4.3 Unification

When proving a proposition of the form $\exists x.\ A$ by its right rule in the sequent or focusing calculus, we must supply a term t and then prove $[t/x]A$. The domain of quantification may include infinitely many terms (such as the natural numbers), so this choice cannot be resolved simply by trying all possible terms t. Similarly, when we use a hypothesis of the form $\forall x.\ A$ we must supply a term t to substitute for x. We refer to this a *existential non-determinism*.

Fortunately, there is a technique called *unification* which is sound and complete for syntactic equality between terms. The basic idea is quite simple: we postpone the choice of t and instead substitute a new *existential variable* (often called *meta-variable* or *logic variable*) X for x and continue with the bottom-up construction of a derivation. When we reach initial sequents we check if there is a substitution for the existential variables such that the hypothesis matches the conclusion. If so, we apply this instantiation globally to the partial derivation and continue to search for proofs of other subgoals. Finding an instantiation for existential variables under which two propositions or terms match is called *unification*. It is decidable if a unifying substitution or *unifier* exists, and if so, we can effectively compute it in linear time. Moreover, we can do so with a minimal commitment and we do not need to choose between various possible unifiers.

Because of its central importance in both backward- and forward-directed search, unification has been thoroughly investigated. Herbrand [Her30] is given credit for the first description of a unification algorithm in a footnote of his thesis, but it was not until 1965 that it was introduced into automated deduction through the seminal work by Alan Robinson [Rob65, Rob71]. The first algorithms were exponential, and later almost linear [Hue76, MM82] and linear algorithms [MM76, PW78] were discovered. In the practice of theorem proving, generally variants of Robinson's algorithm are still used, due to its low constant overhead on the kind of problems encountered in practice. For further discussion and a survey of unification, see [Kni89]. We describe a variant of Robinson's algorithm.

Before we describe the unification algorithm itself, we relate it to the problem of proof search. We use here the sequent calculus with atomic initial sequents, but it should be clear that precisely the same technique of *residuation* applies to focused derivations. We enrich the judgment $\Gamma \overset{-}{\Longrightarrow} A$ by a *residual proposition* F such that

1. if $\Gamma \overset{-}{\Longrightarrow} A$ then $\Gamma \overset{-}{\Longrightarrow} A \setminus F$ and F is true, and

2. if $\Gamma \overset{-}{\Longrightarrow} A \setminus F$ and F is true then $\Gamma \overset{-}{\Longrightarrow} A$.

Generally, we cannot prove such properties directly by induction, but we need to generalize them, exhibiting the close relationship between the derivations of the sequents and residual formulas F.

Residual formulas F are amenable to specialized procedures such as unification, since they are drawn from a simpler logic or deductive system than the general propositions A. In practice they are often solved *incrementally* rather than collected throughout a derivation and only solved at the end. This is important for the early detection of failures during proof search. Incremental solution of residual formulas is the topic of Exercise **??**.

What do we need in the residual propositions so that existential choices and equalities between atomic propositions can be expressed? The basic proposition is one of equality between atomic propositions, $P_1 \doteq P_2$. We also have conjunction $F_1 \wedge F_2$, since equalities may be collected from several subgoals, and $\top$ if there are no residual propositions to be proven. Finally, we need the existential quantifier $\exists x.\ F$ to express the scope of existential variables, and $\forall x.\ F$ to express the scope of parameters introduced in a derivation. We add equality between terms, since it is required to describe the unification algorithm itself. We refer to the logic with these connectives as *unification logic*, defined via a deductive system.

$$\textit{Formulas} \quad F \quad ::= \quad P_1 \doteq P_2 \mid t_1 \doteq t_2 \mid F_1 \wedge F_2 \mid \top \mid \exists x.\ F \mid \forall x.\ F$$

The main judgment "F *is valid*", written $\models F$, is defined by the following rules, which are consistent with, but more specialized than the rules for these connectives in intuitionistic natural deduction (see Exercise **??**).

$$\frac{}{\models P \doteq P} \doteq\mathrm{I} \qquad\qquad \frac{}{\models t \doteq t} \doteq\mathrm{I}'$$

$$\frac{\models F_1 \qquad \models F_2}{\models F_1 \wedge F_2} \wedge\mathrm{I} \qquad\qquad \frac{}{\models \top} \top\mathrm{I}$$

$$\frac{\models [t/x]F}{\models \exists x.\ F} \exists\mathrm{I} \qquad\qquad \frac{\models [a/x]F}{\models \forall x.\ F} \forall\mathrm{I}^a$$

The $\forall\mathrm{I}^a$ rule is subject to the usual proviso that a is a new parameter not occurring in $\forall x.\ F$. There are no elimination rules, since we do not need to consider hypotheses about the validity of a formula F which is the primary reason for the simplicity of theorem proving in the unification logic.

We enrich the sequent calculus with residual formulas from the unification logic, postponing all existential choices. Recall that in practice we merge residuation and solution in order to discover unprovable residual formulas as soon as possible. This merging of the phases is not represented in our system.

Initial Sequents. Initial sequents residuate an equality between its principal propositions. Any solution to the equation will unify P' and P, which means that this will translate to a correct application of the initial sequent rule in the original system.

$$\frac{}{\Gamma, P' \overset{-}{\Longrightarrow} P \setminus P' \doteq P} \mathrm{init}$$

Propositional Connectives. We just give a few sample rules for the connectives which do not involve quantifiers, since all of them simply propagate or combine unification formulas, regardless whether they are additive, multiplicative, or exponential.

$$\frac{\Gamma, A \overset{-}{\Longrightarrow} B \setminus F}{\Gamma \overset{-}{\Longrightarrow} A \supset B \setminus F}\supset\mathrm{R} \qquad \frac{}{\Gamma \overset{-}{\Longrightarrow} \top \setminus \top}\top\mathrm{R}$$

$$\frac{\Gamma, A \supset B \overset{-}{\Longrightarrow} A \setminus F_1 \qquad \Gamma, A \supset B, B \overset{-}{\Longrightarrow} C \setminus F_2}{\Gamma, A \supset B \overset{-}{\Longrightarrow} C \setminus F_1 \wedge F_2}\supset\mathrm{L}$$

Quantifiers. These are the critical rules. Since we residuate the existential choices entirely, the $\exists$R and $\forall$L rules instantiate a quantifier by a new *parameter*, which is existentially quantified in the residual formula in both cases. Similarly, the $\forall$R and $\exists$L rule introduce a parameter which is universally quantified in the residual formula.

$$\frac{\Gamma \overset{-}{\Longrightarrow} [a/x]A \setminus [a/x]F}{\Gamma \overset{-}{\Longrightarrow} \forall x.\ A \setminus \forall x.\ F}\forall\mathrm{R}^a \qquad \frac{\Gamma, \forall x.\ A, [a/x]A \overset{-}{\Longrightarrow} C \setminus [a/x]F}{\Gamma, \forall x.\ A \overset{-}{\Longrightarrow} C \setminus \exists x.\ F}\forall\mathrm{L}^a$$

$$\frac{\Gamma \overset{-}{\Longrightarrow} [a/x]A \setminus [a/x]F}{\Gamma \overset{-}{\Longrightarrow} \exists x.\ A \setminus \exists x.\ F}\exists\mathrm{R}^a \qquad \frac{\Gamma, \exists x.\ A, [a/x]A \overset{-}{\Longrightarrow} C \setminus [a/x]F}{\Gamma, \exists x.\ A \overset{-}{\Longrightarrow} C \setminus \forall x.\ A}\exists\mathrm{L}^a$$

The soundness of residuating equalities and existential choices in this manner is straightforward.

Theorem 4.9 (Soundness of Equality Residuation)
If $\Gamma \overset{-}{\Longrightarrow} A \setminus F$ *and* $\models F$ *then* $\Gamma \overset{-}{\Longrightarrow} A$.

Proof: By induction on the structure of the given derivation $\mathcal{R}$. We show the critical cases. Note how in the case of the $\exists$R rule the derivation of $\models \exists x.\ F$ provides the essential witness term t.

Case:

$$\mathcal{R} = \frac{}{\Gamma, P' \overset{-}{\Longrightarrow} P \setminus P' \doteq P}\mathrm{init}$$

$\models P' \doteq P$	By assumption
$P' = P$	By inversion
$\Gamma, P' \overset{-}{\Longrightarrow} P$	By rule init

Case:

$$\mathcal{R} = \frac{\begin{array}{c}\mathcal{R}_1 \\ \Gamma \stackrel{-}{\Longrightarrow} [a/x]A_1 \setminus [a/x]F_1\end{array}}{\Gamma \stackrel{-}{\Longrightarrow} \exists x.\, A_1 \setminus \exists x.\, F_1} \exists\mathrm{R}^a$$

$\models \exists x.\, F_1$	By assumption
$\models [t/x]F_1$ for some t	By inversion
$\Gamma \stackrel{-}{\Longrightarrow} [t/x]A_1 \setminus [t/x]F_1$	By substitution for parameter a
$\Gamma \stackrel{-}{\Longrightarrow} [t/x]A_1$	By i.h.
$\Gamma \stackrel{-}{\Longrightarrow} \exists x.\, A_1$	By rule $\exists$R

Case:

$$\mathcal{R} = \frac{\begin{array}{c}\mathcal{R}_1 \\ \Gamma \stackrel{-}{\Longrightarrow} [a/x]A_1 \setminus [a/x]F_1\end{array}}{\Gamma \stackrel{-}{\Longrightarrow} \forall x.\, A_1 \setminus \forall x.\, F_1} \forall\mathrm{R}^a$$

$\models \forall x.\, F_1$	By assumption
$\models [b/x]F_1$ for a new parameter b	By inversion
$\models [a/x]F_1$	By substititution of a for b
$\Gamma \stackrel{-}{\Longrightarrow} [a/x]A_1$	By i.h.
$\Gamma \stackrel{-}{\Longrightarrow} \forall x.\, A_1$	By rule $\forall$R

□

The opposite direction is more difficult. The desired theorem:

If $\Gamma \stackrel{-}{\Longrightarrow} A$ *then* $\Gamma \stackrel{-}{\Longrightarrow} A \setminus F$ *for some* F *with* $\models F$

cannot be proved directly by induction, since the premisses of the two derivations are different in the $\exists$R and $\forall$L rules. However, one can be obtained from the other by substituting terms for parameters. Since this must be done simultaneously, we introduce a new notation.

$$\textit{Parameter Substitution} \quad \rho \quad ::= \quad \cdot \mid \rho, t/a$$

We assume all the parameters a substituted for by ρ are distinct to avoid ambiguity. We write $A[\rho]$, $F[\rho]$, and $\Gamma[\rho]$, for the result of applying the substitution ρ to a proposition, formula, or context, respectively.

Lemma 4.10 *If* $\Gamma \stackrel{-}{\Longrightarrow} A$ *where* $A = A'[\rho]$, $\Gamma = \Gamma'[\rho]$ *then* $\Gamma' \stackrel{-}{\Longrightarrow} A' \setminus F$ *for some* F *such that* $\models F[\rho]$.

Proof: The proof proceeds by induction on the structure of the given derivation $\mathcal{D}$. We show only two cases, the second of which required the generalization of the induction hypothesis.

Case:

$$\mathcal{D} = \frac{}{\Gamma_1, P \overset{-}{\Longrightarrow} P}\,\text{init}$$

$\Gamma_1 = \Gamma_1'[\rho]$, $P = P'[\rho]$, and $P = P''[\rho]$	Assumption
$\Gamma_1', P' \overset{-}{\Longrightarrow} P'' \setminus P' \doteq P''$	By rule init
$\models P'[\rho] \doteq P''[\rho]$	By rule $\doteq$ I

Case:

$$\mathcal{D} = \frac{\begin{array}{c}\mathcal{D}_1\\ \Gamma \overset{-}{\Longrightarrow} [t/x]A_1\end{array}}{\Gamma \overset{-}{\Longrightarrow} \exists x.\ A_1}\,\exists\text{R}$$

$\exists x.\ A_1 = A'[\rho]$	Assumption
$A' = \exists x.\ A_1'$ for a new parameter a with	
$[a/x]A_1 = ([a/x]A_1')[\rho, a/a]$	By definition of substitution
$[t/x]A_1 = ([a/x]A_1')[\rho, t/a]$	By substitution for parameter a
$\Gamma = \Gamma'[\rho]$	Assumption
$\Gamma'[\rho] = \Gamma'[\rho, t/a]$	Since a is new
$\Gamma' \overset{-}{\Longrightarrow} [a/x]A_1' \setminus [a/x]F_1$, and	
$\models ([a/x]F_1)[\rho, t/a]$	By i.h.
$\Gamma' \overset{-}{\Longrightarrow} \exists x.\ A_1' \setminus \exists x.\ F_1$	By rule $\exists$R
$\models (\exists x.\ F_1)[\rho]$	By rule $\exists$R and definition of substitution

□

Theorem 4.11 (Completeness of Equality Residuation)
If $\Gamma \overset{-}{\Longrightarrow} A$ *then* $\Gamma \overset{-}{\Longrightarrow} A \setminus F$ *for some* F *and* $\models F$.

Proof: From Lemma 4.10 with $A' = A$, $\Gamma' = \Gamma$, and ρ the identity substitution on the parameters in Γ and A. □

Next we describe an algorithm for proving residuated formulas, that is, an algorithm for unification. We do this in two steps: first we solve the problem in the fragment without parameters and universal quantifiers and then we extend the solution to the general case.

There are numerous ways of describing unification algorithsm in the literature. We view it here as a process of transformation on a collection of constraints. In the first instance we consider global unification, where we are given a single constraint formula (as generated by equality residuation, for example) and we have to determine if it is true. Later, we will generalize the view in order just partially transform the constraints to a normal form which is easily seen to have most general solutions. This latter view will be particularly useful when constraints are generated incrementally during proof search.

A collection of equational constraints is simply a collection of formulas in the unification logic or an indication that the constraints are inconsistent (#).

$$\text{Constraints} \quad C \quad ::= \quad \cdot \mid F, C \mid \#$$

We will freely exchange formulas among the constraints, just as we freely exchange assumptions in the sequent calculus. The empty constraint "·" corresponds to success, a contradiction to failure of proving the unification formula. Constraints may contain free *unification variables* X which are interpreted existentially. They are also known as *existential variables* or *logic variables.* Note that unification variables are never bound. We group the rules into several classes. The first, breaks down the structure of the formulas in C.

$$\begin{array}{lcll}
F_1 \wedge F_2, C & \mapsto & F_1, F_2, C & \\
\top, C & \mapsto & C & \\
\exists x.\, F, C & \mapsto & [X/x]F, C & \text{where } X \text{ not free in } F \text{ or } C
\end{array}$$

The second group of rules breaks down equalities into simpler equalities.

$$\begin{array}{lcll}
p(t_1, \ldots, t_n) \doteq p(s_1 \ldots, s_n), C & \mapsto & t_1 \doteq s_1, \ldots, t_n \doteq s_n, C & \\
f(t_1, \ldots, t_n) \doteq f(s_1 \ldots, s_n), C & \mapsto & t_1 \doteq s_1, \ldots, t_n \doteq s_n, C & \\
p(t_1, \ldots, t_n) \doteq q(s_1 \ldots, s_n), C & \mapsto & \# & \text{where } p \neq q \\
f(t_1, \ldots, t_n) \doteq g(s_1 \ldots, s_n), C & \mapsto & \# & \text{where } f \neq g
\end{array}$$

Note that equations of predicate or function symbols without arguments ($n = 0$) will either be simply removed or be inconsistent.

Finally, we will be left with equations where one of the two sides is a unification variable (we are not yet considering parameters). In that case, we must consider the right-hand side and distinguish several cases:

$$\begin{array}{lcll}
X \doteq X, C & \mapsto & C & \\
X \doteq t, C & \mapsto & [t/X]C & \text{provided } X \text{ not free in } t \\
t \doteq X, C & \mapsto & [t/X]C & \text{provided } X \text{ not free in } t \\
X \doteq t, C & \mapsto & \# & \text{if } t \neq X \text{ and } X \text{ free in } t \\
t \doteq X, C & \mapsto & \# & \text{if } t \neq X \text{ and } X \text{ free in } t
\end{array}$$

The conditions on these rules are necessary in order to recognize cases such as $X \doteq f(X)$, which has no solution: No matter which term we substitute for X, the right-hand side will always have one more function symbol than the left-hand side, so the equation cannot be satisfied. We refer to the condition "*X not free in t*" as the *occurs-check*.

Note that the whole algorithm depends critically on the function symbols being uninterpreted. As a trivial example, consider $+(3, 4) \doteq +(2, 5)$ on which the above algorithm would fail. Slighly trickier is something like $X \doteq -(-(X))$ which is true for any integer X, but violates the occurs-check.

As a first step in the correctness proof we can verify that a unification will always terminate.

Lemma 4.12 (Termination of Unification) *Any sequence of reductions $C \mapsto C' \mapsto C'' \ldots$ must terminate and yield either $\#$ or the empty set of constraints $(\cdot)$.*

Proof: By nested induction, first on the number of variables (unification variables X or bound variables $\exists x$) in C, second on the total size of the constraint, counting quantifiers, logical connectives, and variables occurrences.

The first set of rules for structural decomposition and the rule for eliminating $X \doteq X$ decreases the size of the constraints, without increasing the number of variables. The set of rules for variables (except for $X \doteq X$) reduces the number of variables in C by substitution for all occurrences of a variable (possibly increasing the total size of the constraint). □

In order to show the correctness of the unification algorithm, we would like to show that each step preserves provability. That is, if $C \mapsto C'$ then C is provable iff C' is provable. However, a difficulty arises in the case of existential quantification, since we step from $\exists x.\ F$ to $[X/x]F$ and we have not defined what it means for a formula with a unification variable to be provable. Intuitively, it should mean that not the formula itself, but some instance of it is provable. Hence we define that a constraint is *satisfiable* to mean that there is an instance that is provable. In order to define the concept of an *instance* we define simultaneous substitution for the unification variables of a term.

The second concept we need is that of a *substitution* for existential variables. We use a new notation, because this form of substitution is quite different from substitutions for bound variables x or parameters a.

$$\textit{Substitutions} \quad \theta \quad ::= \quad \cdot \mid \theta, t/X$$

We require that all variables X defined by a substitution are distinct. We write $\mathrm{dom}(\theta)$ for the variables defined by a substitution and $\mathrm{cod}(\theta)$ for all the variables occuring in the terms t. For a ground substitution $\mathrm{cod}(\theta)$ is empty. For the technical development it is convenient to assume that the domain and co-domain of a substitution share no variables. This rules out "circular" substitutions such as $f(X)/X$ and it also disallows identity substitutions X/X. The latter restriction can be dropped, but it does no harm and is closer to the implementation. As for contexts, we consider the order of the definitions in a substitution to be irrelevant.

We write $t[\theta]$, $A[\theta]$, and $\Gamma[\theta]$ for the application of a substitution to a term, proposition, or context. This is defined to be the identity on existential variables that are not explicitly defined in the substitution.

We also need an operation of composition, written as $\theta_1 \circ \theta_2$ with the property that $t[\theta_1 \circ \theta_2] = (t[\theta_1])[\theta_2]$ and similarly for propositions and contexts. Composition is defined by

$$\begin{aligned} (\cdot) \circ \theta_2 &= \theta_2 \\ (\theta_1, t/X) \circ \theta_2 &= (\theta_1 \circ \theta_2), t[\theta_2]/X \end{aligned}$$

In order for composition to be well-defined and have the desired properties we require that $\mathrm{dom}(\theta_1)$, $\mathrm{dom}(\theta_2)$ and $\mathrm{cod}(\theta_2)$ are disjoint, but of course variables in the co-domain of θ_1 can be defined by θ_2.

Now we define that *constraint* $C = F_1, \ldots, F_n$ *is satisfiable* if there exists a substitution θ for unification variables in C such that $\models F_i[\theta]$ for all $1 \leq i \leq n$. We write C *sat* if C is satisfiable.

Theorem 4.13 (Preservation of Satisfiability)
If $C \mapsto C'$ *then* C *sat iff* C' *sat*

Proof: In both directions, the proof is by cases on the definition of $C \mapsto C'$. We show a three cases from left-to-right. The other cases and opposite direction are similar.

Assume $C \mapsto C'$ and C *sat*. We have to show the C' *sat*.

Case: $\exists x.\, F, C_1 \mapsto [X/x]F, C_1$.

$\exists x.\, F, C_1$ *sat*	Assumption
For some θ, $\models (\exists x.\, F)[\theta]$	
and $\models F_1[\theta]$ for every F_1 in C_1	By defn. of *sat*
$\models \exists x.\, (F[\theta])$	By defn. of substitution
$\models [t/x](F[\theta])$	By inversion
$\models ([t/x]F)[\theta]$	By props. of substitution
$\models ([X/x]F)[\theta, t/X]$	Since X not in F or t
$\models F_1[\theta, t/X]$ for any F_1 in C_1	Since X not in C_1
$[X/x]F, C_1$ *sat*	By defn. of *sat*

Case: $X \doteq t, C_1 \mapsto [t/X]C_1$ where X not in t.

$X \doteq t, C_1$ *sat*	Assumption
For some θ, $\models (X \doteq t)[\theta]$	
and $\models F_1[\theta]$ for every F_1 in C_1	By defn. of *sat*
$\models X[\theta] \doteq t[\theta]$	By defn. of substitution
$X[\theta] = t[\theta]$	By inversion
$\theta = (\theta', t[\theta]/X)$	By defn. of substitution
$t[\theta]/X = t[\theta']/X$	Since X not in t
$\models F_1[\theta', t[\theta']/X]$ for any F_1 in C_1	From above
$\models ([t/X]F_1)[\theta']$	By props. of substitution
$[t/X]C_1$ *sat*	By defn. of *sat*

Case: $X \doteq t, C_1 \mapsto \#$ where X in t, $X \neq t$.

$X \doteq t, C_1$ *sat*	Assumption
$\models (X \doteq t)[\theta]$ for some θ	By defn. of *sat*
$\models X[\theta] \doteq t[\theta]$	By defn. of substitution
$X[\theta] = t[\theta]$	By inversion
$X[\theta] = f(\ldots X \ldots)[\theta]$	Since X in t, $X \neq t$

$X[\theta] = f(\ldots X[\theta] \ldots)$	By defn. of substitution
Contradiction	Right-hand side has more function symbols than left-hand side

This case is impossible

□

The argument above requires some elementary reasoning about substitution. Those proofs are usually straightforward by induction on the structure of the term we substitute in, as long as the right condition on occurrences of variables are known.

Termination of unification together with preservation of satisfiability gives us the correctness of unification as a procedure.