi, B}{\Gamma, \Sigma \vdash \Delta, \Pi, A \wedge B}\, \wedge R \quad \mapsto \quad \frac{\dfrac{\dfrac{\Gamma \vdash \Delta, A}{\Gamma, \neg A \vdash \Delta}\, \neg L \quad \dfrac{\Sigma \vdash \Pi, B}{\Sigma, \neg B \vdash \Pi}\, \neg L}{\Gamma, \Sigma, \neg A \vee \neg B \vdash \Delta, \Pi}\, \vee L}{\Gamma, \Sigma \vdash \Delta, \Pi, \neg(\neg A \vee \neg B)}\, \neg R$$

3. Now let's show we can simulate (using de Morgan equivalence) the rules for the $\exists$ quantifier, with the rules of $\forall$ quantifier, in the presence of the negation rules. Again we show only one direction of the simulation.

$$\frac{\Gamma, A[y/x] \vdash \Delta}{\Gamma, \exists x A \vdash \Delta}\, \exists L \quad \mapsto \quad \frac{\dfrac{\dfrac{\Gamma, A[y/x] \vdash \Delta}{\Gamma \vdash \Delta, \neg A[y/x]}\, \neg R}{\Gamma \vdash \Delta, \forall x \neg A}\, \forall R}{\Gamma, \neg\forall x \neg A \vdash \Delta}\, \neg L \qquad \frac{\Gamma \vdash \Delta, , A[t/x]}{\Gamma \vdash \Delta, \exists x A}\, \exists L \quad \mapsto \quad \frac{\dfrac{\dfrac{\Gamma \vdash \Delta, A[t/x]}{\Gamma, \neg A[t/x] \vdash \Delta}\, \neg L}{\Gamma, \forall x \neg A \vdash \Delta}\, \forall L}{\Gamma \vdash \Delta, \neg\forall x \neg A}\, \neg R$$

4. Finally we show that the rules for the true constant can be simulated by the rules of the false constant, still in the presence of the negation rules and up to the de Morgan equivalence. Again we only show one direction of the simulation.

$$\frac{}{\vdash \top}\, \top R \quad \mapsto \quad \frac{\dfrac{}{\bot \vdash}\, \bot L}{\vdash \neg\bot}\, \neg R \qquad \frac{\Gamma \vdash \Delta}{\Gamma, \top \vdash \Delta}\, \top L \quad \mapsto \quad \frac{\dfrac{\Gamma \vdash \Delta}{\Gamma \vdash \Delta, \bot}\, \bot R}{\Gamma, \neg\bot \vdash \Delta}\, \neg L$$

□

Each formula of classical logic can be expressed using only the symbols from $\{\rightarrow, \neg, \forall\}$. This is a really useful feat to reduce the number of rules needed for a subfragment to simulate **LK**. To do so we must use the arrow translation that we define next.

Definition 1.13 (The arrow translation)**.** The arrow translation is the equivalence and compatible closure of the following binary relation.

$$A \vee B \equiv \neg A \rightarrow B \equiv \neg B \rightarrow A \qquad A \wedge B \equiv \neg(A \rightarrow \neg B) \equiv \neg(B \rightarrow \neg A)$$

Proposition 1.2. *We have the following simulation relations with respect to the equivalence relation that is the arrow translation.*

- $\{\vee L, \vee R1, \vee R2\}$ *and* $\{\rightarrow L, \rightarrow R\}$ *are bisimilar up to the arrow translation, in the presence of the rules* $\{wR, cR, exR, \neg L, \neg R\}$.
- $\{\wedge R, \wedge L1, \wedge L2\}$ *and* $\{\rightarrow L, \rightarrow R\}$ *are bisimilar up to the arrow translation in the presence of the rules* $\{wR, cR, exR, \neg L, \neg R\}$.

Proof. **1.** Let's simulate the rules of the $\vee$–fragment using the rules of implication and the structural rules.

$$\frac{\Gamma, A \vdash \Delta \quad \Sigma, B \vdash \Pi}{\Gamma, \Sigma, A \vee B \vdash \Delta, \Pi}\vee L \quad \mapsto \quad \frac{\dfrac{\Gamma, A \vdash \Delta}{\Gamma \vdash \Delta, \neg A}\neg R \quad \Sigma, B \vdash \Pi}{\Gamma, \Sigma, \neg A \rightarrow B \vdash \Delta, \Pi}\rightarrow L \qquad \frac{\Gamma \vdash \Delta, A}{\Gamma \vdash \Delta, A \vee B}\vee R1 \quad \mapsto \quad \frac{\dfrac{\dfrac{\Gamma \vdash \Delta, A}{\Gamma \vdash \Delta, A, B}wR}{\Gamma, \neg A \vdash \Delta, B}\neg L}{\Gamma \vdash \Delta, \neg A \rightarrow B}\rightarrow R$$

Now we show that the implication rules may be simulated by the rules of the $\vee$ connective in pair with the negation and structural rules.

$$\frac{\Gamma \vdash \Delta, A \quad \Sigma, B \vdash \Pi}{\Gamma, \Sigma, A \rightarrow B \vdash \Delta, \Pi}\rightarrow L \quad \mapsto \quad \frac{\dfrac{\Gamma \vdash \Delta, A}{\Gamma, \neg A \vdash \Delta}\neg L \quad \Sigma, B \vdash \Pi}{\Gamma, \Sigma, \neg A \vee B \vdash \Delta, \Pi}\vee L \qquad \frac{\Gamma, A \vdash B, \Delta}{\Gamma \vdash \Delta, A \rightarrow B}\rightarrow R \quad \mapsto \quad \frac{\dfrac{\dfrac{\dfrac{\Gamma, A \vdash B, \Delta}{\Gamma \vdash \neg A, B, \Delta}\neg R}{\Gamma \vdash \neg A \vee B, B, \Delta}\vee R2}{\Gamma \vdash \neg A \vee B, \neg A \vee B, \Delta}\vee R1}{\Gamma \vdash \Delta, \neg A \vee B}cR$$

2. Let's simulate the $\wedge$–fragment with the implication and negation rules. We omit the case for $\wedge L2$ since it is almost identical to $\wedge L1$.

$$\frac{\Gamma, A \vdash \Delta}{\Gamma, A \wedge B \vdash \Delta}\wedge L1 \quad \mapsto \quad \frac{\dfrac{\dfrac{\dfrac{\Gamma, A \vdash \Delta}{\Gamma, A, B \vdash \Delta}wL}{\Gamma, A \vdash \Delta, \neg B}\neg R}{\Gamma \vdash \Delta, A \rightarrow \neg B}\rightarrow R}{\Gamma, \neg(A \rightarrow \neg B) \vdash \Delta}\neg L \qquad \frac{\Gamma \vdash \Delta, A \quad \Sigma \vdash \Pi, B}{\Gamma, \Sigma \vdash \Delta, \Pi, , A \wedge B}\wedge R \quad \mapsto \quad \frac{\Gamma \vdash \Delta, A \quad \dfrac{\Sigma \vdash \Pi, B}{\Sigma, \neg B \vdash \Pi}\neg L}{\dfrac{\Gamma, \Sigma, A \rightarrow \neg B \vdash \Delta, \Pi}{\Gamma, \Sigma \vdash \Delta, \Pi, \neg(A \rightarrow \neg B)}\neg R}\rightarrow L$$

□

Proposition 1.3. ***LKm*** *and* ***LK*** *are bisimilar up to de morgan equivalence and arrow translation (i.e. for the relation corresponding to the closure of their union).*

Proof. The proof is kind of straightforward, it is not particularly short but it contains alot of redundancy. We present it as a derivation of the bisimilation using the previous proposition and reflexivity of the bisimulation as axioms, then our only deductive rule is the previous proposition stating that if $S \sim Q$ and $R \sim T$ we can deduce $S \cup R \sim Q \cup T$.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\mathsf{IMP} \cup \mathsf{S_R} \cup \mathsf{NEG} \sim \mathsf{OR} \cup \mathsf{S_R} \cup \mathsf{NEG}}prop \quad \dfrac{}{\mathsf{IMP} \cup \mathsf{S_R} \cup \mathsf{NEG} \sim \mathsf{AND} \cup \mathsf{S_R} \cup \mathsf{NEG}}prop}{\mathsf{IMP} \cup \mathsf{S_R} \cup \mathsf{NEG} \sim \mathsf{OR} \cup \mathsf{AND} \cup \mathsf{S_R} \cup \mathsf{NEG}}\cup \quad \dfrac{}{\mathsf{S_R} \cup \mathsf{NEG} \sim \mathsf{S_L} \cup \mathsf{NEG}}prop \quad \dfrac{\dfrac{}{\mathsf{C_\perp} \cup \mathsf{NEG} \sim \mathsf{C_\top} \cup \mathsf{NEG}}prop \quad \dfrac{}{\mathsf{Q_\forall} \cup \mathsf{NEG} \sim \mathsf{Q_\exists} \cup \mathsf{NEG}}prop}{\mathsf{C_\perp} \cup \mathsf{Q_\forall} \cup \mathsf{NEG} \sim \mathsf{C_\top} \cup \mathsf{Q_\exists} \cup \mathsf{NEG}}\cup}{\mathsf{C_\perp} \cup \mathsf{Q_\forall} \cup \mathsf{S_R} \cup \mathsf{NEG} \sim \mathsf{C_\top} \cup \mathsf{Q_\exists} \cup \mathsf{S_L} \cup \mathsf{NEG}}\cup}{\mathsf{IMP} \cup \mathsf{S_R} \cup \mathsf{C_\perp} \cup \mathsf{Q_\forall} \cup \mathsf{NEG} \sim \mathsf{OR} \cup \mathsf{AND} \cup \mathsf{S_R} \cup \mathsf{S_L} \cup \mathsf{C_\top} \cup \mathsf{Q_\exists} \cup \mathsf{NEG}}add(\mathsf{IMP} \cup \mathsf{C_\perp} \cup \mathsf{Q_\forall})}{\mathsf{IMP} \cup \mathsf{S_R} \cup \mathsf{C_\perp} \cup \mathsf{Q_\forall} \cup \mathsf{NEG} \sim \mathsf{OR} \cup \mathsf{AND} \cup \mathsf{IMP} \cup \mathsf{S_R} \cup \mathsf{S_L} \cup \mathsf{C_\top} \cup \mathsf{C_\perp} \cup \mathsf{Q_\exists} \cup \mathsf{Q_\forall} \cup \mathsf{NEG}}add(\mathsf{ax} \cup \mathsf{cut})}{\mathsf{ax} \cup \mathsf{cut} \cup \mathsf{IMP} \cup \mathsf{S_R} \cup \mathsf{C_\perp} \cup \mathsf{Q_\forall} \cup \mathsf{NEG} \sim \mathsf{ax} \cup \mathsf{cut} \cup \mathsf{OR} \cup \mathsf{AND} \cup \mathsf{IMP} \cup \mathsf{S_R} \cup \mathsf{S_L} \cup \mathsf{C_\top} \cup \mathsf{C_\perp} \cup \mathsf{Q_\exists} \cup \mathsf{Q_\forall} \cup \mathsf{NEG}}=$$
$$LKm \sim LK$$

Where the rules $add(Q)$ with Q a set of rules, are defined as the following deductions.

$$\frac{S \sim T \quad \dfrac{}{Q \sim Q}reflexivity}{S \cup Q \sim T \cup Q}\cup$$

□

Definition 1.14 (Negation as implication of falsehood, intuitionistic translation)**.** The intuitionistic translation $\equiv_\bot$ is the equivalence and syntactical closure of the relation; $\neg A \equiv_\bot A \to \bot$ and $\top \equiv_\bot \bot \to \bot$

Proposition 1.4 (Simulation of the negation fragment up to intuitionistic translation)**.** *The fragment* $\{\to R, \to L, \bot R, \bot L\}$ *can simulate the fragment* $\{\neg L, \neg R, \top L, \top R\}$ *up to the intuitionistic translation.*

Proof. Let us show that the implication and falsehood fragment can simulate the left and right negation rules (as well as the $\top$ rules). This can be done in the following way.

$$\frac{\Gamma, A \vdash \Delta}{\Gamma \vdash \Delta, \neg A}\,\neg R \quad \mapsto \quad \frac{\dfrac{\Gamma, A \vdash \Delta}{\Gamma, A \vdash \Delta, \bot}\,\bot R}{\Gamma \vdash \Delta, A \to \bot}\,\to R \qquad \frac{\Gamma \vdash \Delta, A}{\Gamma, \neg A \vdash \Delta}\,\neg L \quad \mapsto \quad \frac{\Gamma \vdash \Delta, A \quad \dfrac{}{\bot \vdash}\,\bot L}{\Gamma, A \to \bot \vdash \Delta}\,\to L$$

$$\frac{}{\vdash \top}\,\top R \quad \mapsto \quad \frac{\dfrac{\dfrac{}{\bot \vdash}\,\bot L}{\bot \vdash \bot}\,\bot R}{\vdash \bot \to \bot}\,\to R \qquad \frac{\Gamma \vdash \Delta}{\Gamma, \top \vdash \Delta}\,\top L \quad \mapsto \quad \frac{\dfrac{}{\bot \vdash}\,\bot L \quad \dfrac{\Gamma \vdash \Delta}{\Gamma \vdash \Delta, \bot}\,\bot R}{\Gamma, \bot \to \bot \vdash \Delta}\,\to L$$

□

Proposition 1.5 (**LK**'s equivalent subfragment)**.** ***LKm*** *is equivalent to its subfragment* $\mathbf{LKm}^*$*, with respect to the equivalence relation that is the union of the de Morgan translation, the arrow translation and the intuitionistic translation.*

Proof. Again we will obtain the relation of simulation between the two fragments using derivation rules. From the previous proposition we will use an axiom, since we proved $\mathsf{NEG} \cup \mathsf{C}_\top \lesssim \mathsf{IMP} \cup \mathsf{C}_\bot$ we may aswell claim $\mathsf{NEG} \lesssim \mathsf{IMP} \cup \mathsf{C}_\bot$. We will then apply the *add* rules defined as the following,

$$\frac{\dfrac{}{T \lesssim T}\,reflexivity \qquad S \lesssim Q}{S \cup T \lesssim Q \cup T}\,\cup$$

$$\frac{\dfrac{\dfrac{}{\mathsf{NEG} \lesssim \mathsf{IMP} \cup \mathsf{C}_\bot}\,prop}{\mathsf{NEG} \cup \mathsf{IMP} \cup \mathsf{C}_\bot \cup \mathsf{ax} \cup \mathsf{cut} \cup \mathsf{S}_\mathsf{R} \cup \mathsf{Q}_\forall \lesssim \mathsf{IMP} \cup \mathsf{C}_\bot \cup \mathsf{ax} \cup \mathsf{cut} \cup \mathsf{S}_\mathsf{R} \cup \mathsf{Q}_\forall}\,add(\mathsf{IMP} \cup \mathsf{C}_\bot \cup \mathsf{ax} \cup \mathsf{cut} \cup \mathsf{S}_\mathsf{R} \cup \mathsf{Q}_\forall)}{LKm \lesssim LKm^*}\,=$$

From $LKm \lesssim LKm^*$ noticing that $LKm^* \subset LKm$ (i.e. is one of its sub–fragment) it is straightforward that $LKm^* \subset LKm$ (since each deduction of $\mathbf{LKm}^*$ is a deduction of **LKm**). From the two simulation we conclude the bisimulation $LKm \sim LKm^*$ □

At this point we can ensure that the three sequent calculus **LK**, **LKm** and $\mathbf{LKm}^*$, are equivalent with respect to the translations.

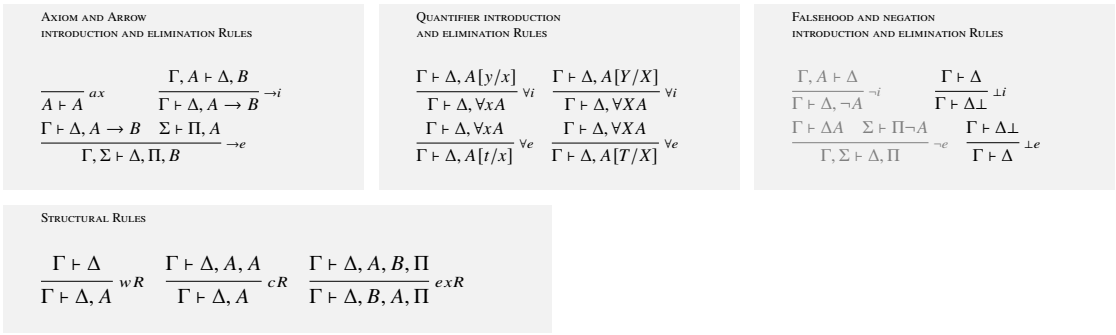


Figure 2: Classical Multiple–Conclusion Natural Deduction with names $\mathbf{NK}_\mu$ (as sequent calculus). It corresponds to a focused sequent calculus.

1.3 Classical Natural Deduction

Proposition 1.6. *The sequent calculus **LKm** is equivalent to the calculus **NK** (with respect to equality).*

Proof. **1**. $\mathbf{NK} \lesssim \mathbf{LKm}$. The introduction rules $\{ax, \neg_i, \to_i, \forall_i, \bot_i\}$ of **NK** are identical to the rules of **LKm** $\{ax, \neg_R, \to_R, \forall_R, \bot_R\}$. Also, the structural rules of **NK** are the identical to the structural rules of **LKm**. As a consequence, we only have to show that the elimination rules of **NK** that is $\{\neg_e, \to_e, \forall_e, \bot_e\}$ can be simulated by **LKm**.

$$\frac{\Gamma \vdash A, \Delta \quad \Sigma \vdash \neg A, \Pi}{\Gamma, \Sigma \vdash \Delta, \Pi}\,\neg_e \quad \mapsto \quad \frac{\dfrac{\Sigma \vdash \Pi, \neg A}{\Sigma, A \vdash \Pi}\,\neg_L \quad \Gamma \vdash A, \Delta}{\Gamma, \Sigma \vdash \Delta, \Pi}\,cut \qquad \frac{\Gamma \vdash A \to B, \Delta \quad \Sigma \vdash A, \Pi}{\Gamma, \Sigma \vdash B, \Delta, \Pi}\,\to_e \quad \mapsto \quad \frac{\Gamma \vdash \Delta, A \to B \quad \dfrac{\Sigma \vdash \Pi, A \quad \dfrac{}{B \vdash B}\,ax}{\Sigma, A \to B \vdash \Pi, B}\,\to_L}{\Gamma, \Sigma \vdash \Delta, \Pi, B}\,cut$$

$$\frac{\Gamma \vdash \Delta, \forall x A}{\Gamma \vdash \Delta, A[t/x]}\,\forall e \quad \mapsto \quad \frac{\Gamma \vdash \Delta, \forall x A \qquad \dfrac{\dfrac{}{A[t/x] \vdash A[t/x]}\,ax}{\forall x A \vdash A[t/x]}\,\forall L}{\Gamma \vdash \Delta, A[t/x]}\,cut \qquad \frac{\Gamma \vdash \Delta, \bot}{\Gamma \vdash \Delta}\,\bot e \quad \mapsto \quad \frac{\Gamma \vdash \Delta, \bot \qquad \dfrac{}{\bot \vdash}\,\bot L}{\Gamma \vdash \Delta}\,cut$$

2. $\mathbf{LKm} \lesssim \mathbf{NK}$. The introduction rules $\{ax, \neg_i, \to_i, \forall_i, \bot_i\}$ of **NK** are identical to the rules of **LKm** $\{ax, \neg_R, \to_R, \forall_R, \bot_R\}$. Also, the structural rules of **NK** are the identical to the structural rules of **LKm**. As a consequence, we only have to show that the logical left–hand rules of **LKm** that is $\{\neg L, \to L, \forall L, \bot L\}$ can be simulated by **NK**.

$$\frac{\Gamma \vdash \Delta, A}{\Gamma, \neg A \vdash \Delta}\,\neg L \quad \mapsto \quad \frac{\Gamma \vdash A, \Delta \quad \dfrac{}{\neg A \vdash \neg A}\,ax}{\Gamma, \neg A \vdash \Delta}\,\neg e \qquad \frac{\Gamma, A[t/x] \vdash \Delta}{\Gamma, \forall x A \vdash \Delta}\,\forall L \quad \mapsto \quad \frac{\dfrac{\dfrac{}{A[y/x] \vdash A[y/x]}\,ax}{A[y/x] \vdash \forall x A}\,\forall i \qquad \Gamma, A[t/x] \vdash \Delta}{\dfrac{\Gamma \vdash \Delta, \neg A[t/x]}{\Gamma, \forall x A \vdash \Delta}\,\neg e}\,\neg i$$

$$\frac{\Gamma \vdash A, \Delta \quad \Sigma, B \vdash \Pi}{\Gamma, \Sigma, A \to B \vdash \Delta, \Pi}\,\to L \quad \mapsto \quad \frac{\dfrac{\dfrac{}{A \to B \vdash A \to B}\,ax \qquad \Gamma \vdash \Delta, A}{\Gamma, A \to B \vdash \Delta, B}\,\to e \qquad \dfrac{\Sigma, B \vdash \Pi}{\Sigma \vdash \Pi, \neg B}\,\neg i}{\Gamma, A \to B, \Sigma \vdash \Delta, \Pi}\,\neg e \qquad \frac{}{\bot \vdash}\,\bot L \quad \mapsto \quad \frac{\dfrac{}{\bot \vdash \bot}\,ax}{\bot \vdash}\,\bot e \qquad \square$$

Proposition 1.7. *The sequent calculus **NK** is equivalent its sub–fragment* $\mathbf{NK}^*$ *that consists of **NK** rules without the rules of introduction and elimination of negation.*

Proof. To do so we must only simulate the introduction and elimination of the negation of **NK** in $\mathbf{NK}^*$. It is trivial that $\mathbf{NK}^*$ is simulated by **NK** as it one of its sub–fragment.

$$\frac{\Gamma, A \vdash \Delta}{\Gamma \vdash \Delta, \neg A}\,\neg i \quad \mapsto \quad \frac{\dfrac{\Gamma, A \vdash \Delta}{\Gamma, A \vdash \Delta, \bot}\,\bot i}{\Gamma \vdash \Delta, A \to \bot}\,\to i \qquad \frac{\Gamma \vdash \Delta, A \quad \Sigma \vdash \Pi, \neg A}{\Gamma, \Sigma \vdash \Delta, \Pi}\,\neg e \quad \mapsto \quad \frac{\dfrac{\Gamma \vdash \Delta, A \quad \Sigma \vdash \Pi, A \to \bot}{\Gamma, \Sigma \vdash \Delta, \Pi, \bot}\,\to e}{\Gamma, \Sigma \vdash \Delta, \Pi}\,\bot e$$

□

2 the $\lambda\mu$–calculus and classical natural deduction with names

2.1 Classical Natural deduction with names

In this section and from now on, we are given a set of names denoted $\mathcal{N}$. It contains a special element called the *empty name* denoted ϵ. A name is *proper* if it is not the empty name. Names are denoted by greek letters $\alpha, \beta, \ldots$.

Definition 2.1 (Named multiset)**.** A *named multiset* Δ is a multiset over the product $\mathcal{N} \times \mathcal{F}$.

- The *support* of a named multiset Δ is the set of proper names α such that, there exists a formula $A \neq \bot$ such that (α, A) occurs in A
- A named multiset is *correct* is its support is a finished set.
- Similarly we define named multiset over the set of variable $\mathcal{V}$ as multiset over $\mathcal{V} \times \mathcal{F}$.

Representation of a named sequent. Given a named multiser $\Delta\ (\alpha_1, A_1), ..., (\alpha_n, A_n), (\epsilon, A)$ we denote Δ as $\alpha_1 : A_1, \ldots, \alpha_n : A_n\ \langle A \rangle$, Where A is the name associated to the empty name ϵ.

The pairs (α, A) are occurring in the sequent in the form $\alpha : A$ where α is a name and A a classical formula. A pair $\alpha : A$ or $x : A$ is called a *judgment*. If we read it as a function, this denotes the fact that to the name α the multiset Δ associate the formula A.

Definition 2.2 (Occurence equivalence)**.** Given Γ and Δ two named multisets ranging over $\mathcal{N} \times \mathcal{F}$. We say that they are occurence–equivalent if, $\Gamma(\alpha : A) \neq 0$ holds if and only if $\Delta(\alpha : A) \neq 0$ holds.

NB. Reasoning relatively to occurence equivalence, is the same as using sets instead of multisets.

Definition 2.3 (μ–equivalence)**.** Two named multisets Γ and Δ are *μ–equivalent* if, for any formula $A \neq \bot$ we have $\Gamma(\alpha : A) = \Gamma(\alpha : A)$. We denote the relation by $\Gamma \equiv_\mu \Delta$.

Definition 2.4 (Functional Multiset)**.** We say a multiset is *functional* if given a name α and a formula A if the following inequality holds $\Gamma(\alpha : A) \neq 0$ then $\Gamma(\alpha : B) = 0$ for any formula B that isn't A.

Definition 2.5 (Translation)**.** Given a multiset Γ we can translate it into a named multiset (assuming the set of names to be ordered or the axiom of choice)

$$A_1, ..., A_n \mapsto \alpha_1 : A_1, ..., \alpha_n : A_n\ \langle\rangle$$

We can also translate any named sequent into a regular one.

$$\alpha_1 : A_1, ..., \alpha_n : A_n\ \langle A \rangle \mapsto A_1, ..., A_n, A$$

Proposition 2.1. *The calculus **NK** is equivalent to* $\mathbf{NK}_\mu$.

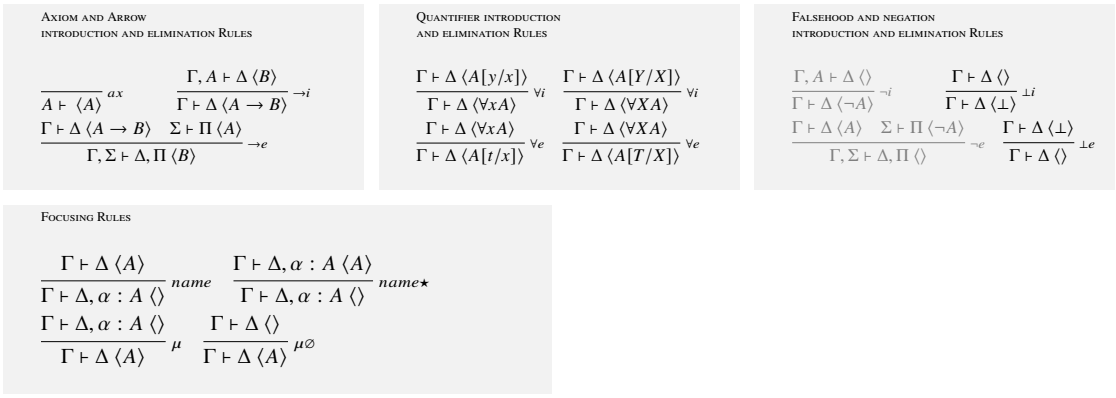


Figure 3: Classical Multiple–Conclusion Natural Deduction with names $\mathbf{NK}_\mu$ (as sequent calculus). It corresponds to a focused sequent calculus.

Proof. **1.** Our first step is to show that $\mathbf{NK}_\mu$ can simulate **NK**.

$$\frac{\Gamma \vdash \Delta}{\Gamma \vdash \Delta, A}\, wR \quad \mapsto \quad \frac{\Gamma \vdash \Delta \langle\rangle}{\Gamma \vdash \Delta \langle A\rangle}\, \mu* \qquad \frac{\Gamma \vdash \Delta, A, A}{\Gamma \vdash \Delta, A}\, cR \quad \mapsto \quad \frac{\dfrac{\Gamma \vdash \Delta, \alpha : A, \beta : A \langle\rangle}{\Gamma \vdash \Delta, \alpha : A \langle A\rangle}\, \mu}{\Gamma \vdash \Delta, \alpha : A \langle\rangle}\, name\star$$

2. Now we show that **NK** can simulate $\mathbf{NK}_\mu$. To do we mainly have to simulate the naming rules, this is quite similar to the previous step.

$$\frac{\Gamma \vdash \Delta \langle\rangle}{\Gamma \vdash \Delta \langle A\rangle}\, \mu* \quad \mapsto \quad \frac{\Gamma \vdash \Delta}{\Gamma \vdash \Delta, A}\, wR \qquad \frac{\Gamma \vdash \Delta, \alpha : A \langle A\rangle}{\Gamma \vdash \Delta, \alpha : A \langle\rangle}\, name\star \quad \mapsto \quad \frac{\Gamma \vdash \Delta, A, A}{\Gamma \vdash \Delta, A}\, cR$$

□

Proposition 2.2. *In* $\mathbf{NK}_\mu$ *the rules for falsehood (i.e.* $\bot$*) are redundant (up to sequent* μ*–equivalency).*

Proof. We show the redundancy using the following maps.

$$\frac{\Gamma \vdash \Delta \langle\rangle}{\Gamma \vdash \Delta \langle\bot\rangle}\, \bot_i \quad \mapsto \quad \frac{\Gamma \vdash \Delta \langle\rangle}{\Gamma \vdash \Delta \langle\bot\rangle}\, name\star \qquad \frac{\Gamma \vdash \Delta \langle\bot\rangle}{\Gamma \vdash \Delta \langle\rangle}\, \bot_e \quad \mapsto \quad \frac{\Gamma \vdash \Delta \langle\bot\rangle}{\Gamma \vdash \Delta, \alpha : \bot \langle\rangle}\, \mu\star$$

To conclude notice that the sequents $\Gamma \vdash \Delta, \alpha : \bot$; and $\Gamma \vdash \Delta$; are μ–equivalent. □

Proposition 2.3. *The rule* $name\star$ *can be simulated by the rules* $\{\mu, name\}$ *up to occurence equivalence.*

Proof. The proposition is straightforward, the key point is to see that

- The rules $\{\mu, name\}$ are compatible with occurence equivalence.
- the named sequents $\Gamma \vdash \Delta, \alpha : A \langle\rangle$ and $\Gamma \vdash \Delta, \alpha : A, \alpha : A \langle\rangle$ are occurence equivalent.

Then consider the following translation.

$$\frac{\Gamma \vdash \Delta, \alpha : A \langle A\rangle}{\Gamma \vdash \Delta, \alpha : A \langle\rangle}\, name\star \quad \mapsto \quad \frac{\Gamma \vdash \Delta, \alpha : A \langle A\rangle}{\Gamma \vdash \Delta, \alpha : A, \alpha : A \langle\rangle}\, name$$

□

From the previous proposition we have shown that $\mathbf{NK} \sim \mathbf{NK}_\mu$. And that $\mathbf{NK}_\mu \sim \mathbf{NK}_\mu*$. Where $\mathbf{NK}_\mu* = \{ax, \to_i, \to_e, \mu, \mu\varnothing, name\}$. We will use the subfragment $\mathbf{NK}_\mu*$ to develop the $\lambda\mu$–calculus in the next section.

2.2 Substitution and detour elimination in classical natural deduction

Definition 2.6 (Alive variable occurence)**.** A judgement $x : A$ is *alive* in a proof π of $\mathbf{NK}_\mu$ if the following inductive property holds:

- If $\pi = \dfrac{}{y : Y \vdash \langle Y \rangle}\, ax$ $\quad x : A$ is alive in π if and only if, $(x : A) = (y : Y)$.
- If $\pi = \dfrac{\begin{array}{c}\Vert\pi_0\\ \Gamma, y : Y \vdash \Delta \langle B \rangle\end{array}}{\Gamma \vdash \Delta \langle Y \to B \rangle} \to i$ $\quad x : A$ is alive in π if and only if, $(x : A) \neq (y : Y)$.
- If $\pi = \dfrac{\begin{array}{cc}\Vert\pi_1 & \Vert\pi_2\\ \Gamma \vdash \Delta \langle B \to C \rangle & \Sigma \vdash \Pi \langle B \rangle\end{array}}{\Gamma, \Sigma \vdash \Delta, \Pi \langle C \rangle} \to e$ $\quad x : A$ is alive in π if and only if, $x : A$ is alive in π_1 or π_2
- If π is of the form $\dfrac{\begin{array}{c}\Vert\pi_0\\ \Gamma \vdash \Delta \langle B \rangle\end{array}}{\Gamma \vdash \Delta, \beta : B \langle\rangle}\, name$, $\dfrac{\begin{array}{c}\Vert\pi_0\\ \Gamma \vdash \Delta, \beta : B \langle\rangle\end{array}}{\Gamma \vdash \Delta \langle B \rangle}\, \mu$ *or* $\dfrac{\begin{array}{c}\Vert\pi_0\\ \Gamma \vdash \Delta \langle\rangle\end{array}}{\Gamma \vdash \Delta \langle B \rangle}\, \mu\varnothing$ $\quad x : A$ is alive in π if and only if, $x : A$ is alive in the subproof π_0

An occurence that is not alive will be said to be *killed, dead* or *bounded*.

Definition 2.7 (Alive and dead leafs of a proof)**.** We define the set of *alive* leafs of a proof π – that we denote $\mathsf{al}(\pi)$ – in a similar fashion, inductively.

- If $\pi = \dfrac{}{y : Y \vdash \langle Y \rangle}\, ax$ $\quad$ Denoting n the only node of π we have $\mathsf{al}(\pi) = \{n\}$.
- If $\pi = \dfrac{\begin{array}{c}\Vert\pi_0\\ \Gamma, x : A \vdash \Delta \langle B \rangle\end{array}}{\Gamma \vdash \Delta \langle A \to B \rangle} \to i$ $\quad$ then $\mathsf{al}(\pi) = \mathsf{al}(\pi_0) \setminus \{n \in \mathsf{L}(\pi_0) \mid \ell(n) = x : A \vdash \langle A \rangle\}$
- If $\pi = \dfrac{\begin{array}{cc}\Vert\pi_1 & \Vert\pi_2\\ \Gamma \vdash \Delta \langle B \to C \rangle & \Sigma \vdash \Pi \langle B \rangle\end{array}}{\Gamma, \Sigma \vdash \Delta, \Pi \langle C \rangle} \to e$ $\quad$ then $\mathsf{al}(\pi) = \mathsf{al}(\pi_1) \cup \mathsf{al}(\pi_2)$.
- If π is of the form $\dfrac{\begin{array}{c}\Vert\pi_0\\ \Gamma \vdash \Delta \langle B \rangle\end{array}}{\Gamma \vdash \Delta, \beta : B \langle\rangle}\, name$, $\dfrac{\begin{array}{c}\Vert\pi_0\\ \Gamma \vdash \Delta, \beta : B \langle\rangle\end{array}}{\Gamma \vdash \Delta \langle B \rangle}\, \mu$ *or* $\dfrac{\begin{array}{c}\Vert\pi_0\\ \Gamma \vdash \Delta \langle\rangle\end{array}}{\Gamma \vdash \Delta \langle B \rangle}\, \mu\varnothing$ then simply $\mathsf{al}(\pi) = \mathsf{al}(\pi_0)$.

A leaf that is not a alive is said to be *dead* or *killed*.

Definition 2.8 (Free and bounded occurence)**.** We say that a leaf n is a *free occurence* of the judgement $x : A$ if n is labelled by $x : A \vdash \langle A \rangle$ and n is alive in π. On the other hand, We say that a leaf n is a *bounded occurence* of the judgement $x : A$ if n is labelled by $x : A \vdash \langle A \rangle$ and n is not alive in π.

We say that a judgement $x : A$ is *occuring free (resp. bounded)* in a proof π if there exists a leaf in π that is a free (resp. bounded) occurence of $x : A$

Proposition 2.4 (Judgement occuring on the left–side of the concluding sequent correspond to free occurences)**.** *If the judgement $x : A$ is occuring free in a proof π in $\mathbf{NK}_\mu$ if and only if, π proves a sequent of the form $\Gamma, x : A \vdash \Delta \langle B \rangle$.*

Proof. **1 → 2**. We prove this proposition by induction on π.

1. If π is an axiom, the only way to make it so that π contains a free occurence of $x : A$ is that π must be of the following form. In that case π indeed proves a sequent of the form $\Gamma, x : A \vdash \Delta \langle B \rangle$, where $\Gamma = \Delta = \emptyset$ and $A = B$.

$$\pi = \frac{}{x : A \vdash \langle A \rangle}\, ax$$

2. If the last rule of π is the introduciton of the implication, then it must be of the following form.

$$\pi = \frac{\begin{array}{c}\Big\| \pi_0 \\ \Gamma, y : Y \vdash \Delta \langle B \rangle\end{array}}{\Gamma \vdash \Delta \langle Y \to B \rangle} \to i$$

   By assumption $x : A$ is occuring free in π this means that $x : A$ must be occuring free in π_0. Therefore applying the induction hypothesis, $\Gamma, y : Y$ contains the judgement $x : A$. But, $(y : Y)$ must be distinct from $(x : A)$ since otherwise, $(x : A)$ would not be free in π. Therefore $x : A$ occurs in Γ i.e. in the left part of the conclusion–sequent of π
3. If the last rule in π is the elimination of the implication, π must be of the following form.

$$\pi = \frac{\begin{array}{cc}\Big\| \pi_1 & \Big\| \pi_2 \\ \Gamma \vdash \Delta \langle B \to C \rangle & \Sigma \vdash \Pi \langle B \rangle\end{array}}{\Gamma, \Sigma \vdash \Delta, \Pi \langle C \rangle} \to e$$

   Assuming that $x : A$ occurs free in π. By definition this means that $x : A$ occurs free in π_1 or π_2. Assuming it occurs free in π_1, applying the induction hypothesis, then Γ must contain the judgement $x : A$. So it follows that this is true aswell for the multiset Γ, Σ.
4. If π is of the form

$$\frac{\begin{array}{c}\Big\| \pi_0 \\ \Gamma \vdash \Delta \langle B \rangle\end{array}}{\Gamma \vdash \Delta, \beta : B \langle\rangle}\, name \quad , \quad \frac{\begin{array}{c}\Big\| \pi_0 \\ \Gamma \vdash \Delta, \beta : B \langle\rangle\end{array}}{\Gamma \vdash \Delta \langle B \rangle}\, \mu \quad or \quad \frac{\begin{array}{c}\Big\| \pi_0 \\ \Gamma \vdash \Delta \langle\rangle\end{array}}{\Gamma \vdash \Delta \langle B \rangle}\, \mu\varnothing$$

   then assuming $x : A$ occurs free in π means that it also occurs free in the subproof π_0. Applying the induction hypothesis the subproof π_0, we can ensure that Γ contains the judgement $x : A$, which allows us to conclude.

2 → 1. We also operate by induction.

1. Assume π is an axiom which has a conclusion of the form $\Gamma, x : A \vdash \Delta \langle B \rangle$. Then π may only be an axiom of the sequent $x : A \vdash \langle A \rangle$, in which by definition $x : A$ occurs free.
2. If the last rule in π is the introduction of the implication, then π is of the following form.

$$\pi = \frac{\begin{array}{c}\Big\| \pi_0 \\ \Gamma, y : Y \vdash \Delta \langle B \rangle\end{array}}{\Gamma \vdash \Delta \langle Y \to B \rangle} \to i$$

   assume that Γ contains an occurence of $x : A$, then clearly so does $\Gamma, y : Y$ (this also ensure that $(x : A) \neq (y : Y)$), applying then the induction hypothesis on π_0 we can ensure that $x : A$ occurs free in π_0 and so since $(x : A) \neq (y : Y)$ it also occurs free in π.

3. If the last rule in π is an elimination of implication, i.e.

$$\pi = \dfrac{\begin{array}{cc}\Vert\pi_1 & \Vert\pi_2\\ \Gamma \vdash \Delta\ \langle B \to C\rangle & \Sigma \vdash \Pi\ \langle B\rangle\end{array}}{\Gamma, \Sigma \vdash \Delta, \Pi\ \langle C\rangle}\ {\to}e$$

assuming that Γ, Σ contains an occurence of $x : A$, this ensure that either Γ or Δ contains the judgement $x : A$. Assume that $x : A$ is contained in Γ, by induction on π_1 then we can ensure that $x : A$ occurs free in π_1, and so by definition it follows that it also occurs free in π.

4. If π is of the form

$$\dfrac{\begin{array}{c}\Vert\pi_0\\ \Gamma \vdash \Delta\ \langle B\rangle\end{array}}{\Gamma \vdash \Delta, \beta : B\ \langle\rangle}\ name \quad , \quad \dfrac{\begin{array}{c}\Vert\pi_0\\ \Gamma \vdash \Delta, \beta : B\ \langle\rangle\end{array}}{\Gamma \vdash \Delta\ \langle B\rangle}\ \mu \quad or \quad \dfrac{\begin{array}{c}\Vert\pi_0\\ \Gamma \vdash \Delta\ \langle\rangle\end{array}}{\Gamma \vdash \Delta\ \langle B\rangle}\ \mu\varnothing$$

then assuming that Γ contains $x : A$ it follows that the statement is also true for the subproof π_0, we can therefore apply the induction hypothesis, claiming that $x : A$ occurs free in π_0, and by definition $x : A$ occurs free in π.

□

Definition 2.9 (Variable substitution for $\mathbf{NK}_\mu$ deductions)**.** Given a proof π a judgement $x : A$ and a proof π' of the sequent $\Sigma \vdash \Pi\ \langle A\rangle$. We define the operation of substitution $\pi[x : A/\pi']$ to be the identity if π does not contain a free occurence of $x : A$. Otherwise we define the substitution inductively.

1. $\left(\dfrac{}{x : A \vdash\ \langle A\rangle}\ ax\right)[x : A/\pi'] = \pi'$
2. $\left(\dfrac{\begin{array}{c}\Vert\pi_0\\ \Gamma, x : A, y : Y \vdash \Delta\ \langle B\rangle\end{array}}{\Gamma, x : A \vdash \Delta\ \langle Y \to B\rangle}\ {\to}i\right)[x : A/\pi'] = \dfrac{\begin{array}{c}\Vert\pi_0[x:A/\pi']\\ \Gamma, \Sigma, y : Y \vdash \Delta, \Pi\ \langle B\rangle\end{array}}{\Gamma, \Sigma \vdash \Delta, \Pi\ \langle Y \to B\rangle}\ {\to}i$
3. $\left(\dfrac{\begin{array}{cc}\Vert\pi_1 & \Vert\pi_2\\ \Gamma, x : A \vdash \Delta\ \langle B \to C\rangle & \Gamma', x : A \vdash \Delta'\ \langle B\rangle\end{array}}{\Gamma, \Gamma', x : A \vdash \Delta, \Delta'\ \langle C\rangle}\ {\to}e\right)[x : A/\pi'] = \dfrac{\begin{array}{cc}\Vert\pi_1[x:A/\pi'] & \Vert\pi_2[x:A/\pi']\\ \Gamma, \Sigma \vdash \Delta, \Pi\ \langle B \to C\rangle & \Gamma', \Sigma \vdash \Delta', \Pi\ \langle B\rangle\end{array}}{\Gamma, \Gamma', \Sigma \vdash \Delta, \Delta', \Pi\ \langle C\rangle}\ {\to}e$
4. $\left(\dfrac{\begin{array}{cc}\Vert\pi_1 & \Vert\pi_2\\ \Gamma, x : A \vdash \Delta\ \langle B \to C\rangle & \Gamma' \vdash \Delta'\ \langle B\rangle\end{array}}{\Gamma, \Gamma', x : A \vdash \Delta, \Delta'\ \langle C\rangle}\ {\to}e\right)[x : A/\pi'] = \dfrac{\begin{array}{cc}\Vert\pi_1[x:A/\pi'] & \Vert\pi_2\\ \Gamma, \Sigma \vdash \Delta, \Pi\ \langle B \to C\rangle & \Gamma' \vdash \Delta'\ \langle B\rangle\end{array}}{\Gamma, \Gamma', \Sigma \vdash \Delta, \Delta', \Pi\ \langle C\rangle}\ {\to}e$
5. $\left(\dfrac{\begin{array}{cc}\Vert\pi_1 & \Vert\pi_2\\ \Gamma \vdash \Delta\ \langle B \to C\rangle & \Gamma', x : A \vdash \Delta'\ \langle B\rangle\end{array}}{\Gamma, \Gamma', x : A \vdash \Delta, \Delta'\ \langle C\rangle}\ {\to}e\right)[x : A/\pi'] = \dfrac{\begin{array}{cc}\Vert\pi_1 & \Vert\pi_2[x:A/\pi']\\ \Gamma \vdash \Delta\ \langle B \to C\rangle & \Gamma', \Sigma \vdash \Delta', \Pi\ \langle B\rangle\end{array}}{\Gamma, \Gamma', \Sigma \vdash \Delta, \Delta', \Pi\ \langle C\rangle}\ {\to}e$
6. $\left(\dfrac{\begin{array}{c}\Vert\pi_0\\ \Gamma, x : A \vdash \Delta\ \langle B\rangle\end{array}}{\Gamma, x : A \vdash \Delta, \beta : B\ \langle\rangle}\ name\right)[\pi'/x] = \dfrac{\begin{array}{c}\Vert\pi_0[x:A/\pi']\\ \Gamma, \Sigma \vdash \Delta, \Pi, \Pi\ \langle B\rangle\end{array}}{\Gamma, \Sigma \vdash \Delta, \Pi, \beta : B\ \langle\rangle}\ name$
7. $\left(\dfrac{\begin{array}{c}\Vert\pi_0\\ \Gamma, x : A \vdash \Delta, \beta : B\ \langle\rangle\end{array}}{\Gamma, x : A \vdash \Delta\ \langle B\rangle}\ \mu\right)[x : A/\pi'] = \dfrac{\begin{array}{c}\Vert\pi_0[x:A/\pi']\\ \Gamma, \Sigma \vdash \Delta, \Pi, \Pi\ \langle B\rangle\end{array}}{\Gamma, \Sigma \vdash \Delta, \Pi, \beta : B\ \langle\rangle}\ name$
8. $\left(\dfrac{\begin{array}{c}\Vert\pi_0\\ \Gamma, x : A \vdash \Delta\ \langle\rangle\end{array}}{\Gamma, x : A \vdash \Delta\ \langle B\rangle}\ \mu\varnothing\right)[x : A/\pi'] = \dfrac{\begin{array}{c}\Vert\pi_0[x:A/\pi']\\ \Gamma, \Sigma \vdash \Delta, \Pi\ \langle\rangle\end{array}}{\Gamma, \Sigma \vdash \Delta, \Pi\ \langle B\rangle}\ \mu\varnothing$

Definition 2.10 (Structural substitution in $\mathbf{NK}_\mu$ proofs)**.** Given two proofs π , π' and a name α. We define the structural substitution inductively. If π proves a sequent$\Gamma \vdash \Delta \langle X \rangle$ such that Δ does not contain any assignement for α then $\pi[\alpha \bullet \pi'] = \pi$. Otherwise Δ contains a judgement $\alpha : A \to B$. and π' has for conclusion a sequent of the form $\Sigma \vdash \Pi \langle A \rangle$

1. If π is an axiom, it is unchanged by this substitution.
2. $$\left(\dfrac{\begin{array}{c}\Vdots\,\pi_0\\ \Gamma, y : Y \vdash \Delta, \alpha : A \to B \langle Z \rangle\end{array}}{\Gamma \vdash \Delta, \alpha : A \to B \langle Y \to Z \rangle} \to i\right)[\alpha \bullet \pi'] = \dfrac{\begin{array}{c}\Vdots\,\pi_0[\alpha \bullet \pi']\\ \Gamma, y : Y, \Sigma \vdash \Delta, \Pi, \alpha : B \langle Z \rangle\end{array}}{\Gamma\Sigma \vdash \Delta, \Pi, \alpha : B \langle Y \to Z \rangle} \to i$$
3. $$\left(\dfrac{\begin{array}{cc}\Vdots\,\pi_1 & \Vdots\,\pi_2\\ \Gamma \vdash \Delta, \alpha : A \to B \langle X \to Y \rangle & \Theta \vdash \Psi, \alpha : A \to B \langle X \rangle\end{array}}{\Gamma, \Theta \vdash \Delta, \Psi, \alpha : A \to B \langle Y \rangle} \to e\right)[\alpha \bullet \pi'] = \dfrac{\begin{array}{cc}\Vdots\,\pi_1[\alpha \bullet \pi'] & \Vdots\,\pi_2[\alpha \bullet \pi']\\ \Gamma, \Sigma \vdash \Delta, \Pi, \alpha : B \langle X \to Y \rangle & \Theta, \Sigma \vdash \Psi, \Pi, \alpha : B \langle X \rangle\end{array}}{\Gamma, \Theta, \Sigma \vdash \Delta, \Psi, \Pi, \alpha : B \langle Y \rangle} \to e$$
4. $$\left(\dfrac{\begin{array}{c}\Vdots\,\pi_0\\ \Gamma \vdash \Delta, \alpha : A \to B \langle X \rangle\end{array}}{\Gamma \vdash \Delta, \alpha : A \to B, \beta : X \langle\rangle}\, name\right)[\alpha \bullet \pi'] = \dfrac{\begin{array}{c}\Vdots\,\pi_0[\alpha \bullet \pi']\\ \Gamma, \vdash \Delta, \Pi, \alpha : B \langle X \rangle\end{array}}{\Gamma, \Sigma \vdash \Delta, \Pi, \alpha : B, \beta : X \langle\rangle}\, name$$
5. $$\left(\dfrac{\dfrac{\begin{array}{c}\Vdots\,\pi_0\\ \Gamma \vdash \Delta, \alpha : A \to B \langle A \to B \rangle\end{array}}{\Gamma \vdash \Delta, \alpha : A \to B, \alpha : A \to B \langle\rangle}\, name}{}\right)[\alpha \bullet \pi'] = \dfrac{\dfrac{\begin{array}{cc}\Vdots\,\pi_0[\alpha \bullet \pi'] & \Vdots\,\pi'\\ \Gamma, \Sigma \vdash \Delta, \Pi, \alpha : B \langle A \to B \rangle & \Sigma \vdash \Pi \langle A \rangle\end{array}}{\Gamma, \Sigma \vdash \Delta, \Pi, \alpha : B \langle B \rangle}\, app}{\Gamma, \Sigma \vdash \Delta, \Pi, \alpha : B, \alpha : B \langle\rangle}\, name$$
6. $$\left(\dfrac{\dfrac{\begin{array}{c}\Vdots\,\pi_0\\ \Gamma \vdash \Delta \langle A \to B \rangle\end{array}}{\Gamma \vdash \Delta, \alpha : A \to B \langle\rangle}\, name}{}\right)[\alpha \bullet \pi'] = \dfrac{\dfrac{\begin{array}{cc}\Vdots\,\pi_0[\alpha \bullet \pi'] & \Vdots\,\pi'\\ \Gamma, \Sigma \vdash \Delta, \Pi \langle A \to B \rangle & \Sigma \vdash \Pi \langle A \rangle\end{array}}{\Gamma, \Sigma \vdash \Delta, \Pi \langle B \rangle}\, app}{\Gamma, \Sigma \vdash \Delta, \Pi, \alpha : B \langle\rangle}\, name$$
7. $$\left(\dfrac{\begin{array}{c}\Vdots\,\pi_0\\ \Gamma \vdash \Delta, \alpha : A \to B, \beta : X \langle\rangle\end{array}}{\Gamma \vdash \Delta, \alpha : A \to B \langle X \rangle}\, \mu\right)[\alpha \bullet \pi'] = \dfrac{\begin{array}{c}\Vdots\,\pi_0[\alpha \bullet \pi']\\ \Gamma, \Sigma \vdash \Delta, \alpha : B, \beta : X \langle\rangle\end{array}}{\Gamma, \Sigma \vdash \Delta, \Pi, \alpha : B \langle X \rangle}\, \mu$$
8. $$\left(\dfrac{\begin{array}{c}\Vdots\,\pi_0\\ \Gamma \vdash \Delta, \alpha : A \to B \langle\rangle\end{array}}{\Gamma \vdash \Delta \langle A \to B \rangle}\, \mu\right)[\alpha \bullet \pi'] = \dfrac{\begin{array}{c}\Vdots\,\pi_0\\ \Gamma \vdash \Delta, \alpha : A \to B \langle\rangle\end{array}}{\Gamma \vdash \Delta \langle A \to B \rangle}\, \mu$$
9. $$\left(\dfrac{\begin{array}{c}\Vdots\,\pi_0\\ \Gamma \vdash \Delta, \alpha : A \to B \langle\rangle\end{array}}{\Gamma \vdash \Delta, \alpha : A \to B \langle X \rangle}\, \mu\varnothing\right)[\alpha \bullet \pi'] = \dfrac{\begin{array}{c}\Vdots\,\pi_0[\alpha \bullet \pi']\\ \Gamma, \Sigma \vdash \Delta, \Pi, \alpha : B \langle\rangle\end{array}}{\Gamma, \Sigma \vdash \Delta, \alpha : B \langle X \rangle}\, \mu\varnothing$$

Last rule of a proof of an implication. Note that in the $\mathbf{NK}_\mu$ sequent calculus, considering a proof π of a sequent of the form $\Gamma \vdash \Delta \langle A \to B \rangle$. The last rule of π is restricted to a certain type of rules, this is because the introduction and elimination rules of $\mathbf{NK}_\mu$ only operate on unnamed i.e. focused formula's. For instance the last rule may not be an introduction rule that is not the introduction of the implication rule i.e. $\to_i$, since in such cases the focused formula cannot be an implication. Also, it cannot be a rule that results in sequent having no focused formula (or equivalently focusing $\bot$). This means it may not be a the elimination of falsehood $\bot_e$ or a naming rule (i.e. unfocusing rule), as these rules result in sequents without a focused formula. From this case analysis we can in fact conclude that the last rule in π may only be of one of the following.

$$\dfrac{\Gamma, x : A \vdash \Delta \langle B \rangle}{\Gamma \vdash \Delta \langle A \to B \rangle} \to i \qquad \dfrac{\Gamma \vdash \Delta \langle\rangle}{\Gamma \vdash \Delta \langle A \to B \rangle}\, \mu\varnothing \qquad \dfrac{\Gamma \vdash \Delta, \alpha : A \to B, \langle\rangle}{\Gamma \vdash \Delta \langle A \to B \rangle}\, \mu$$

Notation. Given a proof π of a sequent $\Gamma \vdash \Delta, \alpha : A\ \langle\rangle$ we denote by $\mu\alpha.\pi$ the proof

$$\dfrac{\overset{\pi}{\triangledown}\quad \Gamma \vdash \Delta, \alpha : A\ \langle\rangle}{\Gamma \vdash \Delta\ \langle A\rangle}\ \mu$$

Definition 2.11 (Detour elimination)**.** We define the *elementary detour elimination* $\rightsquigarrow_0$ as the following binary relation on $\mathbf{NK}_\mu$ proofs.

$$\dfrac{\overset{\pi'}{\triangledown}\quad \Sigma \vdash \Pi\ \langle A\rangle \qquad \dfrac{\overset{\pi}{\triangledown}\quad \Gamma, x : A \vdash \Delta\ \langle B\rangle}{\Gamma \vdash \Delta\ \langle A \to B\rangle}\ {\to}i}{\Gamma, \Sigma \vdash \Delta, \Pi\ \langle B\rangle}\ {\to}e \quad \rightsquigarrow \quad \pi[x : A/\pi']$$

$$\dfrac{\overset{\pi'}{\triangledown}\quad \Sigma \vdash \Pi\ \langle A\rangle \qquad \dfrac{\overset{\pi}{\triangledown}\quad \Gamma \vdash \Delta\ \langle\rangle}{\Gamma \vdash \Delta\ \langle A \to B\rangle}\ \mu\varnothing}{\Gamma, \Sigma \vdash \Delta, \Pi\ \langle B\rangle}\ {\to}e \quad \rightsquigarrow \quad \dfrac{\overset{\pi}{\triangledown}\quad \Gamma \vdash \Delta\ \langle\rangle}{\Gamma \vdash \Delta\ \langle B\rangle}\ \mu\varnothing$$

$$\dfrac{\overset{\pi'}{\triangledown}\quad \Sigma \vdash \Pi\ \langle A\rangle \qquad \dfrac{\overset{\pi}{\triangledown}\quad \Gamma \vdash \Delta, \alpha : A \to B\ \langle\rangle}{\Gamma \vdash \Delta\ \langle A \to B\rangle}\ \mu}{\Gamma, \Sigma \vdash \Delta, \Pi\ \langle B\rangle}\ {\to}e \quad \rightsquigarrow \quad \mu\alpha.\pi[\alpha \bullet \pi']$$

The relation of *detour elimination* denoted $\rightsquigarrow$ then corresponds to the compatible closure of $\rightsquigarrow_0$ (i.e. the sub–tree closure).

2.3 The $\lambda\mu$–calculus

We are given a set of variables $\mathcal{V}$ and a set of names $\mathcal{N}$. For each formula A, $\varnothing_A$ is a special name belonging to $\mathcal{N}$. As a convention we make the variables range over $x, y, z, \ldots$ and the names range over $\alpha, \beta, \gamma, \ldots$. The *simple terms* or *λ–terms* of the $\lambda\mu$–calculus are defined as the regular ones from the λ–calculus. *Guarded terms* or *named terms* are terms of the form $[\alpha]t$ where t is a $\lambda\mu$–term and α is a name. Finally we add the *μ–abstraction* as terms $\mu\alpha.e$ where e is a guarded term. We will make guarded terms range over $e, f, g, \ldots$ and the $\lambda\mu$–terms range over $u, v, t, \ldots$.

$$t, u \quad := \quad x \quad | \quad \lambda x.t \quad | \quad (t)u \quad | \quad [\alpha]t \quad | \quad \mu\alpha.e \quad | \quad \mu\varnothing_A.e$$

Here we have the inequality $\alpha \neq \varnothing$. The reductions of the $\lambda\mu$–calculus are of three kind *logical, structural, or renaming*.

$$\dfrac{}{(\lambda x.u)v \to u[v/x]}\ log \qquad \dfrac{}{(\mu\beta.u)v \to \mu\beta.u[[\beta]wv/[\beta]w]}\ str \qquad \dfrac{}{(\mu\varnothing_{A\to B}.u)v \to \mu\varnothing_B.u}\ str \qquad \dfrac{}{[\alpha]\mu\beta.u \to u[\alpha/\beta]}\ nam$$

We also have the usual rules describing how the $\lambda\mu$–reduction can be passed from a subterm to a term (this is known as the *compatible closure*).

$$\dfrac{(t)u \to (t')u}{t \to t'} \qquad \dfrac{(t)u \to (t)u'}{u \to u'} \qquad \dfrac{\lambda x.t \to \lambda x.t'}{t \to t'} \qquad \dfrac{[\alpha]t \to [\alpha]t'}{t \to t'} \qquad \dfrac{\mu\alpha.e \to \mu\alpha.e'}{e \to e'} \qquad \dfrac{\mu\varnothing.e \to \mu\varnothing.e'}{e \to e'}$$

$$\frac{}{x_A : A \vdash \langle x : A\rangle}\ var \qquad \frac{\Gamma, x : A \vdash \Delta,\ \langle t : B\rangle}{\Gamma \vdash \Delta,\ \langle \lambda x.t : A \to B\rangle}\ abs \qquad \frac{\Gamma \vdash \Delta,\ \langle t : A \to B\rangle \quad \Sigma \vdash \Pi,\ \langle u : A\rangle}{\Gamma, \Sigma \vdash \Delta, \Pi,\ \langle (t)u : B\rangle}\ app$$

$$\frac{\Gamma \vdash \Delta,\ \langle t : A\rangle}{\Gamma \vdash \Delta, \alpha : A,\ \langle [\alpha]t : \bot\rangle}\ name \qquad \frac{\Gamma \vdash \Delta, \alpha : A,\ \langle e : \bot\rangle}{\Gamma \vdash \Delta,\ \langle \mu\alpha.e : A\rangle}\ \mu \qquad \frac{\Gamma \vdash \Delta,\ \langle e : \bot\rangle}{\Gamma \vdash \Delta,\ \langle \mu\varnothing_A.e : A\rangle}\ \mu\varnothing$$

Figure 4: Typing rules for the $\lambda\mu$–calculus, this type system is denoted $\Lambda\mu^{\to}$.

The μ operator and control operators The curry–howard correspondence for classical logic was discovered years later the correspondence between intuitionistic logic and simply typed lambda calculus. This correspondence between classical logic and $\lambda\mu$–calculus, is constructed thanks to the addition of naming schemes and the μ–operator to the regular λ–calculus. Computationally the μ operator is understood has a *control operator* [SU10; Par92a; Mim20], while from the logic point of view it allows us to perform a cut after having performed contractions/weakenings on the right side of a sequent. A regular example of control operator is the call/cc operator, called *call–with–current–continuation*. Note that it is possible to translate the $\lambda\mu$–calculus into the λ–calculus, this enlights the meaning of the μ–abstraction and of control operators, as P. de Groote states in his paper [Gro94], a μ-abstraction is interpreted as a λ-abstraction whose bound variable stands for some possible continuation. The existence of this translation stands for the untyped term of the $\lambda\mu$–term, it does not mean that the simply typed terms of the $\lambda\mu$–calculus are the same as the simply typed term of the λ–calculus, since this would mean the provability of the intuitionistic and classical logic are the same, which we know to be false. Although we know we can embed classical logic into intuitionistic using the Kolmogorov translation, that is double negation. This phenomenon is also present in the translation given by P. de Groote in [Gro94] of $\lambda\mu$–calculus into λ–calculus.

Definition 2.12 (Simply typable $\lambda\mu$–term)**.** A $\lambda\mu$–term t will be said to be *simply typed* or *simply typable* (or if there is no ambiguity merely *typable*), if there exists a derivation in the type system of the previous figure of a sequent $\Gamma \vdash \Delta,\ \langle t : A\rangle$, where A is a propositional formula.

We will denote by $\lambda\mu^{\to}$ the set of simply typed lambda terms of the $\lambda\mu$–calculus.

Variable and Structural Substitutions. In the $\lambda\mu$–calculus two types of substitution exists, each corresponding the respective detour elimination in the classical natural deduction $\mathbf{NK}_\mu$. To define the substitutions we will proceed by induction. For the variable substitution, we proceed like J.L. Krivine does in his book [Kri02] (although we consider only substitutions with one parameter instead of n). To define the *structural substitution* we proceed by induction similarly to [GKM13].

Definition 2.13 (Variable substitution)**.** Given a term u and a variable x we define the substitution $t[u/x]$ in some term t inductively.

1. $x[u/x] = u$, and $y[u/x] = y$.
2. $(\lambda y.t)[u/x] = \lambda y.(t[u/x])$.
3. $(t)v[u/x] = (t[u/x])v[u/x]$
4. $([\alpha]t)[u/x] = [\alpha](t[u/x])$
5. $(\mu\alpha.t)[u/x] = \mu\alpha.t[u/x]$
6. $(\mu\varnothing_{A\to B}.t)[u/x] = \mu\varnothing_{A\to B}.t[u/x]$

Definition 2.14 (Structural Substitution)**.** Given a term u of type A and α a name. We define the name–application $t[\alpha \bullet u]$ inductively as follow

1. $x[\alpha \bullet u] = x$.
2. $(\lambda y.t)[\alpha \bullet u] = \lambda y.(t[\alpha \bullet u])$.

3. $(t)v[u/x] = (t[\alpha \bullet u])v[\alpha \bullet u]$
4. $([\alpha]t)[\alpha \bullet u] = [\alpha]((t[\alpha \bullet u])u)$
5. $(\mu\alpha.t)[\alpha \bullet u] = \mu\alpha.t$ and $(\mu\beta.t)[\alpha \bullet u] = \mu\beta.t[\alpha \bullet u]$.
6. $(\mu\varnothing_{A\to B}.t)[\alpha \bullet u] = \mu\varnothing_{A\to B}.t[\alpha \bullet u]$

Definition 2.15 (Free variables, free names.)**.** Given a term t, we denote the set of its free variables by $FV(t)$, and the set of its free names by $FN(t)$, we define these sets inductively:

- $FV(x) = \{x\}$ and $FN(x) = \varnothing$.
- $FV(\lambda x.t) = FV(t) \setminus \{x\}$ and $FN(\lambda x.t) = FN(t)$.
- $FV((t)u) = FV(t) \cup FV(u)$ and $FN((t)u) = FN(t) \cup FN(u)$.
- $FV([\alpha]t) = FV(t)$ and $FN([\alpha]t) = FN(t) \cup \{\alpha\}$
- $FV(\mu\alpha t) = FV(t)$ and $FN(\mu\alpha t) = FN(t) \setminus \{\alpha\}$
- $FV(\mu\varnothing_A t) = FV(t)$ and $FN(\mu\varnothing_A t) = FN(t)$

Proposition 2.5 (Derivable sequent in $\Lambda\mu^{\to}$)**.** *Given t a term of the $\lambda\mu$–calculus. If t is simply typable then for each derivation in the $\lambda\mu$ type system of root $\Gamma \vdash \Delta\ \langle t : A\rangle$, Γ corresponds to the types of the free variable in t, and Δ corresponds to the types of the free names of t.*

Proof. To do so we reason by induction on t.

1. If $t = x$ is a variable. Then a type derivation of x is of the following form.

$$\frac{}{x : A \vdash \langle x : A\rangle}\ ax$$

   Such a derivation indeed has the property we seek.
2. If $t = (u)v$ is an application. A derivation of t must be of the following form.

$$\frac{\begin{array}{cc}\Big\Vert \pi_u & \Big\Vert \pi_v \\ \Gamma \vdash \Delta\ \langle u : A \to B\rangle & \Sigma \vdash \Pi\ \langle v : A\rangle\end{array}}{\Gamma, \Sigma \vdash \Delta, \Pi\ \langle (u)v : B\rangle}\ app$$

   We call the induction hypothesis on u and v and doing so we can claim that π_u and π_v are such that Γ (resp. Σ) correspond to the free variable of u (resp. v). Δ (resp. Π) correspond to the named subterms of u (resp. v). From there we merely remark that the free variable of $(u)v$ corresponds to the free variable of u and v, but also, that the named subterm of $(u)v$ are the one of u and v.
3. If $t = \lambda x.u$ is an abstraction. Then a derivation of t must be of the following form.

$$\frac{\Big\Vert \pi_u \\ \Gamma, x : A \vdash \Delta\ \langle u : B\rangle}{\Gamma \vdash \Delta\ \langle \lambda x.u : A \to B\rangle}\ abs$$

   Using our induction hypothesis we can deduce that Γ, A^x corresponds to the types of the free variable in u, now note that in this case Γ corresponds to the types of the free variable in $\lambda x.u$. Also we note that the named subterm of in $\lambda x.u$ and u are the same.
4. If $t = [\alpha]u$ is a named term. Then a derivation of t may be of two forms.

$$\frac{\Big\Vert \pi_u^1 \\ \Gamma \vdash \Delta\ \langle u : A\rangle}{\Gamma \vdash \Delta, \alpha : A\ \langle [\alpha]u : \bot\rangle}\ name \qquad \frac{\Big\Vert \pi_u^2 \\ \Gamma \vdash \Delta, \alpha : A\ \langle u : A\rangle}{[\Gamma \vdash \Delta, \alpha : A\ \langle [\alpha]u : \bot\rangle}\ name\star$$

We apply the induction on u. And two case occur then, if u contains subterm named by α the its typing derivation must be π_u^2. Now we just note that the free variables in u and $[\alpha]u$ are the same, and that the named subterm in u and $[\alpha]u$ are of the same type.
If, on the other hand, u does not contain any α–names subterm then its corresponding type derivation if π_u^1. Then we note that the free variables in u and $[\alpha]u$ are the same, and that the named subterm in u and $[\alpha]u$ differs only in the fact that $[\alpha]u$ contains a subterm named by α.

5. If $t = \mu\alpha.e$ is a μ–abstraction, then its derivation may be of the following two forms.

$$\frac{\begin{array}{c}\Big\| \pi_1 \\ \Gamma \vdash \Delta, \alpha : A \ \langle e : \bot \rangle\end{array}}{\Gamma \vdash \Delta;\ \langle \mu\alpha.e : A \rangle}\ unname \qquad \frac{\begin{array}{c}\Big\| \pi_2 \\ \Gamma \vdash \Delta,\ \langle e : \bot \rangle\end{array}}{\Gamma \vdash \Delta;\ \langle \mu\varnothing_A.e : A \rangle}\ unname$$

In both cases we obtain a smaller subterm on which we can apply induction. In both case notice that Γ is unchanged, but also that the free variables in $\mu\alpha.e$ and e are the same.
Now let us treat the case of named subterm. In the first case, by induction the types of the occuring name in e are given by the map Δ, A^α. Now by definition the type of the named subterm in $\mu\alpha.e$ are in Δ (i.e. we removed the type of the name α, as it is no longer free).
In the second case the types of the name is Δ for e and e has no subterm named by α. Therefore the abstraction $\mu\alpha.e$ has the same name–to–type map.

□

2.4 The Isomorphism

We want to point out an important detail: to put in light in an easier manner the isomorphism, it is easier to consider the typing of the terms *a la Curry*. Meaning each variable is attached to a type and so a variable x of type A is written as x_A. Thanks to this we can identify a term with its typing derivation in the typing system from the previous figure. This is not a new way of presenting the isomorphism, in fact J.Y. Girard presents the Curry Howard correspondence in the same way in his book Proofs and Types [GTL89].

If we do not type 'a la Curry' we cannot identify one derivation to one term as indeed: any variable x could correspond to any axiom of some type A. But, in that case since there is an infinite amount of types, we could not construct a *canonical* surjective map from $\lambda\mu$–terms to $\mathbf{NK}_\mu$ derivations.

Proposition 2.6 (Canonical Maps)**.** *We can exhibit and define two canonical invertible maps.*

1. *There is a canonical bijection $[\![\cdot]\!]_1$ between type derivations of $\lambda\mu^{\to}$ and proofs in the sequent calculus $\mathbf{NK}_\mu$.*
2. *There exists a canonical bijection $[\![\cdot]\!]_2$ wich maps each simply typed $\lambda\mu$–terms t to a type derivation of $\Gamma \vdash \Delta\ \langle t : A \rangle$. Where Γ is the typing set of the free variables occuring in t and Δ is the typing set of the free names occuring in t.*

Proof. First we show how to map each type derivation to proof in $\mathbf{NK}_\mu$. The process is straightforward.

$$\frac{}{x_A : A \vdash \langle x : A \rangle}\ var \ \mapsto\ \frac{}{x : A \vdash \langle A \rangle}\ ax$$

$$\frac{\Gamma, x : A \vdash \Delta,\ \langle t : B \rangle}{\Gamma \vdash \Delta,\ \langle \lambda x.t : A \to B \rangle}\ abs \ \mapsto\ \frac{\Gamma, x : A \vdash \Delta,\ \langle B \rangle}{\Gamma \vdash \Delta,\ \langle A \to B \rangle}\ {\to}i$$

$$\frac{\Gamma \vdash \Delta,\ \langle t : A \to B \rangle \quad \Sigma \vdash \Pi,\ \langle u : A \rangle}{\Gamma, \Sigma \vdash \Delta, \Pi,\ \langle (t)u : B \rangle}\ app \ \mapsto\ \frac{\Gamma \vdash \Delta,\ \langle A \to B \rangle \quad \Sigma \vdash \Pi,\ \langle A \rangle}{\Gamma, \Sigma \vdash \Delta, \Pi,\ \langle B \rangle}\ {\to}e$$

$$\frac{\Gamma \vdash \Delta,\ \langle e : \bot \rangle}{\Gamma \vdash \Delta,\ \langle \mu\varnothing_A.e : A \rangle}\ \mu\varnothing \ \mapsto\ \frac{\Gamma \vdash \Delta,\ \langle\rangle}{\Gamma \vdash \Delta,\ \langle A \rangle}\ \mu\varnothing$$

$$\frac{\Gamma \vdash \Delta,\ \langle t : A \rangle}{\Gamma \vdash \Delta, \alpha : A,\ \langle [\alpha]t : \bot \rangle}\ name \ \mapsto\ \frac{\Gamma \vdash \Delta,\ \langle A \rangle}{\Gamma \vdash \Delta, \alpha : A,\ \langle\rangle}\ name$$

$$\frac{\Gamma \vdash \Delta, \alpha : A,\ \langle e : \bot \rangle}{\Gamma \vdash \Delta,\ \langle \mu\alpha.e : A \rangle}\ \mu \ \mapsto\ \frac{\Gamma \vdash \Delta, \alpha : A,\ \langle\rangle}{\Gamma \vdash \Delta,\ \langle A \rangle}\ \mu$$

Inductively one can show that the map is injective, starting with the base case and doing each induction step. The same goes for surjectivity.

Now we show there exists an invertible map from $\lambda\mu$–terms to type derivations. Again we define it by induction, this time on the size of the term.

$$x_A \mapsto \frac{}{x_A : A \vdash \langle x : A\rangle}\, var$$

$$\lambda x.t \mapsto \frac{\Gamma, x : A \vdash \Delta,\ \langle t : B\rangle}{\Gamma \vdash \Delta,\ \langle \lambda x.t : A \to B\rangle}\, abs$$

$$(t)u \mapsto \frac{\Gamma \vdash \Delta,\ \langle t : A \to B\rangle \quad \Sigma \vdash \Pi,\ \langle u : A\rangle}{\Gamma, \Sigma \vdash \Delta, \Pi,\ \langle (t)u : B\rangle}\, app$$

$$\mu\varnothing_A.e \mapsto \frac{\Gamma \vdash \Delta,\ \langle e : \bot\rangle}{\Gamma \vdash \Delta,\ \langle \mu\varnothing_A.e : A\rangle}\, \mu\varnothing$$

$$[\alpha]t \mapsto \frac{\Gamma \vdash \Delta,\ \langle t : A\rangle}{\Gamma \vdash \Delta, \alpha : A,\ \langle [\alpha]t : \bot\rangle}\, name$$

$$\mu\alpha.e \mapsto \frac{\Gamma \vdash \Delta, \alpha : A,\ \langle e : \bot\rangle}{\Gamma \vdash \Delta,\ \langle \mu\alpha.e : A\rangle}\, \mu$$

The inverse of this map is merely the map that given a derivation returns its root term. □

NB. From the previous proposition, we can immediatly conclude that there also exists an invertible map $[\![\cdot]\!] = [\![\cdot]\!]_2 \circ [\![\cdot]\!]_1$ from the simply typed $\lambda\mu$–term to the derivations in $\mathbf{NK}_\mu$.

Corollary 2.0.1 (Free occurences)**.** *Given t a $\lambda\mu$–term, we have the equivalency:*

1. *the variable x_A occurs free in t if and only if the variable x_A occurs free in $[\![t]\!]$.*
2. *the name α occurs free in t if and only if the name α occurs free in $[\![t]\!]$.*

Proof. This is a direct application of the previous propositions 2.6 and 2.5. □

Definition 2.16 (Reduction for type derivations.)**.** We define a relation for typing derivations, given two derivations $\mathcal{D}$ and $\mathcal{D}'$ we have $\mathcal{D} \rightsquigarrow \mathcal{D}'$ if $[\![\mathcal{D}]\!]_1 \rightsquigarrow [\![\mathcal{D}']\!]_2$.

Lemma 2.1 (Substitutions compatibility)**.** *Given two $\lambda\mu$–terms t and u, $[\![t[u/x]]\!]_2 = [\![t]\!]_2[[\![u]\!]_2/x]$ and $[\![t[\alpha \bullet u]]\!]_2 = [\![t]\!]_2[\alpha \bullet [\![u]\!]_2]$.*

Proof. See the proof A in the appendix. □

Theorem 2.1 (Isomorphism)**.** *The maps $[\![\cdot]\!]_1$ and $[\![\cdot]\!]_2$ are isomorphisms, meaning:*

1. *Given two proofs π, ρ in $\mathbf{NK}_\mu$ the reduction $\pi \rightsquigarrow \rho$ holds if and only if $[\![\pi]\!]_1 \rightsquigarrow [\![\rho]\!]_1$*
2. *Given two $\lambda\mu$–terms t and u the reduction $t \to_\beta u$ holds if and only if, $[\![t]\!]_2 \rightsquigarrow [\![u]\!]_2$*

Proof. **1.** This follows directly from the definition of $\rightsquigarrow$ on the derivation in the type system $\Lambda\mu^{\to}$.

2. For this we operate by case analysis on the definition of the reduction in the $\lambda\mu$–calculus.

1. If the reduction if of the form $(\lambda x.t)u \to t[u/x]$ Since we know by definition that $[\![(\lambda x.t)u]\!] \rightsquigarrow [\![t]\!][[\![u]\!]/x]$, and that thanks to the previous lemma we can ensure $[\![t]\!][[\![u]\!]/x] = [\![t[u/x]]\!]$ we can conclude.
2. If the reduction if of the form $(\mu\alpha.e)u \to \mu\alpha.e[\alpha \bullet u]$ Since we know by definition that $[\![(\mu\alpha.e)u]\!] \rightsquigarrow \mu\alpha.[\![e]\!][\alpha \bullet [\![u]\!]]$, and that thanks to the previous lemma we can ensure, $\mu\alpha.[\![e]\!][\alpha \bullet [\![u]\!]] = [\![\mu\alpha.e[\alpha \bullet u]]\!]$ and so we can conclude.
3. If the reduction if of the form $(\mu\varnothing_{A\to B}.e)u \to \mu\varnothing_B.e$ Notice that by definition we have:

$$[\![(\mu\varnothing_{A\to B}.e)u]\!]_2 = \frac{\dfrac{\Vert}{\Gamma \vdash \Delta\ \langle e : \bot\rangle}\, \mu\varnothing \qquad \dfrac{\Vert}{\Sigma \vdash \Pi\ \langle A\rangle}}{\Gamma, \Sigma \vdash \Delta, \Pi\ \langle (\mu\varnothing_{A\to B}.e)u : B\rangle}\, app \qquad [\![\mu\varnothing_B.e]\!]_2 = \frac{\dfrac{\Vert}{\Gamma \vdash \Delta\ \langle e : \bot\rangle}}{\Gamma \vdash \Delta\ \langle \mu\varnothing_B.e : B\rangle}\, \mu\varnothing$$

To conclude merely remark that by definition $[\![(\mu\varnothing_{A\to B}.e)u]\!]_2 \rightsquigarrow [\![\mu\varnothing_B.e]\!]_2$

□

This allows us to conclude the main theorem relating the model of computation that is the $\lambda\mu$–calculus and the classical natural deduction with names $\mathbf{NK}_\mu$.

Theorem 2.2 (Parigot (1992))**.** *The map $[\![\cdot]\!]$ is a bijection from simply typable $\lambda\mu$–terms to derivations in $\mathbf{NK}_\mu$.*

And $[\![\cdot]\!]$ is such that, given two $\lambda\mu$–terms t and u, $t \to u$ if and only if $[\![t]\!] \rightsquigarrow [\![u]\!]$

Proof. Simply using the previous theorem, recalling that the composition of isomorphism is an isomorphism. □

Bibliography

[GTL89] Jean-Yves Girard, Paul Taylor, and Yves Lafont. *Proofs and Types*. USA: Cambridge University Press, 1989. ISBN: 0521371813.

[Gri89] Timothy G. Griffin. "A Formulae-as-Type Notion of Control". In: *Proceedings of the 17th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*. POPL '90. San Francisco, California, USA: Association for Computing Machinery, 1989, pp. 47–58. ISBN: 0897913434. DOI: 10.1145/96709.96714. URL: https://doi.org/10.1145/96709.96714.

[FH92] Matthias Felleisen and Robert Hieb. "The Revised Report on the Syntactic Theories of Sequential Control and State". In: *Theor. Comput. Sci.* 103 (1992), pp. 235–271.

[Par92a] Michel Parigot. "$\lambda\mu$-Calculus: An algorithmic interpretation of classical natural deduction". In: *Logic Programming and Automated Reasoning*. Ed. by Andrei Voronkov. Berlin, Heidelberg: Springer Berlin Heidelberg, 1992, pp. 190–201. ISBN: 978-3-540-47279-7.

[Par92b] Michel Parigot. "Free deduction: An analysis of "Computations" in classical logic". In: *Logic Programming*. Ed. by A. Voronkov. Berlin, Heidelberg: Springer Berlin Heidelberg, 1992, pp. 361–380. ISBN: 978-3-540-47083-0.

[Gro94] Philippe de Groote. "A CPS-translation of the $\lambda\mu$-calculus". In: *Trees in Algebra and Programming — CAAP'94*. Ed. by Sophie Tison. Berlin, Heidelberg: Springer Berlin Heidelberg, 1994, pp. 85–99. ISBN: 978-3-540-48373-1.

[Kri02] Jean-Louis Krivine. "Lambda-calculus types and models". DEA. Lecture. Université Paris 7, Feb. 2002. URL: https://cel.archives-ouvertes.fr/cel-00574575.

[SU10] Morten Sørensen and Paweł Urzyczyn. "Lectures on the Curry-Howard Isomorphism". In: *Studies in Logic and the Foundations of Mathematics* 149 (Oct. 2010). DOI: 10.1016/S0049-237X(06)80005-4.

[GKM13] Herman Geuvers, Robbert Krebbers, and James McKinna. "The $\lambda\mu^T$ -calculus". In: *Annals of Pure and Applied Logic* 164.6 (2013). Classical Logic and Computation 2010(CLAC 2010), pp. 676–701. ISSN: 0168-0072. DOI: https://doi.org/10.1016/j.apal.2012.05.005. URL: https://www.sciencedirect.com/science/article/pii/S0168007212000814.

[Mim20] Samuel Mimram. "Program = Proof". In: *Lecture Notes* (July 2020).

A Proofs

Proof. Lemma 2.1 on substitution compatibility. **1.** As usual we proceed by induction.

1. If t is a variable, then two case may occur if $t = x_A$ then $t[u/x] = u$ And $[\![u]\!]_2 = [\![x_A]\!]_2[[\![u]\!]_2/x_A]$ by definition.
   If $t = y_B$ then $t[u/x] = y_B$ and $[\![y_B]\!]_2 = [\![y_B]\!]_2[[\![u]\!]_2/x_A]$.
2. Now if $t = \lambda y.v$ by definition $t[u/x] = \lambda y.v[u/x]$ and so

$$[\![t[u/x]]\!]_2 = [\![\lambda y.v[u/x]]\!]_2 =_{def} \dfrac{\Gamma, y : B \vdash \Delta \ \langle v[u/x] : C\rangle}{\Gamma \vdash \Delta \ \langle \lambda y.v[u/x] : B \to C\rangle}\ abs \ =_{def} \ [\![\lambda y.v]\!]_2[[\![u]\!]_2/x]$$

   since by definition $[\![\lambda y.v]\!]_2 = \dfrac{\Gamma', y : B \vdash \Delta' \ \langle v : C\rangle}{\Gamma' \vdash \Delta' \ \langle \lambda y.v : B \to C\rangle}\ abs$ where $\Gamma = \Gamma'[\Theta/x]$ and $\Delta = \Delta' \cup \mathbf{1}_\Gamma(x) \cdot \Psi$
3. If $t = (w)v$ then $(w)v[u/x] = (w[u/x])v[u/x]$, and so

$$[\![(w)v[u/x]]\!]_2 = [\![(w[u/x])v[u/x]]\!]_2 = \dfrac{\Gamma \vdash \Delta, \ \langle t[u/x] : A \to B\rangle \quad \Sigma \vdash \Pi, \ \langle u[u/x] : A\rangle}{\Gamma, \Sigma \vdash \Delta, \Pi, \ \langle (t[u/x])u[u/x] : B\rangle}\ app = [\![(w)v]\!]_2[[\![u]\!]_2/x]$$

   since by definition $\dfrac{\Gamma' \vdash \Delta', \ \langle t : A \to B\rangle \quad \Sigma' \vdash \Pi', \ \langle u : A\rangle}{\Gamma', \Sigma' \vdash \Delta', \Pi', \ \langle (t)u : B\rangle}\ app$ where $\Gamma = \Gamma'[\Theta/x]$ and $\Delta = \Delta' \cup \mathbf{1}_\Gamma(x) \cdot \Psi$
4. $$[\![([\alpha]v)[u/x]]\!]_2 = [\![([\alpha]v[u/x]]\!]_2 = \dfrac{\Gamma \vdash \Delta, \ \langle v : B\rangle}{\Gamma \vdash \Delta, \alpha : B \ \langle [\alpha]v : \bot\rangle}\ name = [\![[\alpha]v]\!]_2[[\![u]\!]_2/x]$$

   since by definition $[\![[\alpha]v]\!]_2 = \dfrac{\Gamma' \vdash \Delta', \ \langle v : B\rangle}{\Gamma' \vdash \Delta', \alpha : B \ \langle [\alpha]v : \bot\rangle}\ name$ where $\Gamma = \Gamma'[\Theta/x]$ and $\Delta = \Delta' \cup \mathbf{1}_\Gamma(x) \cdot \Psi$
5. $$[\![(\mu\alpha.v)[u/x]]\!]_2 = [\![(\mu\alpha.v[u/x]]\!]_2 = \dfrac{\Gamma \vdash \Delta, \alpha : B \ \langle v : \bot\rangle}{\Gamma \vdash \Delta \ \langle \mu\alpha.v : B\rangle}\ \mu = [\![\mu\alpha.v]\!]_2[[\![u]\!]_2/x]$$

   since by definition $[\![\mu\alpha.v]\!]_2 = \dfrac{\Gamma' \vdash \Delta', \alpha : B \ \langle v : \bot\rangle}{\Gamma' \vdash \Delta' \ \langle \mu\alpha.v : B\rangle}\ \mu$ where $\Gamma = \Gamma'[\Theta/x]$ and $\Delta = \Delta' \cup \mathbf{1}_\Gamma(x) \cdot \Psi$
6. $$[\![(\mu\varnothing_B v)[u/x]]\!]_2 = [\![\mu\varnothing_B v[u/x]]\!]_2 = \dfrac{\Gamma \vdash \Delta, \ \langle v : \bot\rangle}{\Gamma \vdash \Delta \ \langle \mu\varnothing_A v : B\rangle}\ \mu\varnothing = [\![\mu\varnothing_B v]\!]_2[[\![u]\!]_2/x]$$

since by definition $[\![\mu\varnothing_B v]\!]_2 = \dfrac{\Big\|[\![v]\!]_2}{\dfrac{\Gamma' \vdash \Delta',\ \langle v : \bot\rangle}{\Gamma' \vdash \Delta' \langle \mu\varnothing_A v : B\rangle}\,\mu\varnothing}$ where $\Gamma = \Gamma'[\Theta/x]$ and $\Delta = \Delta' \cup \mathbf{1}_\Gamma(x) \cdot \Psi$

2. Now let us do the same for the structural substitution. If t does not contain a α has a free name then so does $[\![t]\!]_2$ (thanks to the corollary 2.0.1) and both substitution correspond to the identity and therefore the proposition is true, so we assume t contains a free occurence of the name $\alpha : A \to B$.

1. If $t = x_X$ is a variable then $x_X[\alpha \bullet u] = x_X$ and indeed by definition aswell $[\![x_X]\!]_2[\alpha \bullet [\![u]\!]_2] = [\![x_X]\!]_2$.
2. If $t = \lambda y.v$ then by definition $t[\alpha \bullet [\![u]\!]_2] = (\lambda y.v)[\alpha \bullet [\![u]\!]_2] = \lambda y.v[\alpha \bullet [\![u]\!]_2]$. Therefore
$$[\![(\lambda y.v)[\alpha \bullet [\![u]\!]_2]]\!]_2 = [\![\lambda y.v[\alpha \bullet [\![u]\!]_2]]\!]_2 = \dfrac{\Big\|_{v[\alpha\bullet\pi']}}{\Gamma, y : Y, \Sigma \vdash \Delta, \Pi, \alpha : B\ \langle v[\alpha \bullet [\![u]\!]_2] : Z\rangle}\to i = \left(\dfrac{\Big\|_{v}}{\Gamma, y : Y \vdash \Delta, \alpha : A \to B\ \langle v : Z\rangle}\to i\right)[\alpha \bullet \pi'] = [\![\lambda x.v]\!]_2[\alpha \bullet \pi']$$
3. If $t(w)v$ then $(w)v[\alpha \bullet u] = (w[\alpha \bullet u])v[\alpha \bullet u]$. Now to conclude notice that, (we show only the case where α occurs free in both v and u the other case are straightforward and similar):
$$[\![(w[\alpha \bullet u])v[\alpha \bullet u]]\!]_2 = \dfrac{\Big\|_{w[\alpha\bullet u]} \qquad \Big\|_{v[\alpha\bullet u]}}{\Gamma, \Sigma \vdash \Delta, \Pi, \alpha : B\ \langle X \to Y\rangle \quad \Theta, \Sigma \vdash \Psi, \Pi, \alpha : B\ \langle X\rangle}\to e = \left(\dfrac{\Big\|[\![w]\!]_2 \qquad \Big\|[\![w]\!]_2}{\Gamma \vdash \Delta, \alpha : A \to B\ \langle X \to Y\rangle \quad \Theta \vdash \Psi, \alpha : A \to B\ \langle X\rangle}\to e\right)[\alpha \bullet [\![u]\!]_2] = [\![(w)v]\!]_2[\alpha \bullet [\![u]\!]_2]$$
4. If $t = [\beta]v$ then $t[\alpha \bullet u] = [\beta](v[\alpha \bullet u])$.
$$[\![[\beta](v[\alpha \bullet u])]\!]_2 = \dfrac{\Big\|[\![v[\alpha\bullet u]]\!]_2}{\Gamma, \Sigma \vdash \Delta, \Pi, \alpha : B\ \langle X\rangle}\,name = \left(\dfrac{\Big\|[\![v]\!]_2}{\Gamma \vdash \Delta, \alpha : A \to B\ \langle X\rangle}\,name\right)[\alpha \bullet [\![u]\!]_2] = [\![[\beta]v]\!]_2[\alpha \bullet [\![u]\!]_2]$$
5. If $t = [\alpha]v$ then $t[\alpha \bullet u] = [\alpha]((v[\alpha \bullet u])u)$.
$$[\![[\alpha]((v[\alpha \bullet u])u)]\!]_2 = \dfrac{\dfrac{\Big\|[\![v[\alpha\bullet u]]\!]_2 \qquad \Big\|[\![u]\!]_2}{\Gamma, \Sigma \vdash \Delta, \Pi, \alpha : B\ \langle A \to B\rangle \quad \Sigma \vdash \Pi\ \langle A\rangle}\,app}{\Gamma, \Sigma \vdash \Delta, \Pi, \alpha : B\ \langle B\rangle}\,name = \left(\dfrac{\Big\|[\![v]\!]_2}{\Gamma \vdash \Delta, \alpha : A \to B\ \langle A \to B\rangle}\,name\right)[\alpha \bullet [\![u]\!]_2] = [\![[\alpha]v]\!]_2[\alpha \bullet [\![u]\!]_2]$$
6. $[\![\mu\beta.v[\alpha \bullet u]]\!]_2 = \dfrac{\Big\|[\![v[\alpha\bullet u]]\!]_2}{\Gamma, \Sigma \vdash \Delta, \alpha : B, \beta : X\ \langle\rangle}\,\mu = \left(\dfrac{\Big\|[\![v]\!]_2}{\Gamma \vdash \Delta, \alpha : A \to B, \beta : X\ \langle\rangle}\,\mu\right)[\alpha \bullet [\![u]\!]_2] = [\![\mu\beta.v]\!]_2[\alpha \bullet [\![u]\!]_2]$
7. $[\![\mu\alpha.v]\!]_2 = \dfrac{\Big\|[\![v]\!]_2}{\Gamma \vdash \Delta, \alpha : A \to B\ \langle\rangle}\,\mu = \left(\dfrac{\Big\|[\![v]\!]_2}{\Gamma \vdash \Delta, \alpha : A \to B\ \langle\rangle}\,\mu\right)[\alpha \bullet [\![u]\!]_2] = [\![\mu\alpha.v]\!]_2[\alpha \bullet [\![u]\!]_2]$
8. $[\![\mu\varnothing_X.v[\alpha \bullet u]]\!]_2 = \dfrac{\Big\|[\![v[\alpha\bullet u]]\!]_2}{\Gamma, \Sigma \vdash \Delta, \Pi, \alpha : B\ \langle\rangle}\,\mu\varnothing = \left(\dfrac{\Big\|[\![v]\!]_2}{\Gamma \vdash \Delta, \alpha : A \to B\ \langle\rangle}\,\mu\varnothing\right)[\alpha \bullet [\![u]\!]_2] = [\![\mu\varnothing_X.v]\!]_2[\alpha \bullet [\![u]\!]_2]$

□