DEIVID VALE

ON SOLVING NOMINAL DISUNIFICATION PROBLEMS

ON SOLVING NOMINAL DISUNIFICATION PROBLEMS

DEIVID VALE

A dissertation
submitted and presented to the Department of Mathematics
of University of Brasília
in partial fulfillment of the requirements for the Degree of
Master of Mathematics

v1.2.1

On Solving Nominal Disunification Problems

SUPERVISOR:
Daniele Nantes

LOCATION:
Brasília – Brazil

I dedicate this work to Bruno Caxito, Cássia Cristiane, Tiago Araújo, Maria Lúcia, Marina Sousa (the best friends I could ever wish for), Danubia Vale, Thiago Vale, and Johne Vale (the best siblings I could ever wish for), Edna Silva (my dear girlfriend), and to my mother who is the genesis of it all.

"A morte pode sorrir para você de muitas maneiras, mas sorrir de volta cabe a você decidir. Em um lugar como aquele, era possível ver sorrisos que iam de orelha a orelha, mas para vê-los era necessário fechar os olhos . . . "

Bruno Caxito

ABSTRACT

An extension of first-order disunification problems is proposed by taking into account binding operators according to the nominal approach. In this approach, bindings are implemented through nominal atoms used instead of binding variables and renaming of atoms are implemented by atom permutations. In the nominal setting, unification problems consist of equational questions of the form $s \approx_\alpha^? t$ (read: is s α-equivalent to t?) considered under *freshness* problems $a\#^? t$ (read: is a fresh for t?) that restrict solutions by forbidding free occurrences of atoms in the instantiations of variables.

In addition to equational and *freshness* problems, nominal disunification problems also include nominal disunification constraints in the form of disequations $s \not\approx_\alpha^? t$ (read: is s α-different to t?) and their solutions consist of pairs of a substitution σ and a finite set of *freshness* constraints in the form of $a\#X$ such that under these restrictions the σ-instantiation of the equations, disequations, and *freshness* problems holds.

By re-using nominal unification techniques, it is shown how to decide whether two nominal terms can be made different modulo α-equivalence. This is done by extending previous results on first-order disunification and by defining the notion of *solutions with exceptions* in the nominal syntax. A discussion on the semantics of disunification constraints is also given.

I do not aim with my hand; he who aims with his hand has forgotten the face of his father.
I aim with my eye.
I do not shoot with my hand; he who shoots with his hand has forgotten the face of his father.
I shoot with my mind.
I do not kill with my gun; he who kills with his gun has forgotten the face of his father.
I kill with my heart.

King, Stephen — The Gunslinger

ACKNOWLEDGMENTS

I want to express my gratitude to everyone in my life that made this dissertation possible. Enumerating all of you is impossible; notwithstanding, I shall try my best to do so.

I am grateful to my advisor, Daniele Nantes, for having dedicated so many years to teach me how to study and appreciate mathematics. You have shown me the beauty of solving problems and shared with me the excitement of learning something new. Thank you.

I am grateful to my mother, Cleonice de Fátima, the only person in the entire world that dared to sacrifice her education in praise of mine. All that is good in me began with you, mother.

I thank my family that has contributed to this work mostly by making my life worth living. Especially my stepfather, Doraci Dias, for being the father I did not have. Without your support, I would never be able to move to another state to focus only on studying mathematics.

Many thanks to my girlfriend, Edna, for spending time with me and making my life better. Without your constant support, this journey could have been much harder. I am incomplete without you.

I thank my friends outside academia for providing me enough distraction from mathematics. Especially, Bruno Caxito. Thanks, Caxito, for all the good things that arise from a good friendship.

Many thanks go to friends embedded in academic life: Bruna Nunes, Cláudia Orduz, Dino da Silva Sauro, Gabriel Bufolo, Gabriel Silva — the *Maximal Element*, for every ball with positive radius centered in Edna's head—, Junior Cabiludo, and George Leitão.

I thank the professors in the department of mathematics for the excellent education provided over the years. Its worth mention Célius Magalhães and Luciana Ávila for having participated in my academic and personal life since the very beginning of this story.

I give my thanks to the examination committee for the valuable suggestions and improvements on this work. An especial thanks to Professors Mauricio Ayala and Maribel Fernández for the participation in the research done in this dissertation.

Lastly, but unquestionably not least important, I give my thanks to the millions of Brazilians who day after day paid their taxes to fund my education, even not having any choice for it. I shall give my best to return your investments in the form of teaching future generations and perhaps creating scientific knowledge to advance our society.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1

INTRODUCTION

Nominal techniques can be used to reason about systems with binders. The binding structure of these systems always requires a method to deal with α-equivalence between objects in the system, i.e., objects (usually the parse tree representation of the concrete syntax) are considered equal if they differ only by the name of bound variables. For instance, in the syntax of λ-calculus, terms like $\lambda x.x\,y$ and $\lambda z.z\,y$ should be considered equivalent, despite their syntactical differences.

It is common in the literature to consider the α-equivalence relation as part of the syntactical structure of terms of the language. One often says "terms are considered syntactically identical if they are α-convertible". This means that one considers the quotient of the set of terms by the α-equivalence relation. So one has the problem of which representative of α-equivalence classes should be chosen. One of the most popular strategies to solve this problem is called the "Barendregt Variable Convention": choose representatives for which the bound variables are mutually distinct and distinct from any *free* variable in the current context. This strategy solves the problem for handwritten proofs and calculations, but not for implementations. Another treatment of α-equivalence is to get rid of equivalence classes by considering *de Bruijn indices* instead of variable names. Using de Bruijn indices, free and bound variables are indexed as naturals and thus all objects have unique representations so that one does not need to worry about representatives of equivalence classes. This approach facilitates the implementation of systems with binders but at the cost of readability.

The nominal approach diverges from those above in two important ways: first, one can reason about α-equivalence in a readable way, very close to informal practice, while still remaining fully formal since α-renaming is embedded in the nominal syntax; second, nominal α-equivalence is easy to implement in computer systems.

Nominal terms have *atoms* ($a, b, c \ldots$) used to represent object-level variables, and *variables, or unknowns* ($X, Y, \ldots$) used to express variables on the meta-level. Atoms can be abstracted by a binder operator but cannot be instantiated by a substitution, whereas variables cannot be abstracted but can be instantiated by a substitution. For instance, the nominal term $[a]\,t$ represents the abstraction of atom a in t. To rename an atom a to another atom b we make use of an *atom permutation*. Permutations are built as lists of atom *swappings* of the form $(a\ b)$. The action of $\pi = (a\ b)$ over $[a]\,t$, denoted by $\pi \cdot t$ gives as result the term $[b]\,t'$ where t' is obtained by the replacement of all occurrences of a by b and all occurrences of b by a in t. The action of an atom permutation π over a meta-variable X will be 'suspended' in X, written as $\pi \cdot X$, and will be 'executed' only when X is instantiated. The α-equivalence relation over nominal terms is built using permutations and a *freshness*

relation between atoms and terms, written as $a\#t$, which means that a cannot occur *free* in t.

Nominal techniques have been widely explored and investigated for the last years [3, 18, 26]. Nominal unification has also been developed [13, 29], and more recently, works on unification modulo equational theories [1, 2, 4] have also been developed. Unification is the problem of finding a substitution σ that makes two terms 'equal' i.e., $s\sigma \approx_\alpha t\sigma$. In the nominal setting, α-equality comes with freshness conditions for atoms, which should be taken into account when dealing with nominal unification problems. For instance, the problem of unifying $\lambda[a]\,X$ and $\lambda[b]\,Y$ reduces to the problem of unifying $X \approx_\alpha^? (b\ a)\cdot Y$ under the condition that $a\#Y$. Therefore, a solution to a nominal unification problem will be a pair $\langle \Gamma, \sigma\rangle$ consisting of a set of freshness constraints Γ, and a substitution σ over the variables appearing in the problem. Several applications of nominal unification exist, for instance, in logic programming [14], automatic deduction and theorem proving [24], among others.

This work is about *nominal disunification*, that is, the problem of solving nominal unification questions enriched with *disequations*, i.e., constraints of the form $s \not\approx_\alpha^? t$. For example, consider the unification problem $\lambda[a]\,X \approx_\alpha^? \lambda[b]\,Y$ as above, but imposing the condition that solutions can neither map X to the atom a nor Y to the atom b. This condition may be given as a set of disequations, and solutions should be computed in such a way that they (and therefore, their instances) satisfy the imposed restriction. The *nominal disunification constraint* is then represented as a pair of equational and disequational problems:

$$\left\langle \lambda[a]\,X \approx_\alpha^? \lambda[b]\,Y \;\;||\;\; X \not\approx_\alpha^? a,\;\; Y \not\approx_\alpha^? b \right\rangle$$

Imposing such conditions has some side-effects that need to be addressed in order to be able to formally state a definition of solution. In a more general view, a nominal disunification problem is given as $\mathcal{P} = \langle \Delta \vdash s_1 \approx_\alpha^? t_1, \ldots, s_n \approx_\alpha^? t_n \mid \nabla \vdash u_1 \not\approx_\alpha^? v_1, \ldots, u_m \not\approx_\alpha^? v_m\rangle$, and a solution to such problem is a pair $\langle \Gamma, \sigma\rangle$ of a context Γ and a substitution σ, such that σ makes terms of each equation equal, but leaves those of the disequations different, while satisfying the freshness constraints Δ and ∇.

The strategy proposed by Buntine and Bürckert [9] to solve systems containing first-order equations and disequations is followed in the current work. But its extension to the nominal setting is not straightforward since the notions of equality and disequality are different and the freshness side conditions add extra constraints to the problem. The standard nominal unification algorithm [29] can be reused to provide solutions to nominal unification problems, and following Buntine and Bürckert's approach, we show that nominal disequations can be treated in a nominal term-algebra.

RELATED WORK. Disunification problems have been studied extensively in the first-order framework [6, 9, 15–17, 28] and also in the higher-order one [25].

Buntine and Bürckert [9] solve systems of equations and disequations in equational theories with a finitary unification type; they investigate E-disunification problems with two main applications in mind: the first application is to give a generalization for logic programming to include negation clauses in such a way that solution to queries can be expressed as substitutions other than the limited form of negation, called negation as failure.

The second applications is related with the use of E-disunification as a mechanism to drastically reduce the solution space of the unification algorithm for some equational theories. For instance, they showed that associative-commutative unification problems (a.k.a. AC-unification problems) are in fact a kind of so called $AC1$-disunification problems (associative-commutative functions with a unity 1) that have a solution space considerably smaller than the solution space of standard AC-problems. Differently, Comon and Lescanne [15, 16] consider more general problems, called *equational problems*, which include universally and existentially quantified variables in the algebra of rational trees or in the quotient term-algebra $T(F, X)$ by a congruence $=_E$. They propose a set of transformation rules on equational problems of the form $\exists \overline{w} \forall \overline{y} : P_1 \wedge \ldots \wedge P_n$, where P_i, for $i = 1..n$, is a called a system, that is, an equation of the form $s = t$ or $\top$, or a disequation $s \neq t$ or $\bot$, or a disjunction $P_{i1} \vee \ldots \vee P_{in_i}$ of systems. Their strategy consists of applying transformation rules to the equational problem until a kind of *solved form* is reached. These problems have applications in *sufficient completeness* for algebraic specifications defined by sets of rewriting rules.

In [17], Fernández shows that E-disunification is semi-decidable when the theory E is presented by a ground convergent rewrite system, and gives a sound and complete E-disunification procedure based on narrowing. Baader and Schulz [6] show that solvability of disunification problems in the free algebra of the combined theory $E_1 \cup \ldots \cup E_n$ is decidable if solvability of disunification problems with linear constant restrictions in the free algebras of the theories $E_i (1 \leq i \leq n)$ is decidable. Lugiez [25] introduces higher-order disunification problems and gives some decidable cases for which equational problems can be extended to higher-order systems.

CONTRIBUTIONS. The main contributions presented in this dissertation is summarized as follows.

1. We include proofs for all important results, complementing the related work.

2. We extend first-order disunification problems to the nominal framework introducing *nominal constraint problems*.

3. We extend the notion of substitution with exceptions to *solution pairs* that consists of a freshness context and a substitution-with-exceptions, Definition 4.2. In addition, a version of the Consistency Test Algorithm (Algorithm 1) to deal with *the consistency problem* of pairs with exceptions is proposed.

4. We propose a sound, complete, and terminating (provided nominal unification is finitary) procedure (Algorithm 2) to solve nominal disunification constraints that reuses the nominal unification algorithm.

5. We prove that the *Representation Theorem* holds in the nominal approach to disunification. This, Theorem 4.1, is the main result of the dissertation.

6. From the semantics point of view, we show that Birkhoff's HSP Theorem (Theorem 3.12), as in the first-order case, does not hold for nominal disequations, see Example 3.6.

The results obtained in this work, presented mainly in Chapter 4, have been accepted for publication in the Proceedings of LSFA 2019 [5].

OUTLINE OF THE DISSERTATION. Chapter 2 establishes the main notions from nominal syntax, α-equality, and unification. Chapter 3 introduces notions from nominal universal algebra by working on a detailed revision of all the construction needed to prove the nominal version of the so called HSP Theorem, Theorem 3.12. Chapter 4 introduces the nominal constraint problems as well as a generalized notion of instantiation and proves some results on the consistency of pairs with exceptions. Section 4.2 shows how to solve nominal constraint problems by reusing the nominal unification algorithm. Chapter 5 concludes the dissertation.

2

NOMINAL ABSTRACT SYNTAX

This chapter introduces the basic concepts and definitions on nominal terms, constraint solving, and nominal unification. The notation is consistent with standard works on nominal techniques such as [18] and [22].

2.1 NOMINAL TERMS

Fix countable infinite disjoint sets of *variables* $\mathbb{X} = \{X, Y, Z, \dots\}$ and *atoms* $\mathbb{A} = \{a, b, c, d, \cdots\}$. Variables represent meta-level *unknowns* and atoms object level variable symbols. Atoms are identified by their name, so it is redundant to say two atoms a and b are different. A signature Σ is a set of term-formers such that each $f \in \Sigma$ is assigned a unique non-negative integer n, called the *arity* of f, written as $f : n$.

A permutation π is a bijection $\mathbb{A} \to \mathbb{A}$ with finite domain, i.e., the set $\mathtt{supp}(\pi) := \{a \in \mathbb{A} \mid \pi(a) \neq a\}$ is finite. Write $\mathtt{id}$ for the identity permutation. The composition of two permutations π and π' is denoted as $\pi \circ \pi'$. We also write a permutation π as a list of swappings $\pi = (a_1\ b_1)(a_2\ b_2)\dots(a_n\ b_n)$ and denote by $(\mathbb{P}, \circ)$ the group of all permutations with finite support. We may abbreviate $(\mathbb{P}, \circ)$ as $\mathbb{P}$.

Definition 2.1 (Nominal Terms)**.** Let Σ be a signature disjoint from $\mathbb{A}$ and $\mathbb{X}$. The set $T(\Sigma, \mathbb{A}, \mathbb{X})$ of all nominal terms is inductively generated by the following grammar:

$$s, t, u, v \ ::= \ a \mid \pi \cdot X \mid [a]\, t \mid f(t_1, \dots, t_n)$$

Terms are called, respectively, **atoms**, **moderated variables**, **abstractions**, and **functions application**. Nominal terms are like first-order terms with the addition of atoms, abstractions, and a built-in notion of binder. Atoms are just members of $\mathbb{A}$, as said earlier they represent object-level variables. A moderated variable is a variable that has a *suspended* permutation, intuitively, a suspension $\pi \cdot X$ represent 'permute π in whatever X is instantiated to'. Note that X is not a nominal term but $\mathtt{id} \cdot X$ is. However, the abbreviation of $\mathtt{id} \cdot X$ to X is used when no ambiguity arises. An abstraction $[a]\, t$ is intended to represent a term t (the scope of $[a]$) with all occurrences of the atom a bounded by the binder operator $[\cdot]$. Atoms not in the scope of a binder $[\cdot]$ are called *free*.

Example 2.1. Let $\Sigma_\lambda := \{\mathtt{lam} : 1, \mathtt{app} : 2\}$ be the signature for the λ-calculus (for a complete axiomatization of the λ-calculus within the nominal syntax the reader is referred to [21]). If one consider λ-variables as atoms λ-terms can be inductively generated by the grammar:

$$e ::= a \mid \mathtt{lam}([a]\, e) \mid \mathtt{app}(e, e)$$

To simplify notation, write $\mathtt{app}(s,t)$ as $s\,t$ and $\mathtt{lam}([a]\,s)$ as $\lambda[a]\,s$. The following are examples of nominal terms:

$$(\lambda[a]\,a)\,X \qquad (\lambda[a]\,(\lambda[b]\,b\,a)\;\;c)\,d$$

Definition 2.2. The pair $s \equiv t$ denote syntactic equality between terms.

Remark 2.1. Note that if $\pi = \gamma$ then $\pi \cdot X \equiv \gamma \cdot X$, since permutations are bijections. In nominal syntax there is no quotient by abstraction, e.g. $[a]\,a \not\equiv [b]\,b$.

2.1.1 *Substitutions and Permutation Action*

Substitution is not an easy notion to define in a system with binders because it needs to avoid the *capture* of free variables occurrences. For example, consider the λ-term $(\lambda y.yx)$, one can think of this term as the function that 'takes the input y and apply it to x'. The substitution $[x/y]$ applied to $(\lambda y.yx)$ without any restriction give as output the λ-term $(\lambda y.yy)$ which has the function interpretation of 'takes the input y and apply it to itself'. Notice that in the resultant term the former x, substituted for y, is now bounded: it has been *captured* by the abstractor λ. The capture of free variables changes the semantics of λ-terms, in the sense that λ-terms are abstractions for the behavior of functions: changing how functions behave also changes its semantics.

Binders appears not only in the λ-calculus context. Maybe a more iluminating example[1] would be a function like $f(y) = y + x$ which becomes $f(y) = y + y$ if x is renamed for y. Perhaps is now more easy to see what we mean by a change on the semantic of a term by free variable capturing. The same happens in first-order logic when we consider formulas like $\forall y.\, y > x$ which does not have the same semantics as $\forall y.\, y > y$. These requirements could be summarized as the following:

Substitutions of free variables in a term cannot change the semantics of that term.

One approach to tackle 'the binding problem' is by defining a special kind of substitution, called *renaming*, that only map bound variables in a term to other fresh variables, and then builds the notion of a general substitution on top of this renaming operation. For instance, before the application of $[x/y]$ in $(\lambda y.yx)$ rename the binding variable y in $(\lambda y.yx)$ to a fresh variable z obtaining $(\lambda z.zx)$. Finally, apply the substitution $(\lambda z.zx)[x/y]$ to get $\lambda z.zy$.

Another attempt is to define an equivalence relation on terms, called α-equivalence, and put the renaming side conditions inside this definition. Again, renaming is the basis of α-equivalence. For a more detailed discussion the reader is referred to [8, Definition 2.1.11].

The nominal approach differs from the previous by making a distinction between two kinds of variables: first, the meta-level variables (just call them *variables*); second, the object-level variables (call them *atoms*).

1 For a more mathematical inclined audience.

Meta-level variables are unknowns that can be instantiated to other nominal terms. In constrast, atoms are constant-like variables that represent object-level unknowns which can be bounded by a binder operator, and renaming is implemented via the use of permutations of names.

We shall now define how permutations and substitutions act on general terms, denoted by $\pi \cdot t$ and $t\sigma$, respectively.

Definition 2.3 (Action of Permutation)**.** The **object-level** action of a permutation π on a term t is defined by induction on the structure of t, as follows:

$$\pi \cdot a \equiv \pi(a) \qquad \pi \cdot (\gamma \cdot X) \equiv (\pi \circ \gamma) \cdot X \qquad \pi \cdot [a]\, t \equiv [\pi \cdot a]\, (\pi \cdot t)$$
$$\pi \cdot f(t_1, \ldots, t_n) \equiv f(\pi \cdot t_1, \ldots, \pi \cdot t_n)$$

Intuitively, π propagates through the structure of t until it reaches an atom or a moderated variable. It can be proven (by a simple induction on the structure of terms and using the definition given above) that composition and identity of permutations extend to terms, that is: $(\pi \circ \gamma) \cdot t \equiv \pi \cdot (\gamma \cdot t)$ and $\mathtt{id} \cdot t \equiv t$. This fact will be used extensively in the reasoning that follows.

As usual, substitutions are defined as maps from variables to nominal terms. They are the way we instantiate variables, that is, replace the occurrence of a variable X by a nominal term t, denoted as $[X/t]$. The action of a substitution can be extended from variables to terms according to the definition below.

Definition 2.4 (Action of substitutions)**.** The META-LEVEL ACTION of a substitution σ on a term t, denoted as $t\sigma$, is inductively defined by:

$$a\sigma \equiv a \qquad (\pi \cdot X)\sigma \equiv \pi \cdot (X\sigma) \qquad ([a]\, t)\sigma \equiv [a]\, (t\sigma)$$
$$f(t_1, \ldots, t_n)\sigma \equiv f(t_1\sigma, \ldots, t_n\sigma)$$

Intuitively, σ propagates through the structure of a term until it reaches an atom or a moderated variable. Atoms are treated like *constants* by the substitution, and when it reaches a suspension $\pi \cdot X$ then σ acts on X and afterwards π acts on $X\sigma$. For example, take $\pi = (a\ b)$ and $\sigma = [X/b]$ then $((a\ b) \cdot X)\sigma \equiv (a\ b) \cdot b \equiv a$.

Remark 2.2. The composition of two substitutions will be denoted by $\sigma \circ \tau$ (abbreviated by $\sigma\tau$) and means: *first apply σ and then apply τ to the resultant term*.

$$t(\sigma \circ \tau) \equiv t\sigma\tau \equiv (t\sigma)\tau$$

Remark 2.3. Substitutions do not avoid capture of variables by itself. For instance, take $([a]\, t)\sigma \equiv [a]\, (t\sigma)$ the substitution σ does not avoid capture of a in t by the abstraction.

The next lemma guarantees that substitutions and permutation actions commute.

Lemma 2.1 (Commutation Lemma)**.** $\pi \cdot (t\sigma) \equiv (\pi \cdot t)\sigma$.

Proof. The proof is by induction on the structure of t and using Definitions 2.3 and 2.4. The base case is trivial since atoms are not affected by substitution actions. For moderated variables:

$$\begin{aligned}\pi \cdot ((\pi' \cdot X)\sigma) &\equiv \pi \cdot (\pi' \cdot (X\sigma)) \\ &\equiv (\pi \circ \pi') \cdot X\sigma \\ &\equiv (\pi \circ \pi' \cdot X)\sigma \\ &\equiv (\pi \cdot (\pi' \cdot X))\sigma\end{aligned}$$

For abstractions:

$$\begin{aligned}\pi \cdot ([a]\, t\sigma) &\equiv [\pi \cdot a]\, \pi \cdot (t\sigma) \\ &\overset{\text{I.H}}{\equiv} [\pi \cdot a]\, (\pi \cdot t)\sigma \\ &\equiv (\pi \cdot [a]\, t)\sigma.\end{aligned}$$

Finally, for function application:

$$\begin{aligned}\pi \cdot f(t_1, \ldots, t_n)\sigma &\equiv f((\pi \cdot t_1)\sigma, \ldots, (\pi \cdot t_n))\sigma \\ &\overset{I.H}{\equiv} f(\pi \cdot (t_1\sigma), \ldots, \pi \cdot (t_n\sigma)) \\ &\equiv \pi \cdot (f(t_1, \ldots, t_n)\sigma).\end{aligned}$$

□

Definition 2.5. The META-LEVEL PERMUTATION ACTION of a permutation π on a term t, denoted t^π, is inductively defined by:

$$a^\pi \equiv \pi(a) \qquad (\gamma \cdot X)^\pi \equiv (\pi\gamma\pi^{-1}) \cdot X \qquad ([a]\, t)^\pi \equiv [a^\pi]\, t^\pi$$
$$f(t_1, \cdots, t_n)^\pi \equiv f(t_1^\pi, \cdots, t_n^\pi)$$

Remark 2.4. Note that the meta-level action of a permutation on a suspension is the conjugation (as in abstract algebra) of γ by π. In fact, we can view the definition given above as the action of the group $\mathbb{P}$ by conjugation on terms. The conjugation of two elements from $\mathbb{P}$, as usual, is denoted by $\rho^\pi = \pi\rho\pi^{-1}$.

The next lemma translates the predicted behavior of compositions $\pi \circ \rho$ acting on terms. They act like $\mathbb{P}$-conjugation on $\mathbb{A}$.

Lemma 2.2. For any term t and permutations π and ρ, $t^{\pi\circ\rho} \equiv t^{\rho^\pi}$.

Proof. The proof is by induction on the structure of t and using Definition 2.5.

The base case follows by $a^{\pi\circ\rho} \equiv \pi(\rho(a)) \equiv \pi(a^\rho) \equiv a^{\rho^\pi}$.

For abstractions we have:

$$\begin{aligned}([a]\, t)^{\pi\circ\rho} &\equiv [a^{\pi\circ\rho}]\, t^{\pi\circ\rho} \\ &\equiv \left[a^{\rho^\pi}\right] t^{\pi\rho} \\ &\overset{I.H}{\equiv} \left[a^{\rho^\pi}\right] t^{\rho^\pi} \\ &\equiv ([a]\, t)^{\rho^\pi}\end{aligned}$$

The most interesting case is:

$$\begin{aligned}(\gamma \cdot X)^{\pi\circ\rho} &\equiv (\pi\rho)\gamma(\pi\rho)^{-1} \cdot X \\ &\equiv \pi(\rho\gamma\rho^{-1})\pi^{-1} \cdot X \\ &\equiv \pi\gamma^{\rho}\pi^{-1} \cdot X \\ &\equiv (\gamma^{\rho})^{\pi} \cdot X \\ &\equiv (\gamma \cdot X)^{\rho^{\pi}}\end{aligned}$$

Finally the case for function application follows by:

$$\begin{aligned}f(t_1,\ldots,t_n)^{\pi\circ\rho} &\equiv f(t_1^{\pi\circ\rho},\ldots,t_n^{\pi\circ\rho}) \\ &\overset{I.H}{\equiv} f(t_1^{\rho^{\pi}},\ldots,t_n^{\rho^{\pi}}) \\ &\equiv f(t_1,\ldots,t_n)^{\rho^{\pi}}\end{aligned}$$

□

The next lemma states that Definitions 2.3 and 2.5 are definable in term of each other, in the presence of substitution.

Lemma 2.3. Given a term t and a permutation π, let σ be a substitution that maps each X mentioned in t to $\pi \cdot X$ and σ' that maps each X mentioned in t to $\pi^{-1} \cdot X$. Then:

$$\pi \cdot t \equiv t^{\pi}\sigma \qquad \text{and} \qquad t^{\pi} \equiv (\pi \cdot t)\sigma'$$

Proof. The proof follows by induction on the structure of t.

The base case is trivial since atoms are not affected by substitutions.

For abstractions we have: $\pi \cdot [a]\, t \equiv [\pi(a)]\, \pi \cdot t$ and by induction hypothesis follows that $[a^{\pi}]\, \pi \cdot t \equiv [a^{\pi}]\, t^{\pi}\sigma \equiv ([a]\, t)^{\pi}\sigma$.

The case for suspensions follows by:

$$\begin{aligned}\pi \cdot (\gamma \cdot X) &\equiv \pi\gamma \cdot X \\ &\equiv \pi\gamma\pi^{-1} \cdot (\pi \cdot X) \\ &\equiv \pi\gamma\pi^{-1} \cdot (X\sigma) \\ &\equiv (\pi\gamma\pi^{-1} \cdot X)\sigma \\ &\equiv (\gamma \cdot X)^{\pi}\sigma.\end{aligned}$$

For function application $f(\pi \cdot t_1,\ldots,\pi \cdot t_1)$ induction hypothesis gives $\pi \cdot t_i \equiv t_i^{\pi}\sigma$, and we conclude $\pi \cdot f(t_1,\ldots,t_n) \equiv f(t_1,\ldots,t_n)^{\pi}\sigma$.

The proof of $(\gamma \cdot X)^{\pi} \equiv (\pi \cdot (\gamma \cdot X))\sigma'$ follows the same reasoning:

$$\begin{aligned}(\gamma \cdot X)^{\pi} &\equiv \pi\gamma\pi^{-1} \cdot X \\ &\equiv \pi\gamma \cdot (\pi^{-1} \cdot X) \\ &\equiv \pi \cdot (\gamma \cdot X)\sigma'\end{aligned}$$

□

2.2 EQUALITY AND DERIVABILITY

This section concentrates on the study of nominal α-equivalence. The predicate for α-equality ($\approx_\alpha$) in nominal terms relies on the notion of 'fresh variable' (#):

- $a\#t$ means that atom a cannot occur free in t;
- $s \approx_\alpha t$ means that s is α-equivalent to t.

Constraints are generated by the grammar

$$P, Q, C := a\#t \mid s \approx_\alpha t.$$

The first constraint $a\#t$ is called a *freshness constraint* whereas the second $s \approx_\alpha t$ is called an *α-equality constraint*. A freshness constraint of the form $a\#a$ or $a\#X$ is called *primitive* (or reduced). We may drop set brackets and write $a\#t, b\#u$ for $\{a\#t, b\#u\}$.

A context, denoted by greek letters $\Delta, \Gamma, \ldots$ is a set of primitive freshness constraints, it is called *consistent* if it does not contain any constraint of the form $a\#a$.

With the predicates for freshness (#) and α-equality ($\approx_\alpha$), Fernández and Gabbay [18] present a derivation system defined by the rules in Figures 2.1 and 2.2 to express validity of freshness and α-equality constraints. This system generates an *equality theory*, in the sense of Theorem 2.4.

Definition 2.6. An *α-equality* is a pair $s \approx_\alpha t$ where s and t are terms.

$$\frac{}{a\#b}\,(\#ab) \qquad \frac{\pi^{-1}(a)\#X}{a\#\pi\cdot X}\,(\#X) \qquad \frac{}{a\#[a]\,t}\,(\#a) \qquad \frac{a\#t}{a\#[b]\,t}\,(\#b)$$

$$\frac{a\#t_1 \cdots a\#t_n}{a\#f(t_1,\ldots,t_n)}\,(\#f)$$

Figure 2.1: Derivation rules for freshness

To define $\approx_\alpha$ consider the *difference set* of two permutations:

$$\mathtt{ds}(\pi,\gamma) := \{a \in \mathbb{A} \mid \pi(a) \neq \gamma(a)\}.$$

In the rules defining $\approx_\alpha$ below, $\mathtt{ds}(\pi,\gamma)\#X$ denotes the set of constraints $\{a\#X \mid a \in \mathtt{ds}(\pi,\gamma)\}$.

Definition 2.7. Define a notion of α-derivability by the deduction rules in Figures 2.1 and 2.2.

1. Write $\Delta \vdash a\#t$ when a derivation of $a\#t$ exists using the elements of Δ as assumptions. We say $\Delta \vdash a\#t$ is *derivable* or just $\Delta \vdash a\#t$.
2. Write $\Delta \vdash s \approx_\alpha t$ when $s \approx_\alpha t$ can be derived such that the derivation uses (at most) assumptions from Δ. We say $\Delta \vdash s \approx_\alpha t$ is derivable or just $\Delta \vdash s \approx_\alpha t$ and write $\Delta \nvdash t \approx_\alpha u$ (or even $\Delta \vdash s \not\approx_\alpha t$) when $\Delta \vdash t \approx_\alpha u$ is *not derivable*.

$$\frac{}{a \approx_\alpha a}\ (\approx_\alpha a) \qquad \frac{\mathtt{ds}(\pi,\gamma)\#X}{\pi \cdot X \approx_\alpha \gamma \cdot X}\ (\mathrm{Ds})$$

$$\frac{t_1 \approx_\alpha u_1 \ \cdots\ t_n \approx_\alpha u_n}{f(t_1,\ldots,t_n) \approx_\alpha f(u_1,\ldots,u_n)}\ (\mathrm{F})$$

$$\frac{t \approx_\alpha u}{[a]\,t \approx_\alpha [a]\,u}\ (\text{Abs-a}) \qquad \frac{(b\ a)\cdot t \approx_\alpha u \quad b\#t}{[a]\,t \approx_\alpha [b]\,u}\ (\text{Abs-b})$$

Figure 2.2: Derivation rules for α-equivalence

Example 2.2. Below we give some examples of derivation trees, in the first we derive $a\#(X\lambda[a]\,Y)$ in the syntax of untyped λ-calculus with the assumption $a\#X$. The second is a trivial α-equivalence equality, we use it as an example to show how derivations are constructed.

$$\dfrac{a\#X \qquad \dfrac{\dfrac{}{a\#[a]\,Y}\,(\#a)}{a\#\lambda[a]\,Y}\,(\#f)}{a\#(X\,\lambda[a]\,Y)}\,(\#f) \qquad \text{(Abs-b)}\ \dfrac{(\approx_\alpha a)\ \dfrac{}{b \approx_\alpha b} \qquad \dfrac{}{b\#a}\,(\#a)}{[a]\,a \approx_\alpha [b]\,b}$$

The dashed line above represents a computation with the permutation action $(b\ a)\cdot a$, not an application of a derivation rule.

Definition 2.8. We extend notation for t^π, $\pi \cdot t$ and $t\sigma$ to freshness contexts Δ as follows:

$$\begin{aligned}\Delta^\pi &:= \{\pi \cdot a\#X \mid a\#X \in \Delta\}\\ \pi \cdot \Delta &:= \{\pi \cdot a\#\pi \cdot X \mid a\#X \in \Delta\}\\ \Delta\sigma &:= \{a\#X\sigma \mid a\#X \in \Delta\}\end{aligned}$$

Note that Δ^π is a freshness context, but $\pi \cdot \Delta$ and $\Delta\sigma$ need not be.

2.2.1 *Properties of # and $\approx_\alpha$*

In this subsection some syntactical properties of freshness and α-equality constraints will be developed.

Definition 2.9. The functions $\mathtt{atms}(t)$ and $\mathtt{var}(t)$ will be used to compute the set of atoms and unknowns in a term, respectively. They are defined by:

$$\begin{aligned}\mathtt{atms}(a) &= \{a\} & \mathtt{atms}(\pi \cdot X) &= \mathtt{supp}(\pi)\\ \mathtt{atms}([a]\,t) &= \mathtt{atms}(t) \cup \{a\} & \mathtt{atms}(f(t_1,\ldots,t_n)) &= \cup_i \mathtt{atms}(t_i)\end{aligned}$$

$$\begin{aligned}\mathtt{var}(a) &= \emptyset & \mathtt{var}(\pi \cdot X) &= \{X\}\\ \mathtt{var}([a]\,t) &= \mathtt{var}(t) & \mathtt{var}(f(t_1,\ldots,t_n)) &= \cup_i \mathtt{var}(t_i)\end{aligned}$$

This definition can be extended to constructions more complex than terms, e.g. a list (as in $\mathtt{atms}(\Delta, s, t)$) or a substitution. By this we mean the atoms or variables appearing anywhere within the brackets. More

specifically, $\mathtt{atms}(\Delta, s, t)$ means $\{a \mid a\#X \in \Delta\} \cup \mathtt{atms}(s) \cup \mathtt{atms}(t)$. Note that this extension will also be used for the function $\mathtt{var}(\cdot)$.

The next lemma states that if two permutations are identified in their action (that is, $\mathtt{ds}(\pi, \gamma) = \emptyset$) then they are also equivalent, in the sense that they are logically indistinguishable.

Lemma 2.4. Suppose Δ is a context. If $\mathtt{ds}(\pi, \gamma) = \emptyset$ then:

1. $\Delta \vdash \pi \cdot a\#t$ iff $\Delta \vdash \gamma \cdot a\#t$,
2. $\Delta \vdash a\#\pi \cdot t$ iff $\Delta \vdash a\#\gamma \cdot t$,
3. $\Delta \vdash \pi \cdot s \approx_\alpha t$ iff $\Delta \vdash \gamma \cdot s \approx_\alpha t$,
4. $\Delta \vdash s \approx_\alpha \pi \cdot t$ iff $\Delta \vdash s \approx_\alpha \gamma \cdot t$.

Proof. Consider $\pi \neq \mathtt{id}$. If $\pi = \mathtt{id}$ the result is trivially true.

1. Note that $\mathtt{ds}(\pi, \gamma) = \emptyset$ precisely when $\pi(a) = \gamma(a)$ for every atom a, so $\pi \cdot a \equiv \gamma \cdot a$ and the result follows.
2. We work by induction on the derivation of $a\#\pi \cdot t$ from Δ. There are five cases to consider and the analysis is based on the last rule applied in the derivation.
   a) The derivation concludes in $(\#ab)$: then $\pi \cdot t \equiv b$ so $t \equiv c$, for some atom c, and since $\pi \cdot b \equiv \gamma \cdot b$, the base case follows.
   b) The derivation concludes in $(\#X)$:

      Then $\Delta \vdash a\#\pi \cdot X$ and $\pi^{-1}(a)\#X \in \Delta$. Since $\mathtt{ds}(\pi, \gamma) = \emptyset$, one has $\gamma^{-1}(a) = \pi^{-1}(a)$, therefore, there is a proof of $\Delta \vdash a\#\gamma \cdot X$.
   c) The derivation concludes in $(\#a)$:

      Then $\Delta \vdash a\#\pi \cdot ([b]\, s)$ and $\pi \cdot b \equiv a$. Since $\gamma \cdot b \equiv a$ we use $(\#a)$ to build a derivation of $a\#\gamma \cdot [b]\, s$.
   d) The derivation concludes in $(\#b)$:

      Then $\Delta \vdash a\#\pi \cdot [b]\, s$, $\pi \cdot b \equiv b' \not\equiv a$ (otherwise the rule $(\#a)$ is applied), and $\Delta \vdash a\#\pi \cdot s$ is derivable. By induction hypothesis, $\Delta \vdash a\#\gamma \cdot s$ is derivable, and by rule $(\#b)$, there is a derivation of $\Delta \vdash a\#\gamma \cdot [b]s$.
   e) The derivation concludes in $(\#f)$:

      Then $\Delta \vdash a\#\pi \cdot f(t_1, \ldots, t_n)$ and $\Delta \vdash a\#\pi \cdot t_i$ for $1 \leq i \leq n$ is also derivable. By induction hypothesis, $\Delta \vdash a\#\gamma \cdot t_i$. From rule $(\#f)$, $\Delta \vdash a\#\gamma \cdot f(t_1, \ldots, t_n)$ is derivable.
3. The proof is by induction on the derivation of $\Delta \vdash \pi \cdot s \approx_\alpha t$, by analysing the last rule applied.
   a) The base case is trivial since $\pi \cdot a \equiv \gamma \cdot a$ for all atoms a.

b) The derivation concludes in (Ds):

Derive $\Delta \vdash \pi \cdot \tau \cdot X \approx_\alpha \tau' \cdot X$ from $\Delta \vdash a\#X$ for every $a\#X$ in $\mathtt{ds}(\pi \circ \tau, \tau')$. Since $\mathtt{ds}(\pi, \gamma) = \emptyset$, the identity $(\pi \circ \tau) \cdot a \equiv (\gamma \circ \tau) \cdot a$ holds, for every atom a. Therefore, $\mathtt{ds}(\pi\tau, \tau') = \mathtt{ds}(\gamma\tau, \tau')$. We can now write a derivation concluding in $\Delta \vdash \tau \cdot \tau \cdot X \approx_\alpha \tau' \cdot X$, from the same hypothesis, by applying (Ds).

c) The derivation concludes in (Abs-a):

Then we derive $\Delta \vdash [\pi \cdot b]\, \pi \cdot s \approx_\alpha [a]\, t$ from $\Delta \vdash \pi \cdot s \approx_\alpha t$, with $\pi \cdot b \equiv a$. By induction hypothesis, $\Delta \vdash \gamma \cdot s \approx_\alpha t$, besides $\gamma \cdot b \equiv a$, therefore, we use (Abs-a) to conclude $\Delta \vdash [\gamma \cdot b]\, \gamma \cdot s \approx_\alpha [a]\, t$.

d) The derivation concludes in (Abs-b):

Then $\Delta \vdash [\pi \cdot a]\, \pi \cdot s \approx_\alpha [b]\, t$ is derivable from $\Delta \vdash (b\ \pi(a)) \circ \pi \cdot s \approx_\alpha t$ and $\Delta \vdash b\#\pi \cdot s$, with $\pi \cdot a \not\equiv b$. Since $\mathtt{ds}(\pi, \gamma) = \emptyset$ and $\pi \cdot a \equiv \gamma \cdot a$, it follows that $\mathtt{ds}((b\ \pi \cdot a) \circ \pi, (b\ \gamma \cdot a) \circ \gamma) = \emptyset$. By induction hypothesis, $\Delta \vdash (b\ \gamma \cdot a) \cdot \gamma \cdot s \approx_\alpha t$ and by part 2, $\Delta \vdash b\#\gamma \cdot s$. By applying rule (Abs-b) we conclude $\Delta \vdash [\gamma \cdot a]\, \gamma \cdot s \approx_\alpha [b]\, t$.

e) If the derivation concludes in (F) just apply the induction hypothesis and (F).

4. Follows the same reasoning as in item 3.

□

The next theorem shows that nominal logic is *equivariant*, in the sense that derivable freshness ($a\#t$) and equality ($s \approx_\alpha t$) are invariant under the action of permutations.

Theorem 2.1 (Object-level equivariance)**.** For any permutation π:

1. if $\Delta \vdash a\#t$ then $\Delta \vdash \pi \cdot a\#\pi \cdot t$;
2. if $\Delta \vdash s \approx_\alpha u$ then $\Delta \vdash \pi \cdot s \approx_\alpha \pi \cdot u$.

Proof. Consider $\pi \neq \mathtt{id}$. If $\pi = \mathtt{id}$ the result is trivially true.

1. The proof is by induction on the derivation rules from Table 2.1, by analysing the last rule applied in $\Delta \vdash a\#t$.
   a) Base case is trivial.
   b) The last rule is $(\#X)$:

   Then $\Delta \vdash a\#\gamma \cdot X$ is derivable and $\gamma^{-1} \cdot a\#X \in \Delta$. Notice that

   $$(\pi \circ \gamma)^{-1} \cdot \pi \cdot a \equiv (\gamma^{-1} \circ \pi^{-1} \circ \pi) \cdot a \equiv \gamma^{-1} \cdot a$$

   Therefore, $(\pi \circ \gamma)^{-1} \cdot \pi \cdot a\#X \equiv \gamma^{-1} \cdot a\#X \in \Delta$, and the result follows from

   $$\frac{(\pi \circ \gamma)^{-1} \cdot \pi \cdot a\#X \in \Delta}{\Delta \vdash \pi \cdot a\#(\pi \circ \gamma) \cdot X}\ (\#X)$$

c) The last rule is $(\#a)$:

Notice that $\Delta \vdash \pi \cdot a\#\pi \cdot ([a]\, t)$ is the same as $\Delta \vdash \pi \cdot a\#[\pi \cdot a]\, \pi \cdot t$, which is derivable using $(\#a)$.

d) The last rule is $(\#b)$:

Then $\Delta \vdash a\#[b]\, t$ is derivable from $\Delta \vdash a\#t$. By induction hypothesis $\Delta \vdash \pi \cdot a\#\pi \cdot t$, therefore, there is a derivation Π

$$\frac{\begin{array}{c}\Pi\\ \Delta \vdash \pi \cdot a\#\pi \cdot t\end{array}}{\Delta \vdash \pi \cdot a\#\left[\pi \cdot b'\right]\, \pi \cdot t}\ (\#b)$$

where $\pi \cdot b' \equiv b$, and the result follows.

e) The last rule is $(\#f)$:

Then $\Delta \vdash a\#f(t_1, \ldots, t_n)$ is derivable from $\Delta \vdash a\#t_1, \ldots, a\#t_n$. By induction hypothesis, $\Delta \vdash \pi \cdot a\#\pi \cdot t_1, \ldots, \pi \cdot a\#\pi \cdot t_n$ and $\Delta \vdash \pi \cdot a\#\pi \cdot f(t_1, \ldots, t_n)$ is derivable using $(\#f)$.

2. The proof is by induction on the rules from Table 2.2, by analysing the last rule applied in $\Delta \vdash s \approx_\alpha u$.

a) For the base case:

From $\Delta \vdash a \approx_\alpha a$ one can trivially deduce $\Delta \vdash \pi \cdot a \approx_\alpha \pi \cdot a$.

b) The last rule is (Ds):

Then $\Delta \vdash \tau \cdot X \approx_\alpha \gamma \cdot X$ is derivable from $\Delta \vdash \mathtt{ds}(\tau, \gamma)\#X$. Note that $\mathtt{ds}(\pi \circ \tau, \pi \circ \gamma) = \mathtt{ds}(\tau, \gamma)$, therefore, $\Delta \vdash (\pi \circ \tau) \cdot X \approx_\alpha (\pi \circ \gamma) \cdot X$ is derivable using (Ds).

c) The last rule is (Abs-a):

Then $\Delta \vdash [a]\, s \approx_\alpha [a]\, u$ is derivable from $\Delta \vdash s \approx_\alpha u$. By induction hypothesis there is a derivation of $\Delta \vdash \pi \cdot s \approx_\alpha \pi \cdot u$, and we build a derivation for $\Delta \vdash [\pi \cdot a]\, \pi \cdot s \approx_\alpha [\pi \cdot a]\, \pi \cdot u$ using (Abs-a), which is the same as $\pi \cdot [a]\, s \approx_\alpha \pi \cdot [a]\, u$.

d) The last rule is (Abs-b):

Then $\Delta \vdash [a]\, s \approx_\alpha [b]\, u$ is derivable from $\Delta \vdash (b\ a) \cdot t \approx_\alpha u$ and $\Delta \vdash b\#t$. By induction hypothesis, $\Delta \vdash (\pi \circ (b\ a)) \cdot t \approx_\alpha \pi \cdot u$, and $\Delta \vdash \pi \cdot b\#\pi \cdot t$ is derivable from the first part. Since $\mathtt{ds}(\pi \circ (b\ a), (\pi \cdot b\ \pi \cdot a) \circ \pi) = \emptyset$ and by part 3 of Lemma 2.4, there are derivations Π_1 and Π_2 such that

$$\frac{\begin{array}{c}\Pi_1\\ \Delta \vdash ((\pi \cdot b\ \pi \cdot a) \circ \pi) \cdot t \approx_\alpha \pi \cdot u\end{array} \qquad \begin{array}{c}\Pi_2\\ \Delta \vdash \pi \cdot b\#\pi \cdot t\end{array}}{\Delta \vdash [\pi \cdot a]\, \pi \cdot t \approx_\alpha [\pi \cdot b]\, \pi \cdot u}\ (\text{Abs-b})$$

which proves this case.

e) If the derivatoin ends with (F):

By induction hypothesis, $\Delta \vdash \pi \cdot s_i \approx_\alpha \pi \cdot t_i$, for $1 \leq i \leq n$. Finally, with (F) we conclude $\Delta \vdash \pi \cdot f(s_1, \ldots, s_n) \approx_\alpha \pi \cdot f(u_1, \ldots, u_n)$.

□

Corollary 2.1. For any permutation π: $\Delta \vdash a\#\pi \cdot t$ if, and only if, $\Delta \vdash \pi^{-1} \cdot a\#t$. Similarly, $\Delta \vdash \pi \cdot t \approx_\alpha u$ if, and only if, $\Delta \vdash t \approx_\alpha \pi^{-1} \cdot u$.

Proof. Follows by directly by equivariance, Theorem 2.1. □

The next lemma states that freshness derivability is preserved under α-equality.

Lemma 2.5. If $\Delta \vdash t \approx_\alpha u$ then $\Delta \vdash a\#t$ if, and only if, $\Delta \vdash a\#u$.

Proof. The proof is by induction on the structure of t. The base case is trivial.

(a) If $t = \pi \cdot X$, then $\Delta \vdash \pi \cdot X \approx_\alpha \gamma \cdot X$ is derivable from $\Delta \vdash \mathtt{ds}(\pi, \gamma)\#X$. Now if $\Delta \vdash a\#\pi \cdot X$ then $\Delta \vdash \pi^{-1}(a) \cdot X$. There are two cases to consider: first, $\pi(a) \equiv \gamma(a)$ and trivially $\gamma^{-1}(a)\#X$; second, $\pi(a) \not\equiv \gamma(a)$ but since $\Delta \vdash \mathtt{ds}(\pi, \gamma)\#X$ by assumption, the result follows.

(b) Suppose $t = [b]\, t'$ then u is either $[b]\, u'$ or $[c]\, u'$. In the first case, $\Delta \vdash t \approx_\alpha u$ is derivable using (Abs-a), so $\Delta \vdash t' \approx_\alpha u'$. It follows from I.H that if $\Delta \vdash a\#t'$ then $\Delta \vdash a\#u'$, and by using $(\#b)$ whe have that $\Delta \vdash a\#[b]\, t'$ implies $a\#[b]\, u'$. For the latter case, note that $\Delta \vdash t \approx_\alpha u$ is derivable using (Abs-b) so both $\Delta \vdash (c\ b) \cdot t' \approx_\alpha u'$ and $c\#t'$ are derivable. By I.H $a\#(c\ b) \cdot t'$ implies $a\#u'$. Equivariance, using $\pi = (c\ b)$, gives $a\#t'$ and by $(\#b)$ $a\#[c]\, u'$.

(c) Supoose $t = f(t_1, \ldots, t_n)$, then $u = f(u_1, \ldots, u_n)$ and $\Delta \vdash t_i \approx_\alpha u_i$. By using I.H we get $\Delta \vdash a\#t_i$ implies $\Delta \vdash a\#u_i$. The conclusion follows using $(\#f)$.

□

The next result summarizes an important property of derivations: we do not need any assumption $a\#X$ if a does not appear in the syntax of s, for $a\#t$, or in the syntax of s, t for derivations of $s \approx_\alpha t$.

Theorem 2.2 (Strengthening)**.** Suppose $a \notin \mathtt{atms}(s, t)$. Then:

1. $\Delta, a\#X \vdash b\#s$ implies $\Delta \vdash b\#s$.
2. $\Delta, a\#X \vdash s \approx_\alpha t$ implies $\Delta \vdash s \approx_\alpha t$.

Proof. In both cases, the proof is by induction on the derivation rules for freshness Figure 2.1 and/or α-equality (Figure 2.2).

1. The analysis follows by checking the last rule applied in the derivation of $\Delta, a\#X \vdash b\#s$.
   a) Base case: when the last rule is $(\#ab)$ or $(\#a)$ the result follows trivially.
   b) The last rule is $(\#X)$:

      Then $\Delta, a\#X \vdash b\#\pi \cdot X$. If $a \notin \mathtt{atms}(\pi \cdot X) = \mathtt{supp}(\pi)$ then $\pi^{-1} \cdot b \not\equiv a$, and by Corollary 2.1, $\pi^{-1} \cdot b\#X \in \Delta$. Therefore, by applying $(\#X)$ $\Delta \vdash b\#\pi \cdot X$, and the result follows.

c) The last rule is $(\#b)$:

Hence, $\Delta, a\#X \vdash b\#[c]\,s$ is derivable from $\Delta, a\#X \vdash b\#s$. By the induction hypothesis $\Delta \vdash b\#s$, and we derive $\Delta \vdash b\#[c]\,s$ using $(\#b)$.

d) The last rule if $(\#f)$:

Then by induction hypothesis, $\Delta \vdash b\#s_i$ for $1 \leq i \leq n$. We derive $a\#f(s_1, \ldots, s_n)$ with $(\#f)$.

2. The analysis follows by checking the last rule applied in the derivation of $\Delta, a\#X \vdash s \approx_\alpha t$.

a) The base case is trivial, since $\vdash a \approx_\alpha a$ is always derivable.

b) Suppose the last rule is (Ds):

Then $\Delta, a\#X \vdash \pi \cdot X \approx_\alpha \gamma \cdot X$, hence $\Delta, a\#X \vdash \mathtt{ds}(\pi, \gamma)\#X$. The result will follow if we prove that $\Delta \vdash \mathtt{ds}(\pi, \gamma)\#X$ is derivable. In fact, since $a \notin \mathtt{atms}(\pi \cdot X, \gamma \cdot X)$ follows that $a \notin \mathtt{supp}(\pi) \cup \mathtt{supp}(\gamma)$. Therefore, $\pi \cdot a \equiv \gamma \cdot a \equiv a$ and $a \notin \mathtt{ds}(\pi, \gamma)$. Hence, $\Delta \vdash \mathtt{ds}(\pi, \gamma)$, and $\Delta \vdash \pi \cdot X \approx_\alpha \gamma \cdot X$ follows by (Ds).

c) The last rule is (Abs-a):

Then $\Delta, a\#X \vdash [c]\,t \approx_\alpha [c]\,u$ is derived from $\Delta, a\#X \vdash t \approx_\alpha u$. Induction hypothesis gives us a derivation for $\Delta \vdash t \approx_\alpha u$ and, we conclude $\Delta \vdash [c]\,t \approx_\alpha [c]\,u$ using (Abs-a).

d) The last rule is (Abs-b):

Then $\Delta, a\#X \vdash [c]\,t \approx_\alpha [d]\,u$ can be derived from $\Delta, a\#X \vdash (d\ c) \cdot t \approx_\alpha u$ and $\Delta, a\#X \vdash d\#t$. By the inductive hypothesis we have $\Delta \vdash (d\ c) \cdot t \approx_\alpha u$, and by the first part $\Delta \vdash d\#t$. It is now possible to derive $\Delta \vdash [c]\,t \approx_\alpha [d]\,u$ using (Abs-b) from the above premises.

e) The last rule is (F):

Then $\Delta, a\#X \vdash f(t_1, \ldots, t_n)$ and by induction hypothesis $\Delta \vdash t_i$, for $1 \leq i \leq n$. Therefore, by (F), $\Delta \vdash f(t_1, \ldots, t_n)$.

□

Lemma 2.6. Fix π and γ permutations. If $\Delta \vdash a\#s$ for each $a \in \mathtt{ds}(\pi, \gamma)$ then $\Delta \vdash \pi \cdot s \approx_\alpha \gamma \cdot s$.

Proof. The proof is by induction on the structure of terms.

1. For the base case suppose $s \equiv c$, for some atom c. If $c \in \mathtt{ds}(\pi, \gamma)$ then $\Delta \vdash c\#c$ contradicting our assumption that $\Delta \vdash a\#s$. (Observe that no context can prove a freshness constraint of the form $a\#a$ for any atom a). So $c \notin \mathtt{ds}(\pi, \gamma)$ and $\Delta \vdash \pi \cdot c \approx_\alpha \gamma \cdot c$.

2. If $s \equiv \tau \cdot X$ then $\Delta \vdash a\#\tau \cdot X$ for all $a \in \mathtt{ds}(\pi, \gamma)$. By applying rule (Ds), one has the derivation:

$$\frac{\Delta \vdash \mathtt{ds}(\pi\tau, \gamma\tau)\#X}{\Delta \vdash (\pi\tau) \cdot X \approx_\alpha (\gamma\tau) \cdot X}\ (\mathrm{Ds})$$

This derivation has the proof obligation $\Delta \vdash \mathtt{ds}(\pi\tau, \gamma\tau)\#X$. Let $a \in \mathtt{ds}(\pi\tau, \gamma\tau)$ and consider the atom $\tau(a)$. We have two cases to analyze:

a) If $\tau(a) \in \mathtt{ds}(\pi, \gamma)$ then $\pi(\tau(a)) \not\equiv \gamma(\tau(a))$. Note that in this case $a \in \mathtt{ds}(\pi\tau, \gamma\tau)$. It remains to show a derivation to $\Delta \vdash a\#X$. By assumption we have $\Delta \vdash \tau(a)\#\tau \cdot X$, and by $(\#X)$ $\Delta \vdash a\#X$.

b) If $\tau(a) \notin \mathtt{ds}(\pi, \gamma)$ then $\pi(\tau(a)) \equiv \gamma(\tau(a))$ and $a \notin \mathtt{ds}(\pi\tau, \gamma\tau)$, and we have nothing to prove.

3. Suppose $s \equiv [b]\, t$. Then we have that either $b \in \mathtt{ds}(\pi, \gamma)$ or not. In the first case, observe that the derivation step

$$\frac{\Delta \vdash (\gamma \cdot b\ \pi \cdot b) \cdot (\pi \cdot t) \approx_\alpha \gamma \cdot t \qquad \Delta \vdash \gamma \cdot b\#\pi \cdot t}{\Delta \vdash [\pi \cdot b]\,(\pi \cdot t) \approx_\alpha [\gamma \cdot b]\,\gamma \cdot t}\ \text{(Abs-b)}$$

has two proof obligations: $\Delta \vdash (\gamma \cdot b\ \pi \cdot b) \cdot (\pi \cdot t) \approx_\alpha \gamma \cdot t$ and $\Delta \vdash \gamma \cdot b\#\pi \cdot t$. We derive them in the items below.

a) Notice that $\mathtt{ds}((\gamma \cdot b\ \pi \cdot b) \circ \pi, \gamma) = \mathtt{ds}(\pi, \gamma) \setminus \{b\}$[2]. Now the inductive hypothesis can be applied which give us

$$\frac{\mathtt{ds}((\gamma \cdot b\ \pi \cdot b) \circ \pi, \gamma)\#t}{\Delta \vdash (\gamma \cdot b\ \pi \cdot b) \cdot (\pi \cdot t) \approx_\alpha \gamma \cdot t}\ \text{(I.H)}$$

The dashed line above represent a shorthand notation for the use of the induction hypothesis from the premise $\mathtt{ds}((\gamma \cdot b\ \pi \cdot b) \circ \pi, \gamma)\#t$.

b) Since $b \in \mathtt{ds}(\pi, \gamma)$ then:

$$\begin{aligned} \pi \cdot b \not\equiv \gamma \cdot b &\implies b \not\equiv (\pi^{-1}\gamma) \cdot b \\ &\implies \gamma \cdot b \not\equiv (\gamma\pi^{-1}\gamma) \cdot b \\ &\implies (\pi\pi^{-1}\gamma) \cdot b \not\equiv (\gamma\pi^{-1}\gamma) \cdot b \\ &\implies \pi(\pi^{-1}\gamma) \cdot b \not\equiv \gamma(\pi^{-1}\gamma) \cdot b \end{aligned}$$

So, $\pi^{-1}\gamma \cdot b \in \mathtt{ds}(\pi, \gamma)$ and by assumption $\Delta \vdash \pi^{-1}\gamma \cdot b\#[b]\, t$. Since $\pi^{-1} \neq \mathtt{id}$ we derive $\Delta \vdash \pi^{-1}\gamma \cdot b\#t$ by using $(\#b)$. Finally, by equivariance, $\Delta \vdash \gamma \cdot b\#\pi \cdot t$.

Combining items (1) and (2) above the derivation is done.

In the second case, $b \notin \mathtt{ds}(\pi, \gamma)$, write $c \equiv \pi \cdot b \equiv \gamma \cdot b$ and construct a derivation as follows:

$$\frac{\dfrac{\mathtt{ds}(\pi, \gamma)\#t}{\Delta \vdash \pi \cdot t \approx_\alpha \gamma \cdot t}\ \text{(I.H)}}{\Delta \vdash [c]\,\pi \cdot t \approx_\alpha [c]\,\gamma \cdot t}\ \text{(Abs-a)}$$

4. Finally, if $s \equiv f(t_1, \ldots, t_n)$, then by induction hypothesis, if $\Delta \vdash a\#t_i$ for each $a \in \mathtt{ds}(\pi, \gamma)$ and all $1 \leq i \leq n$ then $\Delta \vdash \pi \cdot t_i \approx_\alpha \gamma \cdot t_i$. Hence, the result follows from (F).

2 Observe that in the composition, $(\gamma \cdot b\ \pi \cdot b) \circ \pi$, b is mapped as $b \mapsto \pi \cdot b \mapsto \gamma \cdot b$ while it is mapped as $b \mapsto \gamma \cdot b$ by γ.

□

Theorem 2.3 (Weakening)**.** Suppose $\Delta \vdash \nabla\sigma$, i.e., $\Delta \vdash a\#X\sigma$, for all $a\#X \in \nabla$. Then

1. $\nabla \vdash b\#s$ implies $\Delta \vdash b\#s\sigma$.
2. $\nabla \vdash s \approx_\alpha t$ implies $\Delta \vdash s\sigma \approx_\alpha t\sigma$.

Proof.

1. The proof is by induction on the derivation of $\Delta \vdash b\#s$.
   a) The base case, rule $(\#ab)$, and the rule $(\#a)$ are straightforward.
   b) The derivation concludes in $(\#X)$:

      Then $\nabla \vdash b\#\pi \cdot X$ is derivable from $\nabla \vdash \pi^{-1} \cdot b\#X$. Using this fact we have $\pi^{-1} \cdot b\#X \in \nabla$. By assumption, $\Delta \vdash \nabla\sigma$. Therefore, in particular, $\Delta \vdash \pi^{-1} \cdot b\#X\sigma$. Equivariance, Corollary 2.1, ensures the existence of a derivation for $\Delta \vdash b\#\pi \cdot (X\sigma)$, and by the Commutation Lemma (Lemma 2.1), $\Delta \vdash b\#(\pi \cdot X)\sigma$.
   c) The derivation concludes in $(\#b)$:

      Then $\Delta \vdash b\#[c]\,s$ and by induction hypothesis $\Delta \vdash b\#s\sigma$. Therefore, by $(\#b)$ derive $\Delta \vdash b\#[c]\,(s\sigma)$.
   d) If the derivation concludes in $(\#f)$ just apply the induction hypothesis and use $(\#f)$.
2. The proof is by induction on the derivation of $\nabla \vdash s \approx_\alpha t$.

   The base case is trivial. We prove the result for the case (Ds), the other cases are similar and simpler.

   Suppose the last rule is (Ds):

   Then $\nabla \vdash \pi \cdot X \approx_\alpha \gamma \cdot X$ is derivable from $\nabla \vdash \mathtt{ds}(\pi, \gamma)\#X$. By assumption, $\Delta \vdash \nabla\sigma$. Hence, $\Delta \vdash a\#X\sigma$ for all $a \in \mathtt{ds}(\pi, \gamma)$. The result now follows from Lemma 2.6.

□

Fix a consistent context Δ. As usual, $\approx_\alpha$ is an equivalence relation if it is reflexive, symmetric, and transitive. Also, $\approx_\alpha$ is a *congruence* if it is an equivalence relation closed by formation of terms, i.e., if $\Delta \vdash s \approx_\alpha t$ then $\Delta \vdash fs \approx_\alpha ft$ and $\Delta \vdash [a]\,s \approx_\alpha [a]\,t$.

The next theorem states that $\approx_\alpha$ is a congruence under a consistent context. That is, the logic generated by $\approx_\alpha$ is an *equality theory* over the set of nominal term $T(\Sigma, \mathbb{A}, \mathbb{X})$.

Theorem 2.4. $\approx_\alpha$ is a congruence in a consistent context.

Proof. Let Δ be a consistent context. We first prove that $\approx_\alpha$ is an equivalence relation.

1. Reflexivity: $\Delta \vdash s \approx_\alpha s$, for all terms s.

   The proof is by induction on s. For a complete proof see Lemma B.1.

2. Symmetry: if $\Delta \vdash s \approx_\alpha t$ then $\Delta \vdash t \approx_\alpha s$.

   The proof is by induction on the depth of the derivation $\Delta \vdash s \approx_\alpha t$, see Lemma B.2.

3. Transitivity: if $\Delta \vdash s \approx_\alpha t$ and $\Delta \vdash t \approx_\alpha u$ then $\Delta \vdash s \approx_\alpha u$.

   This is Lemma B.3

Now that $\approx_\alpha$ is an equivalence relation, it follows by (Abs-a) and (F) that $\approx_\alpha$ is a congruence. □

2.3 NOMINAL CONSTRAINT SOLVING AND UNIFICATION

We first look for a systematic way to answer the questions 'is a fresh for t?', denoted as $a\#^? t$, and 'does there exist a proof for $s \approx_\alpha t$?', denoted as $s \approx_\alpha{}^? t$. We call these questions *constraint problems*, and to answer these questions we study the *Nominal Unification Theory*, which is concerned about finding (if it exists) a substitution σ that solves a constraint problem $s \approx_\alpha{}^? t$ (resp. $a\#^? t$), that is, σ such that $s\sigma \approx_\alpha t\sigma$ (resp. $a\#t\sigma$).

2.3.1 *Nominal Constraint Solving*

Note that since syntactic equality '$\equiv$' between nominal terms s and t is not considered *up-to-α-equivalence*, answering the questions $a\#^? t$ and $s \approx_\alpha{}^? t$ for ground terms ($\mathtt{var}(s,t) = \emptyset$) is nothing but a structural analysis: for instance, a is fresh for a ground term t, if whenever a occur in t it appears under the scope of an abstraction $[a]t'$ in t. However, for non-ground terms ($\mathtt{var}(t) \neq \emptyset$), $a\#^? t$ may depend on freshness assumptions made on variables occurring in t. In this subsection, we give an algorithm to decide these questions for arbitrary terms s and t.

In the definition below we decorate the predicates # and $\approx_\alpha$ with "? " to emphasize that we are interested in the decision problem generated by these predicates.

Definition 2.10. A *constraint problem* Pr is a finite set of freshness constraints of the form $a\#^? t$ (read: "is a fresh in t?") and α-equality constraints of the form $s \approx_\alpha{}^? t$ (read: "is s α-equivalent to t?).

Example 2.3. Consider the constraint problem:

$$Pr = \{a\#^? f(a), [b]\, f(a) \approx_\alpha{}^? [c]\, f(c)\}$$

By using rules in Figures 2.1 and 2.2 we obtain derivations for the constraints (without the question mark):

$$\text{(\#f)}\ \frac{a\#a}{a\#f(a)} \qquad \text{(F)}\ \frac{\dfrac{a \approx_\alpha c}{f(a) \approx_\alpha f(c)} \quad \dfrac{\dfrac{}{c\#a}\,(\#ab)}{c\#f(a)}\,(\#f)}{[b]\, f(a) \approx_\alpha [c]\, f(c)}\,\text{(Abs-b)}$$

- In the first derivation we need to show $a\#a$, which is trivially false.
- In the second, the premise from the left ask us to show that $a \approx_\alpha c$ which again is impossible. From the right we solve the freshness constraint by ending the proof with the axiom $(\#ab)$.

We may conclude that the constraints in the problem are false. Let us work on another example.

Example 2.4. Consider the problem $Pr = \{a\#^?(X\ \lambda[b]\,Y)\}$ in the signature of untyped λ-calculus. There is a derivation of the constraint (without the '?'):

$$\dfrac{a\#X \qquad \dfrac{\dfrac{\dfrac{a\#Y}{a\#[b]\,Y}\,(\#b)}{a\#\lambda[b]\,Y}\,(\#f)}{a\#(X\ \lambda[b]\,Y)}\,(\#f)}$$

Note that the leaves of the derivation tree are the freshness constraints $a\#X$ and $a\#Y$. Therefore, if $a\#X$ and $a\#Y$ then we would get the desired constraint, that is, $a\#X, a\#Y \vdash a\#(X\ \lambda[b]\,Y)$.

Instead of going back-and-forth from $\#^?$ to ${\approx_\alpha}^?$ and $\#$ and $\approx_\alpha$, we introduce the set of Simplification Rules, first proposed by Urban, Pitts, and Gabbay in [29] and later by Fernández and Gabbay in [18], which implements a decision algorithm for solving nominal constraint problems.

$$\begin{aligned}
a\#^?b, Pr &\implies Pr \\
a\#^?\pi\cdot X, Pr &\implies \pi^{-1}(a)\#^?X, Pr \quad \pi \neq \mathtt{id} \\
a\#^?[a]\,t, Pr &\implies Pr \\
a\#^?[b]\,t, Pr &\implies a\#^?t, Pr \\
a\#^?f(t_1,\cdots,f_n), Pr &\implies a\#^?t_1,\cdots,a\#^?t_n, Pr
\end{aligned}$$

$$\begin{aligned}
a \approx_\alpha{}^?\ a, Pr &\implies Pr \\
\pi\cdot X \approx_\alpha{}^?\ \gamma\cdot X, Pr &\implies \mathtt{ds}(\pi,\gamma)\#X, Pr \\
f(s_1,\cdots,s_n) \approx_\alpha{}^?\ f(t_1,\cdots,t_n), Pr &\implies s_1 \approx_\alpha{}^?\ t_1,\cdots,s_n \approx_\alpha{}^?\ t_n, Pr \\
[a]\,t \approx_\alpha{}^?\ [a]\,u, Pr &\implies t \approx_\alpha{}^?\ u, Pr \\
[b]\,l \approx_\alpha{}^?\ [a]\,r, Pr &\implies (a\ b)\cdot l \approx_\alpha{}^?\ r, a\#l, Pr
\end{aligned}$$

Figure 2.3: Simplification rules for unification problems

The rules from Fig. 2.3 defines a reduction relation on problems. Write $Pr \implies Pr'$ when Pr' is obtained from Pr by the application of a simplification rule, and write $\overset{*}{\implies}$ for the transitive and reflexive closure of $\implies$.

Example 2.5. Recalling Examples 2.3 and 2.4 and using the Simplification Rules (Figure 2.3) we obtain the following reductions:

$$\{a\#^? f(a), [b]\, f(a) \approx_\alpha [c]\, f(c)\} \stackrel{*}{\Longrightarrow} \{a\#a, a \approx_\alpha c\}$$

$$\{a\#^? (X\ \lambda[b]\, Y)\} \Longrightarrow \{a\#^? X, a\#^? \lambda[b]\, Y\} \Longrightarrow$$
$$\{a\#^? X, a\#^? [b]\, Y\} \Longrightarrow \{a\#^? X, a\#^? Y\}$$

The entailment relation $\Delta \vdash Pr$ establishes a notion of derivability for problems under a consistent context. As we shall see in the next definition.

Definition 2.11. Let Δ be a consistent context. Write $\Delta \vdash Pr$ when a proof of C exist, for all $C \in Pr$, using the derivation rules from Figure 2.1 and 2.2, and elements of Δ as assumptions. We say that Δ *entails* Pr, written $\Delta \vdash Pr$. In this case we say that Δ is a *solution* of Pr.

TERMINATION OF SIMPLIFICATION RULES. We now prove the termination of the procedure generated by the application of simplification rules.

Definition 2.12 (Size of terms/constraints)**.**

1. The *size of a nominal term* t, denoted as $|t|$, is defined inductively by:

$$\begin{array}{c} |a| = 1 \\ |[a]\, t| = 1 + |t| \end{array} \qquad \begin{array}{c} |\pi \cdot X| = 1 \\ |f(t_1, \ldots, t_n)| = 1 + |t_1| + \cdots + |t_n| \\ |\pi \cdot t| = |t| \end{array}$$

2. The *size of a constraint* is given by

$$|a\#^? t| = |t| \qquad \text{and} \qquad |t \approx_\alpha^? u| = |t| + |u|$$

Lemma 2.7. The relation $\Longrightarrow$ defined by the Simplification Rules in Figure 2.3 is terminating.

Proof. We consider the measure $\mu(Pr)$ of a constraint problem given by the multiset of sizes of the constraints in Pr. We must show that whenever $Pr \Longrightarrow Pr'$ the measure strictly decreases, i.e., $\mu(Pr) >_{mul} \mu(Pr')$. In fact, the rules decreases the measure since each rule either replace a constraint by possible many simple ones or erase the constraint from the problem. The interesting cases are for suspension and abstraction:

$$\begin{aligned} \mu(\pi \cdot X \approx_\alpha^? \gamma \cdot X, Pr) &= \{|\pi \cdot X| + |\gamma \cdot X|\} \cup \mu(Pr) \\ &>_{mul} \{|a\#X| \mid a \in \mathtt{ds}(\pi, \gamma)\} \cup \mu(Pr) \end{aligned}$$

$$\begin{aligned} \mu([b]\, l \approx_\alpha^? [a]\, r, Pr) &= \{|[b]\, l| + |[a]\, r|\} \cup \mu(Pr) \\ &>_{mul} \{|(a\ b) \cdot l| + |r|, |l|\} \cup \mu(Pr) \end{aligned}$$

The other cases follows from the same reasoning. □

Corollary 2.2. The relation $\Longrightarrow$ is confluent.

Proof. Since the simplification rules do not overlap they are locally confluent, by the Newman's Lemma confluence follows by strong normalization, Lemma 2.7. □

As a consequence, the simplification rules define a function from constraint problems Pr to their unique normal forms, denoted as $\langle Pr \rangle_{\mathtt{nf}}$.

Definition 2.13. We say an α-equality constraint $u \approx_\alpha{}^? v$ is *reduced* when one of the following holds:

1. u and v are distinct atoms. For example, $a \approx_\alpha{}^? b$ is a reduced α-equality.
2. u and v are applications with different term-formers (e.g. $f(t) \approx_\alpha{}^? g(s)$).
3. u and v are two different variables (e.g $\pi \cdot X \approx_\alpha{}^? \gamma \cdot Y$ are reduced.)
4. u and v are different term constructors at the root. For example, $[a]\, s \approx_\alpha{}^? f(t)$.

Definition 2.14. We say a freshness constraint $a\#t$ is reduced when it is of the form $a\#a$ or primitive, e.g. $a\#X$. We call the first *inconsistent*. Say a problem Pr is *reduced* when it consists of reduced constraints, and *inconsistent* when it contains an inconsistent constraint.

Lemma 2.8. [Characterization of Normal Forms]

1. $\left\langle a\#{}^? s \right\rangle_{\mathtt{nf}}$ is a context Δ.
   Note that Δ need not be consistent, i.e., $\langle a\#f(a) \rangle_{\mathtt{nf}} = \{a\#a\}$.
2. $\left\langle t \approx_\alpha{}^? u \right\rangle_{\mathtt{nf}}$ is of the form $\Delta \cup Contr \cup Eq$, where Δ is a consistent context, $Contr$ is a set of inconsistent reduced freshness constraints, and Eq is a set of reduced equality constraints.
3. $\langle Pr \rangle_{\mathtt{nf}} = \Delta \cup Contr \cup Eq$, as above.

Proof. We check the simplification rules from Fig. 2.3 and note that a problem is reduced if, and only if, it is in normal form. □

Lemma 2.9.

1. If $Pr \overset{*}{\Longrightarrow} Pr'$, then $\Gamma \vdash Pr$ if, and only if, $\Gamma \vdash Pr'$.
2. $\Gamma \vdash Pr$ if, and only if, $\Gamma \vdash \langle Pr \rangle_{\mathtt{nf}}$.

Proof.

1. The proof follows by induction on the number n of steps in the simplification $Pr \overset{*}{\Longrightarrow} Pr'$. In each inductive step there are 12 rules to consider.

   The base case, $n = 0$, trivially holds.

For the inductive step, suppose $Pr \overset{n-1}{\Longrightarrow} Pr'' \implies Pr'$. There are 12 rules to consider, so we work on one case as an example. The others follow the same reasoning:

By induction hypothesis $\Gamma \vdash Pr$ iff $\Gamma \vdash Pr''$, suppose that Pr'' has the form

$$[b]\, l \approx_\alpha^? [a]\, r, Pr \implies (a\ b) \cdot l \approx_\alpha^? r, a\#l, Pr.$$

By inspection of the derivation rules we can conclude that $\Gamma \vdash [b]\, l \approx_\alpha [a]\, r$ can only be derived by the rule (Abs-b), and it has as proof obligation exactly $\Gamma \vdash (a\ b) \cdot l \approx_\alpha r$ and $\Gamma \vdash a\#l$.

2. Follows directly from the first part since $\Longrightarrow$ is convergent, one can reach the normal form after finitely many steps.

□

Lemma 2.10. If Γ is consistent and $\Gamma \vdash Pr$, then Pr is consistent. Moreover, if Pr is in normal form, then it does not contain equality constraints.

Proof. By Lemma 2.9, a consistent context cannot derive an inconsistent Pr. □

Theorem 2.5. Let Pr be a problem and consider $\langle Pr \rangle_{\mathtt{nf}}$ its formal form, i.e., $\langle Pr \rangle_{\mathtt{nf}} = \Delta \cup Contr \cup Eq$, and Γ be a consistent context. Then $\Gamma \vdash Pr$ if, and only if, $Contr$ and Eq are empty.

Proof. By Lemma 2.9, $\Gamma \vdash Pr$ if, and only if, $\Gamma \vdash \Delta, Contr, Eq$. The result now follows from Lemma 2.10 since Γ is consistent. □

Theorem 2.6. Let Γ and Δ be consistent contexts, and Pr and Pr' be any problems. Then

1. The algorithm generated by application of simplification rules is correct: $\Gamma \vdash Pr$ if, and only if, $\langle Pr \rangle_{\mathtt{nf}} = \Delta$ and $\Gamma \vdash \Delta$.

2. The entailment relation on problems satisfies a form of the cut rule:

$$\frac{\Gamma \vdash \Delta \qquad \Gamma, \Delta \vdash \Psi}{\Gamma \vdash \Psi}$$

Proof.

1. Suppose $\Gamma \vdash Pr$. By Theorem 2.5, $\langle Pr \rangle_{\mathtt{nf}} = \Delta$ and, by Lemma 2.9, $\Gamma \vdash \Delta$. Conversely, if $\langle Pr \rangle_{\mathtt{nf}} = \Delta$ and $\Gamma \vdash \Delta$, then using the same results, it follows $\Gamma \vdash Pr$.

2. Suppose $\Gamma \vdash \Delta$ and $\Gamma, \Delta \vdash \Psi$. Then, $\Gamma \vdash C$ for each $C \in \Delta$. Hence, $\Delta \subseteq \Gamma$ and $\Psi \subseteq \Gamma$. Therefore, $\Gamma \vdash \Psi$, as required.

□

2.3.2 *Equational Problems*

In general, *Equational Problems*, i.e., problems in which the main predicate is equality, usually splits into three cases:

- *equality checking* asks if two terms s and t are *equal*. A solution is a value `true` or `false`.
- *unification* asks if two terms s and t *can be made equal*. A solution is a substitution σ such that $s\sigma = t\sigma$.
- *matching* asks if a term s can be *matched* with some term t. A solution is a substitution σ such that $t = s\sigma$.

In the nominal setting there is a version of these problems above as well. This is expected since nominal is 'equational' in its very nature. Note that we have studied the first problem in the previous subsection, constraint solving solves the *nominal equality check* (α-equality check). Since α-equality is defined in terms of the freshness predicate, all solutions of the nominal versions of the problems above are restricted to a freshness context. For instance, the α-equality checking problem $[a]\,X \approx_\alpha [b]\,X$ is `true` when restricted to the context $a\#X, b\#X$.

Several works have been done towards the solvability/decidability of α-equality problems, and due to its importance it is useful to have at least α-equality check and matching algorithms that run as fast as possible.

In a nominal equational problem, terms s and t are called *ground* if $\mathtt{var}(s,t) = \emptyset$, *linear* if each variable occurs at most once and *non-linear* if a variable can occur multiple times in s or t. In Table 2.1, we transcribe the complexity of each one of these cases proposed by Calvèz and Fernández [11].

Case	Alpha-equivalence	Matching
Ground	linear	linear
Non-ground and linear	log-linear	log-linear
Non-ground and non-linear	log-linear	quadratic

Table 2.1: Complexity of Matching and α-equality check.

2.3.3 *Nominal Unification*

As discussed above, nominal unification is the problem of deciding whether two nominal terms can be made α-equivalent by instantiating their variables. We usually are not interested only in a decision procedure but also in constructively building a substitution that solves the problem. Urban, Pitts and Gabbay in [29] showed that nominal unification is decidable, and gave an algorithm which finds the *most general solution* (Definition 2.16) to a nominal unification problem, if one exists. The nominal unification algorithm in [29] is very similar

with the classical Martelli and Montanari [7] approach to fist-order unification: nominal unification problems are transformed by a set of simplification rules on problems and the solution is computed along the way.

Definition 2.15. A *nominal unification problem* Pr is a pair $\langle \nabla, P \rangle$ consisting of a consistent freshness context ∇ and a finite set P of freshness constraints of the form $a\#^? t$ and α-equality constraints (for short, α-equations) of the form $s \approx_\alpha^? t$.

Definition 2.16. A *solution* for a nominal unification problem $Pr = \langle \nabla, P \rangle$ is a pair of the form $\langle \Gamma, \sigma \rangle$ where Γ is a consistent context and σ a substitution such that the following conditions hold:

1. $\Gamma \vdash \nabla\sigma$;
2. $\Gamma \vdash a\#t\sigma$, for all $a\#^? t \in P$;
3. $\Gamma \vdash t\sigma \approx_\alpha s\sigma$, for all $s \approx_\alpha^? t \in P$;
4. $X\sigma \equiv X\sigma\sigma$.

If there is no such $\langle \Gamma, \sigma \rangle$ we say that Pr is *unsolvable*.

Let Pr be a unification problem as above, $\mathcal{U}(Pr)$ denote the set of all solutions of Pr. Solutions are compared by the following partial order, called *instantiation ordering*.

Definition 2.17. Let Γ_1 and Γ_2 be consistent contexts, and σ_1 and σ_2 substitutions. Then $\langle \Gamma_2, \sigma_2 \rangle$ is *an instance* of $\langle \Gamma_1, \sigma_1 \rangle$ on a set of variables $\mathbb{V} \subseteq \mathbb{X}$, denoted by $\langle \Gamma_1, \sigma_1 \rangle \leq^{\mathbb{V}} \langle \Gamma_2, \sigma_2 \rangle$, when there exists some substitution δ such that

$$\text{for all } X \in \mathbb{V},\ \Gamma_2 \vdash X\sigma_1\delta \approx_\alpha X\sigma_2 \qquad \text{and} \qquad \Gamma_2 \vdash \Gamma_1\delta$$

If we want to be more specific we may write $\langle \Gamma_1, \sigma_1 \rangle \leq_\delta^{\mathbb{V}} \langle \Gamma_2, \sigma_2 \rangle$.

In the rest of this work the set of variables $\mathbb{V}$ will consist of the variables of the problem, i.e., $\mathbb{V} = \mathtt{var}(Pr)$, and the superscript will be omitted.

Lemma 2.11. The instantiation ordering $\leq$ defines a partial order on $\mathcal{U}(Pr)$.

Proof. The proof can be found in [18], Lemma 29. □

A *least element* of a partially ordered set is one which is related via $\leq$ to every element of the set.

Definition 2.18. A *most general* (or principal) solution to a problem Pr is a least element of $\mathcal{U}(Pr)$.

UNIFICATION ALGORITHM. The nominal unification algorithm, called `unify` in the next sections, consists of applying the simplification rules, enriched with the instantiation rules given below, to a problem $Pr = \langle \Delta, P \rangle$ until no more rules can be applied.

$$\pi \cdot X \approx_\alpha^? t, Pr \overset{[X/\pi^{-1}\cdot t]}{\Longrightarrow} Pr[X/\pi^{-1} \cdot t], \text{ if } X \notin \mathtt{vars}(t)$$
$$t \approx_\alpha^? \pi \cdot X, Pr \overset{[X/\pi^{-1}\cdot t]}{\Longrightarrow} Pr[X/\pi^{-1} \cdot t], \text{ if } X \notin \mathtt{vars}(t)$$

Remark 2.5. The instantiating rules above only deals equational constraints. If we have an equality constraint $X \approx_\alpha^? t$ it is obvious we should instantiate X to t, but there is no obvious most general instantiation making the freshness constraints $a\#^? X$ true since a freshness constraint $a\#X$ have the meaning: a cannot occur free in any instance of X. This induces infinitely many substitutions, for instance, $X/f(b), X/g(f(c)), \ldots$, therefore, we represent these infinitely many possible substitutions by the constraint $a\#X$, which says that we can instatiate X to any term as long as it does not have free occurrences of a.

Example 2.6. Consider the signature of lambda-calculus as in Example 2.1 and the problem below. We apply the algorithm `unif` to the problem to get:

$$\begin{aligned}
\{(\lambda[a]\, X)Z \approx_\alpha^? (\lambda[b]\, Y)b\} &\Longrightarrow \{\lambda[a]\, X \approx_\alpha^? \lambda[b]\, Y, Z \approx_\alpha^? b\} \\
&\overset{[Z/b]}{\Longrightarrow} \{\lambda[a]\, X \approx_\alpha^? \lambda[b]\, Y\} \\
&\overset{*}{\Longrightarrow} \{(b\ a) \cdot X \approx_\alpha^? Y, b\#^? X\} \\
&\overset{[Y/(b\ a)\cdot X]}{\Longrightarrow} \{b\#^? X\}
\end{aligned}$$

To form the solution pair $\langle \Gamma, \sigma \rangle$, we form Γ with the remaining constraints and take the composition of the labelled substitutions used along the way of computation as σ.

Solution: $\langle b\#X, [Z/b, Y/(b\ a) \cdot X] \rangle$

Lemma 2.12. The unification procedure given by the simplification rules from Fig. 2.3 and instantiating rules terminates.

Proof. We form a lexicographic measure on unification problems given by $\mu'(Pr) = (n, \mu(P))$ where

- n is the number of variables of the problem,
- $\mu(P)$ is the multiset measure defined in Lemma 2.7.

Note that at each application of rules in a unification problem either the number of variables decreases by one (decreasing the measure) or the number of variales on the problem remains unchanded and the complexity $\mu(P)$ decreases, as we have showed in Lemma 2.7. □

Definition 2.19. A unification problem $s \approx_\alpha^? t$ is *reduced* when one of the following holds:

1. s and t are distinct atoms.

2. Precisely one of s and t is a moderated variable and the other term mentions that variable.
3. s and t are applications with different term-formers.
4. s and t have different term constructors at the root and neither is a moderated variable.

We call all reduced unification constraints *inconsistent*.

Normal forms are unique modulo renaming of variables, as in standard first-order unification. The normal form of a unification problem Pr by $\Longrightarrow$ is defined as expected and denoted by $\langle Pr \rangle_{\mathtt{nf}}$. It consists of a set of equations and freshness constraints in reduced form. Pr has a solution iff $\langle Pr \rangle_{\mathtt{nf}}$ contains only consistent reduced freshness constraints, i.e., freshness constraints of the form $a\#X$.

Lemma 2.13 (Unification Normal Forms)**.**

- $\langle a\#s \rangle_{\mathtt{nf}} = \langle \langle a\#s \rangle_{\mathtt{nf}}, \mathtt{id} \rangle$
- $\left\langle s \approx_\alpha^? t \right\rangle_{\mathtt{nf}} = \langle \Delta \cup Contr \cup Eq, \sigma \rangle$, where Δ is a consistent freshness context, $Contr$ is an inconsistent freshness context, Eq is a set of inconsistent unification constraints, and σ is a substitution.

Regarding the complexity of nominal unification, Calvès and Fernández [10] gave an polynomial implementation for the nominal unification problem. However, the actual complexity of the problem is still a open problem since Higher-order pattern unification, which is a closely related problem [12], is linear [27].

MATCHING PROBLEMS In Chapter 3, we solve a *matching problem (in context)*. We briefly describe this kind of problems here. For an efficient implementation the reader is referred to [11].

Definition 2.20. A *matching problem (in context)* is a pair $(\Delta \vdash t) \approx_? (\nabla \vdash r)$, where ∇, Δ are consistent contexts and t, r are nominal terms. The solution to this matching problem, if it exists, is a substitution θ such that:

- $\left\langle \nabla, t \approx_\alpha^? r \right\rangle_{\mathtt{nf}} = \langle \Delta', \theta \rangle$.
- $\Delta \vdash \Delta'$.
- $X\theta \equiv X$ for $X \in \mathtt{vars}(\Delta, t)$

We say that θ solves the matching problem.

A matching problem can be seen as a particular kind of unification problem. The conditions in the definition above ensure that: $\Delta \vdash t \approx_\alpha r\theta$ and $\Delta \vdash \nabla\theta$, and so $\langle \Delta, \theta \rangle \in \mathcal{U}\left(\nabla, t \approx_\alpha^? r\right)$. We can think of the solution to $(\Delta \vdash t) \approx_? (\nabla \vdash r)$ as a unification problem such that it solves $\left\langle \Delta, t \approx_\alpha^? r \right\rangle$ without instantiating any variables in t. Also, denote by $\mathtt{matching}((\Delta \vdash t) \approx_? (\nabla \vdash r)) = \theta$ the call for this algorithm with solution θ.

Example 2.7.

1. Consider the matching problem $b\#X \vdash X \approx_? Y, b\#X \vdash (a\ b) \cdot X \approx_? Y$.

$$\{b\#X, X \approx_? Y, (a\ b) \cdot X \approx_? Y\} \overset{[Y/X]}{\Longrightarrow} \{b\#X, (a\ b) \cdot X \approx_? Y\}$$
$$\Longrightarrow \{a\#X, b\#X\}$$

Solution for the Unification problem: $\langle \{a\#X, b\#X\}, [Y/X] \rangle$

Notice that by the second condition of Definition 2.20 this matching problem does not have a solution. In fact, one has $b\#X \nvdash a\#X, b\#X$.

2. Consider the problem $b\#X, X \approx_? a, (b\ a) \cdot X \approx_? Y$.

$$\{b\#X, X \approx_? a, (b\ a) \cdot X \approx_? Y\} \overset{[Y/(a\ b)\cdot X]}{\Longrightarrow} \{b\#X, X \approx_? a, (b\ a) \cdot X \approx_? (a\ b) \cdot X\}$$
$$\Longrightarrow \{b\#X, X \approx_? a\}$$

This problem also does not have a solution.

Remark 2.6. From the above examples one can see that Definition 2.20 does not express only matching modulo α-equivalence since we can use ∇ to specify constraints which must be satisfied by the matching solution. When the conditions in ∇ are satisfied we say the matching **is triggered**.

3

NOMINAL UNIVERSAL ALGEBRA

This chapter concentrates on two important tasks: the first is that of defining *nominal algebras* and give to them a semantic in nominal sets [22]; the second is to show that a collection of these algebras is equational (Definition 3.28) if, and only if, they form a *variety* [19] (Definition 3.27); this is called HSP Theorem (Theorem 3.12).

3.1 A MORE GENERAL DERIVATION SYSTEM

An equality assertion is a pair $t = u$ where t and u are terms. A nominal theory T is a pair $T = (\Sigma, Ax)$ consisting of a signature Σ and a set of axioms (in the form of equality judgements) $\nabla \vdash t = u$, with terms t and u built over Σ, and ∇ a consistent context.

Derivations in a theory $T = (\Sigma, Ax)$ are defined by the rules in Figures 2.1 and 3.3. We say Π is a *valid* derivation in T when the following two conditions are satisfied:

- Π mentions only terms built over Σ.
- Π mentions only instances of $(ax_{\nabla \vdash t=u})$ such that $(\nabla \vdash t = u) \in Ax$.

As in Chapter 2, $\Delta, \nabla, \Gamma \ldots$ represent *consistent* freshness contexts. Write $\Delta \vdash s = t$, if there exists a valid derivation Π using the elements in Δ as assumptions. We may abbreviate '$\Delta \vdash s = t$ is derivable in T' to the affirmative statement '$\Delta \vdash_T s = t$'.

Example 3.1. Let us consider some classical examples of nominal theories.

(a) `CORE` is a family of theories with no axioms. Theorem 3.5 shows that derivability in `CORE` (using rules from Figs. 2.1 and 3.3), i.e., $\Delta \vdash_{\texttt{CORE}} s = t$, is equivalent to derivability in the theory of α-equivalence (using the rules from Figs. 2.1 and 2.2), $\Delta \vdash s \approx_\alpha t$, as defined in Chapter 2.

(b) In some applications of nominal algebra it is useful to explicitly express the behavior of a capture avoiding substitution operator, denoted by `sub`, in an algebraic treatment. Therefore, we define below in Fig. 3.1 the axioms of the theory `SUB`, first introduced by Gabbay in [20].

(c) Using the signature from Example 2.1 the theory `LAM` gives the first complete algebraic treatment for the λ-calculus [21]. The theory `LAM` is defined by the axioms for `SUB` together with a (β)-axiom, to represent β-conversion, and an (η)-axiom, to represent η-conversion, as defined below in Fig. 3.2.

$\mathrm{var}{\mapsto}$	$\vdash$	$a[a \mapsto X] = X$
$(\# \mapsto)$	$a\#Y \vdash$	$Y[a \mapsto Y] = Y$
$(f \mapsto)$	$\vdash$	$f(Y_1,\ldots,Y_n)[a \mapsto X] = f(Y_1[a \mapsto X],\ldots,Y_n(a \mapsto X))$
$(\mathrm{abs} \mapsto)$	$b\#X \vdash$	$([b]\,Y)[a \mapsto X] = [b]\,(Y[a \mapsto X])$
$(\mathrm{id} \mapsto)$	$\vdash$	$Y[b \mapsto b] = Y$
$(\eta \mapsto)$	$a\#X \vdash$	$[a]\,\mathtt{sub}(X,a) = X$

For each term-former f (including sub), there is one axiom $(f \mapsto)$.

Figure 3.1: Axioms for the theory SUB.

(β)	$\vdash$	$(\lambda[a]\,Y)X = Y[a \mapsto X]$
(η)	$a\#X \vdash$	$\lambda[a]\,(X\,a) = X$

Figure 3.2: Axioms for (β) and (η) conversion.

Observe that in the (η)-axiom the freshness side-conditions of η-conversion

$$\lambda x.(M\ x) =_\eta M \qquad \text{if } x \text{ is not free in } M$$

is expressed, in nominal syntax, as the freshness judgement $a\#X \vdash \lambda[a]\,(X\ a) = X$.

$$\frac{}{t = t}\ (\mathrm{refl}) \qquad \frac{t = u}{u = t}\ (\mathrm{symm})$$

$$\frac{s = u \qquad u = t}{s = t}\ (\mathrm{trans})$$

$$\frac{t = u}{[a]\,t = [a]\,u}\ (\mathrm{cong}[])$$

$$\frac{t = u}{f(t_1,\ldots,t,\ldots,t_n) = f(t_1,\ldots,u,\ldots,t_n)}\ (\mathrm{cong}f)$$

$$\frac{\nabla^\pi\sigma}{t^\pi\sigma = u^\pi\sigma}\ (ax_{\nabla\vdash t=u}) \qquad \frac{a\#t \qquad b\#t}{(a\ b)\cdot t = t}\ (\mathrm{perm})$$

$$\begin{array}{c} [a\#X_1,\ldots,a\#X_n] \quad \Delta \\ \vdots \\ t = u \end{array}$$
$$\overline{\qquad\qquad t = u}\ (\mathrm{fr})\ {\scriptstyle n \geq 1,\ a \notin \mathtt{atms}(t,u,\Delta)}$$

Figure 3.3: Derivation rules for equality

The (fr) rule permit the introduction of freshness assumptions $[a\#X_1, \ldots, a\#X_n]$ into the derivation of $t = u$ from assumptions Δ, for the newly choosen atom a not occuring in t, u and Δ. These assumptions behave like in the introduction rule $\rightarrow_i$ in natural deduction, they need to be *discharged*. Assumptions are usually labelled with natural numbers and get the same label for all assumptions on the same atom. The rules (ref), (symm), and (trans) ensure that equality is in fact an equivalence relation, whereas (cong[]) and (cong$f$) states that equality is a congruence.

3.1.1 *Instantiating Axioms*

The reader may now have more intuition on the motivations for the 'permutative convention': atoms are identified by its name, so two atoms a and b are always considered different objects. This convention is embedded in the α-equality theory itself for instance $a \approx_\alpha b$ cannot be derived in CORE. But nominal theories permit the use of equality axioms to generate theories that may have provable equality between two atoms[1]. For example, from the theory T with a single axiom $\vdash a = f(X)$ one may derive:

$$\dfrac{\dfrac{(ax_{\vdash a=f(X)})}{a = f(X)[X/c]}\quad \dfrac{\dfrac{}{(a\ b)\cdot a = (a\ b)\cdot f(X)[X/c]}(ax_{\vdash a=f(X)})}{b = f(c)}}{\dfrac{a = f(c)}{a = b}\ (\text{trans})}\ (\text{symm})$$

In the derivation above dashed lines represent the computation of substitutions and permutation actions when we instantiate the axiom $\vdash a = f(X)$.

Note that by the form of the axiom rule, $\dfrac{\nabla^\pi\sigma}{t^\pi\sigma = u^\pi\sigma}\,(ax_{\nabla\vdash t=u})$, in order to 'instantiate' the axiom $\Delta \vdash t = u$ using a permutation π and substitution σ one first is required to derive the proof-obligation $\nabla^\pi\sigma$. For instance, the (η)-axiom for η-conversion in Fig. 3.2;

$$a\#X \vdash \lambda[a]\,(X\,a) = X$$

states that 'provided a is fresh for any instance of X one can derive $\lambda[a]\,(X\,a) = X$'. The rule $(ax_{a\#X\vdash\lambda[a](X\,a)=X})$ (abbreviated as (ax_η) for simplicity) derives an instantiated version of (η) provided one can give derivations of the freshness condition $(a\#X)^\pi\sigma$. For example;

$$(\eta)\ \dfrac{(\#b)\ \dfrac{}{a\#b}}{\lambda[a]\,(b\ a) = b} \qquad\qquad \dfrac{a\#a}{\lambda[a]\,(a\ a) = a}$$

the derivation on the left is valid, whereas the derivation on the right is not.

1 The permutative convention still remains valid: two atoms can be 'provable equal' in a nominal theory but in our metamathematical setting for nominal techniques these atoms are still considered different.

3.1.2 *Permutating Atoms: α-equivalence*

The (perm) rule is used to express α-equivalence. To see this, the following derivations are valid in CORE (the theory with no axioms):

$$\dfrac{(\#b)\ \dfrac{(\#a)\ \dfrac{}{a\#b}}{a\#[b]\,b} \qquad \dfrac{}{b\#[b]\,b}\,(\#b)}{[a]\,a = [b]\,b}\,(\text{perm})$$

$$\dfrac{(\#b)\ \dfrac{a\#X}{a\#[b]\,X} \qquad \dfrac{}{b\#[b]\,X}\,(\#a)}{[a]\,(b\ a)\cdot X = [b]\,X}\,(\text{perm})$$

Then $\vdash_{\texttt{CORE}} [a]\,a = [b]\,b$ and $a\#X \vdash_{\texttt{CORE}} [a]\,(b\ a)\cdot X = [b]\,X$. To see that the last step of the derivations above are really instances of (perm), note that $[a]\,a \equiv (b\ a)\cdot[b]\,b$ and $[a]\,(b\ a)\cdot X \equiv (b\ a)\cdot[b]\,X$.

3.1.3 *Proof-theoretical Results*

In this section, we basically states the same results from Subsection 2.2.1. The proofs follow the same spirit as before but now in a more general setting.

EQUIVARIANCE. The first important property about nominal theories is that derivations are closed under applications of permutations.

Definition 3.1. We extend notation for t^π and Δ^π to nominal theories: given a nominal theory $T = (\Sigma, Ax)$ we write $T^\pi := (\Sigma, Ax^\pi)$ where Ax^π is such that $\nabla^\pi \vdash t^\pi = u^\pi \in Ax^\pi$ if, and only if, $\nabla \vdash t = u \in Ax$.

We use the technical lemma below to reason about meta-level equivariance. That is, derivability of $\Delta \vdash_T t = u$ is closed under the meta-action T^π of a permutation in the the theory T.

Lemma 3.1. If $\Delta \vdash_T t = u$ then $\Delta \vdash_{T^\pi} t = u$.

Proof. By induction on derivations, all cases are done by an analysis of the last rule applied in the derivation of $\Delta \vdash_T t = u$. We work on one non-trivial case, that is, the case for the axiom rule ($ax_{\nabla\vdash t=u}$). Therefore, if the last rule applied is the axiom rule our derivation has the form

$$\dfrac{\nabla^\gamma\sigma}{t^\gamma\sigma = u^\gamma\sigma}$$

and by induction hypothesis we need to show that

$$\dfrac{\text{I.H}}{\dfrac{\Delta \vdash_{T^\pi} \nabla^\gamma\sigma}{\Delta \vdash_{T^\pi} t^\gamma\sigma = u^\gamma\sigma}}$$

By Lemma 2.2, this is equivalent to show that

$$\dfrac{\Delta \vdash_{T^\pi} \nabla^{\pi^{\gamma\pi^{-1}}}\sigma}{\Delta \vdash_{T^\pi} t^{\pi^{\gamma\pi^{-1}}}\sigma = u^{\pi^{\gamma\pi^{-1}}}\sigma}$$

Finally, notice that this is nothing but an instance of $(ax_{\nabla^{\pi} \vdash t^{\pi} = u^{\pi}})$ taking as permutation $\gamma\pi^{-1}$ and substitution σ, and using Lemma 2.2 in the same way as before. □

Theorem 3.1 states that derivable equality is closed under permutation at the *meta-level*.

Theorem 3.1 (Meta-level equivariance)**.** For any permutation π:

1. if $\Delta \vdash a\#t$ then $\Delta^{\pi} \vdash \pi(a)\#t^{\pi}$,
2. if $\Delta \vdash_T t = u$ then $\Delta^{\pi} \vdash_T t^{\pi} = u^{\pi}$.

Proof.

1. Follows directly from Equivariance (Theorem A.1).
2. Suppose $\Delta \vdash_T t = u$. By Equivariance (Theorem A.1) follows that

   $$\Delta^{\pi} \vdash_{T^{\pi}} t^{\pi} = u^{\pi}.$$

   Now using Lemma 3.1 we obtain

   $$\Delta^{\pi} \vdash_{T^{\pi\pi^{-1}}} t^{\pi} = u^{\pi}$$

   Using Lemma 2.2 we conclude that $T^{\pi\pi^{-1}} \equiv T^{\pi\pi^{-1}} \equiv T$, and then

   $$\Delta^{\pi} \vdash_T t^{\pi} = u^{\pi}.$$

□

The next theorem is the analogous of Theorem 2.1 to general nominal theories.

Theorem 3.2 (Object Level Equivariance)**.** For any permutation π:

1. if $\Delta \vdash a\#t$ then $\Delta \vdash \pi(a)\#\pi \cdot t$,
2. if $\Delta \vdash_T t = u$ then $\Delta \vdash_T \pi \cdot t = \pi \cdot u$.

Proof. The proof is by induction on derivations, by an analysis of the last rule applied.

1. It follows by the same proof given in Theorem 2.1, since freshness derivations does not depend on the equational theory T we are considering.
2. The proof follows the same lines of the proof for Theorem 2.1. We work on the non-trivial case, that is, the case for axiom rule.

   Suppose that the last rule applied is $(ax_{\nabla \vdash t=u})$:

   Them $\Delta \vdash_T t^{\gamma}\sigma = u^{\gamma}\sigma$ is derived from $\Delta \vdash \nabla^{\gamma}\sigma$. The goal is to derive

   $$\Delta \vdash_T \pi \cdot t^{\gamma}\sigma = \pi \cdot u^{\gamma}\sigma$$

By the Commutation Lemma (Lemma 2.1) this is equivalent to derive $\Delta \vdash_T (\pi \cdot t^{\pi})\sigma = (\pi \cdot u^{\gamma})\sigma$. Now define a substitution σ' that maps each $X \in \mathtt{var}(\Delta, t, u)$ to $\pi \cdot X$, then by Lemma 2.3 we have to derive

$$\Delta \vdash_T t^{\gamma^{\pi}}(\sigma'\sigma) = u^{\gamma^{\pi}}(\sigma'\sigma).$$

By Lemma 2.2 this is equivalent to $t^{\pi\gamma}(\sigma'\sigma) = u^{\pi\gamma}(\sigma'\sigma)$, and it follows from $(ax_{\nabla \vdash t=u})$ by taking

$$\frac{\nabla^{\pi\gamma}(\sigma'\sigma)}{t^{\pi\gamma}(\sigma'\sigma) = u^{\pi\gamma}(\sigma'\sigma)}\,(ax_{\nabla \vdash t=u})$$

this is equivalent to

$$\frac{\pi \cdot \nabla^{\gamma}\sigma}{t^{\pi\gamma}(\sigma'\sigma) = u^{\pi\gamma}(\sigma'\sigma)}\,(ax_{\nabla \vdash t=u})$$

Finally, by the induction hypothesis, $\Delta \vdash \pi \cdot \nabla^{\pi}\sigma$ and the result follows.

□

SUBSTITUTION AND STRENGTHENING. We can apply a substitution of terms for variables provided those terms violate no freshness assumptions imposed on the variables.

Theorem 3.3. Suppose Δ', Δ and σ are such that $\Delta' \vdash a\#t\sigma$, for every $a\#t \in \Delta$. Then, the following hold:

1. If $\Delta \vdash a\#t$ then $\Delta' \vdash a\#t\sigma$.
2. If $\Delta \vdash_T t = u$ then $\Delta' \vdash t\sigma = t\sigma$.

Proof.

1. It follows from the proof for the first part of Theorem 2.3, since freshness derivations do not depends on the theory T.
2. The proof follows by induction on the derivation of $\Delta \vdash_T t = u$. For (fr) we use Equivariance (Theorem A.1) to rename the freshly chosen atom a if it is mentioned by $\Delta', t\sigma$ or $u\sigma$.

□

With the next theorem we can remove freshness assumptions that does not appear anywhere in s or t. This property is called *strengthening*.

Theorem 3.4 (Strengthening)**.** Suppose $a \notin \mathtt{atms}(s, t)$. Then:

1. $\Delta, a\#X \vdash_T b\#s$ implies $\Delta \vdash_T b\#s$.
2. $\Delta, a\#X \vdash_T s \approx_{\alpha} t$ implies $\Delta \vdash_T s \approx_{\alpha} t$.

Proof. The freshness case follows from the proof for Theorem 2.2, since freshness derivations do not depend on the theory T, and note that the equational case is exactly the rule (fr). □

THE EQUIVALENCE (UNDER DERIVABILITY) OF $\approx_\alpha$ AND CORE. Example 3.1-(a) has described a nominal theory called CORE, i.e., the theory without any axiom. The next theorem establishes connection between derivability in the deduction system defining $\approx_\alpha$ (rules in Figs. 2.1 and 2.2) and derivability in the theory for CORE (rules in Figs. 2.1 and 3.3). This equivalence has an important application since $\approx_\alpha$ is practical for implementation purposes, on the other hand, CORE is more suitable for theoretical purposes.

Theorem 3.5. $\Delta \vdash_{\texttt{CORE}} t = u$ is derivable if, and only if, $\Delta \vdash t \approx_\alpha u$ is derivable.

Proof. Suppose $\Delta \vdash_{\texttt{CORE}} t = u$, the proof now follows from an inductive reasoning on the depth of derivations. By using the induction hypothesis it suffices to show:

1. $\approx_\alpha$ is an equivalence relation and a congruence. This follows by Theorem 2.4.

2. If $\Delta \vdash a\#t$ and $\Delta \vdash b\#t$ then $\Delta \vdash (a\ b) \cdot t \approx_\alpha t$.

   The proof can be found in Appendix B, Lemma B.4.

3. $\Delta, a\#X_1, \ldots, a\#X_n \vdash_{\texttt{CORE}} t = u$ where $a \notin \texttt{atms}(t, u, \Delta)$ then $\Delta \vdash t \approx_\alpha u$. By induction hypothesis,

   $$\Delta, a\#X_1, \ldots, a\#X_n \vdash t \approx_\alpha u$$

   and the result follows by Strengthening (Theorem 2.2).

Conversely, the proof is by induction on the derivation of $\Delta \vdash t \approx_\alpha u$. As usual, we do an analysis of the last rule applied in the derivation of $\Delta \vdash t \approx_\alpha u$, we work on some cases:

1. The base case, $\Delta \vdash a \approx_\alpha a$, is proved in CORE as an instance of (refl).

2. Suppose the last rule is (F): then by induction hypothesis we get $\Delta \vdash_{\texttt{CORE}} t_i = u_i$ for $1 \leq i \leq n$, then $\Delta \vdash_{\texttt{CORE}} f(t_1, \ldots, t_n) = f(u_1, \ldots, u_n)$ is derivable from instances of (trans) and (cong f).

3. The last rule if (Abs-a): then by induction hypothesis, $\Delta \vdash_{\texttt{CORE}} t = u$ then derive $\Delta \vdash_{\texttt{CORE}} [a]\, t = [a]\, u$ by using (cong []).

4. The last rule is (Abs-b): the inductive hypothesis provide derivations Π of $\Delta \vdash_{\texttt{CORE}} (b\ a) \cdot t = u$ and Π' of $\Delta \vdash b\#t$. The following derivation is a proof of $\Delta \vdash_{\texttt{CORE}} [a]\, t = [b]\, u$.

$$\dfrac{\dfrac{\dfrac{\dfrac{\Pi'}{b\#t}(\#b) \qquad \dfrac{}{a\#[a]\, t}(\#a)}{b\#[a]\, t}(\text{perm})}{[b]\, (b\ a) \cdot t = [a]\, t}(\text{symm}) \qquad \dfrac{\Pi}{(b\ a) \cdot t = u}}{[a]\, t = [b]\, u}\ \begin{matrix}(\text{cong}[]) \\ (\text{tran})\end{matrix}$$

□

3.2 DENOTATIONAL SEMANTICS

We give the basic notions about *nominal sets*, which provide the underlying set theory for which an interpretation of nominal terms and algebras is more suitable. Nominal Sets enable the study of properties which are invariant under permutation of names, and were originally introduced by Gabbay and Pitts [23] but the mathematical foundations goes back to the 30's set theory and the logic of ZFA (based on the Zermelo-Fraenkel). For more details, we adress the reader to [26].

3.2.1 *Nominal Sets*

Recall that we denote by $\mathbb{A}$ the set of all atoms and by $\mathbb{P}$ the set of all permutations $\pi : \mathbb{A} \to \mathbb{A}$ with finite support. We give to $\mathbb{P}$ a group structure by taking $\circ$ (permutation composition) as the operation in $\mathbb{P}$. To ease notation denote the group $(\mathbb{P}, \circ)$ just by $\mathbb{P}$.

Definition 3.2. A $\mathbb{P}$-action $\cdot$ on a set X is a function $\cdot : \mathbb{P} \times X \to X$, written in infix notation as $\pi \cdot x$, satisfying:

- $\mathtt{id} \cdot x = x$,
- $\pi \cdot (\gamma \cdot x) = (\pi \circ \gamma) \cdot x$, for all $\pi, \gamma \in \mathbb{P}$ and for all $x \in X$.

In this case, call the pair $(X, \cdot)$ a $\mathbb{P}$-set. We also say that the action $\cdot$ *fix* $x \in X$ when $\cdot$ acts trivially on x, i.e., $\pi \cdot x = x$.

Example 3.2.

1. If $\mathbb{P}'$ is any subgroup of $\mathbb{P}$, we get a $\mathbb{P}'$-action on the set of atoms $\mathbb{A}$ by defining $\cdot$ as function application such that $\pi \cdot a = \pi(a)$.
2. In a more general setting we let Σ be a (single-sorted) algebraic signature.Thus $\Sigma = (\Sigma_n \mid n \in \mathbb{N})$ is a countable infinite family of sets. The elements of each Σ_n are the n-ary operations of the signature. We can then inductively build the set $T(\Sigma, \mathbb{X})$ of algebraic terms with variables from $\mathbb{X}$ using the rules:

$$\frac{\dfrac{x \in \mathbb{X}}{x \in T(\Sigma, \mathbb{X})} \quad t_1 \in T(\Sigma, \mathbb{X}) \quad \cdots \quad t_n \in T(\Sigma, \mathbb{X}) \qquad f \in \Sigma_n}{f(t_1, \ldots, t_n) \in T(\Sigma, \mathbb{X})}$$

   This is the way we usually construct terms for instance in first-order languages.

   There is an action of $Sym(\mathbb{X})$ (the symmetric group of $\mathbb{X}$) on Σ given by applying a permutation to variables where they occur in algebraic terms; as defined below:

$$\begin{aligned} \pi \cdot x &= \pi(x) \\ \pi \cdot f(t_1, \ldots, t_n) &= f(\pi \cdot t_1, \ldots, \pi \cdot t_n) \end{aligned}$$

   Note that in the case for the inductive construction of $T(\Sigma, \mathbb{A}, \mathbb{X})$ this example agrees with Definition 2.3.

The next definition plays an important role in the theory of nominal sets, that is, the *support* of a element in a set equipped with an action.

Definition 3.3. Let $(X, \cdot)$ be a $\mathbb{P}$-set. A set of *atomic names* $A \subseteq \mathbb{A}$ is a *support* for an element $x \in X$ if, and only if, for all permutations $\pi \in \mathbb{P}$

$$(\forall a \in A.\ \pi(a) = a) \implies \pi \cdot x = x.$$

By considering a bijection between $\mathbb{A} \to \mathbb{N}$ we can use results from the theory of groups to prove results about nominal sets. Consider the classical theorem below.

Theorem 3.6 (Factorization of Permutations)**.** Every permutation $\pi \in \mathbb{S}_n$ (the group of finite permutations of the elements $\{1, 2, \ldots, n\}$) is a product of transpositions.

Proof. It suffices to factor cycles:

$$(1\ 2\ \ldots\ r) = (1\ r)(1\ r-1)\ldots(1\ 2)$$

□

This theorem can be translated via the bijection $\mathbb{A} \to \mathbb{N}$ to the following statement about permutation of names.

Theorem 3.7 (Factorization of Nominal Permutations)**.** Every permutation $\pi \in \mathbb{P}$ is a product of swappings

$$\pi = (a_1\ a_r)(a_1\ a_{r-1})\ldots(a_1\ a_2)$$

As a corollary,

$$\pi(a_1) \neq a_1,\ a_1 \neq a_i,\ 1 < i \leq r, \text{ and } a_i \neq \pi(a_i) \tag{3.1}$$

We use the theorem above to derive the following characterization of support in terms of swappings.

Lemma 3.2. Let X be a $\mathbb{P}$-set and $x \in X$. A subset $A \subseteq \mathbb{A}$ supports x, if and only if,

$$\forall a_1, a_2 \in \mathbb{A} \setminus A.\ (a_1\ a_2) \cdot x = x. \tag{3.2}$$

Proof. Notice that if $a_1, a_2 \in \mathbb{A} \setminus A$ then $(a_1\ a_2) \cdot a = a$ holds for any $a \in A$. Therefore, if A supports x then for all permutation π that fix every element of A, π also fix x. In particular, $(a_1\ a_2) \cdot x = x$ since $(a_1\ a_2)$ fix A, and hence (3.2) holds.

Conversely, suppose A satisfies (3.2) and take a permutation π fixing every element of A. We must show that π fixes x. By Theorem 3.7, π can be written as the composition $\pi = (b_1\ b_{r-1})(b_1\ b_{r-2})\ldots(b_1\ b_2)$ satisfying (3.1). Since π fixes every elements of A, each swapping is such that $a_1, b_i \notin A$ and hence by (3.2), $(a_1\ b_i)$ fixes x. Therefore, π must also fix x since it is the composition of all the $(a_1\ a_i)$, and hence A supports x. □

Clearly each element of a $\mathbb{P}$-set is supported by $\mathbb{A}$, which is an infinite set. We will be interested in elements that are finitely supported in the sense that there is some finite set of atomic names that is a support for the element.

It can be shown [26, Theorem 2.7] that if an element x of a $\mathbb{P}$-set X has finite support then there is a unique least finite set of atoms $A \subseteq \mathbb{A}$ that supports x.

Definition 3.4. Let $(X, \cdot)$ be a $\mathbb{P}$-set and $x \in X$. Denote by $\mathtt{supp}(x)$ the least set of atoms supporting x.

We also use the following characterization of support:

$$\mathtt{supp}(x) = \{a \in \mathbb{A} \mid \{b \in \mathbb{A} \mid (a\ b) \cdot x \neq x\} \text{ is not finite}\} \tag{3.3}$$

Thus, for any $a \in \mathbb{A}$, $a \notin \mathtt{supp}(x)$ holds if, and only if, $(a\ b) \cdot x = x$ holds for all, but finitely many $b \in \mathbb{A}$.

Definition 3.5. A nominal set X is a $\mathbb{P}$-set $(X_S, \cdot)$ such that all elements $x \in X$ have finite support. In this case we also say that X is finitely supported.

NOTATION: We may denote a nominal set $X = (X_S, \cdot)$ just by X when no confusion arises, the set X_S is called *the underlying set* of the nominal set X.

Example 3.3.

1. The set $\mathbb{A}$ of all atoms, with action defined by $\pi \cdot a = \pi(a)$ is a nominal set; the support of a is $\{a\}$.

2. The set Λ of all λ-terms (Example 2.1), with action defined by Definition 2.3, has finite support. It turns out that the support of a λ-term is the set of atoms occuring in it.

3. The powerset $\mathcal{P}(\mathbb{A}) = \{X \mid X \subseteq \mathbb{A}\}$, with action $\pi \cdot X = \{\pi \cdot a \mid a \in X\}$, is not a nominal set. To see this let us order $\mathbb{A}$ as $\{a_1, a_2, a_3, \dots\}$ and consider $X = \{a_1, a_3, a_5, \dots\} \in \mathcal{P}(\mathbb{A})$. We claim that there is no finite set A of atoms supporting X. To prove this claim, we suppose that is not the case, i.e., there is $A = \{b_1, \dots, b_n\}$ that supports X. Then, choose any pair $c_{2k}, c_{2k+1} \in \mathbb{A} \setminus A$, it follows that $(c_{2k}\ c_{2k+1}) \cdot X \neq X$, a contradiction with Lemma 3.2.

4. We say that $X \subseteq \mathbb{A}$ is *cofinite* if $\mathbb{A} \setminus X$ is finite. The set $\mathcal{P}_{\mathtt{fs}}(\mathbb{A})$ of finite and cofinite subsets of $\mathbb{A}$ with action inherited from $\mathcal{P}(\mathbb{A})$ is a nominal set. The support is calculated in Example 3.4.4.

5. If X and Y are nominal sets then their disjoint union $X + Y := \{x \in X \mid x \notin Y\} \cup \{y \in Y \mid y \notin X\}$ is a nominal set.

6. If X and Y are nominal sets then the product $X \times Y = \{(x, y) \mid x \in X \wedge y \in Y\}$ with action defined as in Definition 3.7 is a nominal set. The support of (x, y) is the union of the supports of x and y.

7. The empty set is a nominal set.

3.2.2 *Semantic Freshness*

In many applications of nominal sets one is more interested (like in our case) in the complementary notion of an atom a being in the support of an element $x \in X$.

Definition 3.6. Given nominal sets X and Y, and elements $x \in X$ and $y \in Y$. Define the predicate $\#_{\mathtt{sem}}$, written as $x \#_{\mathtt{sem}} y$, and say that x is fresh for y if the two elements have disjoint support, that is:

$$x \#_{\mathtt{sem}} y \iff \mathtt{supp}(x) \cap \mathtt{supp}(y) = \emptyset$$

Write $a \#_{\mathtt{sem}} x$ when $a \notin \mathtt{supp}(x)$, and say a is fresh for x.

Example 3.4.

1. We have distinct notations for freshness: the first is freshness in the nominal syntax, written just as #, and the second we denote by $\#_{\mathtt{sem}}$ to describe *semantic freshness*.
2. The set $\mathbb{A}$ of all atoms with action $\pi \cdot a = \pi(a)$ is a nominal set; the support of an atom $a \in \mathbb{A}$ is just $\{a\}$. Note that for $x, y \in \mathbb{A}$, $x \#_{\mathtt{sem}} y$ precisely when $x \neq y$.
3. If $a \in \mathbb{A}$ and X is a nominal set with $x \in X$, we write $a \#_{\mathtt{sem}} x$ when $a \notin \mathtt{supp}(x)$.
4. The support of $\mathbb{A} \setminus \{a\} \in \mathcal{P}_{\mathtt{fs}}(\mathbb{A})$ is $\{a\}$ so $b \#_{\mathtt{sem}} \mathbb{A} \setminus \{a\}$ but not $a \#_{\mathtt{sem}} \mathbb{A} \setminus \{a\}$.

The finiteness of support compared with the infiniteness of $\mathbb{A}$ leads to the following principle for *choices of new names*.

CHOOSE-A-FRESH-NAME PRINCIPLE If $X_1, \ldots, X_n$ are finitely many nominal sets and if $x_1 \in X_1, \ldots, x_n \in X_n$ are elements of them, then there is an atomic name $a \in \mathbb{A}$ satisfying $a \#_{\mathtt{sem}} x_1 \wedge \cdots \wedge a \#_{\mathtt{sem}} x_n$.

Lemma 3.3. Let $x \in X$ be an element of a nominal set X. For all $a, b \in \mathbb{A}$, if $a \#_{\mathtt{sem}} x$, $b \#_{\mathtt{sem}} x$ then $(a\ b) \cdot x = x$.

Proof. Since $a \#_{\mathtt{sem}} x$ and $b \#_{\mathtt{sem}} x$, we have that $a, b \notin \mathtt{supp}(x)$. The result now follows from Lemma 3.2. □

The next lemma states that the properties from derivable (syntactic) freshness (#) extends in a natural way to semantic freshness ($\#_{\mathtt{sem}}$).

Lemma 3.4. Suppose $x, y \in X$ and X a nominal set. Then,

1. If $x = y$ then $a \#_{\mathtt{sem}} x$ iff $a \#_{\mathtt{sem}} y$.
2. If $a \#_{\mathtt{sem}} x$ for every $a \in \mathtt{ds}(\pi, \gamma)$ then $\pi \cdot x = \gamma \cdot x$.
3. If $a \#_{\mathtt{sem}} x$ then $\pi(a) = \pi \cdot x$.
4. If $x = y$ then $\pi \cdot x = \pi \cdot y$.

Proof.

1. If $x = y$, then $\mathtt{supp}(x) = \mathtt{supp}(y)$. Therefore, $a \#_{\mathtt{sem}} x$ if, and only if, $a \#_{\mathtt{sem}} y$.
2. We have that $a \notin \mathtt{supp}(x)$ for each $a \in \mathtt{ds}(\pi, \gamma)$. Therefore, $\pi \cdot x = \gamma \cdot x$.
3. Follows directly from Equivariance (Theorem A.1).

□

From Example 3.3-(3) we can seee that $a \#_{\mathtt{sem}} X$ does not imply that $a \#_{\mathtt{sem}} x$ for every $x \in X$. But for those subset that are finitely supported we have:

Lemma 3.5. Let X be a nominal set and $Y \subseteq X$ finitely-supported. If

$$a_1 \#_{\mathtt{sem}} Y, \ldots, a_n \#_{\mathtt{sem}} Y$$

then there exists some $y \in Y$ such that $a_1 \#_{\mathtt{sem}} y, \ldots, a_n \#_{\mathtt{sem}} y$.

Proof. Fix any $y' \in Y$. By the **Choose-a-Fresh-Name-Principle** there is a choice $b_1, \ldots, b_n$ of fresh names such that $b_i \#_{\mathtt{sem}} Y$ and $b_i \#_{\mathtt{sem}} y'$, for $1 \leq i \leq n$. Then by Part 2 of Lemma 3.4 $(b_1\ a_1) \cdots (b_n\ a_n) \cdot Y = Y$. Define $y = (b_1\ a_1) \cdots (b_n\ a_n) \cdot y'$, note that this element is in Y by Definition 3.2, and we conclude $a_i \#_{\mathtt{sem}} x$ for $1 \leq i \leq n$ by Part 3 of Lemma 3.4 and the assumption $b_i \#_{\mathtt{sem}} y$. □

3.2.3 *Products*

We now study one way to build new nominal sets from given nominal sets $X_1, \ldots, X_n$, i.e., by taking the cartesian product of these sets and defining a suitable product action. The cartezian product is used to construct products nominal algebras later in this chapter.

Definition 3.7. Given $\mathbb{P}$-sets $X_1, \ldots, X_n$ we make the cartesian product

$$X_1 \times \cdots \times X_n := \{(x_1, \ldots, x_n) \mid x_1 \in X_1, \ldots, x_n \in X_n\}$$

into a $\mathbb{P}$-set by defining the action coordinate-wise:

$$\pi \cdot (x_1, \ldots, x_n) := (\pi \cdot x_1, \ldots, \pi \cdot x_n)$$

When $X_1, \ldots, X_n$ are the same $\mathbb{P}$-set we write the product as usual: X^n.

Definition 3.8. Let I be a countably infinite set and $(X_i)_{i \in I}$ an I-indexed collection of nominal sets. Write

$$\Pi_{i \in I} X_i := \{(x_i \mid i \in I) \mid \forall i \in I.\ x_i \in X_i\}$$

for the cartesian product of the family $(X_i)_{i \in I}$. Equip this set with the component-wise permutation action,

$$\pi \cdot (x_i)_{i \in I} = (\pi \cdot x_i)_{i \in I}.$$

This next result follows directly from the definition above.

Lemma 3.6. $a \#_{\mathtt{sem}} (x_i)_{i \in I}$ if, and only if, $a \#_{\mathtt{sem}} x_i$ for every $i \in I$.

3.2.4 *Equivariant Functions*

If $(X, \cdot)$ and $(Y, \cdot)$ are $\mathbb{P}$-sets we say that f defines a function $f \in (X, \cdot) \to (Y, \cdot)$ if it is a function on underlying sets, i.e., $f \in X \to Y$.

Definition 3.9. Suppose that X and Y are nominal sets. A function $f \in X \to Y$ is equivariant if for all permutations $\pi \in \mathbb{P}$:

$$\pi \cdot f(x) = f(\pi \cdot x)$$

Lemma 3.7. Suppose $f : X \to Y$ is an equivariant function between the nominal sets X and Y. Then if A is a support for $x \in X$, then it is a support for $f(x) \in Y$. In particular,

$$\mathtt{supp}(f(x)) \subseteq \mathtt{supp}(x).$$

Proof. Suppose $A \subseteq \mathbb{A}$ supports $x \in X$, and let π be a permutation such that π fixes every element of A. By Definition 3.3, $\pi \cdot x = x$ and hence $f(x) = f(\pi \cdot x) = \pi \cdot f(x)$ since f is equivariant. Thus A also supports $f(x)$ in Y. Therefore if X and Y are nominal sets and $x \in X$, then $\mathtt{supp}(x)$ supports $f(x)$ and hence contains the smallest support, $\mathtt{supp}(f(x))$.

□

Corollary 3.1. If $a \#_{\mathtt{sem}} x_i$, for $i \leq i \leq n$, then $a \#_{\mathtt{sem}} f(x_1, \ldots, x_n)$.

3.3 NOMINAL ALGEBRA

We use nominal sets to give a semantics to nominal algebra signature and theories. This choice is adequate, since nominal sets better represent directly the notion of freshness and permutation action. We begin by saying what we mean by *nominal algebra*.

Definition 3.10. [Σ-algebra] A *nominal* Σ*-algebra* $\mathcal{A}$ consists of:

1. A *domain* nominal set $A = (A_S, \cdot)$ with underlying set A_S and action $\cdot$.
2. An equivariant map $\mathtt{atom} : \mathbb{A} \to A_S$ to interpret atoms; we write the interpretation $\mathtt{atom}(a)$ of a as $a^{\mathcal{A}} \in A$.
3. An equivariant map $\mathtt{abs} : \mathbb{A} \times A_S \to A_S$ such that $a \#_{\mathtt{sem}} \mathtt{abs}(a, x)$ always, to interpret abstraction.
4. An equivariant map $f^{\mathcal{A}} : A_S^n \to A_S$ for each term-former $f : n \in \Sigma$ to interpret term-formers.

Nominal Algebras are usually denoted by $\mathcal{A}, \mathcal{B}$.

As expected, a *valuation* ς in a Σ-algebra $\mathcal{A}$ maps unknowns X to elements $\varsigma(X) \in A_S$. Below we define a equivariant function $[\![\cdot]\!]$ to interpret nominal terms.

Definition 3.11. Let $\mathcal{A}$ be a nominal algebra. Suppose that $t \in T(\Sigma, \mathbb{A}, \mathbb{X})$. The interpretation of t with relation to the valuation ς in $\mathcal{A}$, denoted as $[\![t]\!]^{\mathcal{A}}_{\varsigma}$, is defined inductively by:

$$[\![a]\!]_{\varsigma} = a^{\mathcal{A}} \qquad [\![\pi \cdot X]\!]_{\varsigma} = \pi \cdot \varsigma(X) \qquad [\![[a]\, t]\!]_{\varsigma} = \mathtt{abs}(a, [\![t]\!]_{\varsigma})$$
$$[\![f(t_1, \ldots, t_n)]\!]_{\varsigma} = f^{\mathcal{A}}([\![t_1]\!]_{\varsigma}, \ldots, [\![t_n]\!]_{\varsigma})$$

The interpretation $[\![t]\!]^{\mathcal{A}}_{\varsigma}$ may be written just by $[\![t]\!]_{\varsigma}$ if $\mathcal{A}$ is understood.

Lemma 3.8. Let $\mathcal{A}$ be a Σ-algebra and ς a valuation to the domain A_S. Then $\pi \cdot [\![t]\!]_{\varsigma} = [\![\pi \cdot t]\!]_{\varsigma}$.

Proof. The proof is a standard induction on the structure of t, see Lemma B.5 for a complete proof. □

Definition 3.12. Suppose that $\mathcal{A}$ is a Σ-algebra. Define the notion of *validity* by:

a) $[\![\Delta]\!]^{\mathcal{A}}_{\varsigma}$ is *valid* when $a \#_{\mathtt{sem}} \varsigma(X)$ for each $a\#X \in \Delta$.

b) $[\![\Delta \vdash a\#t]\!]^{\mathcal{A}}_{\varsigma}$ is *valid* when the validity of $[\![\Delta]\!]^{\mathcal{A}}_{\varsigma}$ implies $a \#_{\mathtt{sem}} [\![t]\!]^{\mathcal{A}}_{\varsigma}$, and

c) $[\![\Delta \vdash t = u]\!]^{\mathcal{A}}_{\varsigma}$ is valid when the validity of $[\![\Delta]\!]^{\mathcal{A}}_{\varsigma}$ implies $[\![t]\!]^{\mathcal{A}}_{\varsigma} = [\![u]\!]^{\mathcal{A}}_{\varsigma}$.

A *model* of a theory T is a Σ-algebra $\mathcal{A}$ such that $[\![\Delta \vdash t = u]\!]^{\mathcal{A}}_{\varsigma}$ is valid for every axiom $\Delta \vdash t = u$ in Ax and every valuation ς.

Definition 3.13. For a nominal theory T, define the *validity with respect to* T for judgements form as follows:

1. Write $\Delta \models_T a \#_{\mathtt{sem}} t$ when $[\![\Delta \vdash a\#t]\!]^{\mathcal{A}}_{\varsigma}$ for all models $\mathcal{A}$ of T.

2. Write $\Delta \models_T t = u$ when $[\![\Delta \vdash t = u]\!]^{\mathcal{A}}_{\varsigma}$ for all models $\mathcal{A}$ of T.

Note that, in constrast with derivable freshness #, for which the theory T does not matter at all, semantic freshness $\#_{\mathtt{sem}}$ does depend on the theory T. We will see an example later, first we need to build some more structures and prove some results.

3.3.1 *Soundness*

An important property one seeks in a logical framework is that of the equivalence between syntactic operations and operations on semantic models of that language, this is called Soundness.

To ilustrate, consider the theory LAM defined in Example 3.1. If one builds a derivation in LAM showing that $\lambda[a]\,(b\ a) = a$ it is desirable that this equality is valid in all models (Σ-algebras) of LAM. That is the meaning of soundness: whenever manipulation done following the derivation rules of the system (Figs. 2.1 and 3.3) produce *sound* equalities inside all models of T.

Soundness results have also an practical application: its possible to reason about equality on semantic objects (in our case, inside nominal algebras) in an automated way. We put computers to manipulate symbols, they are very good at it, and prove that these syntactic operations are indeed *sound*. The next theorem says that Nominal Algebra is indeed *sound* for freshness and equality derivations.

Theorem 3.8 (Soundness)**.** Let $T = (\Sigma, Ax)$ be a theory. Then:

1. If $\Delta \vdash a\#t$ then $\Delta \models_T a \#_{\mathtt{sem}} t$.
2. If $\Delta \vdash_T t = u$ then $\Delta \models_T t = u$.

Proof. Let $\mathcal{A}$ be a model of T. We must show that if $a\#t$ (resp. $t = u$) is derivable from Δ using the rules from Figs. 2.1 and 3.3, then $[\![\Delta]\!]_\varsigma$ implies $a \#_{\mathtt{sem}} [\![t]\!]_\varsigma$ (resp. $[\![t]\!]_\varsigma = [\![u]\!]_\varsigma$) for any valuation ς. In both cases the proof is by induction on the depth of the derivation $\Delta \vdash a\#t$ and $\Delta \vdash t = u$.

1. Let Π be a derivation of $\Delta \vdash a\#t$.

   For the base case, suppose Π ends with the axiomatic rule $(\#a)$. Then $a\#b$ is derivable (and $a \neq b$). It must be shown that $a \#_{\mathtt{sem}} b^{\mathcal{A}}$, note that this folows from Example 3.4, Item 2.

   For the inductive step:

   a) If Π ends with the application of $(\#X)$:

      The inductive hypothesis gives us $\pi^{-1}(a) \#_{\mathtt{sem}} \varsigma(X)$ and we conclude $a \#_{\mathtt{sem}} \pi \cdot \varsigma(X)$ by equivariance.

   b) If Π ends with $(\#a)$ then $a \#_{\mathtt{sem}} \mathtt{abs}(a, [\![t]\!]_\varsigma)$, by Definition 3.10.

   c) The cases for $(\#b)$ and $(\#f)$ follow directly from the induction hypothesis and by Lemma 3.7.

2. Let Π be a derivation of $\Delta \vdash t = u$.

   The cases for (refl), (symm), (trans), (cong []) and (cong$f$) follow easily by properties of equality.

   For the inductive step:

   a) If Π ends with an application of (perm):

      By induction hypothesis, $a \#_{\mathtt{sem}} [\![t]\!]_\varsigma$ and $b \#_{\mathtt{sem}} [\![u]\!]_\varsigma$. From Lemma 3.3 follows that $(a\ b) \cdot [\![t]\!]_\varsigma = [\![u]\!]_\varsigma$, finally by equivariance (Lemma 3.8) one get $[\![\pi \cdot t]\!]_\varsigma = [\![t]\!]_\varsigma$.

   b) If Π ends with an application of $(ax_{\nabla \vdash t=u})$:

      Suppose $[\![\nabla^\pi \sigma]\!]_\varsigma$ for any ς. Then $\pi(a) \#_{\mathtt{sem}} (X\sigma)\varsigma$ holds for all $a\#X \in \nabla$. By equivariance, $a \#_{\mathtt{sem}} \pi^{-1} \cdot [\![X\sigma]\!]_\varsigma$ for all $a\#X \in \nabla$. Let ς' be defined by

      $$\varsigma'(X) = \pi^{-1} \cdot [\![X\sigma]\!]_\varsigma \text{, for every } X.$$

      Then $a \#_{\mathtt{sem}} \varsigma'(X)$ for all $a\#X \in \nabla$, so $[\![\nabla]\!]_\varsigma$ holds. Since $\nabla \vdash t = u$ is an axiom of T, we know $[\![t]\!]_{\varsigma'} = [\![u]\!]_{\varsigma'}$. By

the Item (4) of Lemma 3.4, $\pi \cdot [\![t]\!]_{\varsigma'} = \pi \cdot [\![u]\!]_{\varsigma'}$ also holds. Equivariance give us $[\![\pi \cdot u]\!]_{\varsigma'} = [\![\pi \cdot u]\!]_{\varsigma'}$. By a *standard induction* reasoning on the syntax of terms one can show $[\![\pi \cdot t]\!]_{\varsigma'} = [\![t^{\pi}\sigma]\!]_{\varsigma}$ and $[\![\pi \cdot u]\!]_{\varsigma'} = [\![t^{\pi}\sigma]\!]_{\varsigma}$ to finally conclude $[\![t^{\pi}\sigma]\!]_{\varsigma} = [\![u^{\pi}\sigma]\!]_{\varsigma}$.

c) Suppose Π ends with (fr):

Then $\Delta \vdash_T t = u$ is derived from $\Delta, a\#X_1, \ldots, a\#X_n \vdash t = u$, where $a \notin \mathtt{atms}(\Delta, t, u)$. By ZFA Equivariance (Theorem A.1), applyed considering the derivability relation $\vdash_T$, the following is provable for a' not occuring in $\mathtt{atms}(\Delta, t, u)$.

$$\Delta, a'\#X_1, \ldots, a'\#X_n \vdash_T t = u$$

We also retain the inductive hypothesis for $\Delta, a'\#X_1, \ldots, a'\#X_n \vdash_T t = u$ by ZFA Equivariance. Now, by the **Choose-a-Fresh-Name Principle** select an atom $a' \notin \mathtt{atms}(\Delta, t, u)$ such that $a' \#_{\mathtt{sem}} \varsigma(X_i)$ for $1 \leq i \leq n$. Then by the inductive hypothesis,

$$[\![\Delta, a'\#X_1, \ldots, a'\#X_n \vdash_T t = u]\!]_{\varsigma}.$$

But this is equivalent to $[\![\Delta \vdash_T t = u]\!]_{\varsigma}$ since $[\![a'\#X_1, \ldots, a'\#X_n \vdash_T t = u]\!]_{\varsigma}$, and the result follows.

□

3.3.2 *The Ground Initial Algebra* $\mathbb{F}(T, \mathcal{D})$

A very known strategy to build models for equational theories is to add constants in the signature to construct an infinite number of elements of the language and then quotient these elements by provable equality. In the nominal setting, adding a term-former with arity zero only give us a supply of elements with empty support ($a\#\mathbf{d}$ is derivable for every $a \in \mathbb{A}$ and $\mathbf{d} : 0$). In this section, we study a way to build an *initial nominal algebra* with such extra elements but with nonempty support.

As in [19], let $\mathcal{D}$ be a set of term-formers disjoint from Σ, they are called 'extra term-formers'. Then the set $\mathbb{F}(\Sigma, \mathcal{D})$ of *ground nominal terms* is generated by the grammar:

$$g ::= a \mid [a]\, g \mid f(g_1, \ldots, g_n) \mid \mathbf{d}(a_1, \ldots, a_m)$$

Where $f : n$ range over elements of Σ^n and $\mathbf{d} : m$ ranges over elements of $\mathcal{D}^m$.

Remark 3.1. Note that the initial ground (nominal) algebra is the analogous in nominal setting to the initial first-order ground algebra $T(\Sigma)/_E$, where E is a set of equational axioms. In what follow we will see how to effectively build a nominal algebra from this set of ground terms.

We first give to $\mathbb{F}(\Sigma, \mathcal{D})$ a $\mathbb{P}$-set structure by defining the action of a permutation π on $g \in \mathbb{F}(\Sigma, \mathcal{D})$ exactly as in Definiton 2.3.

Lemma 3.9. The set $\mathbb{F}(\Sigma, \mathcal{D})$ of ground terms is a nominal set with the $\mathbb{P}$-action as above. Moreover $\mathtt{supp}(g) = \{a \in \mathbb{A} \mid a \in \mathtt{atms}(g)\}$.

Proof. We need to show that every $g \in \mathbb{F}(\Sigma, \mathcal{D})$ has finite support. Consider the equation (3.3) and note that precisely when $a \in \mathtt{atms}(g)$, the set $\{b \in \mathbb{A} \mid (a\ b) \cdot x \neq x\}$ is not finite. □

Corollary 3.2. As a corollary, $a \notin \mathtt{atms}(g)$ if, and only if, $a \#_{\mathtt{sem}} g$.

Proof. Follows directly from Lemma 3.9. □

Lemma 3.10. Let $g \in \mathbb{F}(\Sigma, \mathcal{D})$ then $a \notin \mathtt{atms}(g)$ implies $\vdash a\#g$.

Proof. By induction on the structure of g using the rules from Fig. 2.1. □

Definition 3.14. Write (cong**d**) for an instance of the (cong$f$) rule when $f \in \mathcal{D}$. We denote by $[g]_T$ the set of ground terms g' such that a derivation of $\vdash_T g = g'$ exists that does not mention (cong**d**) for any $\mathbf{d} \in \mathcal{D}$.

Note that the set $[g]_T$ has the meaning of 'derivable ground equality'; we denote by $\mathbb{F}(T, \mathcal{D}) = \{[g]_T \mid g \in \mathbb{F}(\Sigma, \mathcal{D})\}$ the set of all such $[g]_T$, sometimes called *the set of all ground terms up to* T. We give to $\mathbb{F}(T, \mathcal{D})$ a $\mathbb{P}$-set structure by defining the $\mathbb{P}$-action $\pi \cdot [g]_T = [\pi \cdot g]_T$. The finite (Lemma 3.10) support of $[g]_T$ is the set $\{a \in \mathbb{A} \mid \nvdash a\#g\}$. Therefore, $\mathbb{F}(T, \mathcal{D})$ is a nominal set. Note also that if $\vdash a\#g$ then $a \#_{\mathtt{sem}} [g]_T$. Next we prove a technical lemma.

Lemma 3.11. Let $[g]_T \in \mathbb{F}(T, \mathcal{D})$. Then $a_1 \#_{\mathtt{sem}} [g]_T, \ldots, a_n \#_{\mathtt{sem}} [g]_T$ if, and only if, there exists some $g' \in [g]_T$ such that $\vdash a_1\#g', \ldots, \vdash a_n\#g'$ are all derivable.

Proof. Suppose $a_1 \#_{\mathtt{sem}} [g]_T, \ldots, a_n \#_{\mathtt{sem}} [g]_T$. Lemma 3.5 ensures the existence of $g \in [g]_T$ such that $a_1 \#_{\mathtt{sem}} g', \ldots, a_n \#_{\mathtt{sem}} g'$. So by Lemma 3.9 $a_i \notin \mathtt{atms}(g')$ and by Lemma 3.10 we conclude $\vdash a_1\#g', \ldots, \vdash a_n\#g'$. □

Now we have in hand all ingredients to build the so promised 'nominal initial algebra'. The construction is analogous to the first-order definition of algebras out of terms quotiented by derivable equality.

The initial algebra $\mathbb{F}(T, \mathcal{D})$ is defined overloading the notation from the set of all elements $[g]_T$, but this should cause no confusion since one can easily distinguish the algebra $\mathbb{F}(T, \mathcal{D})$ from its domain set $\mathbb{F}(T, \mathcal{D}) = \{[g]_T \mid g \in \mathbb{F}(\Sigma, \mathcal{D})\}$ from the context.

Definition 3.15. Let $T = (\Sigma, Ax)$ be a theory and $\mathcal{D}$ a possible empty set of term-formers disjoint from Σ. The **initial algebra** of T over $\mathcal{D}$ is the Σ-algebra with:

1. Domain nominal set as $\mathbb{F}(T, \mathcal{D}) = \{[g]_T \mid g \in \mathbb{F}(\Sigma, \mathcal{D})\}$.
2. Interpretation of atoms $a^{\mathbb{F}(T,\mathcal{D})} = [a]_T$
3. Interpretation of abstraction $\mathtt{abs}(a, x) = [[a]\, g]_T$ for some $g \in x$.

4. Interpretation of functions given by

$$f^{\mathbb{F}(T,\mathcal{D})}(x_1,\ldots,x_n) = [f(g_1,\ldots,g_n)]_T$$

for some $g_1 \in x_1,\ldots,g_n \in x_n$, for each term-former $f : n \in \Sigma$.

Lemma 3.12. Under the same conditions as in Definition 3.15, $\mathbb{F}(T,\mathcal{D})$ is a (nominal) Σ-algebra.

Proof sketch. We have showed that the domain set $\mathbb{F}(T,\mathcal{D}) = \{[g]_T \mid g \in \mathbb{F}(T,\mathcal{D})\}$ is a nominal set. For the interpretation functions it must be shown that they are well defined, i.e., the choices for $g \in x$ and $g_1 \in x_1,\ldots,g_n \in x_n$ does not matter, and that they are equivariant. Finally, we must show that $a \#_{\mathtt{sem}} \mathtt{abs}(a,x)$ holds always. For a full detailed proof see [19, Lemma 6.11]. □

Lemma 3.13. Consider $t \in \mathbb{F}(\Sigma,\mathcal{D})$ as above. Suppose that $X\sigma \in \mathbb{F}(\Sigma,\mathcal{D})$ for every $X \in \mathtt{vars}(t)$. Lets ς be a valuation to the domain $\mathbb{F}(T,\mathcal{D})$ such that $X\sigma \in \varsigma(X)$ for every $X \in \mathtt{vars}(t)$. Then $[t\sigma]_T = [\![t]\!]_\varsigma$.

Proof. By a standard induction on the structure of t. The proof can be found in Appendix B, see Lemma B.6. □

Theorem 3.9. Let T be a nominal theory. The initial algebra $\mathbb{F}(T,\mathcal{D})$ is a model of T.

Proof. The proof can be found in the Appendix B, see Theorem B.1. □

Example 3.5. Consider the theory ATOM with one axiom $\vdash a = b$. Notice that $a \#_{\mathtt{sem}} [a]_{\mathtt{ATOM}}$, since $[a]_{\mathtt{ATOM}} = \mathbb{A}$. Its not hard to verify that this property holds for all models $[\![\cdot]\!]$ of ATOM. Hence, $\models_{\mathtt{ATOM}} a\#a$ but also $\nvdash a\#a$.

This example has an immediate consequence on completeness of nominal algebra under freshness derivations, i.e., we can show that $a \#_{\mathtt{sem}} [a]_{\mathtt{ATOM}}$ but not $\vdash_T a\#a$. In general, $\Delta \models_T b \#_{\mathtt{sem}} t$ does not imply $\Delta \vdash t$ necessarily.

3.3.3 *Completeness for Equality Derivations*

For this subsection, fix a signature Σ, a theory $T = (\Sigma, Ax)$, Σ-terms t, u, and a freshness context Δ. We study the completeness of nominal algebra for equality derivations, i.e., if $\Delta \models_T t = u$ then $\Delta \vdash_T t = u$. We state this result as Theorem 3.10 below and take the rest of this subsection to prove it.

Theorem 3.10 (Equality Completeness)**.** If $\Delta \models_T t = u$ then $\Delta \vdash_T t = u$.

To prove the completeness theorem we study a somewhat nontrivial construction, but the main idea is quite simple: we only need a specific model and a specific valuation to this model that preserves sufficient information on the semantics of $\Delta \models_T t = u$ to allow us to reconstruct a syntactic derivation of $\Delta \vdash_T t = u$.

We begin by separating the sets of variables and atoms occuring in Δ, t, u, and use the sets below to regain informations about the atoms and variables used in the semantic equality $\Delta \models_T t = u$.

Definition 3.16. Let $\mathcal{X} = \mathtt{var}(\Delta, t, u)$ and let $\mathcal{A} = \mathtt{atms}(\Delta, t, u)$. For each $X_i \in \mathcal{X}$:

- let $a_{X_{i1}}, \ldots, a_{X_{ik}}$ be the atoms in $\mathcal{A}$ (in some arbitrary but fixed order) such that $a_{X_{il}} \# X_i \notin \Delta$; for all $1 \leq l \leq k$.
- let $\mathbf{d}_X : k$ be a term-former.

For each unknown $X \notin \mathcal{X}$, let $\mathbf{d}_X : 0$ be a term-former, and consider the set $\mathcal{D}$ of all $\mathbf{d}_X$, for each unknown $X \in \mathbb{X}$.

Definition 3.17. Let σ be the following substitution:

$$X\sigma = \mathbf{d}_X(a_{X_{i1}}, \ldots, a_{X_{ik}}) \qquad (X \in \mathcal{X})$$
$$X\sigma = d_X() \qquad (X \notin \mathcal{X})$$

The fixed model we shall consider, perhaps with no surprises, is the initial algebra $\mathbb{F}(T, \mathcal{D})$ with $\mathcal{D}$ as defined in Definition 3.16 above and the valuation ς given by $\varsigma(X) = [X\sigma]_T$.

The reader may ask: why do we consider this model? The answer is also quite simple: initial models have the beauty of simplicity. Elements of $\mathbb{F}(T, \mathcal{D})$ are ground representations of provable equality within the theory T and freshness constraints are decidable just by looking in the syntax of the equality representatives, since there is no need to consider freshness assumptions of the form $a\#X$.

The model $\mathbb{F}(T, \mathcal{D})$ is just simple enough and still carry all information about the equality assertions made by the axioms of the theory. Let us begin with some technical results.

Lemma 3.14. $[t\sigma]_T = [\![t]\!]_\varsigma^{\mathbb{F}(T,\mathcal{D})}$ and $[u\sigma]_T = [\![u]\!]_\sigma^{\mathbb{F}(T,\mathcal{D})}$.

Proof. We show that $[t\sigma]_T = [\![t]\!]_\varsigma^{\mathbb{F}(T,\mathcal{D})}$ by induction on the syntax of t, we work on the two most interesting cases:

Suppose $t \equiv \pi \cdot X$, with $X \in \mathcal{X}$:

$$[\pi \cdot X\sigma]_T = [\pi \cdot \mathbf{d}_k(\vec{a})]_T = \pi \cdot [\mathbf{d}_k(\vec{a})]_T \qquad [\![\pi \cdot X]\!]_\varsigma = \pi \cdot \varsigma(X) = \pi \cdot [X\sigma]_T = \pi \cdot [\mathbf{d}_k(\vec{a})]_T$$

Suppose $t \equiv [a]\, t$, we have:

$$[[a]\, t\sigma]_T = \mathtt{abs}(a, [t\sigma]_T) \qquad [\![[a]\, t]\!]_\varsigma = \mathtt{abs}(a, [\![t]\!]_\varsigma) \overset{IH}{=} \mathtt{abs}(a, [t\sigma]_T)$$

□

The initial model validates the freshness constraints in Δ. This is ilustraded in Lemma 3.15 below.

Lemma 3.15. $[\![\Delta]\!]_\varsigma^{\mathbb{F}(T,\mathcal{D})}$ is valid.

Proof. Suppose $a\#X_i \in \Delta$, we must show that $a \#_{\mathtt{sem}} \varsigma(X_i)$. By Definition 3.16, $X_i \in \mathcal{X}$ so $\varsigma(X) = [\mathbf{d}_X(a_{X_{i1}}, \ldots, a_{X_{ik}})]_T$. But also $a \notin \{a_{X_{i1}}, \ldots, a_{X_{ik}}\}$ by construction; trivially, $a \notin \mathtt{atms}(\mathbf{d}_X(a_{X_{i1}}, \ldots, a_{X_{ik}}))$, hence by Lemma 3.10, $\vdash a\#\mathbf{d}_X(a_{X_{i1}}, \ldots, a_{X_{ik}})$ is derivable. Finally, by Lemma 3.11 we get $a \#_{\mathtt{sem}} [\mathbf{d}_X(a_{X_{i1}}, \ldots, a_{X_{ik}})]_T$ as required. □

Definition 3.18. Let Π be a derivation of $\vdash_T t\sigma = u\sigma$ without using (cong**d**). Let $\mathcal{A}'$ be $\mathcal{A}$ extended with:

- atoms mentioned anywhere in Π;
- a set $\mathcal{B}$ of fresh atoms, in bijection with $\mathcal{A}$, fixed as $a_{X_{il}} \mapsto b_{X_{il}}$.
- one fresh atom c such that c doest not occur in $\mathcal{A}, \Pi, \mathcal{B}$.

Remark 3.2. Note that in the definition above the existence of the the atoms in $\mathcal{B}$ and the fresh atom c is due to the **Choose-a-Fresh-Atom-Principle**.

Let Δ' be Δ extended with freshness assumptions $a'\#X$ for every $X \in \mathcal{X}$ and $a' \in \mathcal{A}' \setminus \mathcal{A}$. For the rest of this subsection let g, h range over arbitrary ground terms g and h over $\mathbb{F}(\Sigma, \mathcal{D})$ that mentions only atoms from $\mathcal{A}' \setminus (\mathcal{B} \cup \{c\})$.

Definition 3.19. Define an *inverse mapping* $(-)^{-1}$ from ground terms as above to terms in Σ inductively as follows:

$$a^{-1} \equiv a \qquad ([a]\, g)^{-1} \equiv [a]\, g^{-1} \qquad f(g_1, \ldots, g_n)^{-1} \equiv f(g_1^{-1}, \ldots, g$$
$$\mathbf{d}_X(a'_{X_{i1}}, \ldots, a'_{X_{ik}})^{-1} \equiv \pi_X(a'_{X_{i1}}, \ldots, a'_{X_{ik}}) \cdot X \qquad (X_i \in \mathcal{X})$$
$$\mathbf{d}_X()^{-1} \equiv c \qquad (X_i \notin \mathcal{X})$$

With the abbreviation

$$\pi_X(a'_{X_{i1}}, \ldots, a'_{X_{ik}}) := (a'_{X_{i1}}\ b_{X_{i1}}) \cdots (a'_{X_{ik}}\ b_{X_{ik}})(b_{X_{i1}}\ a_{X_{i1}}) \cdots (b_{X_{ik}}\ a_{X_{ik}})$$

The inverse mapping is equivariant over the atoms we care about, i.e., atoms in $\mathcal{A}' \setminus (\mathcal{B} \cup \{c\})$.

Lemma 3.16. Let π be a permutation that only mentions atoms from $\mathcal{A}' \setminus (\mathcal{B} \cup \{c\})$, then $\Delta' \vdash_{\texttt{CORE}} (\pi \cdot g)^{-1} = \pi \cdot g^{-1}$.

Proof. We reason by induction on the structure of the ground term g. The base case is trivially true using Definition 3.19. The only non-trivial inductive step is when $g \equiv d_X(a'_{Xi1}, \ldots, a'_{Xik})$ with $X \in \mathcal{X}$. We must show

$$\Delta' \vdash_{\texttt{CORE}} \pi_X(\pi(a'_{Xi1}), \ldots, \pi(a'_{Xik})) \cdot X = (\pi \circ \pi_X(a'_{X_{i1}}, \ldots, a'_{X_{ik}})) \cdot X$$

By Lemma 2.6 it suffices to show

$$\Delta' \vdash \texttt{ds}(\pi_X(\pi(a'_{Xi1}), \ldots, \pi(a'_{Xik})), \pi \circ \pi_X(a'_{X_{i1}}, \ldots, a'_{X_{ik}}))\#X.$$

Notice that π doest not mention any of the $b_{X_{il}}$, so π takes an atom $a'_{X_{il}}$ and maps it to $a'_{X_{ij}}$, hence $\texttt{ds}(\pi_X, \pi \circ \pi_X) \subseteq \mathcal{A}'$, and the results follows. □

Lemma 3.17. $\Delta' \vdash_{\texttt{CORE}} (t\sigma)^{-1} = t$ and $\Delta' \vdash_{\texttt{CORE}} (u\sigma)^{-1} = u$

Proof. It suffices to prove $\vdash_{\texttt{CORE}} (v\sigma)^{-1} = v$ for each subterm v of t and u. We do this by induction on the structure of v. The most interesting case is when $v = \pi \cdot X$, i.e., we need to show that $\Delta \vdash_{\texttt{CORE}} (\pi \cdot X\sigma)^{-1} = \pi \cdot X$. This is equivalent to

$$\Delta' \vdash \texttt{ds}(\pi_X(\pi(a_{Xi1}), \ldots, \pi(a_{Xik})), \pi)\#X$$

which is true by an analysis analogous from the Lemma 3.16 above. □

Corollary 3.3. If $\Delta \vdash_T g^{-1} = h^{-1}$ then $\vdash_T g = h$.

Proof. By Lemma 3.17 and transitivity (tran). □

The next Lemma shows how to construct a derivation of $\Delta \vdash_T t = u$ from a ground derivation of $\vdash_T t\sigma = u\sigma$. This is almost what we need for a complete derivation of $\Delta \vdash_T t = u$ using the semantic of $\mathbb{F}(T, \mathcal{D})$.

Lemma 3.18. If $\vdash_T t\sigma = u\sigma$ without using (cong**d**) then $\Delta \vdash_T t = u$. For freshness, if $\vdash a\#t\sigma$ then $\Delta \vdash a\#t$.

Proof. Suppose we could transform a derivation Π of $\vdash_T t\sigma = u\sigma$ ($\vdash a\#t\sigma$) into a derivation of $\Delta' \vdash_T (t\sigma)^{-1} = (u\sigma)^{-1}$ ($\Delta' \vdash a\#(t\sigma)^{-1}$). Given that, the result follows from Lemma 3.17 since we deduce $\Delta' \vdash_T t = u$ and obtain $\Delta \vdash_T t = u$ by the Strengthening Theorem 3.4. For freshness, the result follows from Lemmas 3.17 and 2.5.

The transformation of $\vdash_T t\sigma = u\sigma$ ($\vdash a\#t\sigma$) into $\Delta' \vdash_T (t\sigma)^{-1} = (u\sigma)^{-1}$ ($\Delta' \vdash a\#(t\sigma)^{-1}$) is inductive on Π.

If the derivation ends with an instance of $(\#ab), (\#a), (\#b)$ then the result follows by an instance of the same rule, possibly by using the inductive hypothesis.

The case for $(\#X)$ is impossible by assumption. Let us work on the interesting with more attention. Suppose Π ends with $(\#f)$. There are three cases to consider:

1. The case of $\vdash a\#f(g_1, \ldots, g_n)$ for $f \in \Sigma$: by assumption, $\vdash a\#g_i$ for $1 \leq i \leq n$, and we get $\Delta' \vdash a\#g_i^{-1}$ by induction hypothesis. Now we can build a derivation of $\Delta' \vdash a\#(g_1^{-1}, \ldots, g_n^{-1})$ using $(\#f)$.

2. The case of $\vdash a\#\mathbf{d}_X()$ for $\mathbf{d}_X \in \mathcal{D}$ and $X \notin \mathcal{X}$: it follows from the axiom rule $(\#ab)$ that $\vdash a\#c$ is always derivable.

3. The case of $a\#\mathbf{d}_X(a_{X'_{i1}}, \ldots, a_{X'_{ik}})$ for $\mathbf{d}_X \in \mathcal{D}$ and $X \in \mathcal{X}$: we must show $\Delta' \vdash a\#\pi \cdot X$, where

$$\pi = (a'_{X_{i1}}\ b_{X_{i1}}) \cdots (a'_{X_{ik}}\ b_{X_{ik}})(b_{X_{i1}}\ a_{X_{i1}}) \cdots (b_{X_{ik}}\ a_{X_{ik}}).$$

   By $(\#X)$, this follows form $\Delta' \vdash \pi^{-1}(a)\#X$. Since $a \neq a'_{X_{il}}$ and also $a \neq b_{X_{il}}$ for all l, we have

$$\pi^{-1}(a) = (b_{X_{ik}}\ a_{X_{ik}}) \cdots (b_{X_{i1}}\ a_{X_{i1}})(a)$$

   We proceed by a case distinction on a:

   - If a is in fact $a_{X_{il}}$ for some l, then $\pi^{-1}(a) = b_{X_{il}}$, and the result follows since $b_{X_{il}}\#X_i \in \Delta'$ by construction.
   - If a is none of the $a_{X_{il}}$ for all l, then $\pi^{-1}(a) = a$. By construction $a\#X \in \Delta$ since $a_{X_{il}}$ are the only atoms in $\mathcal{A}$ for which $a_{X_{il}}\#X_i \notin \Delta$. The result follows.

If the derivation Π of $\vdash_T t\sigma = u\sigma$ ends with an instance of (refl), (symm), (tran) or (cong[]) the result trivially follows by an instance of the same rule by possible using the inductive hypothesis. Again we work on the more interesting cases:

Suppose the derivation ends with (perm): by the inductive hypothesis we have $\Delta' \vdash a\#g^{-1}$ and $\Delta' \vdash b\#g^{-1}$. Then

$$\Delta' \vdash_T (a\ b) \cdot g^{-1} = g^{-1}$$

using (perm). Using Lemma 3.16, we conclude

$$\Delta' \vdash_T ((a\ b) \cdot g)^{-1} = g^{-1}.$$

The result now follows from Corollary 3.3.

Suppose the derivation ends with $(ax_{\nabla \vdash v=w})$. Then $\vdash \nabla^\pi\tau$ and $\vdash_T v^\pi\tau = w^\pi\tau$ for some permutation π and substitution τ such that $\nabla\tau, v\tau$ and $w\tau$ do not mention any unknown. We must show $\Delta' \vdash_T (v^\pi\tau)^{-1} = (w^\pi\tau)^{-1}$. Define the substitution τ' such that $X\tau' = (X\tau)^{-1}$ when $X\tau \neq X$ and $X\tau' = X$ when $X\tau = X$. It follows that $(v^\pi\tau)^{-1} \equiv v^\pi\tau'$, $(w^\pi\tau)^{-1} \equiv w^\pi\tau'$ and $(\nabla^\pi\tau)^{-1} = \nabla^\pi\tau'$, so it suffices to show

$$\nabla' \vdash_T v^\pi\tau' = w^\pi\tau'.$$

By $(ax_{\nabla \vdash v=w})$, this follows from $\Delta' \vdash \nabla^\pi\tau'$, i.e., $\Delta' \vdash (\nabla^\pi\tau)^{-1}$, which follows from the inductive hypothesis using the assumption $\vdash \nabla^\pi\tau$. □

Putting the pieces together we finally show the main result of this subsection: semantic equality, $\Delta \models_T t = u$, implies syntactic equality, $\Delta \vdash_T t = u$.

Proof of Theorem 3.10. Suppose $\Delta \models_T t = u$, so $[\![\Delta \vdash t = u]\!]_\varsigma^{\mathbb{F}(T,\mathcal{D})}$ for the initial ground algebra $\mathbb{F}(T,\mathcal{D})$ and the valuation ς defined as above. Now $[\![\Delta]\!]_\varsigma$ is valid by Lemma 3.15, so $[\![t]\!]_\varsigma = [\![u]\!]_\varsigma$. By Lemma 3.14 we have $[t\sigma]_T = [\![t]\!]_\varsigma$ and $[u\sigma]_T = [\![u]\!]_\varsigma$. Therefore, by construction $\vdash_T t\sigma = u\sigma$ without using (cong**d**). It follows by Lemma 3.18 that $\Delta \vdash_T t = u$. □

Now that we have finally proved Theorem 3.10 let us think about completeness for a moment. Recall that in Example 3.5 we showed $\models_{\mathtt{ATOM}} a\ \#_{\mathtt{sem}}\ a$ but not $\vdash a\#a$, as an example for incompleteness of nominal algebra with respect to freshness derivations.

Looking at Definition 3.10 the reader may ask: 'atoms in nominal syntax are identified by its name, so why allow for the interpretation of atoms to interpret two distinct atoms as equal elements in the domain?'

Indeed, this seams to be the source of freshness incompleteness. But, if one add the condition saying $\mathtt{atom} : \mathbb{A} \to A$ is an injection one would have the property that $[\![a]\!]_\varsigma \neq [\![b]\!]_\varsigma$ for all atoms $a, b \in \mathbb{A}$. But this condition would invalidate the soundness of the equality fragment. To see this consider the theory ATOM. So $\vdash_{\mathtt{ATOM}} a = b$ is trivially derivable, but any model of ATOM satisfies $[\![a]\!]_\varsigma \neq [\![b]\!]_\varsigma$. The next subsection shows how to regain freshness completeness without loosing soundness: semantic freshness is hidden inside the equality fragment.

3.3.4 *Completeness for Freshness*

Definition 3.20. Let $\Delta \vdash a\#t$ be a freshness judgement. By the **Choose-a-Fresh-Name-Principle** make a fixed but arbitrary choice of fresh atom b such that b does not occurs in Δ, a, t. Write Δ^+ for the context $\Delta, b\#X_1, \ldots, b\#X_n$ where $\{X_1, \ldots, X_n\} = \mathtt{var}(t)$ and

$$\Delta^+ \vdash (a\ b) \cdot t = t$$

for the equality judgement obtained from $\Delta \vdash a\#t$ as outlined above.

Theorem 3.11 (Completeness for Freshness)**.** $\Delta \models_T a\#t$ if, and only if, $\Delta^+ \vdash_T (a\ b) \cdot t = t$.

Proof. Choose a model $\mathcal{A}$ of T and any valuation ς such that $b \#_{\mathtt{sem}} \varsigma(X_1), \ldots, b \#_{\mathtt{sem}} \varsigma(X_n)$. By a standard induction on the structure of t one can show that $b \#_{\mathtt{sem}} [\![t]\!]_\varsigma$. Lemma 3.3 says that $a \#_{\mathtt{sem}} [\![t]\!]_\varsigma$ if, and only if, $(a\ b) \cdot [\![t]\!]_\varsigma = [\![t]\!]_\varsigma$ and from equivariance of the interpretation, Lemma 3.8, the last part is equivalent to $[\![(a\ b) \cdot t]\!]_\varsigma = [\![t]\!]_\varsigma$.

Now suppose $\Delta \models_T a\#t$. By Definition 3.13, $a \#_{\mathtt{sem}} [\![t]\!]_\varsigma$ for any model of T and all ς such that $[\![\Delta]\!]_\varsigma$ is valid. By the arguments above $[\![(a\ b) \cdot t]\!]_\varsigma = [\![t]\!]_\varsigma$ if $b \#_{\mathtt{sem}} \varsigma(X_1), \ldots, b \#_{\mathtt{sem}} \varsigma(X_n)$. It follows from Equality Completeness, Theorem 3.10, that $\Delta^+ \vdash (a\ b) \cdot t = t$. □

3.4 HOMOMORPHISMS, SUBALGEBRAS AND PRODUCT ALGEBRAS

We now concentrates our study in the interaction between nominal algebras. The first mathematical object that the reader may think of is perhaps the notion of *homomorphism* between algebras. Indeed they can carry a lot of information from one algebra to another, since they preserve interpretation.

3.4.1 *Homomorphisms and Homomorphic Images.*

Definition 3.21. For Σ-algebras $\mathcal{A}$ and $\mathcal{B}$, a Σ-algebra Homomorphism from $\mathcal{A}$ to $\mathcal{B}$ is a equivariant function $\theta : A \to B$ such that

- $\theta a^{\mathcal{A}} = a^{\mathcal{B}}$,
- $\theta \mathtt{abs}^{\mathcal{A}}(a, x) = \mathtt{abs}^{\mathcal{B}}(a, \theta x)$,
- $\theta f^{\mathcal{A}}(x_1, \ldots, x_n) = f^{\mathcal{B}}(\theta x_1, \ldots, \theta x_n)$, for every $f \in \Sigma$.

Suppose $\mathcal{A}$ and $\mathcal{B}$ are Σ-algebras. Call $\mathcal{B}$ a homomorphic image of $\mathcal{A}$ if there is a surjective Homomorphism from $\mathcal{A}$ to $\mathcal{B}$.

The next lemma states that if a homomorphism preserves valuations then it also preserves interpretation. We use it as a tool to prove a more important result, Lemma 3.20.

Lemma 3.19. Let $\theta : \mathcal{A} \to \mathcal{B}$ be a homomorphism. Suppose ς is a valuation to $\mathcal{A}$ and ς' is a valuation to $\mathcal{B}$. Also, suppose that $\theta(\varsigma(X)) = \theta(\varsigma'(X))$, for all unknowns X. Then $\theta([\![t]\!]^{\mathcal{A}}_\varsigma) = \theta([\![t]\!]^{\mathcal{B}}_\varsigma)$.

Proof. We reason by induction on the structure of t. The base case is trivial. We work on two interesting cases:

Suppose $t \equiv \pi \cdot X$,

$$\begin{aligned}\theta([\![\pi \cdot X]\!]^{\mathcal{A}}_{\varsigma}) &\equiv \theta(\pi \cdot \varsigma(X)) \\ &\equiv \pi \cdot \varsigma'(X) \\ &\equiv \theta([\![\pi \cdot X]\!]^{\mathcal{B}}_{\varsigma'}).\end{aligned}$$

Suppose $t \equiv [a]\, t$,

$$\begin{aligned}\theta([\![[a]\, t]\!]^{\mathcal{A}}_{\varsigma}) &\equiv \theta(\mathtt{abs}^{\mathcal{A}}(a, [\![t]\!]^{\mathcal{A}}_{\varsigma})) \\ &\equiv \mathtt{abs}^{\mathcal{B}}(a, \theta([\![t]\!]^{\mathcal{A}}_{\varsigma})) \\ &\overset{I.H}{\equiv} \mathtt{abs}^{\mathcal{B}}(a, \theta([\![t]\!]^{\mathcal{B}}_{\varsigma'})) \\ &\equiv \theta([\![[a]\, t]\!]^{\mathcal{B}}_{\varsigma})\end{aligned}$$

□

Lemma 3.20. Suppose $\mathcal{A}$ and $\mathcal{B}$ are Σ-algebras and that $\mathcal{B}$ is a homomorphic image of $\mathcal{A}$. If $\mathcal{A}$ is a model of T, then so is $\mathcal{B}$.

Proof. Fix some surjective homomorphism $\theta : A \to B$, choose any axiom $(\nabla \vdash t = u) \in Ax$, and a valuation ς to $\mathcal{B}$. It suffices to show that $[\![\nabla \vdash t = u]\!]^{\mathcal{B}}_{\varsigma}$ is valid. Suppose $[\![\nabla]\!]^{\mathcal{B}}_{\varsigma}$ is valid, by definition this means that $a \#_{\mathtt{sem}} \varsigma(X)$ for every $a\#X \in \nabla$.

For each unknown X define the set $\mathcal{X} = \{x \in \mathcal{A} \mid \theta(x) = \varsigma(X)\} \subseteq \mathcal{A}$, by equivariance of θ we get $\pi \cdot \mathcal{X} = \{x \in \mathcal{A} \mid \theta(x) = \pi \cdot \varsigma(X)\}$. Therefore if $\pi \cdot \varsigma(X) = \varsigma(X)$ then $\pi \cdot \mathcal{X} = \mathcal{X}$ and it follows that $\mathtt{supp}(\mathcal{X}) \subseteq \mathtt{supp}(\varsigma(X))$. Define the valuation ς' to $\mathcal{A}$ by setting $\varsigma'(X) = x$ for some choice of $x \in \mathcal{X}$ such that $a \#_{\mathtt{sem}} x$ for every $a\#x \in \nabla$, this choice exists by Lemma 3.5. This construction makes $[\![\nabla]\!]^{\mathcal{A}}_{\varsigma'}$ valid, and so since $\mathcal{A}$ is a model of T we have $[\![t]\!]^{\mathcal{A}}_{\varsigma'} = [\![u]\!]^{\mathcal{A}}_{\varsigma'}$. Applying θ to both sides of this equation we get, using Lemma 3.19,

$$[\![t]\!]^{\mathcal{B}}_{\varsigma} = [\![u]\!]^{\mathcal{B}}_{\varsigma}.$$

Therefore, $[\![\nabla \vdash t = u]\!]_{\varsigma}$ is valid in $\mathcal{B}$, as required. □

3.4.2 *Subalgebras*

Definition 3.22. For Σ-algebras $\mathcal{A}$ and $\mathcal{B}$ call $\mathcal{A}$ a subalgebra of $\mathcal{B}$, written as $\mathcal{A} \leq \mathcal{B}$, if the following conditions are satisfied:

- $A \subseteq B$,
- $a^{\mathcal{A}} = b^{\mathcal{B}}$, for all atoms $a \in \mathbb{A}$,
- $\mathtt{abs}^{\mathcal{A}}(a, x) = \mathtt{abs}^{\mathcal{B}}(a, x)$, for all atoms $a \in \mathbb{A}$ and all $x \in A$.
- The interpretation of functions is preserved on A, that is, for all $f \in \Sigma$, and all $x_1, \ldots, x_n \in A$, $f^{\mathcal{A}}(x_1, \ldots, x_n) = f^{\mathcal{B}}(x_1, \ldots, x_n)$.

What we are saying in this definition is that subalgebras are closed under the interpretation of atoms, abstractions, and function interpretation which it inherits from $\mathcal{B}$.

Lemma 3.21. Let $\mathcal{A}, \mathcal{B}$ be Σ-algebras. Suppose $\mathcal{B}$ is a model of $T = (\Sigma, Ax)$. If $\mathcal{A} \leq \mathcal{B}$ then $\mathcal{A}$ is a model of T.

Proof. Suppose $(\nabla \vdash t = u) \in Ax$ and suppose ς is a valuation to $\mathcal{A}$ such that $a \#_{\mathtt{sem}} \varsigma(X)$ for every $a\#X \in \nabla$. Since ς is also a valuation to $\mathcal{B}$ which is a model of T, it follows that $[\![t]\!]^{\mathcal{A}}_{\varsigma} = [\![u]\!]^{\mathcal{A}}_{\varsigma}$. Therefore, $\mathcal{A}$ satisfies all the axioms of T. □

3.4.3 *Product Algebras*

Definition 3.23. Let $(\mathcal{A}_i)_{i\in I}$ be a countable I-indexed family of Σ-algebras. The product algebra $\mathcal{P} = \Pi_{i\in I}\mathcal{A}_i$ is the Σ-algebra defined as:

- the domain is the product nominal set $P = \Pi_{i\in I} A_i$,
- $a^{\mathcal{P}} = (a^{\mathcal{A}_i})_{i\in I}$,
- $\mathtt{abs}^{\mathcal{P}}(a, (x_i)_{i\in I}) = \Pi_{i\in I}\mathtt{abs}^{\mathcal{A}_i}(a, x_i)$,
- For each term-former $f : n$, the component-wise interpretation function

$$f^{\mathcal{P}}((x_i)^1, \ldots, (x_i)^n) = (f(x_i^1, \ldots, x_i^n))_{i\in I}$$

Using Definition 3.10, one can easily show that the product algebra is indeed a nominal Σ-algebra.

Lemma 3.22. For any I-indexed family of Σ-algebras $(\mathcal{A}_i)_{i\in I}$, if $\mathcal{A}_i$ is a model of $T = (\Sigma, Ax)$ for every $i \in I$ then so is the product algebra $\mathcal{P}$.

Proof. Suppose that ς is a valuation to $\mathcal{P}$, take an axiom $(\nabla \vdash t = u) \in Ax$, and suppose $a \#_{\mathtt{sem}} \varsigma(X)$ for every $a \#_{\mathtt{sem}} X \in \nabla$.

For each $i \in I$ we obtain a valuation ς_i to the domain A_i of $\mathcal{A}_i$ by projecting to the i-th component of $\varsigma(X)$. By Lemma 3.6 we have that $a \#_{\mathtt{sem}} \varsigma_i(X)$ for every $a\#X \in \nabla$. Since for every $i \in I$ each Σ-algebra $\mathcal{A}_i$ is a model of T, we obtain, for every $i \in I$:

$$\mathrm{p}_i([\![t]\!]^{\mathcal{A}_i}_{\varsigma}) = \mathrm{p}_i([\![u]\!]^{\mathcal{A}_i}_{\varsigma})$$

where p_i is the i-th projection. It follows that

$$[\![t]\!]^{\mathcal{P}}_{\varsigma} = [\![t]\!]^{\mathcal{P}}_{\varsigma}\,.$$

□

3.4.4 *Atoms-abstraction*

Fix a nominal set X. Suppose $x \in X$ and $a \in \mathbb{A}$.

Definition 3.24. We define the set *atom-abstraction* $[a]\, x$ by

$$[a]\, x = \{(b, (b\ a) \cdot x) \mid b \#_{\mathtt{sem}} x\} \cup \{(a, x)\}$$

Write $[\mathbb{A}]\, X$ for the $\mathbb{P}$-set such that:

- The underlying set is $[\mathbb{A}]\, X = \{[a]\, x \mid a \in \mathbb{A},\ x \in X\}$,
- The permutation action is $\pi \cdot [a]\, x = [\pi(a)]\, \pi \cdot x$.

Lemma 3.23. If X be a nominal set, $x \in X$ and $a \in \mathbb{A}$, then $\mathtt{supp}([a]\, x) = \mathtt{supp}(x) \setminus \{a\}$.

Proof. The proof can be found in [23, Corollary 5.2]. □

Now we define the abstracted Σ-algebra $[\mathbb{A}]\, \mathcal{A}$ out of $\mathcal{A}$.

Definition 3.25. Suppose that $\mathcal{A}$ is a Σ-algebra. Define $[\mathbb{A}]\, \mathcal{A}$ by:

- The domain set is the nominal set $[\mathbb{A}]\, A$,
- $a^{[\mathbb{A}]\mathcal{A}} = [c]\, a^{\mathcal{A}}$, for any $c \neq a$.
- $\mathtt{abs}^{[\mathbb{A}]\mathcal{A}}(a, [c]\, x) = [c]\, \mathtt{abs}^{\mathcal{A}}(a, x)$, for any $c \neq a$.
- $f^{[\mathbb{A}]\mathcal{A}}([c]\, x_1, \ldots, [c]\, x_n) = [c]\, f^{\mathcal{A}}(x_1, \ldots, x_n)$.

Lemma 3.24. $[\mathbb{A}]\, \mathcal{A}$ is a nominal Σ-algebra.

Proof. The proof can be found in [19, Lemma 8.15] □

Lemma 3.25. If $y_1, \ldots, y_n \in [\mathbb{A}]\, X$ then for any $c \in \mathbb{A}$ such that $c \#_{\mathtt{sem}} y_1, \ldots, c \#_{\mathtt{sem}} y_n$ there exists $x_1, \ldots, x_n \in X$ such that $y_i = [c]\, x_i$ for $1 \leq i \leq n$.

Proof. Since $y_i \in [\mathbb{A}]\, X$, it is of the form $y_i = [d]\, x_i$, for some $d \in \mathbb{A}$ and $x_i \in X$. Also, by Definition 3.24;

$$y_i = \{(b, (b\ d) \cdot x_i) \mid b \#_{\mathtt{sem}} x_i\} \cup \{(d, x_i)\}$$

from $c \#_{\mathtt{sem}} y_i$ it follows, using Lemma 3.23, that $c \notin \mathtt{supp}(y_i) = \mathtt{supp}(x_i) \setminus \{d\}$ so $c \#_{\mathtt{sem}} x_i$ then the pair $(c, (c\ d) \cdot x_i) \in y_i$ and we can take $y_i = [c]\, x_i$, as required. □

Lemma 3.26. If $[a]\, x = [c]\, y$ then $x = y$.

Proof. From $[c]\, x = [c]\, y$ and using Definition 3.24 we have that for every atom b such that $b \#_{\mathtt{sem}} x$: $(b, (b\ c) \cdot x) = (b, (b\ c) \cdot y)$, so $(b\ c) \cdot x = (b\ c) \cdot y$ and finally $x = y$. □

Definition 3.26. If $\mathcal{A}$ is a Σ-algebra and ς is a valuation to A, then write $[c]\, \varsigma$ for the valuation to $[\mathbb{A}]\, A$ such that X maps to $[c]\, \varsigma(X)$.

Lemma 3.27. Suppose that $\mathcal{A}$ is a Σ-algebra and ς a valuation to A. If $c \notin \mathtt{atms}(t)$ then

$$[\![t]\!]^{[\mathbb{A}]\mathcal{A}}_{[c]\varsigma} = [c]\,[\![t]\!]^{\mathcal{A}}_{\varsigma}$$

Proof. By a standard induction on t. □

Corollary 3.4. Suppose that $c \notin \mathtt{atms}(t,u)$. Then $[\![t]\!]^{\mathcal{A}}_{\varsigma} = [\![u]\!]^{\mathcal{A}}_{\varsigma}$ if, and only if, $[\![t]\!]^{[\mathbb{A}]\mathcal{A}}_{[c]\varsigma} = [\![u]\!]^{[\mathbb{A}]\mathcal{A}}_{[c]\varsigma}$.

Proof. If $[\![t]\!]^{\mathcal{A}}_{\varsigma} = [\![u]\!]^{\mathcal{A}}_{\varsigma}$ then $[c]\,[\![t]\!]^{\mathcal{A}}_{\varsigma} = [c]\,[\![u]\!]^{\mathcal{A}}_{\varsigma}$ and the result follows from Lemma 3.27.

Conversely, suppose $[\![t]\!]^{[\mathbb{A}]\mathcal{A}}_{[c]\varsigma} = [\![u]\!]^{[\mathbb{A}]\mathcal{A}}_{[c]\varsigma}$. By Lemma 3.27 $[c]\,[\![t]\!]^{\mathcal{A}}_{\varsigma} = [c]\,[\![u]\!]^{\mathcal{A}}_{\varsigma}$. The result now follows by Lemma 3.26. □

Now we can prove the main result of this subsection.

Lemma 3.28. If $\mathcal{A}$ is a model of T then so is $[\mathbb{A}]\,\mathcal{A}$.

Proof. Suppose $(\nabla \vdash t = u) \in Ax$. Take ς a valuation to $[\mathbb{A}]\,A$ such that $a \#_{\mathtt{sem}} \varsigma(X)$ for every $a\#X \in \nabla$. Now choose some fresh atom c such that $c \notin \mathtt{atms}(\nabla, t, u)$ and such that $c \#_{\mathtt{sem}} \varsigma(X)$ for every X in $\mathtt{var}(\nabla, t, u)$. By using Lemma 3.25, construct a valuation ς' to A such that $\varsigma(X) = ([c]\,\varsigma')(X)$ for every X in $\mathtt{var}(\nabla, t, u)$, and therefore such that

$$[\![t]\!]^{[\mathbb{A}]\mathcal{A}}_{\varsigma} = [\![t]\!]^{[\mathbb{A}]\mathcal{A}}_{[c]\varsigma'} \qquad \text{and} \qquad [\![u]\!]^{[\mathbb{A}]\mathcal{A}}_{\varsigma} = [\![t]\!]^{[\mathbb{A}]\mathcal{A}}_{[c]\varsigma'}.$$

By Lemma 3.27,

$$[\![t]\!]^{[\mathbb{A}]\mathcal{A}}_{[c]\varsigma'} = [c]\,[\![t]\!]^{\mathcal{A}}_{\varsigma'} \qquad \text{and} \quad [\![u]\!]^{[\mathbb{A}]\mathcal{A}}_{[c]\varsigma'} = [c]\,[\![u]\!]^{\mathcal{A}}_{\varsigma'}.$$

By Lemma 3.23 $a \#_{\mathtt{sem}} \varsigma'(X)$ for every $a\#X \in \nabla$. Finally, since $\mathcal{A}$ is a model of T we have $[\![t]\!]^{\mathcal{A}}_{\varsigma'} = [\![u]\!]^{\mathcal{A}}_{\varsigma'}$ and therefore $[c]\,[\![t]\!]^{\mathcal{A}}_{\varsigma'} = [c]\,[\![u]\!]^{\mathcal{A}}_{\varsigma'}$. The result follows. □

3.5 VARIETIES AND EQUATIONAL CLASSES OF ALGEBRAS

Definition 3.27. A (nominal algebra) variety $\mathcal{V}$ for a signature Σ is a collection of Σ-algebras $\in$-closed under homomorphic image, subalgebra, countable product, and atoms-abstraction.

Definition 3.28. The collection $\mathcal{V}$ of Σ-algebras is called *(nominal algebra) equational* if there is a theory $T = (\Sigma, Ax)$ such that $\mathcal{V}$ is the collection of all models of T.

We now are able to state the main result of this chapter: a version of the HSP theorem for nominal algebras.

Theorem 3.12 (Nominal HSP Theorem)**.** A collection of Σ-algebras $\mathcal{V}$ is **equational** if, and only if, it is a **variety**.

The proof of the HSP Theorem takes the rest of this section. Fix a theory $T = (\Sigma, Ax)$ and a collection $\mathcal{V}$ of Σ-algebras[2].

2 For now we do not assume $\mathcal{V}$ is a variety.

3.5.1 *Surjections Out of Initial Algebras*

Finally, we give to $\mathbb{F}(T, \mathcal{D})$ its *initiallity* we talked about earlier. We first show that every Σ-algebra $\mathcal{A}$ is the homomorphic image of some sufficient large initial algebra.

Lemma 3.29. Suppose $\mathcal{A}$ is a model of $T = (\Sigma, Ax)$. Then the following determine a Σ-algebra homomorphism from $\mathbb{F}(T, \mathcal{D})$ to $\mathcal{A}$: for each term-former $\mathcal{D}$

- a choice of n atoms $a_1, \ldots, a_n$, and
- a choice of element $\theta\mathbf{d}(a_1, \ldots, a_n) \in A$ such that $\mathtt{supp}(\theta\mathbf{d}(a_1, \ldots, a_n)) \subseteq \{a_1, \ldots, a_n\}$.

Proof. Define the relation $\theta \subseteq \mathbb{F}(\Sigma, \mathcal{D}) \times A$ by:

- $\theta\mathbf{d}(\pi(a_1), \ldots, \pi(a_n)) = \pi \cdot \theta\mathbf{d}(a_1, \ldots, a_n)$.
- $\theta a = a^{\mathcal{A}}$.
- $\theta[a]\, g = \mathtt{abs}^{\mathcal{A}}(a, \theta[g]_T)$.
- $\theta f(g_1, \ldots, g_n) = f^{\mathcal{A}}(\theta g_1, \ldots, \theta g_n)$.

We first need to show that θ is a well defined function $\theta : \mathbb{F}(T, \mathcal{D}) \to A$. That is, $[g]_T = [h]_T$ implies $\theta g = \theta h$. Indeed, let Π be a ground derivation of $\vdash_T g = h$. Consider the construction of the inverse mapping, Definition 3.19, and take σ a substitution as in Definition 3.17. Let $\varsigma(X) = \theta(X\sigma)$. We claim that $\theta g = \left[\!\left[g^{-1} \right]\!\right]_\varsigma$ and $\theta h = \left[\!\left[h^{-1} \right]\!\right]_\varsigma$.

Proof of the Claim. We prove it by induction on g. The atomic case is trivial. We work on one inductive step: Suppose $g \equiv [a]\, g'$. So

$$\theta[a]\, g' = \mathtt{abs}^{\mathcal{A}}(a, \theta[g']_T) \qquad \begin{aligned} \left[\!\left[([a]\, g')^{-1} \right]\!\right]_\varsigma &= \left[\!\left[[a]\, (g')^{-1} \right]\!\right]_\varsigma \\ &= \mathtt{abs}^{\mathcal{A}}(a, \left[\!\left[(g')^{-1} \right]\!\right]_\varsigma) \\ &= \mathtt{abs}^{\mathcal{A}}(a, \theta g') \end{aligned}$$

□

Now θ can be viewed as a function $\theta : \mathbb{F}(T, \mathcal{D}) \to A$. Also, one can show $\pi \cdot \theta g = \theta(\pi \cdot g)$ by induction on g. It follows that;

- $\pi \cdot (\theta[g]_T) = \pi \cdot \theta g = \theta\pi \cdot g = \theta[g]_T$.
- $\theta a^{\mathbb{F}(T,\mathcal{D})} = a^{\mathcal{A}}$.
- $\theta f^{\mathbb{F}(T,\mathcal{D})}([g_1]_T, \ldots, [g_n]_T) = f^{\mathcal{A}}(\theta[g_1]_T, \ldots, [g_n]_T)$.

So θ is a Σ-algebra homomophism from the initial ground algebra $\mathbb{F}(T, \mathcal{D})$ to $\mathcal{A}$. So by Lemma 3.18, $\Delta \vdash_T g^{-1} = h^{-1}$. Therefore, the result follows by Soundness, Theorem 3.8. □

Theorem 3.13. If $\mathcal{A} \in \mathcal{V}$ then there exists some (sufficiently large) set of fresh term-formers $\mathcal{D}$ such that there exists a surjective Σ-algebra homomophism $\theta : \mathbb{F}(T, \mathcal{D}) \to \mathcal{A}$.

Proof. As usual, we write $|A|$ for the cardinality of A. Suppose that $\mathcal{D}$ is a set of term-formers with cardinality at least $|A|$ for every arity $n > 0$, and with no term-formers with cardinality 0. We shall exhibit a suitable θ from $\mathbb{F}(T, \mathcal{D})$ to $\mathcal{A}$.

For each permutation equivalence class $\{\pi \cdot x \mid \pi \in \mathbb{P}\} \subseteq A$ choose a representative $x \in A$. And for each such x,

- order the support of x as $a_1, \ldots, a_n$, pick a unique n-ary term-former $\mathbf{d} \in \mathcal{D}$, and
- assign $\theta\mathbf{d}(a_1, \ldots, a_n) = x$.

For each remaining unassigned $\mathbf{d}(b_1, \ldots, b_n)$ assign $\theta\mathbf{d}(b_1, \ldots, b_n)$ to $b_1^{\mathcal{A}}$. By Lemma 3.29, this choice extends to a homomophism θ from $\mathbb{F}(T, \mathcal{D})$ to $\mathcal{A}$.

To see that θ is a surjection consider any $x' \in A$. By construction there exists some representative x such that $x' \in \{\pi \cdot x \mid \pi \in \mathbb{P}\}$. So write $x' = \pi \cdot x$. Hence, $x = \theta\mathbf{d}(a_1, \ldots, a_n)$ and so $x' = x = \theta\mathbf{d}(a_1, \ldots, a_n)$. □

3.5.2 *Injections Out of Initial Algebras*

Fix a countable collection $\mathcal{V}$ of Σ-algebras. We now show that there exists a homomophism from the initial algebra $\mathbb{F}(T, \mathcal{D})$ to the product $\Pi_{i \in I} \mathcal{B}_i$.

Definition 3.29. Let $\mathcal{A}$ ($\mathcal{B}_{i \in I}$) be a Σ-algebra (I-indexed family of Σ-algebras) such that $\mathcal{A}$ ($\mathcal{B}_i$) are all models of T, for $i \in I$. Let $\theta_i : \mathcal{A} \to \mathcal{B}_i$ be a I-indexed family of homomophisms. Then write $\theta = \Pi_{i \in I} \theta_i$ for the natural map from $\mathcal{A}$ to $\Pi_{i \in I} \mathcal{B}_i$, mapping $x \in A$ to $(\theta_i x)_{i \in I} \in \Pi_{i \in I} B_i$.

It is easy to verify that θ defined in Definition 3.29 is a Σ-algebra homomophism.

Definition 3.30. Let $T = (\Sigma, Ax)$ where Ax is the collection of judments valid in all $\mathcal{B} \in \mathcal{V}$ for all valuations. Call T the theory generated by $\mathcal{V}$.

Remark 3.3. In the definition above we are saying that $(\nabla \vdash t = u) \in Ax$ exactly when for every $\mathcal{B} \in \mathcal{V}$ and every possible valuation ς to B, it is the case that if $a \#_{\mathtt{sem}} \varsigma(X)$ for every $a\#X \in \nabla$ then $[\![t]\!]^{\mathcal{B}}_{\varsigma} = [\![u]\!]^{\mathcal{B}}_{\varsigma}$.

Theorem 3.14. Let $\mathcal{V}$ be a family of Σ-algebras and T be the theory generated by $\mathcal{V}$. Suppose $\mathcal{D}$ is any set of fresh term-formers (so $\mathcal{D} \cap \Sigma = \emptyset$).

Then there exists a countable I-indexed set of Σ-algebras $\mathcal{B}_i \in \mathcal{V}$, for $i \in I$, such that there exists a injective Σ-algebra homomophism $\theta : \mathbb{F}(T, \mathcal{D}) \to \mathcal{P} = \Pi_{i \in I} B_i$.

Proof. Let I be the set of pairs (g, h) of ground terms in the signature $\Sigma \cup \mathcal{D}$ such that $[g]_T \neq [h]_T$. Note that I is a countable set since $\mathbb{F}(\Sigma, \mathcal{D})$ is countable. Fix some arbitrary but fixed $i = (g, h) \in I$. We assumed that $\nvdash_T g = h$, so by Corollary 3.3 $\Delta \nvdash_T t = u$, where $t = g^{-1}\sigma$ and

$u = h^{-1}\sigma$, as in Lemma 3.18. By our assumption that T is generated by $\mathcal{V}$, there exists some model $\mathcal{B}_i$ in $\mathcal{V}$ and valuation ς such that $[\![\Delta]\!]^{\mathcal{B}_i}_{\varsigma}$ is valid whereas $[\![t]\!]_{\varsigma} \neq [\![u]\!]_{\varsigma}$. Define the set

$$\{c_1, \ldots, c_p\} := \left(\bigcup_{X \in \mathcal{X}} \mathtt{supp}(\varsigma X) \right) \setminus \mathcal{A}$$

where $\mathcal{A}$ and $\mathcal{X}$ are as in Definition 3.16. Write $\mathcal{B}'_i$ for $[\mathbb{A}]\,\mathcal{B}_i$ and write ς' for $[c_1]\ldots[c_p]\varsigma$ (see Definition 3.26). Since $\mathcal{V}$ is closed under atoms-abstraction we have $\mathcal{B}_i \in \mathcal{V}$. By Corollary 3.4

$$[\![t]\!]^{\mathcal{B}'_i}_{\varsigma'} \neq [\![u]\!]^{\mathcal{B}'_i}_{\varsigma'}$$

We construct a choice θ by setting $\theta\mathbf{d}(a_1, \ldots, a_n) = \varsigma'(X)$ for each $\mathbf{d} \in \mathcal{D}$, where $X \in \mathcal{X}$ is the unknown corresponding to $\mathbf{d}$, and $a_1, \ldots, a_n$ is the choice of atoms in order corresponding to $\mathbf{d}$, in the sense of Definition 3.16 and Def. 3.17.

In order to use Lemma 3.29 to obtain a homomophism θ from $\mathbb{F}(T, \mathcal{D})$ to $\mathcal{B}'_i$ we must verify that $\mathtt{supp}(\varsigma'(X)) \subseteq \{a_1, \ldots, a_n\}$. This is done as follows:

Suppose that $a \notin \{a_1, \ldots, a_n\}$. By the rules in Figure 2.1 $\vdash a\#\mathbf{d}(a_1, \ldots, a_n)$. By Lemma 3.18, $\Delta \vdash a\#\mathbf{d}(a_1, \ldots, a_n)^{-1}$. By assumption $[\![\Delta]\!]^{\mathcal{B}'_i}_{\varsigma'}$ so by Soundness (Theorem 3.8) $a \#_{\mathtt{sem}} [\![\mathbf{d}(a_1, \ldots, a_n)]\!]^{\mathcal{B}'_i}_{\varsigma'}$. By using Definition 3.19 we can see that $\mathbf{d}(a_1, \ldots, a_n)^{-1} = X$ and with that we deduce that $a \#_{\mathtt{sem}} \varsigma'(X)$. So it follows that $\mathtt{supp}(\varsigma'(X)) \subseteq \{a_1, \ldots, a_n\}$, as required.

It follows by the choice of $\mathcal{B}_i$ that $\theta := \Pi_{i \in I}\theta_i$, as in Definition 3.29, from $\mathbb{F}(T, \mathcal{D})$ to $\mathcal{P} = \Pi_{i \in I}\mathcal{B}'_i$ is an injective Σ-algebra homomophism. □

The next Lemma is the last piece of the puzzle, it says that the initial algebra $\mathbb{F}(T, \mathcal{D})$ is always inside a variety in which T is the theory generated by it.

Lemma 3.30. Suppose $\mathcal{V}$ is a variety and suppose T is the theory generated by $\mathcal{V}$. Then $\mathbb{F}(T, \mathcal{D}) \in \mathcal{V}$ for every set of term-formers $\mathcal{D}$.

Proof. By Theorem 3.14 there is some I-indexed family of Σ-algebras and a injective homomophism $\theta : \mathbb{F}(T, \mathcal{D}) \to \mathcal{P} = \Pi_{i \in I}\mathcal{B}_i$. Since $\mathcal{V}$ is closed under products, $\mathcal{P} \in \mathcal{V}$. The homomorphic image of $\mathbb{F}(T, \mathcal{D})$ is a subalgebra of $\mathcal{P}$, and since $\mathcal{V}$ is closed under subalgebras and homomorphic images the result follows. □

3.5.3 *The Nominal HSP Theorem*

Finally, we give a proof for the Nominal HSP Theorem using most of the results proved so far.

Proof of the Nominal HSP Theorem. Suppose $\mathcal{V}$ is equational. So, by Lemma 3.22 $\mathcal{V}$ is closed under products. Lemma 3.20 says $\mathcal{V}$ is closed under homomorphic images, by Lemma 3.21 $\mathcal{V}$ is closed under subalgebras,

and from Lemma 3.28 we know $\mathcal{V}$ is closed under atom-abstraction. Therefore, $\mathcal{V}$ is a variety.

Conversely, suppose $\mathcal{V}$ is a variety. Let T be the theory generated by $\mathcal{V}$. Let $\mathcal{B}$ be any model of T. By Theorem 3.13 there exits some $\mathcal{D}$ such that $\mathcal{B}$ is the homomorphic image of $\mathbb{F}(T, \mathcal{D})$. By Lemma 3.30 $\mathbb{F}(T, \mathcal{D}) \in \mathcal{V}$. Since $\mathcal{V}$ is closed under homomophisms, $\mathcal{B} \in \mathcal{V}$ as required. Therefore $\mathcal{V}$ is equational. □

We also consider the term-algebra $\mathcal{T}(\Sigma, \mathbb{A}, \mathbb{X})$, as in the first-order case, this term-algebra is *generic* (see [9]) for solving existentially closed equations.

Corollary 3.5. Let ϕ be the existentially closed equational judgement:

$$\phi ::= \exists \vec{X}(\Delta \vdash s =_T t).$$

Then $\Delta \vdash_{\mathbb{F}(T,\mathcal{D})} \phi$ if, and only if, $\Delta \vdash_{\mathcal{T}(\Sigma,\mathbb{A},\mathbb{X})} \phi$.

Example 3.6. Corollary 3.5 does not hold for disequations, to see this consider the theory $T = (\Sigma, Ax)$ where $\Sigma = \{f(\_)\}$ and $Ax = \{\vdash f(X) = a, \vdash [a]\, f(X) = a\}$. In $\mathcal{T}(\Sigma, \mathbb{A}, \mathbb{X})$ one can derive $\vdash f(b) = a$ and $\vdash f(b) = b$:

$$\dfrac{\dfrac{\dfrac{}{\vdash id \cdot f(X)\sigma = a\sigma}\,(ax_{\Delta' \vdash f(X)=a})}{\vdash f(b) = a}\,(\text{symm}) \qquad \dfrac{}{\vdash (a\ b) \cdot f(X)\sigma' = (a\ b) \cdot a\sigma'}\,(ax_{\Delta' \vdash f(X)=a})}{\vdash a = f(b) \qquad \vdash f(b) = b}\,(\text{trans})$$
$$\vdash a = b$$

by taking $\sigma = \{X/b\}$ in the left branch and $\sigma' = \{X/a\}$ in the right branch. The dashed lines in the derivation above represent the result obtained after the application of substitutions σ and σ', and the swapping of names in the axiom $\vdash f(X) = a$. Therefore, every atom is in the same equivalence class modulo T.

Also, from the axioms of T one can derive $\vdash f(t) = a$ and $\vdash [b]\, f(u) = a$ for any pair of terms t and u. Hence it is possible to derive $\vdash t' = u'$ for any non-variable terms t' and u'. The other equivalence classes are for variables $X, Y, Z \ldots$.

It follows that $\vdash_{\mathcal{T}(\Sigma,\mathbb{A},\mathbb{X})} \exists X.X \neq a$. However, $\mathbb{F}(T, \mathcal{D})$ (with $\mathcal{D} = \emptyset$) has only one equivalence class, i.e., the class of all ground terms and atoms, therefore, $\nvdash_{\mathbb{F}(T,\mathcal{D})} \exists X.X \neq a$ since every ground term and atom are in the same equivalence class modulo T.

4

NOMINAL DISUNIFICATION CONSTRAINTS

In this chapter, we follow the approach proposed by Buntine and Bürckert [9] for solving a system of equations and disequations. Our approach, as in the first-order case, depends on the unification type of a (nominal) theory T. Fix the nominal algebra $\mathcal{T}(\Sigma, \mathbb{A}, \mathbb{X})$ with the empty set of axioms, that is, terms are considered up to the built-in α-equivalence. The results in this section can be extended to any theory T provided unification is decidable and finitary for this theory; pure α-equivalence has been chosen to make examples and proofs easier to follow.

In chapter 3 we study nominal universal algebra and define precisely what we mean by a nominal equational theory.

Definition 4.1. A *(nominal disunification) constraint problem* $\mathcal{P}$ is an ordered pair $\mathcal{P} = \langle E \mid\mid D \rangle$ where E is a nonempty set of nominal equations-in-context $\Delta \vdash s \approx_\alpha t$ and D is a (possible empty) set of nominal disequations-in-context $\nabla \vdash p \not\approx_\alpha q$, as follows:

$$E = \{\Delta_1 \vdash s_1 \approx_\alpha{}^? t_1, \cdots, \Delta_n \vdash s_n \approx_\alpha{}^? t_n\}$$
$$D = \{\nabla_1 \vdash p_1 \not\approx_\alpha{}^? q_1, \cdots, \nabla_m \vdash p_m \not\approx_\alpha{}^? q_m\}$$

The sets $\Delta_1, \ldots, \Delta_n, \nabla_1, \ldots, \nabla_m$ are consistent contexts. We call them the *initial freshness conditions* that are imposed on equations (disequations) in the problem $\mathcal{P}$.

In the case any of the Δ_i or ∇_j of a problem is empty we may write an equation(disequation)-in-context just as $s_i \approx_\alpha{}^? t_i$ ($p_j \not\approx_\alpha{}^? q_j$) instead of $\emptyset \vdash s \approx_\alpha{}^? t$. We also may consider the equations and disequations of the problems under the same context, $\Delta := \cup\Delta_i$ and $\nabla := \cup\nabla_j$.

Remark 4.1. A constraint problem is equivalent to an existentially closed formula:

$$\mathcal{P} := \exists \vec{X} \left(\left(\bigwedge \Delta_i \vdash s_i \approx_\alpha t_i \right) \wedge \left(\bigwedge \nabla_j \vdash p_j \not\approx_\alpha q_j \right) \right).$$

We solve these formulas in the nominal term-algebra $\mathcal{T}(\Sigma, \mathbb{A}, \mathbb{X})$ (see [19, 22]), this is the logical task of finding witnesses/solutions for the variables in $\mathcal{P}$, that is, a pair $\langle \Gamma, \sigma \rangle$ where σ is a substitution for the variables of the formula such that under some (possible empty) consistent context Γ we have $\Gamma \vdash \mathcal{P}\sigma$.

To give some intuition on the construction of solution pairs, consider the constraint problem below:

$$\mathcal{P} = \left\langle (b\ a) \cdot X \approx_\alpha{}^? Y \mid\mid [a]X \not\approx_\alpha{}^? [b]Y \right\rangle \tag{4.1}$$

The intended effect of a solution $\langle \Gamma, \sigma \rangle$ of $\mathcal{P}$ is that it needs to solve the equation $(b\ a) \cdot X \approx_\alpha{}^? Y$ and the disequation $[a]X \not\approx_\alpha{}^? [b]Y$ where

solving this disequation means $\Gamma \nvdash [a]\, X\sigma \approx_\alpha [b]\, Y\sigma$, i.e., $\langle \Gamma, \sigma \rangle$ is not a solution of the equation $[a]\, X \approx_\alpha{}^? [b]\, Y$ which will be called the *associated equation* to the problem $[a]\, X \not\approx_\alpha{}^? [b]\, Y$. Notice that,

$$\langle \Gamma, \sigma \rangle = \langle \emptyset, [Y/(b\ a) \cdot X] \rangle \tag{4.2}$$

solves the constraint problem $\mathcal{P}$ above. The main goal of this section is show how to construct these solutions.

In general, instantiation plays an important role in unification theory. It is by instances of more general unifiers (instantiation closure) that one produces a finite representation of all other solutions of a unification problem. Therefore, it is helpful to have the property of instantiation closure to solutions of constraints problems as well. Unfortunately, this is not the case since we are solving constraints in the nominal term-algebra $\mathcal{T}(\Sigma, \mathbb{A}, \mathbb{X})$. For an example, let $\mathcal{Q}$ be the constraint problem:

$$\mathcal{Q} = \left\langle X \approx_\alpha{}^? Y \;||\; X \not\approx_\alpha{}^? a \right\rangle$$

The pair $\langle \Gamma, \sigma \rangle = \langle \emptyset, [X \mapsto (a\ b) \cdot Z, Y \mapsto (a\ b) \cdot Z] \rangle$ solves $\mathcal{Q}$. However, if we instantiate this solution with $\delta = [Z \mapsto b]$ the instance $\langle \emptyset, [X/a, Y/a] \rangle$ is not a solution of $\mathcal{Q}$.

Example 4.1. Let $\langle \Gamma, \sigma \rangle = \langle \emptyset, [Y/(b\ a) \cdot X] \rangle$, as in (4.2). Consider the pair

$$\langle \Gamma', \sigma \rangle = \langle b\#X, [Y/(b\ a) \cdot X] \rangle.$$

Notice that $\langle \Gamma, \sigma \rangle \leq \langle \Gamma', \sigma \rangle$, therefore $\langle \Gamma', \sigma \rangle$ solves $(b\ a) \cdot X \approx_\alpha{}^? Y$. In addition, $\langle \Gamma', \sigma \rangle$ is a solution of the equation $[a]\, X \approx_\alpha{}^? [b]\, Y$ associated to $[a]\, X \not\approx_\alpha{}^? [b]\, Y$. It can not solve (4.1) since it solves the equations and the associated equation $[a]\, X \approx_\alpha{}^? [b]\, Y$.

The reader may wonder if such an anomaly is caused by the context assumptions added on the initial problem or, reasonably enough, even by the α-equivalence embedded in the theory of nominal terms. Certainly, context assumptions seem to cause some difficulties. Firstly, because that the notion of instantiation may introduce new freshness constraints, as in Example 4.1. Secondly, freshness conditions on the equational part of a constraint problem can interact with solutions and, as showed in the example below, even change the solvability of a problem.

Example 4.2. Consider the following modification of the original problem (4.1):

$$\mathcal{P}' = \left\langle b\#X \vdash (b\ a) \cdot X \approx_\alpha{}^? Y \;||\; [a]\, X \not\approx_\alpha{}^? [b]\, Y \right\rangle$$

Notice that $\mathcal{P}'$ does not have a solution: every time we solve $b\#X \vdash (b\ a) \cdot X \approx_\alpha{}^? Y$ we always solve the equation $[a]\, X \approx_\alpha{}^? [b]\, Y$ associated to $[a]\, X \not\approx_\alpha{}^? [b]\, Y$.

We will work on this type of issues in the remaining of the dissertation. First, we define precisely what we mean by a solution of nominal constraint problems. Keep in mind that our goal is the development of a nominal generalization for the already established notion of instantiation of solutions (Definition 2.17), but this needs to be done in such a way that instantiation closure still holds.

4.1 GENERALIZED INSTANTIATION

In this section, some notions initially established in [9] will be extended into the nominal framework. The main difference is the lifting of the notion of *substitution with exceptions* to pairs of the form $\langle \Gamma, \sigma \rangle$ consisting of a consistent freshness context and a substitution, in addition of course, to the fact that α-equality is axiomatized in nominal terms which adds some complexity when compared to syntactic equality. Besides, we have adapted the Consistency Test Algorithm (Algorithm 1) to deal with pairs with exceptions.

Definition 4.2. A *pair with exceptions*, denoted as $\langle \Gamma, \sigma \rangle - \Psi$, consists of a pair $\langle \Gamma, \sigma \rangle$ and an I-indexed family of the form $\Psi = \{\langle \nabla_l, \psi_l \rangle \mid l \in I\}$.

Pairs with exceptions will be used as a representation of solutions of a constraint problem that has restrictions on how they can be instantiated. For instance, in the problem $\mathcal{Q}$ above, solutions of the equation $X \approx_\alpha Y$ can be instantiated in any way *except* for the instances where X is mapped to a.

Definition 4.3 (Pair instances)**.**

- A pair $\langle \Gamma, \sigma \rangle$ is said to be an *instance of a family* $\Psi = \{\langle \nabla_l, \psi_l \rangle \mid l \in I\}$, denoted by $\Psi \leq \langle \Gamma, \sigma \rangle$, if and only if each instance of $\langle \Gamma, \sigma \rangle$ is an instance of some $\langle \nabla_l, \psi_l \rangle$ in Ψ.
- A pair $\langle \Delta, \lambda \rangle$ is an *instance of a pair with exceptions* $\langle \Gamma, \sigma \rangle - \Psi$, written $\langle \Gamma, \sigma \rangle - \Psi \leq \langle \Delta, \lambda \rangle$, if and only if $\langle \Delta, \lambda \rangle$ is an instance of $\langle \Gamma, \sigma \rangle$ but not of Ψ.

Definition 4.4. A pair with exceptions $\langle \Gamma, \sigma \rangle - \Psi$ is consistent if and only if it has at least one instance.

For example, the pair with exceptions $\langle b\#X, [Y/(b\ a) \cdot X] \rangle - \{(b\#X, [Y/(b\ a) \cdot X])\}$ from Example 4.2 is inconsistent. The following lemma is a useful characterization of consistency for pair with exceptions.

Lemma 4.1 (Inconsistency Lemma)**.** A pair with exceptions $\langle \Gamma, \sigma \rangle - \Psi$ is inconsistent if and only if $\langle \Gamma, \sigma \rangle$ is an instance of Ψ.

Proof.

($\Rightarrow$) If $\langle \Gamma, \sigma \rangle$ is an instance of Ψ then all instances $\langle \Delta, \gamma \rangle \leq \langle \Gamma, \sigma \rangle$ is an instance of some $\langle \nabla_i, \psi_i \rangle$ in Ψ, so by Definition 4.3 $\langle \Gamma, \sigma \rangle - \Psi$ has no instances hence it is inconsistent.

($\Leftarrow$) Conversely, suppose $\langle \Gamma, \sigma \rangle - \Psi$ is consistent and $\langle \Gamma, \sigma \rangle$ is an instance of Ψ. Then there exists an instance $\langle \Delta, \lambda \rangle$ of $\langle \Gamma, \sigma \rangle - \Psi$. Hence $\langle \Gamma, \sigma \rangle \leq \langle \Delta, \lambda \rangle$. Since $\langle \Gamma, \sigma \rangle$ is an instance of Ψ we have $\langle \nabla_i, \psi_i \rangle \leq \langle \Gamma, \sigma \rangle$, by transitivity

$$\langle \nabla_i, \psi_i \rangle \leq \langle \Gamma, \sigma \rangle \leq \langle \Delta, \lambda \rangle$$

a contradiction with Definition 4.3.

□

Recalling Definition 3.11, we say that a pair with exceptions $\langle \Gamma, \sigma \rangle - \psi$ is inconsistent on a Σ-algebra $\mathcal{A}$ iff $\mathit{instances}(\langle \Gamma, \sigma \rangle - \Psi) = \emptyset$ in $\mathcal{A}$, where

$$\mathit{instances}(\langle \nabla, \rho \rangle) = \{\langle \nabla, \rho \rangle \varsigma \mid \text{ for all valuation } \varsigma\}.$$

Corollary 4.1. If $\langle \Gamma, \sigma \rangle - \psi$ is inconsistent on $\mathcal{T}(\Sigma, \mathbb{A}, \mathbb{X})$ then it is inconsistent on the ground algebra $\mathbb{F}(\mathrm{CORE}, \mathcal{D})$.

Proof. If $\langle \Gamma, \sigma \rangle - \psi$ is inconsistent on $\mathcal{T}(\Sigma, \mathbb{A}, \mathbb{X})$ then each instance $\langle \Gamma', \sigma' \rangle$ of $\langle \Gamma, \sigma \rangle$ is in turn an instance of some $\langle \Delta_l, \psi_l \rangle \in \Psi$, i.e., in terms of Definition 2.17, there exists δ such that

$$\text{for all } X \in \mathbb{X}, \qquad \Gamma' \vdash_{\mathcal{T}(\Sigma,\mathbb{A},\mathbb{X})} X\sigma' = X\psi_l\delta \qquad \text{and} \qquad \Gamma' \vdash \Delta_l\delta.$$

The result follows from Corollary 3.5. Notice that the converse is not true in general. For instance, consider the theory T as in Example 3.6, all pair with exceptions are inconsistent on $\mathbb{F}(T, \mathcal{D})$ since it has only one equivalence class but this not happens in $\mathcal{T}(\Sigma, \mathbb{A}, \mathbb{X})$. □

Corollary 4.2. Let $\langle \Gamma, \sigma \rangle - \Psi$ be a pair with exceptions. If there is some $\langle \nabla_l, \psi_l \rangle \in \Psi$ such that there exists a substitution δ satisfying

$$\Gamma \vdash X\sigma \approx_\alpha X\psi_l\delta, \text{ for all } X \in \mathtt{vars}(\mathcal{P}).$$

Then $\langle \Gamma, \sigma \rangle - \Psi$ is inconsistent if and only if $\Gamma \supseteq \langle \nabla_l\delta \rangle_{\mathtt{nf}}$.

Proof. Consider the pair with exceptions $\langle \Gamma, \sigma \rangle - \Psi$ as above. From assumption, σ is an instance of ψ_l over the context Γ. By the inconsistency lemma this pair with exceptions is inconsistent iff $\langle \Gamma, \sigma \rangle$ is an instance of Ψ. The result follows from the fact that $\Gamma \vdash \nabla_l\delta$ iff $\Gamma \supseteq \langle \nabla_l\delta \rangle_{\mathtt{nf}}$. □

The above corollary enables us to algorithmically test if some pair with exceptions is consistent provided that we have already solved the matching-in-context problem $(\Gamma \vdash X\sigma) \approx_? (\Gamma \vdash X\psi)$ (for all variables X appearing in the constraint problem) where $X\sigma$ is the pattern (see [18, Definition 45]). That is, for each ψ_l we solve the unification problem

$$\Gamma \vdash X_1\sigma \approx_\alpha^? X_1\psi_l, \cdots, X_n\sigma \approx_\alpha^? X_n\psi_l$$

without instantiating variables of $X_i\sigma$, for all $1 \leq i \leq n$. The solution of this matching problem (if it exists) will be denoted by δ. In [11], the authors give an efficient implementation for the matching problem.

input : $\langle \Gamma, \sigma \rangle - \psi$ a finite pair with exceptions
output: `true` if the input is consistent `false` otherwise

```
foreach ⟨∇_l, ψ_l⟩ ∈ Ψ do
    if matching(Γ, X_1σ ≈? X_1ψ_l, ··· , X_nσ ≈? X_nψ_l) = δ then
        if Γ ⊇ ⟨(∇_l δ)⟩_nf then
            return false and stop
        end
    end
end
return true
```

Algorithm 1: Consistency Test

4.2 SOLVING NOMINAL CONSTRAINTS

Finally, we give the formal definition of a solution of a nominal constraint problem and also construct a finite representation for the solution set.

Definition 4.5. Let $\mathcal{P} = \left\langle \Delta \vdash s_1 \approx_\alpha^? t_1, \cdots, s_n \approx_\alpha^? t_n \;||\; \nabla \vdash p_1 \not\approx_\alpha^? q_1, \cdots, p_m \not\approx_\alpha^? q_m \right\rangle$ be a nominal disunification constraint problem. A solution of $\mathcal{P}$ is a pair $\langle \Gamma, \sigma \rangle$ of a consistent context Γ and a substitution σ satisfying the following conditions:

1. $\langle \Gamma, \sigma \rangle$ is a solution of the equational part E of $\mathcal{P}$.
2. $\langle \Gamma, \sigma \rangle$ satisfies the disequations in the disequational part D of $\mathcal{P}$, that is:
   a) $\Gamma \nvdash \nabla\sigma$, or
   b) $\Gamma \nvdash p\sigma \approx_\alpha q\sigma$, for all $p \not\approx_\alpha^? q$ in D.

input : A disunification problem $\mathcal{P} = \langle E \;||\; D \rangle$.
output: A finite set S of pairs with exceptions (can be the empty set).

```
let ⟨Γ, σ⟩ := unify(E) ;
let Ψ := ⋃_{p_i ≉_α^? q_i ∈ D} {⟨∇_i, ψ_i⟩ = unify(∇_i, p_i ≈_α^? q_i)}
if consistent(⟨Γ, σ⟩ − Ψ) then
    return ⟨Γ, σ⟩ − Ψ
else
    return ∅
end
```

Algorithm 2: Disunification Algorithm

Definition 4.6. We call a set S of pairs with exceptions a *complete representation* of the solutions of the constraint problem $\mathcal{P}$ iff S satisfies the following conditions:

1. If $\langle \Gamma, \sigma \rangle - \Psi \leq (\Delta, \lambda)$ for some $\langle \Gamma, \sigma \rangle - \Psi$ in S then $\langle \Delta, \lambda \rangle$ solves $\mathcal{P}$.
2. If $\langle \Delta, \lambda \rangle$ solves $\mathcal{P}$ then it is an instance of some $\langle \Gamma, \sigma \rangle - \Psi$ in S.

3. $\langle \Gamma, \sigma \rangle - \Psi$ is consistent for all $\langle \Gamma, \sigma \rangle - \Psi \in S$.

Similar to nominal unification problems, we are interested in generating a complete finite representation for the set of solutions to a constraint problem $\mathcal{P}$. We use Algorithm 2 to compute such a representation in the form of a pair with exceptions $\langle \Gamma, \sigma \rangle - \Psi$ where $\langle \Gamma, \sigma \rangle$ is a solution for the equations in $\mathcal{P}$ and the family $\Psi = \{\langle \Delta_l, \psi_l \rangle\}$ is formed by taking each pair $\langle \Delta_l, \psi_l \rangle$ as the solution of the associated equations $\Delta \vdash p_l \approx_\alpha^? q_l$, $1 \leq l \leq m$. Termination of Algorithm 2 follows from the termination of `unify`, and correctness (soundness and completeness) follows from the Representation Theorem below.

Theorem 4.1 (Representation Theorem)**.** Let

$$\mathcal{P} = \left\langle \Delta \vdash s_1 \approx_\alpha^? t_1, \cdots, s_n \approx_\alpha^? t_n \;||\; \nabla \vdash p_1 \not\approx_\alpha^? q_1, \cdots, p_m \not\approx_\alpha^? q_m \right\rangle$$

be a nominal constraint problem. Define the family

$$\Psi := \bigcup_{p \not\approx_\alpha^? q \in D} \mathcal{U}\left(\nabla, p \approx_\alpha^? q\right).$$

Then the set $S = \{\langle \Gamma, \sigma \rangle - \Psi \mid \langle \Gamma, \sigma \rangle \in \mathcal{U}(E) \text{ and } \Psi \not\leq \langle \Gamma, \sigma \rangle\}$ is a complete representation of solutions for the constraint problem $\mathcal{P}$.

Proof.

1. Take $\langle \Lambda, \lambda \rangle$ an instance of some $\langle \Gamma, \sigma \rangle - \Psi$ in S. Then $\langle \Gamma, \sigma \rangle \leq \langle \Lambda, \lambda \rangle$ and it is not an instance of Ψ. Since unification problems are closed by instantiation it follows that $\langle \Lambda, \lambda \rangle$ solves the equational part of $\mathcal{P}$. It remains to show that $\langle \Lambda, \lambda \rangle$ solves the disequational part of $\mathcal{P}$. Suppose by contradiction that $\langle \Lambda, \lambda \rangle$ satisfies $\nabla \vdash p_l \approx_\alpha q_l$ for some $\nabla \vdash p_l \not\approx_\alpha q_l$ in D. Therefore, $\langle \Lambda, \lambda \rangle$ is an instance of $\langle \nabla_l, \psi_l \rangle$ (a solution of the associated unification problem $\nabla \vdash p_l \approx_\alpha^? q_l$ in D) and every instance of $\langle \Lambda, \lambda \rangle$ is an instance of $\langle \nabla_l, \psi_l \rangle$ then $\Psi \leq \langle \Lambda, \lambda \rangle$, a contradiction.

2. Suppose $\langle \Lambda, \lambda \rangle$ solves $\mathcal{P}$. Then, $\langle \Lambda, \lambda \rangle$ solves the equational (disequational) part of $\mathcal{P}$. Consider $\langle \Gamma, \sigma \rangle \in \mathcal{U}(E)$ a solution of E, then we conclude that $\langle \Gamma, \sigma \rangle \leq \langle \Lambda, \lambda \rangle$. In addition, $\langle \Lambda, \lambda \rangle$ solves the disequational part of $\mathcal{P}$ as well, that is;

$$\Lambda \nvdash \nabla\lambda \qquad \text{or} \qquad \Lambda \nvdash p\lambda \approx_\alpha q\lambda, \quad \text{for all } p \not\approx_\alpha^? q \in D \tag{4.3}$$

   Assume $\langle \Lambda, \lambda \rangle$ is an instance of Ψ. Then all instances of $\langle \Lambda, \lambda \rangle$ is an instance of some $\langle \nabla_l, \psi_l \rangle$ in Ψ. Hence, there is some $\langle \nabla_l, \psi_l \rangle$ in Ψ such that

$$\langle \nabla_l, \psi_l \rangle \leq \langle \Lambda, \lambda \rangle,$$

   a contradiction with (4.3). Therefore, $\langle \Lambda, \lambda \rangle$ cannot be an instance of Ψ and, we conclude that $\langle \Lambda, \lambda \rangle$ is an instance of $\langle \Gamma, \sigma \rangle - \Psi$, as required.

□

Remark 4.2.

1. Note that any ground instance of a pair with exception representing a solution of a constraint problem $\mathcal{P}$ is also a solution of $\mathcal{P}$. We can restrict solutions to ground instances, but this does not mean that if a problem is solvable in the term-algebra $\mathcal{T}(\Sigma, \mathbb{A}, \mathbb{X})$ it is also solvable in the ground algebra $\mathbb{F}(CORE, \mathcal{D})$, as discussed earlier in Example 3.6.

2. If one wants to solve a disunification problem in the initial ground algebra, by Lemma 4.1, one needs to test if all ground instances of the solutions to the equational part E are an instance of the exceptions ψ. For some nominal theories this is not an easy task.

3. We have a restricted instantiation closure, as it is not transitive. In fact, $\langle \emptyset, [X/Z] \rangle$ is an instance of the pair with exception $\langle \emptyset, [X/Y] \rangle - \langle \emptyset, [X/a] \rangle$. Note that $\langle \emptyset, [X/Z] \rangle \leq \langle \emptyset, [X/a] \rangle$ but the latter is not an instance of $\langle \emptyset, [X/Y] \rangle - \langle \emptyset, [X/a] \rangle$.

Example 4.3. Consider the constraint problem $\mathcal{P}$ below:

$$\mathcal{P} = \left\langle \lambda[a]\, X \approx_\alpha{}^? \lambda[b]\, Y \;||\; X \not\approx_\alpha{}^? Y, X \not\approx_\alpha{}^? a \right\rangle.$$

First apply `unif` to the equational part of the problem obtaining as result:

$$\langle \Gamma, \sigma \rangle = \langle b\#X, [Y/(b\ a) \cdot X] \rangle \tag{4.4}$$

Then solve the associated equations of the disequational part to combine them as a family of pairs with exception:

$$\Psi = \{\langle \emptyset, [X/Y] \rangle, \langle \emptyset, [X/a] \rangle\} \tag{4.5}$$

Finally form the pair with exception $\langle \Gamma, \sigma \rangle - \Psi$ by the combination of (4.4) and (4.5). We can check consistency of $\langle \Gamma, \sigma \rangle - \Psi$ using Algorithm (1).

5

CONCLUSIONS AND FUTURE WORK

In this work, we have developed a method to deal with nominal equations constrained by equality constraints in the form of nominal disequations. The approach adapts Buntine and Bürckert's first-order method to solve disequations taking into account the particularities of nominal syntax and semantics. To the best of our knowledge, this is the first work that deals with disequations in the nominal setting. The main result, Theorem 4.1, establishes the soundness and completeness of the proposed approach.

As future work, we intend to investigate more specific applications of nominal constraint problems; inspired from Buntine and Bürckert's work we could seek some direct extensions to nominal logic programming with negated equations and apply our results to more general unification theories (for instance, AC and $AC1$-nominal unification problems). Also, the more general approach to disunification followed by Comon and Lescane [16] using quantified variables will be investigated.

A

ZFA SET THEORY AND EQUIVARIANCE

We use atoms in this dissertation — we introduce them when we say: 'Fix a countable infinite set $\mathbb{A}$ of atoms' in Chapter 2.

We can represent atoms as natural numbers $0, 1, 2, 3, \ldots$ or as sets $\emptyset, \{\emptyset\}, \ldots$ in principle we might use some properties of atoms specific to their representation, such as $a < b$ or $a \in b$. However, in this dissertation we have not used none of these properties because we consider atoms to be *...atomic.*

Here atomic has an specific meaning: atomic objects does not have *internal* structure, in contrast with the examples given above. Set theorists call these atomic objects *urelemente*. A consequence of this construction is that atoms can be distinguished apart — $a \neq b$ is always `true` and $a = b$ is always `false`. This give us the very important ability to *rename* atoms. This is called *equivariance property*, it is an meta-mathematical tool to reason about nominal structures such as nominal algebra, λ-calculus (represented as in Chapter 1).

Definition A.1. For the language of ZFA set thoery, in addition to the basic language of first-order logic with equality, we assume:

- A binary predicate symbol $\in$ called set membership.
- A constant term-former $\mathbb{A}$ called the set of atoms.

(Sets)	$\forall x.((\exists y.y \in x) \to x \notin \mathbb{A})$
(Extensionality)	$\forall x.(x \notin \mathbb{A} \to x = \{z \mid z \in x\})$
(Comprehension)	$\forall x.\exists y.(y \notin \mathbb{A} \wedge y = \{z \in x \mid \phi\}),$ if y is not free in ϕ
($\in$-induction)	$\forall x.(\forall y.(y \in x \to \phi[y/x]) \to \phi) \to \forall x.\phi$
(Replacement)	$\forall x.\exists z.(z \notin \mathbb{A} \wedge z = \{F(y) \mid y \in x\})$
(Pairset)	$\forall x.\forall y.\exists z.(z = \{x, y\})$
(Union)	$\forall x.\exists z.(z \notin \mathbb{A} \wedge z = \{y \mid \exists y'.(y \in y' \wedge y' \in x)\})$
(Powerset)	$\forall x.\exists z.(z = \{y \mid y \subseteq x\})$
(Infinity)	$\exists x.(\emptyset \in x \wedge \forall y.(y \in x \to y \cup \{y\} \in x))$

Figure A.1: Axioms of ZFA Set Theory

In the axioms of Fig. A.1, ϕ range over all predicates, $\phi[x/y]$ denotes the predicate obtained by capture-avoiding substitution of x by y, and $F(y)$ represent any function which can be expressed in the language of ZFA sets. We also use the following sugar:

$x = \{z \mid z \in x\}$	is sugar for	$\forall y.(\forall z.(z \in x \iff z \in y) \to x = y$
$y = \{z \in x \mid \phi\}$	is sugar for	$\forall z.(z \in y \iff (z \in x \wedge \phi))$
$z = \{F(y) \mid y \in x\}$	is sugar for	$\forall u.(u \in z \iff \exists y.(F(y) = u \wedge y \in$
$z = \{x, y\}$	is sugar for	$\forall u.(u \in z \iff (x = u \vee u = y))$
$z = \{y \mid \exists y'.(y \in y' \wedge y' \in x)\}$	is sugar for	$\forall y.(y \in z \iff \exists y'.(y \in y' \wedge y' \in x$
$z = \{y \mid y \subseteq x\}$	is sugar for	$\forall y.(y \in z \iff \forall y'.(y' \in y \to y' \in$
$\emptyset \in x$	is sugar for	$\exists z.(z \in x \wedge \forall z'.z' \notin z)$
$y \cup \{z\} \in x$	is a sugar for	$\exists u.(u \in x \wedge \forall u'.(u' \in u \iff u \in y$

The syntatic sugar in set theory is very rich; in this description itself we use some other sugars that may be 'obvious' for the reader.

Definition A.2. We define a *permutation action* on ZFA sets by:

$$\pi \cdot a = \pi(a) \qquad \pi \cdot X = \{\pi \cdot x \mid x \in X\} \quad (X \notin \mathbb{A})$$

This definition is by $\in$-induction, a standard method in set theory which relies on a well-foundedness property implied by the axiom ($\in$-Induction).

Recall that ϕ ranges over predicates of ZFA. Write $\phi(x_1, x_2, \ldots, x_n)$ to range over predicates which mentions at most $x_1, x_2, \ldots, x_n$ as free variable symbols.

Theorem A.1 (ZFA Equivariance)**.** If $\phi(x_1, \ldots, x_n)$ is a predicate in ZFA set theory then

$$\phi(x_1, \ldots, x_n) \iff \phi(\pi \cdot x_1, \ldots, \pi \cdot x_n)$$

is always provable, for any permutation π.

Corollary A.1. We can interchange $\phi(x_1, \ldots, x_n)$ and $\phi(\pi \cdot x_1, \ldots, \pi \cdot x_n)$ in a proof and in validity on models.

The proofs and more results on ZFA can be found in [22, Appendix A] and [23].

B

COMPLETE PROOFS

This chapter concentrates complete proofs stated in the body of the dissertation.

B.1 CHAPTER 2

Lemma B.1. $\approx_\alpha$ is reflexive, i.e., $\Delta \vdash s \approx_\alpha s$, for all s.

Proof. The proof is by induction on the syntax of s. The atomic case, $\Delta \vdash a \approx_\alpha a$, is always derivable.

If $s \equiv \pi \cdot X$ then since $\mathtt{ds}(\pi, \pi) = \emptyset$ one trivially derive $\pi \cdot X \approx_\alpha \pi \cdot X$ using (Ds).

If $s \equiv [a]\, t$. Then by induction hypothesis, $\Delta \vdash t \approx_\alpha t$. Therefore, $\Delta \vdash [a]\, t \approx_\alpha [a]\, t$ by (Abs-a).

If $s \equiv f(t_1, \ldots, t_n)$ then by induction hypothesis, $\Delta \vdash t_i \approx_\alpha t_i$ for all $1 \leq i \leq n$. The result follows by (F). □

Lemma B.2. $\approx_\alpha$ is symmetric, i.e., if $\Delta \vdash s \approx_\alpha t$ then $\Delta \vdash t \approx_\alpha s$.

Proof. The proof is by induction on the derivation of $\Delta \vdash s \approx_\alpha t$ by an analysis of the last rule applied.

(a) The base case is trivial.

(b) The last rule is (Ds):

Then $\Delta \vdash \pi \cdot X \approx_\alpha \gamma \cdot X$ is derivable from $\mathtt{ds}(\pi, \gamma)\#X$. So, $\Delta \vdash \gamma \cdot X \approx_\alpha \pi \cdot X$ is also derivable from $\mathtt{ds}(\pi, \gamma)\#X$.

(c) The last rule is (Abs-a):

Hence $\Delta \vdash [a]\, s \approx_\alpha [a]\, t$ is derivable from the premise $\Delta \vdash s \approx_\alpha t$. The induction hypothesis gives $\Delta \vdash t \approx_\alpha s$ and we derive $\Delta \vdash [a]\, t \approx_\alpha [a]\, s$ using (Abs-a).

(d) The last rule is (Abs-b):

So $\Delta \vdash [a]\, s \approx_\alpha [b]\, t$ is derivable from the premises $\Delta \vdash (b\ a) \cdot s \approx_\alpha t$ and $\Delta \vdash b\#s$. The derivation step

$$\frac{\Delta \vdash (a\ b) \cdot t \approx_\alpha s \qquad \Delta \vdash a\#t}{\Delta \vdash [b]\, t \approx_\alpha [a]\, s}\ \text{(Abs-b)}$$

has two proof obligations: $\Delta \vdash (a\ b) \cdot t \approx_\alpha s$ and $\Delta \vdash a\#t$ which we derive in the items below.

(i) By equivariance, Theorem 2.1, follows that $\Delta \vdash s \approx_\alpha (a\ b) \cdot t$ is derivable, and by induction hypothesis it follows that $\Delta \vdash (a\ b) \cdot t \approx_\alpha s$.

(ii) Since $\Delta \vdash b\#s$ and by item (i) above $\Delta \vdash s \approx_\alpha (a\ b)\cdot t$, if follows from Lemma 2.5 that $\Delta \vdash b\#(a\ b)\cdot t$ is derivable. Finally, equivariance gives $\Delta \vdash (a\ b)\cdot b\#t$ which is equivalent to $\Delta \vdash a\#t$.

(e) The last rule is (F):

Then $\Delta \vdash f(s_1,\ldots,s_n) \approx_\alpha f(t_1,\ldots,t_n)$ is derivable from $\Delta \vdash s_i \approx_\alpha t_i$, for $1 \leq i \leq n$. Apply the inductive hypothesis to get $\Delta \vdash t_i \approx_\alpha s_i$, for $1 \leq i \leq n$. Finally,

$$\Delta \vdash f(t_1,\ldots,t_n) \approx_\alpha f(s_1,\ldots,s_n)$$

is derivable using (F).

□

Lemma B.3. $\approx_\alpha$ is transitive, i.e., if $\Delta \vdash s \approx_\alpha t$ and $\Delta \vdash t \approx_\alpha u$ then $\Delta \vdash s \approx_\alpha u$.

Proof. The proof is by a simultaneous induction on the depth of the derivations for $\Delta \vdash s \approx_\alpha t$ and $\Delta \vdash t \approx_\alpha u$. First notice that since the rules for $(\approx_\alpha)$ are syntax-directed the derivations for $\Delta \vdash s \approx_\alpha t$ and $\Delta \vdash t \approx_\alpha u$ must end with rules of the same type. The atomic case is trivial.

1. The last rule is (Ds):

   Then $\Delta \vdash \pi\cdot X \approx_\alpha \gamma\cdot X$ and $\Delta \vdash \gamma\cdot X \approx_\alpha \tau\cdot X$ are derivable from $\Delta \vdash \mathtt{ds}(\pi,\gamma)$ and $\Delta \vdash \mathtt{ds}(\gamma,\tau)$, respectively. We need a derivation

   $$\frac{\Delta \vdash \mathtt{ds}(\pi,\tau)}{\Delta \vdash \pi\cdot X \approx_\alpha \tau\cdot X}\ (\text{Ds})$$

   This follows from the fact that $\mathtt{ds}(\pi,\tau) \subseteq \mathtt{ds}(\pi,\gamma)\cup\mathtt{ds}(\gamma,\tau)$. Indeed, let $a \in \mathtt{ds}(\pi,\tau)$, and suppose by contradiction that $a \notin \mathtt{ds}(\pi,\gamma)\cup\mathtt{ds}(\gamma,\tau)$. Hence, $\pi(a) \neq \tau(a)$ and $\pi(a) = \gamma(a) = \tau(a)$ which is a contradiction.

2. The last rule is for abstractions (Abs-a) and/or (Abs-b).

   (a) Both derivations ends with (Abs-a).

   Therefore, $\Delta \vdash [a]\,s \approx_\alpha [a]\,t$ and $\Delta \vdash [a]\,t \approx_\alpha [a]\,u$ are derivable from $\Delta \vdash s \approx_\alpha t$ and $\Delta \vdash t \approx_\alpha u$, respectively. By induction hypothesis we have that $\Delta \vdash s \approx_\alpha u$ is derivable. Hence, $\Delta \vdash [a]\,s \approx_\alpha [a]\,u$ follows using (Abs-a).

   (b) One of the derivations ends with (Abs-b). Suppose, without loss of generality, $\Delta \vdash [a]\,s \approx_\alpha [a]\,t$ and $\Delta \vdash [a]\,t \approx_\alpha [b]\,u$.

   Then $\Delta \vdash [a]\,s \approx_\alpha [a]\,t$ is derivable from $\Delta \vdash s \approx_\alpha t$, and $\Delta \vdash [a]\,t \approx_\alpha [b]\,u$ is derivable from $\Delta \vdash (b\ a)\cdot t \approx_\alpha u$ and $\Delta \vdash b\#t$. The derivation step

   $$\frac{\Delta \vdash (b\ a)\cdot s \approx_\alpha u \qquad \Delta \vdash b\#s}{\Delta \vdash [a]\,s \approx_\alpha [b]\,u}\ (\text{Abs-b})$$

has two proof obligations: $\Delta \vdash (b\ a) \cdot s \approx_\alpha u$ and $\Delta \vdash b\#s$. We prove them in the two items below.

(i) Equivariance, Theorem 2.1, give us a derivation of $\Delta \vdash t \approx_\alpha (a\ b) \cdot u$ and combining this with the derivation of $\Delta \vdash s \approx_\alpha t$ and the induction hypothesis results in a derivation of $\Delta \vdash s \approx_\alpha (a\ b) \cdot u$. Again, by equivariance we have a derivation of $\Delta \vdash (b\ a) \cdot s \approx_\alpha u$.

(ii) By hypothesis we have a derivation of $\Delta \vdash s \approx_\alpha t$ and $\Delta \vdash b\#t$. Hence by Lemma 2.5 it follows that $\Delta \vdash b\#s$.

(c) Both derivations ends with (Abs-b).

Then $\Delta \vdash [a]\,s \approx_\alpha [b]\,t$ and $\Delta \vdash [b]\,t \approx_\alpha [c]\,u$ are derivable from ($\Delta \vdash (b\ a) \cdot s \approx_\alpha t$ and $b\#s$) and ($\Delta \vdash (c\ b) \cdot t \approx_\alpha u$ and $c\#t$), respectively. Now the derivation

$$\frac{\Delta \vdash (c\ a) \cdot s \approx_\alpha u \qquad \Delta \vdash c\#s}{\Delta \vdash [a]\,s \approx_\alpha [c]\,u}\ \text{(Abs-b)}$$

has the proof obligations: $\Delta \vdash (c\ a) \cdot s \approx_\alpha u$ and $\Delta \vdash c\#s$ which we derive in the items below.

(i) Combining the derivation for $\Delta \vdash (b\ a) \cdot s \approx_\alpha t$ and $\Delta \vdash t \approx_\alpha (b\ c) \cdot u$ and using the inductive hypothesis we get the derivation $\Delta \vdash (b\ a) \cdot s \approx_\alpha (b\ c) \cdot u$ and by equivariance:

$$\Delta \vdash (c\ b) \circ (b\ a) \cdot s \approx_\alpha u$$

Now notice that $\mathtt{ds}((c\ b) \circ (b\ a), (c\ a)) = \{b, c\}$. By item (ii) we have $\Delta \vdash b\#s$ and since $\Delta \vdash b\#s$ by hypothesis it follows, from Lemma 2.6, that

$$\Delta \vdash (c\ b) \circ (b\ a) \cdot s \approx_\alpha (c\ a) \cdot s$$

Induction hypothesis one more time gives the desired result, $\Delta \vdash (c\ a) \cdot s \approx_\alpha u$.

(ii) Since $\Delta \vdash c\#t$ and $\Delta \vdash (b\ a) \cdot s \approx_\alpha t$, by Lemma 2.5, $\Delta \vdash c\#(b\ a) \cdot s$. Equivariance give us the derivation $\Delta \vdash (b\ a) \cdot c\#s$, which is equivalent to $\Delta \vdash c\#s$.

3. If the last rule is (F).

Then $\Delta \vdash f(s_1, \ldots, s_n) \approx_\alpha f(t_1, \ldots, t_n)$ and $\Delta \vdash f(t_1, \ldots, t_n) \approx_\alpha f(u_1, \ldots, u_n)$ are derivable from the premises $\Delta \vdash s_i \approx_\alpha t_i$ and $\Delta \vdash t_i \approx_\alpha u_i$, for $1 \leq i \leq n$, respectively. We derive $\Delta \vdash f(s_1, \ldots, s_n) \approx_\alpha f(u_1, \ldots, u_n)$ from the induction hypothesis and using (F).

□

B.2 CHAPTER 3

Lemma B.4. Let Δ be a consistent context. If $\Delta \vdash a\#t$ and $\Delta \vdash b\#t$ then $\Delta \vdash (a\ b) \cdot t \approx_\alpha t$.

Proof. The proof is by induction on the structure of t:

(a) The base case: $\Delta \vdash a\#c$ and $\Delta \vdash b\#c$ can be easily derived since $\Delta \vdash (a\ b) \cdot c \approx_\alpha c$ is equivalent to $\Delta \vdash c \approx_\alpha c$, which is an instance of $(\approx_\alpha a)$.

(b) In the case for $t \equiv \pi \cdot X$ we have $\Delta \vdash \pi^{-1}(a)\#X$ and $\Delta \vdash \pi^{-1}(b)\#X$, hence $\pi^{-1}(a)\#X$ and $\pi^{-1}(b)\#X$ are in Δ. By meta-level equivariance Δ^π contains the freshness assumptions $a\#X$ and $b\#X$. Thefore, the result follows from the derivation below

$$\frac{\Delta \vdash a\#X \qquad \Delta \vdash b\#X}{\Delta \vdash (a\ b) \cdot X \approx_\alpha X}\ (\text{Ds})$$

(c) If t is an abstraction we consider two cases.

(i) The abstracted atom is either a or b, w.l.g, $t \equiv [a]\,t$. Then consider the following derivation:

$$\dfrac{\dfrac{\dfrac{}{\Delta \vdash t \approx_\alpha t}\ (\text{refl}) \qquad \dfrac{\text{H}}{\Delta \vdash b\#t}}{\Delta \vdash (a\ b) \cdot ((a\ b) \cdot t) \approx_\alpha t} \qquad \dfrac{}{\Delta \vdash a\#(a\ b) \cdot t}\ (\text{eqv})}{\Delta \vdash [b]\,(a\ b) \cdot t \approx_\alpha [a]\,t}\ (\text{Abs-b})$$

(ii) The abstracted atom is different from a and b. Then consider the following derivation:

$$\frac{\dfrac{\text{I.H}}{\Delta \vdash (a\ b) \cdot t \approx_\alpha t}}{\Delta \vdash [c]\,(a\ b) \cdot t \approx_\alpha [c]\,t}\ (\text{Abs-b})$$

(d) If $t \equiv f(t_1, \ldots, t_n)$: by induction hypothesis one has derivations for $\Delta \vdash (a\ b) \cdot t_i \approx_\alpha t_i$, for all $1 \leq i \leq n$. Then $\Delta \vdash (a\ b) \cdot f(t_1, \ldots, t_n) \approx_\alpha f(t_1, \ldots, t_n)$ is derivable using (F).

□

Lemma B.5. Let $\mathcal{A}$ be a Σ-algebra and ς a valuation to the domain A_S. Then $\pi \cdot [\![t]\!]_\varsigma = [\![\pi \cdot t]\!]_\varsigma$.

Proof. The proof is by induction on the structure of of t. The base case follows by equivariance of `atom`.

(a) If $t \equiv \gamma \cdot X$, then;

$$\begin{aligned} \pi \cdot [\![\gamma \cdot X]\!]_\varsigma &= \pi \cdot (\gamma \cdot \varsigma(X)) \\ &= \pi\gamma \cdot \varsigma(X) \\ &= [\![\pi \cdot (\gamma \cdot X)]\!]_\varsigma . \end{aligned}$$

(b) If $t \equiv [a]\,t$, then;

$$\begin{aligned} \pi \cdot [\![[a]\,t]\!]_\varsigma &= \pi \cdot \mathtt{abs}(a, [\![t]\!]_\varsigma) \\ &\overset{I.H}{=} \mathtt{abs}(\pi(a), [\![\pi \cdot t]\!]_\varsigma) \\ &= [\![\pi \cdot [a]\,t]\!]_\varsigma \end{aligned}$$

(c) If $t \equiv f(t_1, \ldots, t_n)$ then;

$$\begin{aligned}\pi \cdot [\![f(t_1, \ldots, t_n)]\!]_\varsigma &= \pi \cdot f^{\mathcal{A}}([\![t_1]\!]_\varsigma, \ldots, [\![t_n]\!]_\varsigma) \\ &= f^{\mathcal{A}}(\pi \cdot [\![t_1]\!]_\varsigma, \ldots, \pi \cdot [\![t_n]\!]_\varsigma) \\ &\overset{I.H}{=} f^{\mathcal{A}}([\![\pi \cdot t_1]\!]_\varsigma, \ldots, [\![\pi \cdot t_n]\!]_\varsigma) \\ &= [\![\pi \cdot f(t_1, \ldots, t_n)]\!]_\varsigma\end{aligned}$$

□

Lemma B.6. Consider $t \in \mathbb{F}(\Sigma, \mathcal{D})$ as above. Suppose that $X\sigma \in \mathbb{F}(\Sigma, \mathcal{D})$ for every $X \in \mathtt{vars}(t)$. Let ς be a valuation to the domain $\mathbb{F}(T, \mathcal{D})$ such that $X\sigma \in \varsigma(X)$ for every $X \in \mathtt{vars}(t)$. Then $[t\sigma]_T = [\![t]\!]_\varsigma$.

Proof. The proof is by induction on the structure of terms.

(a) The base case, $t \equiv a$, follows from the fact that $a\sigma \equiv a$ and $[\![a]\!]_\varsigma = [a]_T$.

(b) If $t \equiv \pi \cdot X$:

$$\begin{aligned}[(\pi \cdot X)\sigma]_T &= \pi \cdot [X\sigma]_T \\ &= \pi \cdot \varsigma(X) \\ &= [\![\pi \cdot X]\!]_\varsigma\end{aligned}$$

(c) If $t \equiv [a]\, t$:

$$\begin{aligned}[([a]\, t)\sigma]_T &= [[a]\, t\sigma]_T \\ &= \mathtt{abs}(a, [[a]\, t\sigma]_T) \\ &\overset{I.H}{=} \mathtt{abs}(a, [\![t]\!]_\varsigma) \\ &= [\![[a]\, t]\!]_\varsigma\end{aligned}$$

(d) If $t \equiv f(t_1, \ldots, t_n)$:

$$\begin{aligned}[f(t_1, \ldots, t_n)\sigma]_T &= [f(t_1\sigma, \ldots, t_n\sigma)]_T \\ &= f^{\mathbb{F}}(T, \mathcal{D})([t_1\sigma]_T, \ldots, [t_n\sigma]_T) \\ &\overset{I.H}{=} f^{\mathbb{F}}(T, \mathcal{D})([\![t_1]\!]_\varsigma, \ldots, [\![t_n]\!]_\varsigma) \\ &= [\![f(t_1, \ldots, t_n)]\!]_\varsigma\end{aligned}$$

□

Theorem B.1. Let T be a nominal theory. The initial algebra $\mathbb{F}(T, \mathcal{D})$ is a model of T.

Proof. Let $\Delta \vdash t = u$ be an axiom of T. Take ς a valuation to the domain $\mathbb{F}(T, \mathcal{D})$ and suppose that $[\![\Delta]\!]_\varsigma$ is valid, that is, $a \#_{\mathtt{sem}} X\varsigma(X)$ for every $a\#X$ in Δ. We must show that $[\![t]\!]_\varsigma = [\![u]\!]_\varsigma$.

Let $\mathcal{X}$ be the set of all variables in $\mathtt{var}(\Delta, t, u)$. By Lemma 3.11, for every $X \in \mathcal{X}$ there is an element $g_X \in \varsigma(X)$ such that $\vdash a \#_{\mathtt{sem}} g_X$ for every $a\#X \in \Delta$. Define σ as the substitution such that $X\sigma \equiv g_X$ if $X \in \mathcal{X}$ and $X\sigma \equiv X$ if $X \notin \mathcal{X}$.

By construction $\vdash a\#X\sigma$ for every $a\#X$ in Δ, hence $\vdash_T t\sigma = u\sigma$ by an application of the rule $(ax_{\Delta\vdash t=u})$. It is clear that this is a ground derivation (it does not mention any variable or instance of the rule (fr)), so $[t\sigma]_T = [u\sigma]_T$. Therefore, $[t\sigma]_T = [\![t]\!]_\varsigma$ and $[u\sigma]_T = [\![u]\!]_\varsigma$ by Lemma B.6 and the result follows. □

BIBLIOGRAPHY

[1] Mauricio Ayala-Rincón, Washington de Carvalho Segundo, Maribel Fernández, and Daniele Nantes-Sobrinho. "On Solving Nominal Fixpoint Equations." In: *Proc. 11th International Symposium on Frontiers of Combining Systems, FroCoS*. Vol. 10483. Lecture Notes in Computer Science. Springer, 2017, pp. 209–226. DOI: 10.1007/978-3-319-66167-4_12. URL: https://doi.org/10.1007/978-3-319-66167-4_12.

[2] Mauricio Ayala-Rincón, Washington de Carvalho Segundo, Maribel Fernández, and Daniele Nantes-Sobrinho. "Nominal C-Unification." In: *27th International Symposium on Logic-Based Program Synthesis and Transformation, LOPSTR 2017, Revised Selected Papers*. Vol. 10855. Lecture Notes in Computer Science. Springer, 2018, pp. 235–251. DOI: 10.1007/978-3-319-94460-9_14. URL: https://doi.org/10.1007/978-3-319-94460-9_14.

[3] Mauricio Ayala-Rincón, Maribel Fernández, and Daniele Nantes-Sobrinho. "Nominal Narrowing." In: *1st International Conference on Formal Structures for Computation and Deduction, FSCD 2016, June 22-26, 2016, Porto, Portugal*. Ed. by Delia Kesner and Brigitte Pientka. Vol. 52. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016, 11:1–11:17. ISBN: 978-3-95977-010-1. DOI: 10.4230/LIPIcs.FSCD.2016.11. URL: https://doi.org/10.4230/LIPIcs.FSCD.2016.11.

[4] Mauricio Ayala-Rincón, Maribel Fernández, and Daniele Nantes-Sobrinho. "Fixed-Point Constraints for Nominal Equational Unification." In: *3rd International Conference on Formal Structures for Computation and Deduction, FSCD 2018, July 9-12, 2018, Oxford, UK*. Ed. by Hélène Kirchner. Vol. 108. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2018, 7:1–7:16. ISBN: 978-3-95977-077-4. DOI: 10.4230/LIPIcs.FSCD.2018.7. URL: https://doi.org/10.4230/LIPIcs.FSCD.2018.7.

[5] Mauricio Ayala-Rincón, Maribel Fernández, Daniele Nantes-Sobrinho, and Deivid Vale. "On Solving Nominal Disunification Constraints." In: *Proc. 14th Logical and Semantic Frameworks with Applications - LSFA 2019 To appear.* Eletronic Notes in Theoretical Computer Science. 2019.

[6] Franz Baader and Klaus U. Schulz. "Combination Techniques and Decision Problems for Disunification." In: *Theor. Comput. Sci.* 142.2 (1995), pp. 229–255. DOI: 10.1016/0304-3975(94)00277-0. URL: https://doi.org/10.1016/0304-3975(94)00277-0.

[7] Franz Baader and Jörg H. Siekmann. "Handbook of Logic in Artificial Intelligence and Logic Programming." In: ed. by Dov M. Gabbay, C. J. Hogger, and J. A. Robinson. New York, NY, USA: Oxford University Press, Inc., 1994. Chap. Unification

Theory, pp. 41–125. ISBN: 0-19-853746-8. URL: https://www.cs.bu.edu/~snyder/publications/UnifChapter.pdf.

[8] H. P. Barendregt. *The lambda calculus : its syntax and semantics / H.P. Barendregt*. English. North-Holland Pub. Co. ; sole distributors for the U.S.A. and Canada Elsevier North-Holland Amsterdam ; New York : New York, 1981, xiv, 615 p. : ISBN: 0444854908.

[9] Wray L. Buntine and Hans-Jürgen Bürckert. "On Solving Equations and Disequations." In: *J. ACM* 41.4 (July 1994), pp. 591–629. ISSN: 0004-5411. DOI: 10.1145/179812.179813. URL: http://doi.acm.org/10.1145/179812.179813.

[10] Christophe Calvès and Maribel Fernández. "A polynomial nominal unification algorithm." In: *Theoretical Computer Science* 403.2 (2008), pp. 285 –306. ISSN: 0304-3975. DOI: https://doi.org/10.1016/j.tcs.2008.05.012. URL: http://www.sciencedirect.com/science/article/pii/S0304397508003927.

[11] Christophe Calvès and Maribel Fernández. "Matching and alpha-equivalence check for nominal terms." In: *Journal of Computer and System Sciences* 76.5 (2010). Workshop on Logic, Language, Information and Computation, pp. 283 –301. ISSN: 0022-0000. DOI: https://doi.org/10.1016/j.jcss.2009.10.003. URL: http://www.sciencedirect.com/science/article/pii/S0022000009000920.

[12] James Cheney. "Relating Nominal and Higher-Order Pattern Unification (draft)." In: *Proceedings of UNIF 2005*. 2005, pp. 104–119.

[13] James Cheney. "Equivariant Unification." In: *Journal of Automated Reasoning* 45.3 (2010), pp. 267–300. ISSN: 1573-0670. DOI: 10.1007/s10817-009-9164-3. URL: https://doi.org/10.1007/s10817-009-9164-3.

[14] James Cheney and Christian Urban. "αProlog: A Logic Programming Language with Names, Binding and α-Equivalence." In: *Logic Programming*. Ed. by Bart Demoen and Vladimir Lifschitz. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 269–283. ISBN: 978-3-540-27775-0.

[15] Hubert Comon. "Disunification: a Survey." In: *Computational Logic: Essays in Honor of Alan Robinson*. Ed. by Jean-Louis Lassez and Gordon Plotkin. MIT Press, 1991, pp. 322–359.

[16] Hubert Comon and Pierre Lescanne. "Equational problems and disunification." In: *Journal of Symbolic Computation* 7.3 (1989). Unification: Part 1, pp. 371 –425. ISSN: 0747-7171. DOI: https://doi.org/10.1016/S0747-7171(89)80017-3. URL: http://www.sciencedirect.com/science/article/pii/S0747717189800173.

[17] Maribel Fernández. "Narrowing Based Procedures for Equational Disunification." In: *Appl. Algebra Eng. Commun. Comput.* 3 (1992), pp. 1–26. DOI: 10.1007/BF01189020. URL: https://doi.org/10.1007/BF01189020.

[18] Maribel Fernández and Murdoch J. Gabbay. "Nominal rewriting." In: *Information and Computation* 205.6 (2007), pp. 917 –965. ISSN: 0890-5401. DOI: https://doi.org/10.1016/j.ic.2006.12.002. URL: http://www.sciencedirect.com/science/article/pii/S0890540106001635.

[19] Murdoch J. Gabbay. "Nominal Algebra and the HSP Theorem." In: *Journal of Logic and Computation* 19.2 (Oct. 2008), pp. 341–367. ISSN: 0955-792X. DOI: 10.1093/logcom/exn055. eprint: http://oup.prod.sis.lan/logcom/article-pdf/19/2/341/3510563/exn055.pdf. URL: https://doi.org/10.1093/logcom/exn055.

[20] Murdoch J. Gabbay and Aad Mathijssen. "Capture-Avoiding Substitution as a Nominal Algebra." In: *Formal Aspects of Computing* 20.4-5 (2008), pp. 451–479. DOI: 10.1007/11921240_14.

[21] Murdoch J. Gabbay and Aad Mathijssen. "The lambda-calculus is nominal algebraic." In: *Reasoning in simple type theory: Festschrift in Honour of Peter B. Andrews on his 70th Birthday*. Ed. by Christoph Benzmüller, Chad Brown, Jörg Siekmann, and Rick Statman. Studies in Logic and the Foundations of Mathematics. IFCoLog, 2008.

[22] Murdoch J. Gabbay and Aad Mathijssen. "Nominal universal algebra: equational logic with names and binding." In: *Journal of Logic and Computation* 19.6 (2009), pp. 1455–1508. DOI: 10.1093/logcom/exp033.

[23] Murdoch J. Gabbay and Andrew M. Pitts. "A New Approach to Abstract Syntax with Variable Binding." In: *Formal Aspects of Computing* 13.3 (2002), pp. 341–363. ISSN: 1433-299X. DOI: 10.1007/s001650200016. URL: https://doi.org/10.1007/s001650200016.

[24] Brian Huffman and Christian Urban. "A New Foundation for Nominal Isabelle." In: *Interactive Theorem Proving*. Ed. by Matt Kaufmann and Lawrence C. Paulson. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 35–50. ISBN: 978-3-642-14052-5.

[25] Denis Lugiez. "Higher Order Disunification: Some decidable cases." In: *First International Conference on Constraints in Computational Logics, CCL*. Vol. 845. Lecture Notes in Computer Science. Springer, 1994, pp. 121–135. DOI: 10.1007/BFb0016848. URL: https://doi.org/10.1007/BFb0016848.

[26] Andrew M. Pitts. *Nominal Sets: Names and Symmetry in Computer Science*. New York, NY, USA: Cambridge University Press, 2013. ISBN: 1107017785, 9781107017788.

[27] Zhenyu Qian. "" In: *TAPSOFT'93: Theory and Practice of Software Development*. Ed. by M. C. Gaudel and J. P. Jouannaud. Berlin, Heidelberg: Springer Berlin Heidelberg, 1993, pp. 391–405. ISBN: 978-3-540-47598-9.

[28] Veena Ravishankar, Kimberly A. Gero, and Paliath Narendran. "Asymmetric Unification and Disunification." In: *CoRR* abs/1706.05066 (2017). arXiv: 1706.05066. URL: http://arxiv.org/abs/1706.05066.

[29] Christian Urban, Andrew M. Pitts, and Murdoch J. Gabbay. "Nominal unification." In: *Theoretical Computer Science* 323.1 (2004), pp. 473 –497. ISSN: 0304-3975. DOI: https://doi.org/10.1016/j.tcs.2004.06.016. URL: http://www.sciencedirect.com/science/article/pii/S0304397504004013.

COLOPHON

This version of the thesis (v1.2.1) is a slight update on the version submitted. The change is basically in typographic style and some small typo fixing here and there. Thank you very much for your feedback and contribution.