An Investigation into Nominal Equational Problems (Work in progress)

Mauricio Ayala-Rincón[1], Maribel Fernández[2], Daniele Nantes-Sobrinho[3], and Deivid Vale[4]*

[1] Departamentos de Matemática e Ciência da Computação, Universidade de Brasília, Brazil
ayala@unb.br
[2] Department of Informatics, King's College London, London, UK
maribel.fernandez@kcl.ac.uk
[3] Departamento de Matemática, Universidade de Brasília, Brazil
dnantes@mat.unb.br
[4] Department of Software Science, Radboud University, The Netherlands
deividvale@cs.ru.nl

Abstract

We consider *nominal equational problems* of the form $\exists W \forall Y : P$, where P consists of conjunctions and disjunctions of equations of the form $s \approx_\alpha t$ (read: "s is α-equivalent to t"), freshness constraints of the form $a\#t$ (read: "a is fresh for t") and their negations: $s \not\approx_\alpha t$ and $a \#\!\!\!/ t$, where a is an atom and s, t are nominal terms. When dealing with general nominal equational problems we face the challenge of properly defining their semantics to take into account the interaction between negative freshness constraints and the existential and universal quantifiers. Here we propose a discussion regarding two different approaches: (i) adopting the usual freshness and equational contraints; (ii) the use of the "new" quantifier (И) and fixed point equations instead of freshness constraints; in both cases being careful to obtain the correct meaning.

1 Introduction

Disunification problems have been extended to the nominal setting [2], using a restricted form of constraints called *nominal (disunification) constraints*: equations (judgments $\Delta \vdash s \approx_\alpha{}^? t$) enriched with disequations, i.e., negated equations of the form $s \not\approx_\alpha{}^? t$. In that setting, a nominal constraint problem $\mathcal{P}$ is equivalent to the existentially closed formula:

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

This problem is solved in the nominal term-algebra $\mathcal{T}(\Sigma, \mathbb{A}, \mathbb{X})$ by constructing suitable representation to the witnesses for the variables in $\mathcal{P}$ [2].

Comon and Lescanne [6] investigated the so-called *equational problem*, in their words: "an equational problem is any first-order formula whose only predicate symbol is =", that is, it has the form $\exists w_1, \ldots, w_n \forall y_1, \ldots, y_m : P$ where P is a *system*, i.e., an equation $s = t$, or a disequation $s \neq t$, or a disjunction of systems $\bigvee P_i$, or a conjunction of systems $\bigwedge P_i$, or a failure $\bot$, or success $\top$. The motivation to study such problems was the applicability in pattern-matching for functional languages, sufficient completeness for term rewriting systems, dealing with negation in logic programming languages, etc.

*The author is supported by the NWO TOP project "Implicit Complexity through Higher Order Rewriting" (ICHOR), NWO 612.001.803/7571.

With the development of nominal techniques, including nominal logic [10], nominal unification and rewriting [7], nominal logic programming [4], and nominal (universal) algebra [8], it is natural to extend equational problems into the "nominal world" and consider *nominal equational problems*. Based on Comon and Lescanne's work, the expected form of a nominal extension to the first-order equational problem would be

$$\mathcal{P} ::= \exists W_1 \dots W_n \forall Y_1 \dots Y_m : P$$

with P being a *nominal system*, i.e., a formula consisting of conjunctions and disjunctions of freshness, equality constraints, and their negations.

In this paper, we discuss alternative formulations of nominal equational problems taking into account the kind of constraints used and the model on which they are interpreted. We also discuss a preliminary rule based strategy to solve such problems. This work is a first step towards the generalisation of nominal disunification constraint problems (introduced in [2]) which consist of equations and disequations without universally-quantified variables.

2 Background

We assume the reader is familiar with nominal techniques and recall some concepts and notations that shall be used in the paper; for more details the reader is referred to [7, 11].

Fix countable infinite, pairwise disjoint, sets of *atoms* $\mathbb{A} = \{a, b, c, \dots\}$ and *variables* $\mathbb{X} = \{X, Y, Z, \dots\}$. Atoms follow the permutative convention, i.e., names a, b, and c run permutatively over $\mathbb{A}$, therefore they represent different names. As usual, we form nominal terms with a finite set Σ of *term-formers* — disjoint from $\mathbb{A}$ and $\mathbb{X}$ — such that for each $f \in \Sigma$, a unique non-negative integer n (the arity of f, written as $f : n$) is assigned.

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 and $\pi \circ \pi'$ for the composition of π and π'. The difference set of π and γ is defined by $\mathtt{ds}(\pi, \gamma) = \{a \in \mathbb{A} \mid \pi(a) \neq \gamma(a)\}$.

Nominal terms are given by the following grammar: $s, t := a \mid \pi \cdot X \mid [a]t \mid f(t_1, \dots, t_n)$ where a is an *atom*, $\pi \cdot X$ is a moderated variable, $[a]\, t$ is the *abstraction* of a in the term t, and $f(t_1, \dots, t_n)$ is a *function application* with $f \in \Sigma$ and $f : n$. We abbreviate a ordered sequence $t_1, \dots, t_n$ of terms by $\tilde{t}$.

Example 1. *Let $\Sigma_\lambda := \{\mathtt{lam} : 1, \mathtt{app} : 2\}$ be a signature for the λ-calculus. Using atoms to represent λ-calculus variables, λ-expressions are generated by the grammar: $e := a \mid \mathtt{lam}([a]\, e) \mid \mathtt{app}(e, e)$. As usual, we write $\mathtt{app}(s, t)$ as $s\, t$ and $\mathtt{lam}([a]\, s)$ as $\lambda\, [a]\, s$. The following are examples of nominal terms: $(\lambda\, [a]\, a)\, X$ and $(\lambda\, [a]\, (\lambda\, [b]\, b\, a)\;\, c)\, d$.*

The *action of a permutation* π on a term t is inductively defined by: $\pi \cdot a = \pi(a)$, $\pi \cdot (\pi' \cdot X) = (\pi \circ \pi') \cdot X$, $\pi \cdot ([a]t) = [\pi(a)](\pi \cdot t)$, and $\pi \cdot f(t_1, \dots, t_n) = f(\pi \cdot t_1, \dots, \pi \cdot t_n)$. *Substitutions*, ranging over $\sigma, \gamma, \tau \dots$, are maps (with finite domain) from variables to terms. The *action of a substitution* σ on a term t, denoted $t\sigma$, is inductively defined by: $a\sigma = a, (\pi \cdot X)\sigma = \pi \cdot (X\sigma)$, $([a]t)\sigma = [a](t\sigma)$ and $f(t_1, \dots, t_n)\sigma = f(t_1\sigma, \dots, t_n\sigma)$. Note that $t(\sigma\gamma) = (t\sigma)\gamma$.

Equality and Freshness Constraints A *nominal equation* (disequation) is the symbol $\top$ ($\bot$) or an expression of the form $s \approx_\alpha t$ ($s \not\approx_\alpha t$) where s and t are nominal terms. A *trivial equation* is either of the form $s \approx_\alpha s$ or $\top$. Similarly, a *trivial disequation* is either $s \not\approx_\alpha s$ or $\bot$.

A finite set of *primitive freshness constraints* of the form $a\#X$ is called a *freshness context*, we use Δ, ∇, and Γ to denote them. Equality and freshness constraints are defined by the derivation rules in Figure 1 below.

$$\frac{}{\nabla \vdash a \approx_\alpha a}\ (\text{ax}) \qquad \frac{\nabla \vdash t_1 \approx_\alpha t'_1 \quad \cdots \quad \nabla \vdash t_n \approx_\alpha t'_n}{\nabla \vdash f(t_1, \ldots t_n) \approx_\alpha f(t'_1, \ldots, t'_n)}\ (\text{f}) \qquad \frac{\nabla \vdash t \approx_\alpha t'}{\nabla \vdash [a]t \approx_\alpha [a]t'}\ (\text{abs-a})$$

$$\frac{\nabla \vdash t \approx_\alpha (a\, a') \cdot t' \qquad \nabla \vdash a \# t'}{\nabla \vdash [a]t \approx_\alpha [a']t'}\ (\text{abs-b}) \qquad \frac{a\#X \in \nabla \text{ for all } a \text{ s.t. } \pi \cdot a \neq \pi' \cdot a}{\nabla \vdash \pi \cdot X \approx_\alpha \pi' \cdot X}\ (\text{var})$$

$$\frac{}{\nabla \vdash a\#b}\ (\#\text{-ax}) \qquad \frac{(\pi^{-1} \cdot a\#X) \in \nabla}{\nabla \vdash a\#\pi \cdot X}\ (\#\text{-var}) \qquad \frac{}{\nabla \vdash a\#[a]t}\ (\#\text{-abs-a})$$

$$\frac{\nabla \vdash a\#t}{\nabla \vdash a\#\,[b]\,t}\ (\#\text{-abs-b}) \qquad \frac{\nabla \vdash a\#t_1 \quad \cdots \quad \nabla \vdash a\#t_n}{\nabla \vdash a\#f(t_1, \ldots t_n)}\ (\#\text{-f})$$

Figure 1: Equality and Freshness Rules

3 (NEP) Nominal Equational Problems

Definition 1. *A* nominal system P *is a formula defined by the following grammar*

$$P, P' ::= \top \mid \bot \mid s \approx_\alpha t \mid s \not\approx_\alpha t \mid a\#t \mid a\not\#t \mid P \wedge P' \mid P \vee P'.$$

Although not usual, the negation of freshness — denoted as $a\not\#X$ — means that *a is not fresh for X*, that is, there exists an instance $t = X\sigma$ of X with at least one free occurrence of a.

Definition 2 (NEP-First Version)**.** *A* NEP *is a formula of the form*

$$\mathcal{P} ::= \exists W_1 \ldots W_n \forall Y_1 \ldots Y_m : P$$

where P is a nominal system and $\overline{W} = \{W_1, \ldots, W_n\}, \overline{Y} = \{Y_1, \ldots, Y_m\}$, are sets of mutually distinct variables called respectivley auxiliary unknowns *and* parameters*. $\mathit{fv}(\mathcal{P})$ denotes the set of free variables occurring in $\mathcal{P}$ also called* principal unknowns.

Example 2 (Nominal Disunification Constraints)**.** *Nominal disunification constraints [2] have the form $\mathcal{P} := \exists\overline{X}\langle E \mid\mid D\rangle$, where E is a finite set of nominal equations in context, i.e., $E = \bigcup_{0\leq i\leq n}\{\Delta_i \vdash s_i \approx_\alpha t_i\}$ and D is a finite set of nominal disequations in context, $D = \bigcup_{0\leq j\leq m}\{\nabla_j \vdash u_j \not\approx_\alpha v_j\}$. This problem is a particular case of* NEP*: if one takes the judgment $\Delta \vdash s \approx_\alpha t$ as $\Delta \Rightarrow s \approx_\alpha t$, or yet as $\neg\Delta \vee s \approx_\alpha t$[1], we obtain the following formula:*

$$\mathcal{P} := \exists\overline{X}(\bigwedge_{i=0}^{n}(\neg[\Delta_i] \vee s_i \approx_\alpha t_i)) \wedge (\bigwedge_{j=0}^{m}(\neg[\nabla_j] \vee u_j \not\approx_\alpha v_j)), \tag{1}$$

where $[\Delta_i], [\nabla_j]$ are conjunctions of freshness constraints contained in Δ_i, ∇_j, respectively.

3.1 Solutions of Equational Problems

Let $\mathcal{P} = \exists\overline{W}\forall\overline{Y} : P$ be a NEP. Let $\mathcal{A}$ be an algebra that provides an interpretation for the symbols in the signature. An $\mathcal{A}$-*solution* for $\mathcal{P}$ is a pair $\langle\Gamma, \sigma\rangle$, consisting of a freshness context Γ and a substitution σ, such that $\langle\Gamma, \sigma\rangle$ $\mathcal{A}$-*validates* the system P (as defined below). We will assume that $\mathcal{A}$ is the nominal algebra of terms $\mathcal{T}(\Sigma, \mathbb{A}, \mathbb{X})$, but one could use the ground algebra $\mathcal{T}(\Sigma, \mathbb{A}, \emptyset)$ or a quotient algebra, say $\mathcal{T}(\Sigma, \mathbb{A}, \mathbb{X})/=_E$ for a given equational theory E.

[1]Similarly, for disequations

Definition 3. *Let C range over equality and freshness constraints and D range over negative constraints (disequations and negated freshness). We denote by $C\sigma$ (resp. $D\sigma$) the constraint obtained by instantiating the terms in C (resp. D) with the substitution σ and by $\neg D$ the positive constraint obtained by negating D.*

A pair $\langle\Gamma,\sigma\rangle$ $\mathcal{A}$-validates *a system P iff*

1. *$P = \top$; or*
2. *$P = C$; $\Gamma \vdash C\sigma$ holds in $\mathcal{A}$; or*
3. *$P = D$; $\Gamma \not\vdash \neg D\sigma$ in $\mathcal{A}$; or*
4. *$P = P_1 \wedge \ldots \wedge P_n$ and $\langle\Gamma,\sigma\rangle$ $\mathcal{A}$-validates each P_i, $1 \leq i \leq n$; or*
5. *$P = P_1 \vee \ldots \vee P_m$ and $\langle\Gamma,\sigma\rangle$ $\mathcal{A}$-validates at least one P_i, $1 \leq i \leq m$.*

The definition of solution relies on a pair $\langle\Gamma,\gamma\rangle$ being *away from a set of variables*.

Definition 4. *A pair $\langle\Gamma,\gamma\rangle$ of a freshness context and a substitution is* away from a set of variables $\mathbb{V} \subset \mathbb{X}$ *iff Γ does not contain any $a\#X$ with $X \in \mathbb{V}$ and γ is away from $\mathbb{V}$, i.e., no variable from $\mathbb{V}$ occurs in $\langle\Gamma,\gamma\rangle$.*

Definition 5. *A pair $\langle\Gamma,\gamma\rangle$ is an* $\mathcal{A}$-solution *of the* NEP $\mathcal{P} = \exists\overline{W}\forall\overline{Y} : P$ *if, and only if, the following conditions hold:*

1. *$\langle\Gamma,\gamma\rangle$ is away from $\overline{W} \cup \overline{Y}$ and $\mathtt{dom}(\gamma) = \overline{X} = fv(\mathcal{P})$;*
2. *there is a pair $\langle\Delta,\delta\rangle$ away from $\overline{Y} \cup \overline{X}$ ($\mathtt{dom}(\delta) = \overline{W}$) such that for all pairs $\langle\Lambda,\lambda\rangle$ away from $\overline{W} \cup \overline{X}$ ($\mathtt{dom}(\lambda) = \overline{Y}$), $\langle\Gamma\Delta\Lambda,\gamma\delta\lambda\rangle$ $\mathcal{A}$-validates P.*

3.2 Nominal Equational Solved Forms

The future goal is to develop a procedure to solve NEP based on applications of *simplification rules*, as proposed in [6], that transform problems into simpler ones preserving the set of solutions. Successive applications of such rules lead to a *solved form* from which we know how to extract a solution from. We consider three first-order solved forms: *parameterless*, *unification*, and *definition with constraints*. Below we extend those notions to the nominal setting.

Definition 6 (Solved Forms)**.**

1. *A* NEP $\mathcal{P}$ *is in* unification solved form *if it is equivalent to a nominal unification problem of the form $\langle\Gamma, X_1 \approx_\alpha t_1 \wedge \ldots \wedge X_n \approx_\alpha t_n\rangle$ where all the unknowns $X_1,\ldots,X_n$ are distinct and do not occur in the t_i's and Γ is a freshness context;*
2. *A* NEP $\mathcal{P}$ *is in* parameterless solved form *if it contains no universal quantifiers.*
3. *A* NEP *is a* definition with constraints *if it is either $\top,\bot$, or a problem of the form $\mathcal{P} := \exists\overline{X}(\bigwedge_{i=0}^{n}(\neg[\Delta_i] \vee X_i \approx_\alpha t_i)) \wedge (\bigwedge_{j=0}^{m}(\neg[\nabla_j] \vee X'_j \not\approx_\alpha v_j))$, where variables $X_1,\ldots,X_n$ occur only once in the equational part (left conjunction). Variables X'_j is different from v_j, for $1, \leq j \leq m$. $[\Delta_i]$ and $[\nabla_j]$ are defined as in Example 2.*

It is essential to remark that as in [6] the *definition with constraints* solved form is equivalent to the *disunification problem* introduced in [3], and its extension to the nominal setting is the disunification constraints problem [2] described in Example 2. We discuss rules in the Appendix.

3.3 Discussion: a strict equational approach to (NEP)

It is natural to try to define NEP as nominal formulas whose only predicate symbol is $\approx_\alpha$. For that, we can explore existing results relating freshness and equality constraints. Initially, the freshness predicate was defined using a quantified fixed point equation, in [11]: $a\#t$ iff $\text{И}a'.(a\ a')\cdot t \approx t$.

In another work, a freshness constraint was shown to have a tight relation with a specific equation between abstracted atoms:

Lemma 1 (Lemma 3.1 in [9]). *$P \cup \{a\#^? t\}$ and $P \cup \{[a][b]t \approx^? [b][b]t\}$ have the same solutions.*

The above approaches motivate the following definition for NEP:

Definition 7 (NEP-Second Version). *A NEP is a formula of the form*

$$\mathcal{P} ::= \exists\overline{W}\forall\overline{Y}\text{И}\overline{a} : P$$

where P is a system generated by the grammar $P, P' ::= \top \mid \bot \mid s \approx_\alpha t \mid s \not\approx_\alpha t \mid P \wedge P' \mid P \vee P'$, and $\overline{W} = \{W_1, \ldots, W_n\}, \overline{Y} = \{Y_1, \ldots, Y_m\}$, and $\mathit{fv}(\mathcal{P})$, as in Definition 2, are the auxiliary unknowns*,* parameters *and* principal unknowns*.*

As shown in [1], solving equations via freshness constraints is equivalent to following the approach via fixed point equations when the equational theory is empty. However, when dealing with equational theories that include commutativity, it seems to be more convenient to use a purely equational approach. Therefore, we conjecture that this second approach would be more convenient when dealing quotient algebras, such as $\mathcal{T}(\Sigma, \mathbb{A}, \mathbb{X})/=_C$, or $\mathcal{T}(\Sigma, \mathbb{A}, \mathbb{X})/=_{AC}$, among others.

4 Conclusion

We have considered two approaches to define NEPs as a straightforward nominal version of the problem introduced in [6, 5], using (negated) freshness constraints in addition to $\approx_\alpha$, or using purely $\approx_\alpha$ as predicate but with the "new" quantifier И. As future work we plan to investigate which approach is more convenient when defining the rules for simplifying the NEPs in order to obtain a correct procedure to solve such problems, besides we also intend to investigate NEPs modulo equational theories.

References

[1] 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*, pages 7:1–7:16, 2018. doi:10.4230/LIPIcs.FSCD.2018.7.

[2] Mauricio Ayala-Rincón, Maribel Fernández, Daniele Nantes-Sobrinho, and Deivid Vale. On Solving Nominal Disunification Constraints. *ENTCS*, 348:3 – 22, 2020. Proc. 14th Int. Workshop on Logical and Semantic Frameworks, with Applications LSFA 2019. doi:10.1016/j.entcs.2020.02.002.

[3] Wray L. Buntine and Hans-Jürgen Bürckert. On solving equations and disequations. *J. ACM*, 41(4):591–629, July 1994. URL: http://doi.acm.org/10.1145/179812.179813, doi:10.1145/179812.179813.

[4] James Cheney and Christian Urban. *alpha*-Prolog: A Logic Programming Language with Names, Binding and α-Equivalence. In *Proc. 20th International Conference on Logic Programming ICLP*, volume 3132 of *LNCS*, pages 269–283. Springer, 2004. doi:10.1007/978-3-540-27775-0_19.

[5] Hubert Comon. Disunification: a Survey. In Jean-Louis Lassez and Gordon Plotkin, editors, *Computational Logic: Essays in Honor of Alan Robinson*, pages 322–359. MIT Press, 1991.

[6] Hubert Comon and Pierre Lescanne. Equational problems and disunification. *J. Symb. Comput.*, 7(3/4):371–425, 1989. doi:10.1016/S0747-7171(89)80017-3.

[7] Maribel Fernández and Murdoch J. Gabbay. Nominal rewriting. *Information and Computation*, 205(6):917 – 965, 2007. doi:10.1016/j.ic.2006.12.002.

[8] Murdoch J. Gabbay and Aad Mathijssen. Nominal (Universal) algebra: equational logic with names and binding. *J. of Logic and Computation*, 19(6):1455–1508, 2009. doi:10.1093/logcom/exp033.

[9] Jordi Levy and Mateu Villaret. An efficient nominal unification algorithm. In *Proceedings of the 21st International Conference on Rewriting Techniques and Applications, RTA*, volume 6 of *LIPIcs*, pages 209–226, 2010. doi:10.4230/LIPIcs.RTA.2010.209.

[10] Andrew M. Pitts. Nominal logic, a first order theory of names and binding. *Information and Computation*, 186(2):165 – 193, 2003. doi:10.1016/S0890-5401(03)00138-X.

[11] Andrew M. Pitts. *Nominal Sets: Names and Symmetry in Computer Science*. Cambridge University Press, New York, NY, USA, 2013. doi:10.1017/CBO9781139084673.

A Rule based procedure

In this section we present a preliminary set of simplification rules which will be used in the algorithm for solving NEP. They are still under investigation and proofs of correctness and termination are ongoing work.

Intuitively, a set of transformation rules transforms a problem $\mathcal{P}$ into a problem $\mathcal{P}'$ (denoted as $\mathcal{P} \implies \mathcal{P}'$), which are simpler in some sense. Transformation rules may have conditions (rule controls) in order to be applied. The goal is to reach one of the normal forms defined above. Different strategies can possible lead to different normal forms. Strategies can also vary according to the model where the problem is being solved.

The primary control gives priority for application of rules: we split rules into sets $\mathcal{R}_i$ using the index i as a priority stack, i.e., a rule $R \in \mathcal{R}_i$ can only be applied if no rules from $\mathcal{R}_j$, with $j < i$ can be applied. A *procedure* on a NEP $\mathcal{P}$ is a strategy $R_0, R_1, \ldots, R_k$ for application of rules such that $\mathcal{P} = \mathcal{P}_0 \implies_{R_0} \mathcal{P}_1 \implies_{R_2} \ldots \implies_{R_k} \mathcal{P}_{k+1}$, where $R_i \in \mathcal{R}_j$, for $0 \leq i \leq k$ and some $0 \leq j \leq 6$, satisfying the primary control give above. A problem $\mathcal{P}$ is rewritten in a *pattern-matching* fashion, i.e, rules give the pattern occurring in the problem. Before the application of each rule $\mathcal{P}$ is reduced to its conjunctive normal form.

$\mathcal{R}_0$: Trivial Rules			
(T_1) $t \approx_\alpha t \implies \top$	(T_2) $t \not\approx_\alpha t \implies \bot$	(T_3) $a \approx_\alpha b \implies \bot$	(T_4) $a\#b \implies \top$
(T_5) $a\#a \implies \bot$	(T_6) $a\not\#a \implies \top$	(T_7) $a\not\#b \implies \bot$	
$\mathcal{R}_1$: Clash and Occurrence Check Rules			
(CL_1) $f(\tilde{t}) \approx_\alpha g(\tilde{u}) \implies \bot$	$(CL_2) f(\tilde{t}) \not\approx_\alpha g(\tilde{u}) \implies \top$	where $f \neq g$	
(CL_3) $s \not\approx_\alpha t \implies \top$	(CL_4) $s \approx_\alpha t \implies \bot$	$s\|_\epsilon \neq t\|_\epsilon$ and neither is a moderated variable	
(O_1) $Z \approx_\alpha t \implies \bot$	(O_2) $Z \not\approx_\alpha t \implies \top$	$Z \notin \texttt{vars}(t)$ and $Z \neq t$	
$\mathcal{R}_2$: Elimination of parameters and auxiliary unknowns.			
(C_1) $\forall\overline{Y}, Y : P \implies \forall\overline{Y} : P$	(C_2) $\exists\overline{W}, W : P \implies \exists\overline{W} : P$	$(C_3) \exists\overline{W}, W : W \approx_\alpha t \wedge P \implies \exists\overline{W} : P$	
$W \notin \texttt{vars}(P, t)$ and $Y \notin \texttt{vars}(P)$.			

$\mathcal{R}_3$: Equality and freshness simplification			
(E_1)	$\pi\cdot X \approx_\alpha \gamma\cdot X \implies \wedge\,\mathtt{ds}(\pi,\gamma)\#X$	(F_1)	$a\#\pi\cdot X \implies \pi^{-1}(a)\#X$
(E_2)	$[a]\,t \approx_\alpha [a]\,u \implies t\approx_\alpha u$	(F_2)	$a\#\,[a]\,t \implies \top$
(E_3)	$[a]\,t \approx_\alpha [b]\,u \implies (b\ a)\cdot t\approx_\alpha u \wedge b\#t$	(F_3)	$a\#\,[b]\,t \implies a\#t$
(E_4)	$f(\tilde t)\approx_\alpha f(\tilde u) \implies \wedge_i\, t_i\approx_\alpha u_i$	(F_4)	$a\#f(t_1,\dots,t_n) \implies \wedge_i a\#t_i$

$\mathcal{R}_4$: Instantiation Rules

(I_1) $Z\approx_\alpha t\wedge P \implies Z\approx_\alpha t\wedge P[Z/t]$, where $Z\notin \mathtt{vars}(t)$ and Z is not a parameter.

(I_2) $\pi\cdot Z\approx_\alpha t \implies Z\approx_\alpha \pi^{-1}\cdot t$ $\quad(I_3)$ $\pi\cdot Z\not\approx_\alpha t \implies Z\not\approx_\alpha \pi^{-1}\cdot t, t$ is not a suspension.

$\mathcal{R}_5$: Simplification of Parameters

$$\begin{array}{ll}
(U_1) & \forall\overline{Y},Y: P\wedge Y\not\approx_\alpha t \implies \bot\\
(U_2) & \forall\overline{Y}: P\wedge (Y\not\approx_\alpha t\vee Q) \implies \forall\overline{Y}: P\wedge Q[Y/t],\ \text{if } Y\notin\mathtt{vars}(t),\ Y\in\overline{Y}\\
(U_3) & \forall\overline{Y},Y: P\wedge Y\approx_\alpha t \implies \bot,\ \text{if } Y\not\equiv t\\
(U_4) & \forall\overline{Y}: P\wedge (Z_1\approx_\alpha t_1\vee\cdots\vee Z_n\approx_\alpha t_n\vee Q) \implies \forall\overline{Y}: P\wedge Q\\
(U_5) & \forall\overline{Y},Y: P\wedge a\#Y \implies \bot\\
(U_6) & \forall\overline{Y},Y: P\wedge a\not\#Y \implies \bot
\end{array}$$

Conditions for (U_4)**:** (i) Z_i is a variable and $Z_i\not\equiv t_i$; (ii) each equation in the disjunction contains at least one occurrence of a parameter; (iii) Q does not contain any parameter.

$\mathcal{R}_6$: Terms Disunification			
(DC)	$f(\tilde t)\not\approx_\alpha f(\tilde u) \implies \vee_i\, t_i\not\approx_\alpha u_i$	(NF_1)	$a\not\#\pi\cdot X \implies \pi^{-1}(a)\not\#X$
(D_1)	$\pi\cdot X\not\approx_\alpha \gamma\cdot X \implies \vee_i\,\mathtt{ds}(\pi,\gamma)\not\#X$	(NF_2)	$a\not\#\,[a]\,t \implies \bot$
(D_2)	$[a]\,t\not\approx_\alpha [a]\,u \implies t\not\approx_\alpha u$	(NF_3)	$a\not\#\,[b]\,t \implies a\not\#t$
(D_3)	$[a]\,t\not\approx_\alpha [b]\,u \implies (b\ a)\cdot t\not\approx_\alpha u\vee b\not\#t$	(NF_4)	$a\not\#f(\tilde t) \implies \vee_i a\not\#t_i$

$\mathcal{R}_7$: Explosion Rule

$$\exists\overline{W}\forall\overline{Y}: P \implies \exists W_1,\dots,W_n,\overline{W}\forall\overline{Y}: P\wedge X\approx_\alpha f(W_1,\dots,W_n)$$

Rule Conditions: (i) X is a free or existential variable occurring in P, $W_1,\dots,W_n$ are newly chosen auxiliary variables not occurring anywhere in the problem, and $f\in\Sigma$; (ii) there exists an equation $X=u$ (or disequation $X\not\approx_\alpha u$) in P such that u is not a variable and contains at least one parameter; (iii) no other rule can be applied.

The explosion rule creates a new problem for each $f\in\Sigma$. Given some explosion equation (disequation), all possible constructions with $f\in\Sigma$ must be considered for completeness' sake. Therefore, our procedure will build a finitely branching tree of problems to be solved.

Example 3. *Let $\mathcal{P}$ be a* NEP*, using the signature from Example 1, as follows:*

$$\mathcal{P}=\forall Y:\lambda\,[a]\,X\not\approx_\alpha \lambda\,[a]\,\lambda\,[a]\,Y \overset{dec}{\implies} \forall Y:[a]\,X\not\approx_\alpha [a]\,\lambda\,[a]\,Y \overset{abs}{\implies} \forall Y: X\not\approx_\alpha \lambda\,[a]\,Y$$

Notice that more rules can be applied and the explosion rule results in two parallel problems $\mathcal{P}_1=\exists W_1\forall Y: X\not\approx_\alpha \lambda\,[a]\,Y\wedge X\approx_\alpha \lambda W_1$ and $\mathcal{P}_2=\exists W_1,W_2\forall Y: X\not\approx_\alpha [a]\,Y\wedge X=W_1W_2$.

Successive application of rules gives:

$$\begin{aligned}
\mathcal{P}_1 &\overset{inst}{\Longrightarrow} \exists W_1 \forall Y : \lambda W_1 \not\approx_\alpha \lambda\,[a]\,Y \wedge X \approx_\alpha \lambda W_1 \\
&\overset{dec}{\Longrightarrow} \exists W_1 \forall Y : W_1 \not\approx_\alpha [a]\,Y \wedge X \approx_\alpha \lambda W_1 \\
&\overset{expl}{\Longrightarrow} \exists W_1 W_2 \forall Y : W_1 \not\approx_\alpha [a]\,Y \wedge X \approx_\alpha \lambda W_1 \wedge W_1 \approx_\alpha \lambda W_2 \\
&\overset{inst}{\Longrightarrow} \exists W_1 W_2 \forall Y : \lambda W_2 \not\approx_\alpha [a]\,Y \wedge X \approx_\alpha \lambda W_1 \wedge W_1 \approx_\alpha \lambda W_2 \\
&\overset{dis}{\Longrightarrow} \exists W_1 W_2 \forall Y : X \approx_\alpha \lambda W_1 \wedge W_1 \approx_\alpha \lambda W_2 \\
&\overset{pl,inst}{\overset{*}{\Longrightarrow}} \exists W_1 W_2 : X \approx_\alpha \lambda\lambda W_2 \wedge W_1 \approx_\alpha \lambda W_2.
\end{aligned}$$

Similarly, $\mathcal{P}_2 \overset{*}{\Longrightarrow} \exists W_1, W_2 : X = W_1 W_2$. *Notice that from this point one reaches a parameterless normal form. Solutions to* $\mathcal{P}$ *can be easily obtained by instantiating* W_2 *to any ground term in* $\mathcal{P}_1$ *and* W_1, W_2 *to any term in* $\mathcal{P}_2$ *since* X *only needs to be instantiated to a term headed by an application. It is easy to check that this choice indeed generates solutions for* $\mathcal{P}$.