A Satisfiability Algorithm for a Typed Feature Logic

Stephan Kepser

Contents

I Description of the Problem

1 Introduction

The aim of this Magisterarbeit is to show that Paul King's typed feature logic is satisfiable. King (1989) developed a logic for Head-driven Phrase Structure Grammar in his doctoral thesis. HPSG (Pollard and Sag 1987, Pollard and Sag in press) has become one of the two predominant linguistic theories. Calling HPSG just a linguistic theory, is a bit misleading. Rather it exists of at least four parts which can be bundled into three different groups. First of all there is a specific formalism underlying the linguistic theory, the formalism of typed feature structures. The second group consist of the theories. There is a universal linguistic theory which is assumed to hold for every language of the world. This is HPSG's universal grammar. And there are the language specific theories which complement the universal theory to give grammars for all the specific languages. The third block consists of language specific lexica. Both theories in the second part and lexica of the third have to be formalised using the formalism of the first part. The existence of this formalism made HPSG appealing both for theoretical and computer linguistic work. It is this formalism that concerns us here, too. We will not present the formalism as given in the earlier (Pollard and Sag 1987) or later book (Pollard and Sag in press).[1] Instead we expect the reader to have already certain intuitions and experiences with typed feature structures. We will rather take the logical formalism as provided by King here. It cannot be our task to show that this logic really captures the intuitions and formalisations of the HPSG formalism. This is one of the main claims of King's doctoral thesis (King 1989); and the interested reader is refered to it. There is only one point that we want to raise attention to. In opposite to LFG or other "more or less typed" feature structure formalisms, there is in HPSG a close connection between features and types in a structure. This connection is expressed by so called appropriateness restrictions without which the introduction of types is not really that useful. Appropriateness restrictions state in general for the whole formalism and therefore for each legitimate feature structure certain restrictions on which features and types can co occur. That is, for each type the appropriateness restrictions specify which features may be allowed with this type and for each of these legal types it is stated which values, i.e. types are appropriate. Capturing these requirements in the logic and ensuring their validness for each satisfiable formula of the logic is one of the main tasks.

Let us consider King's logic from a logician's viewpoint. Its expressive power situates it between propositional logic and first order predicate logic. It is well known that propositional logic is decidable. Predicate logic, on the other hand, is not. It is semi-decidable in that validity of a formula can be shown. That is we can proof whether a formula is true in all models. Satisfiability cannot be

[1] The formalism underwent some significant changes mainly in terms of ontology form the first to the second book.

shown. That is given any not inconsistent formula, there is no algorithmic way to find a model in which that formula is true. Since we show that King's logic is satisfiable it is immediately clear that the expressive means of the logic must be significantly reduced as compared to predicate logic. King's logic does not contain any quantification. There is only one variable which cannot be bound or abstracted over. There are constants, one-place function symbols and two two-place relations. All the standard connectives exist and are interpreted in their usual way. Since it is known that the presence of two-place predicates in a logic with standard use of variables as in predicate logic makes such a logic already unsatisfiable, the severe restrictions on the use of that just one variable here are of prime importance. It was not our task to find or only approach the border between decidability and undecidability. As stated before, King's logic is designed for a certain linguistic task. But we take a step into that direction. With type statements and two-place predicates we go beyond the expressive power of propositional logic. We also show that additional requirements on the models such as the appropriateness restrictions set them out do not represent unovercomeable obstacles. Therefore we provide another argument for the use of weak logics speciallised for certain limited purposes and against the full power of predicate logic or even modal logics which by all there their beauty and familiarity lack features which are sometime indispensable.

Let us take a look at the logic from a linguist's perspective. The linguist should of course be an HPSG linguist. One of the advantages of HPSG is that it is quite a formal theory. The linguist can formalise his ideas and thereby see, if they can be spelled out in a clear and consistent way. This, of course, requires minimally that a formalisation's consistency can be checked. Now, while one can directly see, if a small feature structure is consistent or not, this is certainly no longer true for a large grammar with many rules. The feature structure formalism itself does not allow such a consistency check. A logic though may do so. And the claim is that King's logic does. Because that is, what satisfiability does. If a formula or description is satisfiable, then there exists a model for it. And this can only be the case, if the formula is consistent. A claim that one can formalise and check linguistic theories has always been underlying HPSG. But only a satisfiability proof as it will be given here can justify that. There is an even further going demand in HPSG, namely that grammaticality of a linguistic expression with respect to a certain grammar be decidable. We will come back to this point later, after we explained the logic. Another interesting point to be made here is that HPSG makes use of full disjunctions and full negations in feature structures. Grammar rules like the Head Feature Principle are formalised as implications with complex antecedentia which thereby require full negation. But so far there had been no formalised interpretation for typed feature structures with full negation. In his doctoral thesis King proofed that the formalisation that Pollard and Sag developed in *Information Based Syntax and Semantics* is incorrect. Indeed there had even been claims that typed feature structures with full negation can't be modelled in a way that would leave the resulting logic decidable. The satisfiability proof given here proves this claim wrong and thereby justifies the use of full negation in feature structures.

Although we provide an algorithm to obtain a certain normal form for formulas, implementational issues have never been of any concern in this work. Thus the algorithm would be called extremely insufficient and therefore unsuitable for implementation. The aim of the algorithm is a theoretical one, namely to show that a mechanical procedure to reach that normal form does exist at all. It is only after this existence is ensured that one can start asking of how to find the most efficient way to reach the normal form. This problem is left to further, implementationally related work.

2 Syntax and Semantics of the Typed Feature Logic

As stated earlier, the logic we will use is King's typed feature logic. We will now provide the definitions for its syntax and semantics. Keeping in mind that we want to describe typed feature structures, one can already imagine the way the language has to look like. There must be a symbol for the root of a feature structure. Attributes will be concatenated to give terms. We handle structure sharing by a predicate that works like equality modulo definedness, that is without reflexivity. We also need a predicate to assign a type to a term. The semantics will be as closely related to standard semantics of a logic as possible. But there is one shift in point of view. In standard logic a formula is true in a model. But feature structures are designed to describe (linguistic) entities. Therefore we speak about descriptions. And these descriptions are to describe objects in the universes of our structures. Thus descriptions are true of objects and not of structures. We will see what that means later on. We start with the appropriateness restrictions.

1. Definition (Signatures) A **signature** is a total function
$\mathfrak{S} : \mathfrak{V} \times \mathfrak{A} \rightarrow \mathfrak{P}(\mathfrak{V})$ [2] where
  $\mathfrak{V}$ is a finite set of **varieties**, and
  $\mathfrak{A}$ is a set of **attributes**.

Varieties are sometimes called types.[3] Thus $\mathfrak{S}$ gives appropriateness restrictions for types and attributes: Given a certain variety, $\mathfrak{S}$ states which attributes are appropriate for that variety and which value these attributes may

[2] $\mathfrak{P}(M)$ is the power set of M.

[3] The notion of a *type* is used in so many different ways that its use is likely to be more confusing than enlightening. There is for example – to name just one major source of confusion – a major shift in what a type means from (Pollard and Sag 1987) to (Pollard and Sag in press). The above use of *type* is one along the lines of (Pollard and Sag 1987). To avoid adding to this confusion, we use the term as rarely as possible, and if we use it, we always mean types in the sense of (Pollard and Sag 1987). We rather stick with the term variety which King nowadays calls a species.

receive. Note that $\mathfrak{S}$ is a total function. For each pair of a variety and an attribute it states the set of varieties appropriate for this pair. An attribute $\alpha \in \mathfrak{A}$ is therefore inappropriate for a variety $\varphi \in \mathfrak{V}$, iff $\mathfrak{S}\langle\varphi, \alpha\rangle = \emptyset$.

We demand $\mathfrak{V}$ to be finite for purely technical reasons. Since the varieties are thought of to be the categories which allow us to categorise every object in our universes, there are no philosophical reasons as to why there should be only finitely many. — Although there are people who believe that there exists only a finite number of concepts. We will come back to this point at the place where the finiteness of $\mathfrak{V}$ is of importance.

In what follows we write $\{A \rightharpoonup B\}$ for the set of partial functions from A to B.

2. Definition (Structures) If $\mathfrak{S} : \mathfrak{V} \times \mathfrak{A} \to \mathfrak{P}(\mathfrak{V})$ is a signature, then an $\mathfrak{S}$-**structure** is a triple $\langle U, V, A\rangle$, where
U is a set of **objects**,
$V : U \to \mathfrak{V}$ is a **variety assignment** function, and
$A : \mathfrak{A} \to \{U \rightharpoonup U\}$ is an **attribute interpretation** function

such that for each $\alpha \in \mathfrak{A}$ and $u \in U$,
S1: If $A(\alpha)$ is defined on u then $V(A(\alpha)(u)) \in \mathfrak{S}\langle V(u), \alpha\rangle$, and
S2: if $\mathfrak{S}\langle V(u), \alpha\rangle le \neq \emptyset$ then $A(\alpha)$ is defined on u.

U and A together look like a typical model in a classical logics where attributes are interpreted as partial functions with no varieties or types. But our universe U is not flat, it is structured by varieties in that each object $u \in U$ gets a variety $v \in \mathfrak{V}$ assigned. Since varieties and attributes are correlated in a way defined by the signature $\mathfrak{S}$, we need a way to enforce coherence of this correlation in our $\mathfrak{S}$-structure. This is what the two conditions S1 and S2 do. S1 demands obedience to the appropriateness restrictions. Conversely, S2 requires that if a certain attribute α is appropriate for the variety which a certain object u get assigned then the partial function modeling α in the $\mathfrak{S}$-structure must be defined on that object u. It is also important to notice that an $\mathfrak{S}$-structure depends only on its signature $\mathfrak{S}$. Therefore an $\mathfrak{S}$-structure is not the same as a model in classical logic: The existence of an $\mathfrak{S}$-structure does not entail satisfiability of a description.

3. Definition (Terms) Let $:$ be a reserved symbol. If $\mathfrak{S} : \mathfrak{V} \times \mathfrak{A} \to \mathfrak{P}(\mathfrak{V})$ is a signature, then an $\mathfrak{S}$-**term** is a member of the smallest set X such that
$: \in X$, and
if $\tau \in X$ and $\alpha \in \mathfrak{A}$, then $\tau\alpha \in X$.

Write $\mathfrak{T}$ for the set of $\mathfrak{S}$-terms.

The reserved symbol $:$ is the root symbol of a feature structure. We write paths in a feature description from left to right, i.e. the sub paths on the left are closer to the root while those on the right are closer to the leaves. This is the standard notation for feature structures. A notation more common to logicians would stress on the fact that attribute are one-place function symbols and thus use standard function notation. In consequence the leaves would appear to the left while the root symbols were to be found to the right.

4. Definition (Term interpretation) Let $\mathfrak{S} : \mathfrak{V} \times \mathfrak{A} \to \mathfrak{P}(\mathfrak{V})$ be a signature and $S = \langle U, V, A \rangle$ be an $\mathfrak{S}$-structure. Then the S-**term interpretation** function is the function $[]_S : \mathfrak{T} \to \{U \rightharpoonup U\}$, where
$[:]_S$ is the identity function on U, and
if $\tau \in \mathfrak{T}$ and $\alpha \in \mathfrak{A}$ then $[\tau\alpha]_S = A(\alpha) \circ [\tau]_S$.

Terms are interpreted as partial functions. Since terms are rooted concatenations of attributes and attributes are interpreted as partial functions, it is most natural to interpret a term as the concatenation of those partial functions which are the interpretations of the attributes it consists of.

Now we can introduce the full logics of $\mathfrak{S}$-terms and its denotation.

5. Definition (Descriptions) Let $(,), \sim, \approx, \neg, \wedge, \vee$ and $\to$ be reserved symbols. Let $\mathfrak{S} : \mathfrak{V} \times \mathfrak{A} \to \mathfrak{P}(\mathfrak{V})$ be a signature. Then an $\mathfrak{S}$-**description** is a member of the smallest set X such that
if $\tau \in \mathfrak{T}$ and $\varphi \in \mathfrak{V}$ then $\tau \sim \varphi \in X$,
if τ_1 and $\tau_2 \in \mathfrak{T}$ then $\tau_1 \approx \tau_2 \in X$,
if $\delta \in X$ then $\neg\delta \in X$,
if δ_1 and $\delta_2 \in X$ then $(\delta_1 \wedge \delta_2), (\delta_1 \vee \delta_2), (\delta_1 \to \delta_2) \in X$.

Write $\mathfrak{D}$ for the set of $\mathfrak{S}$-descriptions.

There are two basic statements one can make. One can say that a certain term or path receives a certain variety. And one can say that two path are equated. This is not to say that they are equal in the usual sense. Rather this is our way to express what is called a structure sharing or reentrancy in the notions of feature structures. In graph theoretical notions this means that two path meet at a certain node, that this node can be reached equally via both paths. Note also that the $\approx$ predicate is used to express that a certain path is defined. Not all terms we can construct must be defined in a description. Thus $\tau \approx \tau \in \Sigma$ states that the term τ is defined in the description Σ. Out of these two basic formulas complex descriptions can be formed using the standard connectives negation, conjunction, disjunction and implication.

As a notational convention, we will use $\tau, \tau_1, \tau_2, \tau', \ldots$ or κ for terms, α or β for single attributes, σ, σ' for attribute paths and $\varphi, \varphi_1, \varphi_2, \ldots, \varphi'$ for varieties.

6. Definition (Description Denotations) If $\mathfrak{S} : \mathfrak{V} \times \mathfrak{A} \to \mathfrak{P}(\mathfrak{V})$ is a signature and $S = \langle U, V, A \rangle$ is an $\mathfrak{S}$-structure, then the S-**denotation** is the function $[\![]\!]_S : \mathfrak{D} \to \mathfrak{P}(U)$, where $(\tau, \tau_1, \tau_2 \in \mathfrak{T}, \varphi \in \mathfrak{V}, \delta, \delta_1, \delta_2 \in \mathfrak{D})$

$[\![\tau \sim \varphi]\!]_S = \{u \in U \mid [\tau]_S$ is defined on u and $V([\tau]_S(u)) = \varphi\}$,
$[\![\tau_1 \approx \tau_2]\!]_S = \{u \in U \mid [\tau_1]_S$ is defined on u,
$[\tau_2]_S$ is defined on u,
and $[\tau_1]_S(u) = [\tau_2]_S(u)\}$,
$[\![\neg\delta]\!]_S = U \setminus [\![\delta]\!]_S$,
$[\![(\delta_1 \wedge \delta_2)]\!]_S = [\![\delta_1]\!]_S \cap [\![\delta_2]\!]_S$,
$[\![(\delta_1 \vee \delta_2)]\!]_S = [\![\delta_1]\!]_S \cup [\![\delta_2]\!]_S$, and
$[\![(\delta_1 \rightarrow \delta_2)]\!]_S = (U \setminus [\![\delta_1]\!]_S) \cup [\![\delta_2]\!]_S$.

The denotation of a description is the set of objects in the universe U of which the description is true. In other words, descriptions are given to describe certain object in our universe. Thus the denatation of a description is the set of objects for which this description holds, or which it describes. This is the way to understand the denotations of the basic statements. Which objects does $\tau \sim \varphi$ describe? The interpretation of τ must be defined on an object u; and the variety which that object that is given by application of the interpretation function of τ on u receives must be φ. And the denotation of $\tau_1 \approx \tau_2$ is given by those objects u which have the interpretation functions of τ_1 and τ_2 both defined on them and for which these functions yield the same result. The interpretation of the connectives is rather standard. The interpretation of the negation is the complement against the universe. Conjunctions are interpreted as intersections, since both descriptions are supposed to hold for an object; disjunctions are taken as unions, since at least one description of the two shall hold for an object. Finally we note that $[\![(\delta_1 \rightarrow \delta_2)]\!] = [\![((\neg\delta_1) \vee \delta_2]\!]$ which is the standard reading of implications.

7. Definition (Theory) Let $\mathfrak{S}$ be a signature. An $\mathfrak{S}$-**theory** is a subset of $\mathfrak{D}$, the set of $\mathfrak{S}$-descriptions.

A theory is a collection of descriptions which is interpreted conjunctively.

8. Definition (Theory denotation) Let $\mathfrak{S}$ be a signature and let $S = \langle U, V, A\rangle$ be an $\mathfrak{S}$-structure. Then the S-denotation of a theory $T \subseteq \mathfrak{D}$ is the function $[\![]\!]_S : \mathfrak{P}(\mathfrak{D}) \rightarrow \mathfrak{P}(U)$, where
$[\![T]\!] = \bigcap\{[\![\delta]\!]_S \mid \delta \in T\}$

Thus the denotation of some theory is the set of those objects in the universe for which the description given in this theory holds. Notice that $[\![\emptyset]\!]_S = U$.

9. Definition (Satisfiability) Let $\mathfrak{S}$ be a signature and $T \subseteq \mathfrak{D}$ be a theory. T is satisfiable iff there is an $\mathfrak{S}$-structure S with $[\![T]\!]_S \neq \emptyset$.

10. Definition (Model) Let $\mathfrak{S}$ be a signature and $T \subseteq \mathfrak{D}$ be a theory. An $\mathfrak{S}$-structure $S = \langle U, V, A\rangle$ is a **model** of T, if $[\![T]\!]_S = U$.

Thus an $\mathfrak{S}$-structure $S = \langle U, V, A \rangle$ is a model of a theory, if the theory is true of *every* object in the universe U. Since U is supposed to be non-empty, this is a demand stronger than just satisfiability which only requests that the theory is true of at least one object in the universe U.

Since we will be using it later on quite often, we take over the notion of a literal from standard logic.

11. Definition A **set of literals** Σ is a subset of the set of $\mathfrak{S}$-descriptions ($\Sigma \subset \mathfrak{D}$ where each element is an atomic formula or its negation; i.e. if $\delta \in \Sigma$ then it has one of the four forms

$\tau \sim \varphi$ with $\tau \in \mathfrak{T}$ and $\varphi \in \mathfrak{V}$ or
$\tau_1 \approx \tau_2$ with τ_1 and $\tau_2 \in \mathfrak{T}$ or
$\neg\tau \sim \varphi$ with $\tau \in \mathfrak{T}$ and $\varphi \in \mathfrak{V}$ or
$\neg\tau_1 \approx \tau_2$ with τ_1 and $\tau_2 \in \mathfrak{T}$

The terms of a set of literals are all terms and their subterms occurring in one literal or the other in that set.

12. Definition Let Σ be a set of literals. Define

$$\begin{aligned}\textsc{Term}(\Sigma) = \quad & \{\tau \in \mathfrak{T} \mid (\neg)\tau\sigma \approx \tau' \in \Sigma, \sigma \in \mathfrak{A}^*\} \\ & \cup\{\tau \in \mathfrak{T} \mid (\neg)\tau' \approx \tau\sigma \in \Sigma, \sigma \in \mathfrak{A}^*\} \\ & \cup\{\tau \in \mathfrak{T} \mid (\neg)\tau\sigma \sim \varphi \in \Sigma\}\end{aligned}$$

be the set of all terms of Σ.

3 The Method

What we want to show is satisfiability of the logic. With the above definitions of the logic, our task is then: Given any description, we must show that it is inconsistent or provide a structure for it such that the description is true of at least one object in the structure's universe; and we must do so in an algorithmic manour. This is quite a large problem; and finding a satisfier for any consistent formula in any form is very cumbersome, if not impossible. Therefore we split the task into subproblems of handable size. We will define two normal forms for descriptions, one is called *normal sets* and the other *closed sets*. The normal forms are defined in such a way that a normal set is satisfiable. Therefore the task of the algorithm will be to transform any description into something like a normal set. More precisely, a given description will firstly be put into disjunctive normal form. Then we apply the normalisation algorithm to be described later to that formula in DNF. This will give us a set of disjuncts. If this set is empty, then the original formula is unsatisfiable. If there are disjuncts left, then each of them is a normal set. Therefore the formula is satisfiable. Of course one must prove the claim that a normal set is satisfiable.

That is where the closed sets come in. A normal set is the weakest possible normal form that is still satisfiable. Thus one cannot directly give a model for that. A closed set on the other hand is that specific that it is easy to state a structure which satisfies it. Thus the introduction of closed sets is a matter of convenience in that showing their satisfiability is simple. The remaining difficult task is to show that a normal set can be embedded in a closed set. And it is the core work of this thesis to prove that this can be done in a natural way. That means one can enrich a normal set by adding literals to it which are either already more or less implied by the normal set or which we are free to choose to extend the normal set without changing the underlying properties of the set.

4 The Relationship to HPSG: Grammaticality

As stated earlier, it is very important for Pollard and Sag that HPSG is a *formal* theory. To underline that commitment they state the following:

> There is a further condition of *decidability* that we impose upon a linguistic theory, [...]. That is, we require that for a substantial fragment of candidate expressions (i.e. expressions and non-expressions) for a given language under study, it must be determinable by algorithm whether each candidate expression is assigned a well-formed structure by the theory, and if so what the structure is.
>
> Pollard & Sag: HPSG II, page xviii

The question is how can this decidability be achieved. We show here that King's logic which is very strongly tight to the formal side of HPSG is satisfiable and therefore decidable. Now, it would be tempting to say that this is it, but unfortunately, it is not. Decidability of the logic means that there is a way to check consistency of a description. Since grammars are nothing but big descriptions, their consistency can be checked, too. But what Pollard and Sag demand here, is more. It is the decidability of grammaticality. In order to see the problem we must understand the notion of grammaticality in terms of the logic. A phrase is grammatical, if it obeys to all the grammar rules. A typical grammar rule is the Head Feature Principle (appendix of HPSG II):

$$_{phrase}[\text{DTRS } \textit{headed-struc}] \rightarrow \begin{bmatrix} \text{SYNSEM LOC CAT HEAD} & \boxed{1} \\ \text{DTRS HEAD-DTR SYNSEM LOC CAT HEAD} & \boxed{1} \end{bmatrix}$$

How does this rule read? Take a large feature structure. At every node of sort *phrase* with a DTRS feature with value *headed-struc* in this feature structure it must be the case that the values for the SYSEM LOC CAT HEAD attribute path and the DTRS HEAD-DTR SYNSEM LOC CAT HEAD attribute path are structure-shared. The important bit for us here is the inherent recursion: The rule must hold for *every* sub structure of sort *phrase* with DTRS feature having *headed-struc* as value. The recursion is not a direct part of the rule, but this is the way the rule is supposed to be set to action. The translation into the logic gives us the following description:

$(:\sim \textit{phrase} \quad \wedge \quad : \text{DTRS} \sim \textit{headed-struct}) \rightarrow$
$:\text{SYN LOC CAT HEAD} \approx :\text{DTRS HEAD-DTR SYNSEM LOC CAT HEAD}$

The description states almost the same, but it does not contain the recursion. In opposite, there is a direct reference to the root of the description: $:\sim \textit{phrase}$ demands the root of the description to be of sort *phrase*. Now if the phrase to be analised is a whole sentence, how can we ensure in the logic that the Head Feature Principle is not only applied to the outmost phrase but also to the embeded VPs? Or, to put it differently, how do we get the recursion in? The answer is the notion of grammaticality as given in the next definition.

13. Definition (Grammaticality) Let Γ be a theory and δ a description. δ is **grammatical** with respect to Γ, if there is an $\mathfrak{S}$-structure $S = \langle U, V, A \rangle$ with

$$[\![\Gamma]\!]_S = U \text{ and } [\![\delta]\!]_S \neq \emptyset.$$

The recursion comes in by demanding that the denotation of the grammar (Γ) be the *whole* universe. Since every sub phrase of a phrase δ is modeled by some object in the universe, the demand that the grammar be true of every object ensures that all the objects obey to the grammar rules and therefore the sub parts of δ must do so, too. This solves the problem of recursion. But if one compares grammaticality with the earlier introduced notion of satisfiability, one can see that grammaticality demands much more: We must find a structure such that the grammar is true of every object of the universe and not just one, as it was required for satisfiability. Obviously, satisfiability is a first, necessary step. But a lot remains to be done, until Pollard and Sag's demand can be fullfiled.

5 Wedekind's Approach

In this section we describe an ealier approach to the logic of feature structures, namely that by Wedekind (1992). Wedekind develops a logic for linguistic formalisms like LFG. Now, though LFG uses kinds of feature structures, there is a significant difference to feature structures as being used in HPSG. LFG lacks the notion of types, while types are very important in HPSG. Types may at first thought be regarded as just names for nodes. But there is more to them, and this "more" is of course the appropriateness restrictions. They extend the expressive power of the formalism by specifying restrictions on what attributes a type may have and further what values the attributes may receive. In King's logic, it is the signature (as given in definition 1) which encodes the appropriateness restrictions. Now, as there are no types in LFG, nothing corresponding to a signature can be found in Wedekind's approach. Thus one can already imagine that this approach falls short of formalising a linguistic theory like HPSG. But this needs to be shown more clearly. And still, the approach shows in a simpler environment the typical means and mechanisms which we will make use of later. Thus they invite for being studied here.

In principle, Wedekind's approach looks as follows. The logic is a classical logic. It consists of constants as names for the roots of feature structures and one-place function symbols as names for attributes. Attribute path can be equated to give atomic formulas. And then these can be used to construct complex formulas with conjunction, disjunction, implication and negation in the usual manour. The models are rather standard, too. A model consists of a non-empty universe and an interpretation function. This function interprets constants as objects in the universe and function symbols as partial functions on the objects in the universe. The notion of satisfiability, which we are interested in here, follows standard definitions in classical logic, too. What concers us most, is the proof of satisfiability which says that if a formula is consistent, than there is a model for it. This proof proceeds in two steps. Wedekind defines a normal form that stands in the middle of these two steps. The normal form for a formula is defined in such a way, that one can either directly see, that the formula is inconsistent, or one can state a model for it. The model is a version of the classical term model. The important first half of the proof consists of an algorithm which shows that any formula can be transformed into normal form in an algorithmic way. In the second half it is shown that there exists a model for each formula in normal form which is not obviously inconsistent. This strategy is interesting for us, because we want to use a similar one. We will also define normal forms and give an algorithm to obtain a normal form.

Without changing its contents, we amend Wedekinds notation to look as similar to ours as can be to ease comparison.

14. Definition (Wedekind's Logic: The Syntax) Let C be a set of *constants* and F be a set of *unary function symbols*. The set of *terms* $\mathfrak{T}_W$ is given by: Each constant is a term ($C \subseteq \mathfrak{T}_W$). If $f \in F$ and $\tau \in \mathfrak{T}_W$, then $\tau f \in \mathfrak{T}_W$. The set of *atomic formulas* is $\{\tau_1 \approx \tau_2 \mid \tau_1, \tau_2 \in \mathfrak{T}_W\} \cup \{\bot\}$. The set of *formulas* is given by: Each atomic formula is a formula. If φ and ψ are formulas, the so are $(\varphi \wedge \psi), (\varphi \vee \psi), (\varphi \rightarrow \psi)$ and $(\neg\varphi)$.

If α is a well-formed expression, i. e. a term or a formula, then the substitution $\alpha[\tau_1/\tau_2]$ is given by replacing *some* (possibly all or none) occurences of τ_1 in α by τ_2.

15. Definition (Wedekind's Models) A *model* consists of a pair $M = \langle U, I\rangle$, where U is a non-empty universe and $I = I_C \cup I_F$ is a interpretation function such that

(i) $I_C \in \{C \rightharpoonup U\}$
(ii) $I_F \in \{F \rightharpoonup \{U \rightharpoonup U\}\}$
(iii) $\forall f \in F(f \in Dom(I) \Longrightarrow I(f) \neq \emptyset)$

Condition (iii) is there to ensure that partial function symbols which are interpreted, receive a non-empty interpretation. This is done to avoid redundancies between non-interpreted functions and functions which receive an empty interpretation.

16. Definition (Wedekind's Satisfiability) Satisfiability ($M \models \varphi$, read: M models φ, M is a model of φ, φ is true in M) is defined recursively:

$M \not\models \bot$
$M \models \tau \approx \tau'$, iff $I(\tau)$ and $I(\tau')$ are defined and $I(\tau) = I(\tau')$
$M \models (\varphi \wedge \psi)$, iff $M \models \varphi$ and $M \models \psi$
$M \models (\varphi \vee \psi)$, iff $M \models \varphi$ or $M \models \psi$
$M \models (\varphi \rightarrow \psi)$, iff $M \models \varphi$ entails $M \models \psi$
$M \models (\neg\varphi)$, iff $M \not\models \varphi$.

Wedekind's notion of satisfiability is rather standard. The connectives are interpreted the usual way; and the $\approx$ for path equations is interpreted as normal equality, on things that are defined, of course. We can now go ahead and see how one can find a model for every and only the satisfiable formulae. Firstly, we give the normal form mentioned above.

17. Definition (Wedekind's Closed Set) A set of literals Σ is closed, iff

$\Sigma = \{\bot\}$

or

$\bot \notin \Sigma$,
if $\tau_1 \approx \tau_2 \in \Sigma$ then $\tau_2 \approx \tau_1 \in \Sigma$,
if $\tau_1 \approx \tau_2 \in \Sigma$ and $\tau_2 \approx \tau_3 \in \Sigma$, then $\tau_1 \approx \tau_3 \in \Sigma$,
if $\tau\alpha \approx \tau\alpha \in \Sigma$, then $\tau \approx \tau \in \Sigma$,
if $\tau_1 \approx \tau_2 \in \Sigma, \tau_1\alpha \approx \tau_1\alpha \in \Sigma, \tau_2\alpha \in \textsc{Term}(\Sigma)$, the $\tau_1\alpha \approx \tau_2\alpha \in \Sigma$,
if $\neg\delta \in \Sigma$ then $\delta \notin \Sigma$

The demands for a closed set are clearly expressed. A closed set is eighter obviously unsatisfiable ($\Sigma = \{\bot\}$). Or it does not contain the absurdity and is closed under symmetry, transitivity, subterm reflexivity and a version of right invariance. This version here demands that right invariance be restricted to terms already introduced previously ($\tau_2\alpha \in \textsc{Term}(\Sigma)$). This restriction enables the handling of loops without closed sets necessarily becoming infinite. This is important, if one wants to obtain a closed set in an algoritmic manour. And finally a closed set does not contain literals contradicting each other.

Any formula can be put into this normal form. The process to get there consists of two major steps. The first step is rather standard and consists in transforming the formula to disjunctive normal form. In the second step each disjunct is subjected to the following algorithm.

18. Algorithm (Wedekind's Closing Algorithm) The algorithm is given as a sequence of sets of formulas. Let Σ be a set of formulas. The set sequence is defined inductively by:

$$\Sigma_0 := \begin{cases} \{\bot\} & \text{if } \bot \in \Sigma\text{; otherwise} \\ \Sigma \cup \{\tau_2 \approx \tau_1 \mid \tau_1 \approx \tau_2 \in \Sigma\} \cup \{\tau \approx \tau \mid \tau\sigma \approx \tau' \in \Sigma\} \\ \quad \cup \{\tau \approx \tau \mid \tau' \approx \tau\sigma \in \Sigma\} \end{cases}$$

$$\Sigma_{i+1} := \begin{cases} \{\bot\} & \text{if } \exists\delta \in \Sigma_i : \neg\delta \in \Sigma_i\text{; otherwise} \\ \Sigma_i \cup \{(\tau_1 \approx \tau_2)[\tau/\tau'] \mid & \tau_1 \approx \tau_2, \tau \approx \tau' \in \Sigma_i \quad \wedge \\ & \tau_1[\tau/\tau'], \tau_2[\tau/\tau'] \in \textsc{Term}(\Sigma)\} \end{cases}$$

Each step of the algorithm is split into two cases. The first one handles the situation where the set is unsatisfiable. This can happen in the base case (Σ_0), if Σ contains the absurdity, or in the step case (Σ_{i+1}), if Σ_i contains two formulas which contradict each other. Here, the set is simply reduced to $\{\bot\}$. In the second case, where the set is not obviously unsatisfiable, the set is grown a bit. New formulas are added in order to get closer to consistency or to finally hit a contradiction. In the base case, Σ is closed under symmetry and subterm reflexivity. The step case is a bit more complicated. We have two formulas $\tau_1 \approx \tau_2, \tau \approx \tau' \in \Sigma_i$ where τ is thought of to be a proper prefix of τ_1 or τ_2. If the substitution of τ' for τ in τ_1 and τ_2 does not create new terms, then we add new formulas $(\tau_1 \approx \tau_2)[\tau/\tau']$.

What remains is to present a way of how to get a model for every satisfiable closed set. Unsurprisingly the model we give is a version of the term model. As one can see, the path equated by $\approx$ form an equivalence class, hence the symmetry and transitivity demands in the definition of a closed set. Thus it is natural to take this class as the interpretation for each term in the class. More precisely this looks the following way.

19. Definition (Wedekind's Model for a Closed Set) Let $\Sigma \neq \{\bot\}$ be a closed set. Let $P = \{\tau_1 \approx \tau_2 \in \Sigma\}$ the set of path. We destinguish two cases.

1. $P = \emptyset$.

   There are no path defined in the given set. Thus we take $U = \{*\}$ to have a non-empty universe and $I = \emptyset$ since there are no attributes or path to be interpreted. $M = \langle U, I\rangle$ is a model.

2. $P \neq \emptyset$.

   Let
   $$D = \{\tau \mid \tau \approx \tau \in \Sigma\}$$
   the set of all terms defined in Σ. Define
   $$[\tau] = \{\tau' \mid \tau \approx \tau' \in \Sigma\}$$
   the equivalence class of τ. The universe
   $$U = \{[\tau] \mid \tau \in D\}$$

is the set of all equivalence classes which exist in Σ. For each constant symbol c,

$I(c)$ is defined, iff $c \in D$, and
if $I(c)$ is defined, then $I(c) = [c]$.

For each function symbol f,

$I(f)$ is defined, iff $\tau' f \in D$ for some term τ', and
if $I(f)$ is defined, then for each term τ,
$I(f)[\tau]$ is defined, iff $\tau' f \in D$ for some $\tau' \in [\tau]$, and
if $I(f)[\tau]$ is defined, then $I(f)[\tau] = [\tau' f]$ for each $\tau' \in [\tau]$ such that $\tau' f \in D$.

Thus the interpretation function takes every constant symbol to be represented by the class this constant symbol can be found in. A function symbol f is interpreted as the partial function $[\tau] \xrightarrow{I(f)} [\tau f]$. $M = \langle U, I \rangle$ is a model.

One can see that the definitions above constitute a model for any satisfiable closed set. We will not give a full prove of this statement here, it can be found in Wedekind's paper. The following table gives an overview of the strategy employed.

Given any formula φ

i. Convert φ to Disjunctive Normal Form

ii. Apply algorithm 18 to each disjunct

iii. Each disjunct is a Closed set

iv. The formula φ is satisfiable, iff at least one Closed set is not $\{\bot\}$

---

v. To prove the above, give a model for a Closed set $\neq \{\bot\}$

That Wedekind's approach lies completely in the realm of classical logic, is a consequence of the requirements – or lack of those – that the linguistic formalism to be modeled demands. LFG simply does not require a more sophisticated approach. Since the logic we give is supposed to model HPSG, we must cope with additional expectations. LFG and HPSG are both attribute-value formalisms. One of the key difference though is the notion of a type in HPSG, for which there is no counterpart in LFG. Thus the HPSG formalism offers an expressive means that is not present in LFG, namely the appropriateness restrictions. As explained earlier, appropriateness restrictions state generally and globaly which types are allowed to receive which attributes and which values are legal for these attributes. Now although linguistically speaking there are some conventions on the appearence of certain attributes in a feature structure in LFG, these do not have the status of restrictions expressable by the underlying formalism. One could violate these conventions without doing something formally illicit. In HPSG they are part of the formalism and thus must be respected and represented in some way in a logic which claims to model the formalism underlying HPSG. We claimed already that in our logic there is a thing representing the appropriateness restrictions, namely the signature. Them main factor that makes our approach more complicated and the later proofs so long is to show that the consequences of these appropriateness restrictions are really obeyed to. The question here is: Is there a way to express appropriateness restrictions in Wedekind's logic. The apparent answer is negative. One can express, that a certain term τ is not defined, simply by $\neg\tau \approx \tau$. But that is not the same as saying that it is in general illicit to define such a term. A simple example may illustrate that. Let $f \in F$ be a function symbol and $\tau \in \mathfrak{T}_W$ a term not containing f. Say the term τ is defined ($\tau \approx \tau$). How can we expressed that there may be no formula so ever containing the term τf which is another way of saying that feature f is not appropriate for term τ. Some concrete formula could contain a subformula $\neg\tau f \approx \tau f$. But there is no way we can enforce the presence of that subformula in every formula given. We can't prevent that eventually a formula may turn up which contains $\tau f \approx \tau f$ as a subformula. Jet that is exactly what we need to do. Thus there is no way to represent and express appropriateness restrictions in Wedekinds logic. Another way of looking at these problems is to look at models. Taking up the example above, even a formula containing $\tau f \approx \tau f$ as a subformula could have a model, a satisfier. That is another way of saying that the formula is consistent and therefore legal. But we demanded that formulae of this kind be illegal. That means there are models for formulas which do not obey appropriateness restrictions. And this happens for the simple reason that the logic simply does not give us the means to express appropriateness restrictions. Therefore Wedekind's approach does not solve our problem of finding a satisfiability procedure for a logic for HPSG.

II Normalisation of a Description

6 Normalising a Description

This part is concerned with the task of normalising a description. It reports work of a term paper by Thilo Götz (1993). The intention of converting a description into some normal form is typically to show that the description has certain properties which are inherent or a consequence of that normal form. In our case the claim is that a set of literals in normal form is satisfiable. Thus the normalisation is a way to determine if a given description is satisfiable. A description is any set with possibly conjunctive and disjunctive terms. Thus the the normalisation will take place in two steps. In a first step the description is transformed into disjunctive normal form. Thereafter each disjunct is put in a set on its own. The result, called a matrix, is a finite set each element of which represents one disjunct and is a finite set of literals. This transformation step is rather standard. Thus we simply postulate its existence without giving further notice.

The second step of the normalisation is obviously the more important one, since its existence cannot be assumed. Thus we have to present an algorithm to normalise a matrix. The algorithm will be given as a set of rewrite rules, and an application of the algorithm to a set of clauses will correspond to a closure under rule application.

To prove correctness of the algorithm, three things need to be considered:

1. Termination: for finite sets of finite clauses, rule application has to terminate after a finite number of steps.
2. Denotation preservation: each application of a rule must not alter the denotation of the set.
3. Partial correctness: no rules apply iff the set is in normal form.

To make things clearer, we introduce some notions.

20. Definition If τ_1 and τ_2 are terms and φ is a variety, then $\tau_1 \approx \tau_2$ and $\tau_1 \sim \varphi$ are *positive literals*.

If δ is a positive literal, then $\neg\delta$ is a *negative literal*.

A *clause* is a finite (possibly empty) set of literals.

A *matrix* is a finite (possibly empty) set of clauses.

A clause is interpreted *conjunctively*, i.e. if Σ is a clause, then $[\![\Sigma]\!] = \bigcap_{\delta\in\Sigma} [\![\delta]\!]$.

A matrix is interpreted *disjunctively*, i.e. if Γ is a matrix, then $[\![\Gamma]\!] = \bigcup_{\Sigma\in\Gamma} [\![\Gamma]\!]$.

What is called a clause here was previously, in definition 7, coined a theory. This has no significant meaning. The earlier notion stems from the perspective of logics while the notion of a clause is more common when speaking about input to an algorithm. The normal form to be obtained for a clause is given in the following definition.

21. Definition (Normal Clause) A set Σ of literals is **normal** iff the following conditions hold ($\tau, \tau_1, \tau_2 \in \textsc{Term}(\Sigma); \varphi, \varphi', \varphi_1, \varphi_2 \in \mathfrak{V}; \alpha \in \mathfrak{A}; \sigma \in \mathfrak{A}^*$)

(1) $:\approx: \in \Sigma$;
(2) if $\tau_1 \approx \tau_2 \in \Sigma$ then $\tau_2 \approx \tau_1 \in \Sigma$; (symmetry)
(3) if $\tau_1 \approx \tau_2, \tau_2 \approx \tau_3 \in \Sigma$ then $\tau_1 \approx \tau_3 \in \Sigma$; (transitivity)
(4) if $\tau\sigma \approx \tau\sigma \in \Sigma$ then $\tau \approx \tau \in \Sigma$; (subterm reflexivity)
(5) if $\tau_1 \approx \tau_2, \tau_1\sigma \approx \tau_1\sigma, \tau_2\sigma \approx \tau_2\sigma \in \Sigma$ then $\tau_1\sigma \approx \tau_2\sigma \in \Sigma$;
(6) if $\tau \approx \tau \in \Sigma$ then for some $\varphi \in \mathfrak{V}, \tau \sim \varphi \in \Sigma$;
(7) if for some $\varphi \in \mathfrak{V}, \tau \sim \varphi \in \Sigma$ then $\tau \approx \tau \in \Sigma$;
(8) if $\tau_1 \approx \tau_2 \in \Sigma, \tau_1 \sim \varphi_1 \in \Sigma, \tau_2 \sim \varphi_2 \in \Sigma$ then $\varphi_1 = \varphi_2$;
(9) if $\tau \sim \varphi_1 \in \Sigma, \tau\alpha \sim \varphi_2 \in \Sigma$ then $\varphi_2 \in \mathfrak{S}\langle\varphi_1, \alpha\rangle$;
(10) if $\tau \sim \varphi \in \Sigma, \tau\alpha \in \textsc{Term}(\Sigma), \mathfrak{S}\langle\varphi, \alpha\rangle \neq \emptyset$ then $\tau\alpha \approx \tau\alpha \in \Sigma$;
(11) if $\neg\delta \in \Sigma$ then $\delta \notin \Sigma$.

We explain shortly what the requirements for a normal clause mean. Condition (1) says that the description must be rooted, since the colon is our symbol for the root node of a feature structure. Condition (2) demands symmetry and (3) transitivity of $\approx$. Since what is expressed with $\approx$ on path that are defined is some kind of equality, symmetry and transitivity must hold. Of course, we cannot have full reflexivity, because not every path may be defined. Thus $\tau \approx \tau$ is not a vacuous statement – what it were, had we full reflexivity – but rather means that path τ is defined. As a consequence, we need subterm reflexivity as demanded in (4), because if a path is defined, then all its sub paths must be defined, too. Condition (5) is a weaker variant of right invariance. There will be a longer explanation of this condition later in section 9 on page 25 when we introduce a second normal form and explain the differences between these two. Requirements (6) and (7) belong together and speak about assigning varieties to paths. The first one demands that if a path is defined, it must receive a variety. The second one requires the inverse, that is if a variety assignment statement exists for a certain path, then this path must be defined. Condition (8) enforces consistency of the variety assignment. If two paths are defined and said to be equal, than they must of course receive one and the same variety. Condition (9) brings the appropriateness requirements into account. Say a path is defined and that it receives a certain variety. If the path grows by an attribute, then this attribute must be appropriate for the variety, and more the variety that the grown path receives must be appropriate for this attribute. That is what appropriateness specifications demand. Condition (10) spell out circumstances under which a certain path must grow. Again the full meaning of this requirement will become clear, when we compare it to the one in the second normal form. The last condition (11) is the usual demand that you must not have contradictory literals in your clause.

A matrix is normal iff all the clauses it contains are normal and there is at least one nonempty clause. The algorithm may produce an empty matrix; in that case, the original matrix was unsatisfiable.

We first give the algorithm in the next section and then show its correctness in the consecuting one.

7 The Normalisation Algorithm

The algorithm will be given as a set of rewrite rules on sets of clauses. We will use Γ as a variable over sets of clauses and Σ as a variable over clauses.

22. Algorithm (Clause Normalisation) The algorithm consists of a sequence rewrite rule applications. One step of the algorithm is the application of exactly one rewrite rule. The algorithm terminates, if no rule can be applied (any more). A rule applies to a set of clauses Γ' only if the left hand side of the rule matches Γ', and if the right hand side is a valid set description under the same variable assignment. The rewrite rules are

$$
\begin{array}{rcll}
\Gamma \uplus \{\Sigma\} & \longrightarrow & \Gamma \cup \{\Sigma \uplus \{:\approx:\}\} & (1) \\
\Gamma \uplus \{\Sigma \uplus \{\tau_1 \approx \tau_2\}\} & \longrightarrow & \Gamma \cup \{\Sigma \uplus \{\tau_1 \approx \tau_2\} \uplus \{\tau_2 \approx \tau_1\}\} & (2) \\
\Gamma \uplus \{\Sigma \uplus \{\tau_1 \approx \tau_2, \tau_2 \approx \tau_3\}\} & \longrightarrow & \Gamma \cup \{\Sigma \uplus \{\tau_1 \approx \tau_2, \tau_2 \approx \tau_3\} \uplus \{\tau_1 \approx \tau_3\}\} & (3) \\
\Gamma \uplus \{\Sigma \uplus \{\tau\sigma \approx \tau\sigma\}\} & \longrightarrow & \Gamma \cup \{\Sigma \uplus \{\tau\sigma \approx \tau\sigma, \tau \approx \tau\}\} & (4) \\
\Gamma \uplus \{\Sigma \uplus \{\tau_1 \approx \tau_2, \tau_1\sigma \approx \tau_1\sigma, & \longrightarrow & \Gamma \cup \{\Sigma \uplus \{\tau_1 \approx \tau_2, \tau_1\sigma \approx \tau_1\sigma, & (5) \\
\tau_2\sigma \approx \tau_2\sigma\}\} & & \qquad \tau_2\sigma \approx \tau_2\sigma\} \uplus \{\tau_1\sigma \approx \tau_2\sigma\}\} & \\
\Gamma \uplus \{\Sigma \uplus \{\tau \approx \tau\}\} & \longrightarrow & \Gamma \cup \{\Sigma \uplus \{\tau \approx \tau, \tau \sim \varphi\} \mid \varphi \in \mathfrak{V}\}, & (6) \\
 & & \textit{if } \Sigma \textit{ does not contain a literal} & \\
 & & \tau \sim \varphi',\ \varphi' \in \mathfrak{V} & \\
\Gamma \uplus \{\Sigma \uplus \{\tau \sim \varphi\}\} & \longrightarrow & \Gamma \cup \{\Sigma \uplus \{\tau \sim \varphi, \tau \approx \tau\}\} & (7) \\
\Gamma \uplus \{\Sigma \uplus \{\tau_1 \approx \tau_2, \tau_1 \sim \varphi_1, & \longrightarrow & \Gamma, \textit{if } \varphi_1 \neq \varphi_2 & (8) \\
\tau_2 \sim \varphi_2\}\} & & & \\
\Gamma \uplus \{\Sigma \uplus \{\tau \sim \varphi_1, \tau\alpha \sim \varphi_2\}\} & \longrightarrow & \Gamma, \textit{if } \varphi_2 \notin \mathfrak{S}\langle \varphi_1, \alpha \rangle & (9) \\
\Gamma \uplus \{\Sigma \uplus \{\tau \sim \varphi\}\} & \longrightarrow & \Gamma \cup \{\Sigma \uplus \{\tau \sim \varphi, \tau\alpha \approx \tau\alpha\}\}, & (10) \\
 & & \textit{if } \tau\alpha \in \mathrm{T{\small ERM}}(\Sigma) \textit{ and } \mathfrak{S}\langle \varphi, \alpha \rangle \neq \emptyset & \\
\Gamma \uplus \{\Sigma \uplus \{\delta, \neg\delta\}\} & \longrightarrow & \Gamma \textit{ for any positive literal } \delta & (11)
\end{array}
$$

There is an obvious correspondence between a line in the definition of a normal clause and the line in the algorithm suffixed with the same number. Thus, for example, line (3) in the definition demands transitivity. The corresponding line (3) in the rewrite rules picks out a clause and and two literals in it for which transitivity doesn't hold. It then adds the missing literal. Note the use of ordinary and disjoint union. The first rule for instance can only apply to a matrix that contains a clause Σ that doesn't contain $:\approx:$. Rule application will yield the same matrix except for set Σ: it now also contains $:\approx:$.

8 Proof Of Correctness

8.1 Termination

To show termination of the rewrite system, two separate issues need to be considered:

- for any matrix, there's only a finite number of matrices that could be derived from it
- it is not possible to derive any matrix from itself in a non-empty derivation

Each of the rewrite rules applies to exactly one clause in the matrix, there is no interaction between clauses. Therefore we know that at any stage of a derivation, for each clause Σ' there is a clause Σ in the original matrix that Σ' was derived from. On the other hand, Σ' can only be a subset of some finite set determined by Σ.

23. Definition (σ) If Σ is a clause, let

$$\sigma(\Sigma) = \mathfrak{P}(\Sigma \cup \{\tau_1 \approx \tau_2 \mid \tau_1, \tau_2 \in \textsc{Term}(\Sigma) \cup \{:\}\} \\ \cup \{\tau \sim \varphi \mid \tau \in \textsc{Term}(\Sigma) \cup \{:\} \text{ and } \varphi \in \mathfrak{V}\})$$

If Σ is finite, then $\sigma(\Sigma)$ is finite. If Γ is a matrix, let

$$\sigma(\Gamma) = \bigcup\{\sigma(\Sigma) \mid \Sigma \in \Gamma\}$$

If Γ is finite, then $\sigma(\Gamma)$ is finite.

24. Lemma If Γ, Γ' are matrices and Γ' was derived from Γ in a finite number of steps, then $\Gamma' \subseteq \sigma(\Gamma)$

Proof: As noted above, each Σ' in Γ' was derived from a unique Σ in Γ. Furthermore, by inspection of the rules we see that for each negative literal δ, each $\tau, \tau_1, \tau_2 \in \mathcal{T}$ and each $\varphi \in \mathfrak{V}$,

if $\delta \in \Sigma'$ then $\delta \in \Sigma$,

if $\tau_1 \approx \tau_2 \in \Sigma'$ then $\tau_1, \tau_2 \in \textsc{Term}(\Sigma) \cup \{:\}$, and

if $\tau \sim \varphi \in \Sigma'$ then $\tau \in \textsc{Term}(\Sigma) \cup \{:\}$ and $\varphi \in \mathfrak{V}$.

Thus, $\Sigma' \in \sigma(\Sigma)$.

Thus, $\Gamma' \subseteq \sigma(\Gamma)$. ∎

G-spaces and G-relations

To show that there can be no cycles in the computation, we'll use the notion of a *G-relation* kindly provided by Paul King (1993a). G-relations have the property of being irreflexive, and by showing that derivations in the rewrite system are G-relations, we can prove that there are no cycles in derivations.

25. Definition A **G-space** is a pair $\langle S, \mu \rangle$ where

S is a set, and
μ is a function from S to the natural numbers.

26. Definition A **G-relation** on a G-space $\langle S, \mu \rangle$ is a binary relation $\sim$ between $\mathfrak{P}(S)$ and $\mathfrak{P}(S)$, where

if $X \sim Y$ then for some M, A and $B \subseteq S$,
$A \neq \emptyset$
$M \cap A = \emptyset$
$M \cap B = \emptyset$
$M \cup A = X$
$M \cup B = Y$, and
for each $b \in B$, there is some $a \in A$ with $\mu(a) < \mu(b)$.

27. Proposition Let $\sim$ be a G-relation on a G-space $\langle S, \mu \rangle$. Then there is no $X \subseteq S$ with $X \sim X$, i.e. G-relations are irreflexive.

Proof: By contradiction.
Let $X \subseteq S$ and $X \sim X$.
$\Longrightarrow \exists M, A, B \subseteq S : A \neq \emptyset, M \cap A = \emptyset, M \cap B = \emptyset, M \cup A = M \cup B$, and
$\forall b \in B \exists a \in A : \mu(a) < \mu(b)$
$\Longrightarrow \exists A \subseteq S : A \neq \emptyset$ and $\forall b \in A \exists a \in A : \mu(a) < \mu(b)$
$\Longrightarrow$ Contradiction
Therefore, there is no $X \subseteq S, X \sim X$. ∎

28. Definition The composition of the binary relation $\sim$ between X and Y and the binary relation $\simeq$ between Y and Z is the binary relation $\cong$ between X and Z where

$x \cong z$, iff for some $y, x \sim y$ and $y \simeq z$.

29. Proposition For each G-relation $\sim$ and $\simeq$ on a G-space $\langle S, \mu \rangle$, if $\cong$ is the composition of $\sim$ and $\simeq$, then $\cong$ is a G-relation on G-space $\langle S, \mu \rangle$.

Proof: Let $X, Z \subseteq S$ and $X \cong Z$.
$\Longrightarrow \exists Y \subseteq S : X \sim Y, Y \simeq Z$
$\Longrightarrow \exists M, N, A, B, C, D \subseteq S :$
$A \neq \emptyset, C \neq \emptyset,$
$M \cap A = \emptyset, N \cap C = \emptyset,$
$M \cap B = \emptyset, N \cap D = \emptyset,$
$M \cup A = X, N \cup D = Z,$
$M \cup B = N \cup C = Y,$
$\forall b \in B \exists a \in A : \mu(a) < \mu(b),$

$$
\begin{aligned}
& \forall d \in D \exists c \in C : \mu(c) < \mu(d) \\
\Longrightarrow\ & \exists M, N, A, B, C, D \subseteq S : \\
& \quad (A \cup (M \cap C)) \neq \emptyset, \\
& \quad (M \cap N) \cap (A \cup (M \cap C)) = \emptyset, \\
& \quad (M \cap N) \cap (D \cup (N \cap B)) = \emptyset, \\
& \quad (M \cap N) \cup (A \cup (M \cap C)) = X, \\
& \quad (M \cap N) \cup (D \cup (N \cap B)) = Z, \\
& \quad \forall f \in (D \cup (N \cap B)) \exists e \in (A \cup (M \cap C)) : \mu(e) < \mu(f) \\
\Longrightarrow\ & \exists O, E, F \subseteq S : \\
& \quad E \neq \emptyset, \\
& \quad O \cap E = \emptyset, \\
& \quad O \cap F = \emptyset, \\
& \quad O \cup E = X, \\
& \quad O \cup F = Z, \\
& \quad \forall f \in F \exists e \in E : \mu(e) < \mu(f).
\end{aligned}
$$

Thus, $\cong$ is a G-relation on $\langle S, \mu \rangle$. ■

30. Definition If $\mathfrak{A}$ is a set of attributes and $\mathfrak{V}$ is a set of varieties, then $\mathsf{Form}(\mathfrak{A}, \mathfrak{V})$ is the set of clauses over $\mathfrak{A}$ and $\mathfrak{V}$.

31. Lemma If $\mathfrak{A}$ is a set of attributes ,$\mathfrak{V}$ is a set of varieties and for any $X \in \mathsf{Form}(\mathfrak{A}, \mathfrak{V}), \mu(X) = |X|$, then $\langle \mathsf{Form}(\mathfrak{A}, \mathfrak{V}), \mu \rangle$ is a G-space.

32. Definition ($\vdash^1$) $\vdash^1\ \subset \mathsf{Form}(\mathfrak{A}, \mathfrak{V}) \times \mathsf{Form}(\mathfrak{A}, \mathfrak{V})$ is a binary relation on matrices s.t. if Γ, Γ' are matrices, then $\Gamma \vdash^1 \Gamma'$ iff Γ' is the result of having applied exactly one rewrite rule to Γ.

33. Proposition $\vdash^1$ is a G-relation.

Proof: By case by case analysis of the rewrite rules.

a) Suppose $\Gamma \vdash^1 \Gamma'$ by application of rule (0) – (4), (6) or (9). Then there are clauses $\Sigma \in \Gamma$ and $\Sigma' \in \Gamma'$ s.t.

$\mu(\Sigma') = \mu(\Sigma) + 1$ and

$\Gamma \backslash \{\Sigma\} = \Gamma' \backslash \{\Sigma'\}$

b) Suppose $\Gamma \vdash^1 \Gamma'$ by application of rule (5). Then there is a clause $\Sigma \in \Gamma$ and a set of clauses $\Gamma'' \subseteq \Gamma'$ s.t.

$\mu(\Sigma') = \mu(\Sigma) + 1$ for each $\Sigma' \in \Gamma''$ and

$\Gamma \backslash \{\Sigma\} = \Gamma' \backslash \Gamma''$

c) Suppose $\Gamma \vdash^1 \Gamma'$ by application of rule (7), (8) or (10). Then there is a clause $\Sigma \in \Gamma$ s.t.

$\Gamma \backslash \{\Sigma\} = \Gamma'$

Therefore, $\models^1$ is a G-relation. ■

The notion of derivation in a single step can be generalized to derivation in n steps. Since we know that G-relations are closed under composition, $\models^n$ is a G-relation for each n and no n-step derivation can contain cycles.

Thus, for any matrix Γ, only a finite number of other matrices can be derived from it. And since no matrix can be derived from itself, every derivation will terminate after a finite number of steps.

The statement on termination deserves a further note, and it has to do with rewrite rule (6). This rule says approximately, if there is a clause in the matrix that contains a term equation literal for which there is no variety assignment literal, then add a set of clauses each member of which contains the old clause plus some variety assignment literal. The whole set exhausts all possible variety assignment literals for the given term. Thus we must be sure that there is only a *finite* number of varieties. Hence our demand in the definition of a signature that the set of varieties be finite. It is clear that some finiteness restriction must be given implicitely with rewrite rule (6), because otherwise the algorithm may never terminate. But there may be another way to do so. In the given version, the rewrite rule is copletely ignorant in relation to approriateness restrictions. That means most of the variety assignment literals introduced varities which are plain inappropriate for the term in question. This is, of course, no problem for the algorithm as such, since there is the rewrite rule (9) which eliminates clauses that contain appropriateness restriction violating literals. But one could obviously redefine rewrite rule (6) in such a way, that it takes the appropriateness restrictions into account.[4] This also enables us to recast the finiteness restrictions. Since rule (6) would then consider only appropriate varieties, all that needed to be ensured is that the set of varieties appropriate for a certain type be finite. Remember that a signature is a total function $\mathfrak{S} : \mathfrak{V} \times \mathfrak{A} \to \mathfrak{P}(\mathfrak{V})$. We would have to demand that $\mathfrak{P}(\mathfrak{V})$ be restricted to the set of *finite* subsets of $\mathfrak{V}$. Unfortunately the restriction that for the root symbol : there can be only a finite number of varieties appropriate would have to stay in place. This can not be stated via the appropriateness conditions for the general case. This fact is rather sad. We wanted to allow for inifintely many varieties to categorise the objects in our universe. This works for all objects exept for the one that models the root node. But the root node is the least specified thing we have. Thus the thing where it would make most sense to think that it can fall under infinitely many categories just can't do so for technical reasons.

8.2 Denotation Preservation

To see that the application of a rule does not change the denotation of a set, we check the rules one by one.

[4] And any serious trial to implement the algorithm will certainly do so.

1. $[\![:\approx:]\!] = \mathcal{U}$, therefore $[\![\Sigma]\!] \cap [\![:\approx:]\!] = [\![\Sigma]\!]$ for any clause Σ.

2. $[\![\tau_1 \approx \tau_2]\!] = [\![\tau_2 \approx \tau_1]\!]$,
therefore $[\![\Sigma]\!] \cap [\![\tau_1 \approx \tau_2]\!] = [\![\Sigma]\!] \cap [\![\tau_1 \approx \tau_2]\!] \cap [\![\tau_2 \approx \tau_1]\!]$.

3. $[\![\tau_1 \approx \tau_2, \tau_2 \approx \tau_3]\!] = [\![\tau_1 \approx \tau_2, \tau_2 \approx \tau_3, \tau_1 \approx \tau_3]\!]$ by transitivity of path equations.

4. $$\begin{aligned} [\![\tau\sigma \approx \tau\sigma]\!] &= \{u \mid [\tau\sigma] \text{ is defined on } u\} \\ &= \{u \mid \mathfrak{A}(a) \circ [\tau] \text{ is defined on } u\} \\ &\subseteq \{u \mid [\tau] \text{ is defined on } u\} \\ &= [\![\tau \approx \tau]\!]. \end{aligned}$$
Therefore $[\![\tau\sigma \approx \tau\sigma]\!] = [\![\tau\sigma \approx \tau\sigma, \tau \approx \tau]\!]$

5. $$\begin{aligned} &[\![\tau_1 \approx \tau_2, \tau_1\sigma \approx \tau_1\sigma, \tau_2\sigma \approx \tau_2\sigma]\!] \\ &\quad= \{u \mid [\tau_1](u)\!\downarrow, [\tau_2](u)\!\downarrow, [\tau_1\sigma](u)\!\downarrow, [\tau_2\sigma](u)\!\downarrow \text{ and} \\ &\qquad\quad [\tau_1](u) = [\tau_2](u)\} \\ &\quad= \{u \mid [\tau_1](u)\!\downarrow, [\tau_2](u)\!\downarrow, [\tau_1\sigma](u)\!\downarrow, [\tau_2\sigma](u)\!\downarrow, \\ &\qquad\quad [\tau_1](u) = [\tau_2](u) \text{ and } [\tau_1\sigma](u) = [\tau_2\sigma](u)\} \\ &\quad= [\![\tau_1 \approx \tau_2, \tau_1\sigma \approx \tau_1\sigma, \tau_2\sigma \approx \tau_2\sigma, \tau_1\sigma \approx \tau_2\sigma]\!] \end{aligned}$$

6. $$\begin{aligned} [\![\tau \approx \tau]\!] &= \{u \mid [\tau](u)\!\downarrow\} \\ &= \{u \mid [\tau](u)\!\downarrow \text{ and for some } \varphi \in \mathfrak{V},\ V([\tau](u)) = \varphi\} \\ &= \bigcup_{\varphi \in \mathfrak{V}} \{u \mid [\tau](u)\!\downarrow \text{ and } V([\tau](u)) = \varphi\} \\ &= [\![\bigvee_{\varphi \in \mathfrak{V}} (\tau \approx \tau, \tau \sim \varphi)]\!] \end{aligned}$$

7. $$\begin{aligned} [\![\tau \sim \varphi]\!] &= \{u \mid [\tau](u)\!\downarrow \text{ and } V([\tau](u)) = \varphi\} \\ &= [\![\tau \sim \varphi, \tau \approx \tau]\!] \end{aligned}$$

8. If $\varphi_1 \neq \varphi_2$, then
$$\begin{aligned} &[\![\tau_1 \approx \tau_2, \tau_1 \sim \varphi_1, \tau_2 \sim \varphi_2]\!] \\ &\quad= \{u \mid [\tau_1](u)\!\downarrow, [\tau_2](u)\!\downarrow, [\tau_1](u) = [\tau_2](u), \\ &\qquad\quad V([\tau_1](u)) = \varphi_1 \text{ and } V([\tau_2](u)) = \varphi_2\} \\ &\quad= \emptyset, \end{aligned}$$
because every object is of one and only one variety.

9. If $\varphi_2 \notin \mathfrak{S}\langle\varphi_1, \alpha\rangle$ then
$$\begin{aligned} [\![\tau \sim \varphi_1, \tau\alpha \sim \varphi_2]\!] &= \{u \mid [\tau_1](u)\!\downarrow, [\tau_2](u)\!\downarrow, V([\tau_1](u)) = \varphi_1 \\ &\qquad \text{and } V([\tau_2](u)) = \varphi_2\} \\ &= \emptyset, \end{aligned}$$
by condition S1 on structures.

10. If $\mathfrak{S}\langle\varphi,\alpha\rangle \neq \emptyset$ then
$$\begin{aligned}
[\![\tau \sim \varphi]\!] &= \{u \mid [\tau](u)\!\downarrow \text{ and } V([\tau](u)) = \varphi\} \\
&= \{u \mid [\tau](u)\!\downarrow, V([\tau](u)) = \varphi \text{ and } [\tau\alpha](u)\!\downarrow\} \\
&\qquad\qquad\qquad\qquad \text{(condition S2 on structures)} \\
&= [\![\tau \sim \varphi, \tau\alpha \approx \tau\alpha]\!]
\end{aligned}$$

11. If δ is a positive literal, then $[\![\delta, \neg\delta]\!] = [\![\delta]\!] \cap \mathcal{U} \backslash [\![\delta]\!] = \emptyset$.

8.3 Partial Correctness

It is now clear that the algorithm terminates, and that it preserves the denotation of the sets of clauses it is applied to. But it remains to be seen that the algorithm stops iff the set is in normal form. To proof this, we need show that

1. If a set is not in normal form, then we can apply a rule (left to right), and
2. If a set *is* in normal form, then no rule can be applied to it (right to left)

Ad 1: If a set of clauses is not in normal form, then there is at least on clause that violates at least one of the normal form conditions. We could go through the normal form conditions one by one and show that whenever a set of clauses contains a clause violating a normal form condition, one of the rules of the algorithm applies to the set. We will spare the reader this trouble, however, as this is quite obviously true. The algorithm was set up exactly such that there is a one to one correspondence between the conditions and the rules.

Ad 2: A similar argument can be made to show that no rule can apply to a matrix in normal form; whenever a rule applies to a set, it is not in normal form. This exactly as obvious as the other direction.

III Proof of Satisfiability

9 Normal Forms and Class Operators

We will give two definitions of two normal forms for descriptions and a series of class operators which operate on descriptions. The first normal form, called normal, is already introduced as the normal form the algorithm in the last section delivered. It is thus a repetition of definition 21. The second normal form, called closed, is in a sense an extended version of the first in that the conditions and requirements are sharpened. We use two normal forms simply for a division of labour. The reader may remember that two important things must be shown in order to show that any given description is either inconsistent or satisfiable. First one must bring the description into a format which enables one to find a model and second one must find a model – and proof that this is a model, of course. Of course one could try and build a model directly from any description in any format. But that is extremely cumbersome to do. And it wouldn't save any work. Because the issues we face must reappear under any method. We have the advantage of separating blocks of problems. The second normal form, the closed sets, is ideal for finding a model. We can almost immediately go ahead and construct one. The first normal form can be used as the output of the algorithm without the algorithm becoming too complicated and, what is more important, without making the proofs for termination, denotation preservation and partial correctness huge and incomprehensible. Thus we have a gap between the first normal form and the second one. The whole next section which is definitely the heart of the research is devoted to bridging this gap. Unfortunately this is by no means easy or obvious. We will define a number of class operator which enable us to embed a normal set in a closed set. That is we extend the normal set in such a way that the resulting set is in closed form and contains the original normal set as a subset.

34. Definition (Normal set) A set Σ of literals is **normal** iff the following conditions hold ($\tau, \tau_1, \tau_2 \in \mathfrak{T}; \varphi, \varphi', \varphi_1, \varphi_2 \in \mathfrak{V}; \alpha \in \mathfrak{A}; \sigma \in \mathfrak{A}^*$)

(1) $:\approx: \in \Sigma$;
(2) if $\tau_1 \approx \tau_2 \in \Sigma$ then $\tau_2 \approx \tau_1 \in \Sigma$; (symmetry)
(3) if $\tau_1 \approx \tau_2, \tau_2 \approx \tau_3 \in \Sigma$ then $\tau_1 \approx \tau_3 \in \Sigma$; (transitivity)
(4) if $\tau\sigma \approx \tau\sigma \in \Sigma$ then $\tau \approx \tau \in \Sigma$; (subterm reflexivity)
(5) if $\tau_1 \approx \tau_2, \tau_1\sigma \approx \tau_1\sigma, \tau_2\sigma \approx \tau_2\sigma \in \Sigma$ then $\tau_1\sigma \approx \tau_2\sigma \in \Sigma$;
(6) if $\tau \approx \tau \in \Sigma$ then for some $\varphi \in \mathfrak{V}, \tau \sim \varphi \in \Sigma$;
(7) if for some $\varphi \in \mathfrak{V}, \tau \sim \varphi \in \Sigma$ then $\tau \approx \tau \in \Sigma$;
(8) if $\tau_1 \approx \tau_2 \in \Sigma, \tau_1 \sim \varphi_1 \in \Sigma, \tau_2 \sim \varphi_2 \in \Sigma$ then $\varphi_1 = \varphi_2$;
(9) if $\tau \sim \varphi_1 \in \Sigma, \tau\alpha \sim \varphi_2 \in \Sigma$ then $\varphi_2 \in \mathfrak{S}\langle\varphi_1, \alpha\rangle$;
(10) if $\tau \sim \varphi \in \Sigma, \tau\alpha \in \textsc{Term}(\Sigma), \mathfrak{S}\langle\varphi, \alpha\rangle \neq \emptyset$ then $\tau\alpha \approx \tau\alpha \in \Sigma$;
(11) if $\neg\delta \in \Sigma$ then $\delta \notin \Sigma$.

35. Definition (Closed set) A set Σ of literals is **closed** iff ($\tau, \tau_1, \tau_2 \in \mathfrak{T}; \varphi, \varphi', \varphi_1, \varphi_2 \in \mathfrak{V}; \alpha \in \mathfrak{A}; \sigma \in \mathfrak{A}^*$)

$:\approx: \in \Sigma$;
if $\tau_1 \approx \tau_2 \in \Sigma$ then $\tau_2 \approx \tau_1 \in \Sigma$; (symmetry)
if $\tau_1 \approx \tau_2, \tau_2 \approx \tau_3 \in \Sigma$ then $\tau_1 \approx \tau_3 \in \Sigma$; (transitivity)
if $\tau\sigma \approx \tau\sigma \in \Sigma$ then $\tau \approx \tau \in \Sigma$; (subterm reflexivity)
if $\tau_1 \approx \tau_2, \tau_1\sigma \approx \tau_1\sigma \in \Sigma$ then $\tau_1\sigma \approx \tau_2\sigma \in \Sigma$;
if $\tau \approx \tau \in \Sigma$ then for some $\varphi \in \mathfrak{V}, \tau \sim \varphi \in \Sigma$;
if for some $\varphi \in \mathfrak{V}, \tau \sim \varphi \in \Sigma$ then $\tau \approx \tau \in \Sigma$;
if $\tau_1 \approx \tau_2 \in \Sigma, \tau_1 \sim \varphi_1 \in \Sigma, \tau_2 \sim \varphi_2 \in \Sigma$ then $\varphi_1 = \varphi 2$;
if $\tau \sim \varphi_1 \in \Sigma, \tau\alpha \sim \varphi_2 \in \Sigma$ then $\varphi_2 \in \mathfrak{S}\langle\varphi_1, \alpha\rangle$;
if $\tau \sim \varphi \in \Sigma, \mathfrak{S}\langle\varphi, \alpha\rangle \neq \emptyset$ then $\tau\alpha \approx \tau\alpha \in \Sigma$;
if $\neg\delta \in \Sigma$ then $\delta \notin \Sigma$.

Since they appear to be small, please note the differences between these two definitions carefully. Only two pairs of lines differ slightly. The first difference occurs in line (5), which concerns right invariance of path equations. Once two paths are made equal, all extensions of one path must be present on the other as well and must be equal to their counterparts. Thinking in terms of feature *structures* this seems to be no issue. But it isn't an issue only due to the properties of the representation normally underlying feature structures. In graph theory, the demanded right invariance holds by definition of the theory. But we are working with feature *descriptions* instead. A literal doesn't "know" anything about other literals in its set. We can *interpret* two or more literals as belonging together, but there is no way to express this in our representation. Thus right invariance doesn't fall out for free but must rather be forced upon. In the definition of a normal set, we do not demand full right invariance, rather something weaker: If two paths are equated and both grow with the same attributes then the grown paths must be equated, too. In the definition of a closed set, full right invariance is demanded: If two paths are equated and one of them grows then the other must grow the same way and must be equated to the first grown one.

The second difference can be spoted in condition (10). It can be stated using a notion introduced by Carpenter (1992): While closed sets are *totally well-typed*, normal sets may not be so. A feature structure or description is *well-typed*, if the features (attributes) which are present on a type (variety) are licensed by the appropriateness conditions. There may be no inappropriate features for a type. This is demanded in condition (9) equally for both normal and closed sets. A feature structure or description is *totally well-typed*, if every feature (attribute) that is appropriate for a type (variety) is indeed realised at this type (variety). Every appropriate feature must be present. This is exactly what is demanded in condition (10) for closed sets. For a normal set, this condition is weakened. This weakening cannot be expressed in terms of feature structures, because their underlying representation doesn't allow the cases reflected in the weaker condition (10). The presence of a literal in the set stating the presence of an attribute which is appropriate is demanded only if this attribute is introduced already by some other literal. The question is what could this "other literal" be. If it were any positive literal, than all the other conditions would enforce the presence of the literal stating the definedness of the grown path already. Thus the "other literal" can only be a negative one. To achieve consistency of negation, all we demand is that there must not be contradictory literals. That is what condition (11) says. In order to make this small demand be sufficient for full consistency we must be sure that the necessary positive literals to compare a negative literal with are always present. Condition (10) of a normal set takes over part of this job.

Where do feature structures fit in this picture? It is already stated above that a closed set describes a totally well-typed feature structure. From the comments above it is also clear, that a normal set of literals may describe something that is not at all a feature structure. There are properties which come with the notion of a feature structure that are not demanded for a normal set, for example right invariance in its full form. Thus our notion of a normal set is weaker. It should therefore be easier to see, whether a description is normal than trying to find out, if it matches the requirements for a feature structure. Even more, it seems that the normal set is weakest normal form one can give. If one weakens or gives up one of the conditions demanded then it should be possible to come up with a description which obeys to that weaker normal form, but is *unsatisfiable*. Of course such a normal form is useless. If one wanted a definition of feature structures using descriptions, what would be needed were something in between a normal set and a closed set: One could take the definition of a closed set and replace the requirement for totally well-typedness in line (10) with the condition (10) given for normal sets. We cannot just forget about condition (10), because otherwise inconsistencies may appear. Unfortunately this statement cannot be explained by a few hints. It turns out when checking the proofs for lemma 54 and 55.

Now we can define the class operators. Let $\textsc{Choose}$: $\mathfrak{P}(\mathfrak{V} \setminus \emptyset) \to \mathfrak{V}$ be a choice function, that is $\textsc{Choose}(M) \in M$.

36. Definition Let Σ be a set of literals. Let $\tau, \tau_1, \tau_2, \tau_3, \kappa \in \mathfrak{T}, \alpha \in \mathfrak{A}, \varphi, \varphi' \in \mathfrak{V}$. We define

$$\begin{array}{lll}
\Pi_1(\Sigma) & := & \{\ \tau_1\alpha \approx \tau_2\alpha \ \mid \ \tau_1 \approx \tau_2 \in \Sigma, \\
& & \qquad\qquad\quad \tau_1 \sim \varphi \in \Sigma, \\
& & \qquad\qquad\quad \tau_2 \sim \varphi \in \Sigma, \\
& & \qquad\qquad\quad \mathfrak{S}\langle\varphi, \alpha\rangle \neq \emptyset, \\
& & \qquad\qquad\quad \tau_1\alpha \approx \tau_2\alpha \notin \Sigma\} \\
\Pi_2(\Sigma) & := & \{\tau\alpha \sim \varphi' \ \mid \ \tau \sim \varphi \in \Sigma, \\
& & \qquad\qquad\quad \mathfrak{S}\langle\varphi, \alpha\rangle \neq \emptyset, \\
& & \qquad\qquad\quad \tau\alpha \approx \tau\alpha \notin \Sigma \\
& & \qquad\qquad\quad \exists\kappa(\kappa \approx \tau \in \Sigma, \kappa\alpha \sim \varphi' \in \Sigma)\} \\
\Pi_3(\Sigma) & := & \{\tau\alpha \sim \textsc{Choose}(\mathfrak{S}\langle\varphi, \alpha\rangle) \ \mid \ \tau \sim \varphi \in \Sigma, \\
& & \qquad\qquad\qquad\qquad\qquad \mathfrak{S}\langle\varphi, \alpha\rangle \neq \emptyset, \\
& & \qquad\qquad\qquad\qquad\qquad \tau\alpha \approx \tau\alpha \notin \Sigma \\
& & \qquad\qquad\qquad\qquad\qquad \neg(\exists\kappa\exists\varphi'(\kappa \approx \tau \in \Sigma, \kappa\alpha \sim \varphi' \in \Sigma))\} \\
\Pi(\Sigma) & := & \Sigma \cup \Pi_1(\Sigma) \cup \Pi_2(\Sigma) \cup \Pi_3(\Sigma) \\
\mathrm{T}(\Sigma) & := & \Sigma \cup \{\tau_1 \approx \tau_3 \mid \tau_1 \approx \tau_2 \in \Sigma, \tau_2 \approx \tau_3 \in \Sigma, \tau_1 \approx \tau_3 \notin \Sigma\} \\
\mathrm{T}_0(\Sigma) & := & \Sigma \\
\mathrm{T}_{i+1}(\Sigma) & := & \mathrm{T}(\mathrm{T}_i(\Sigma)) \\
\mathrm{T}^*(\Sigma) & := & \bigcup_{i\in\omega} \mathrm{T}_i(\Sigma) \\
\Pi^*(\Sigma) & := & \mathrm{T}^*(\Pi(\Sigma)) \\
\Xi_0(\Sigma) & := & \Sigma \\
\Xi_{i+1}(\Sigma) & := & \Pi^*(\Xi_i(\Sigma)) \\
\Xi(\Sigma) & := & \bigcup_{i\in\omega} \Xi_i(\Sigma)
\end{array}$$

Although the exact operations of these class operators will become clear only in the subsequent sequence of lemmata and their proofs, we try to explain a bit what the individual operators are designed to do. Π makes paths grow. Remember that a closed set is totally well-typed, that is every appropriate attribute must be present. Π adds appropriate attributes one at a time. Π_1 takes over the job of adding one new attribute for each variety where this attribute is appropriate but not jet present. If we add the attribute path, we must assign a variety to it. This is done by Π_2 or Π_3. As one can see from the conditions of application only one of the two can be relevant at any given time. In the case of Π_2 we let a path grow and equate it to a previously existing one. In this case it must necessarily receive that variety that the previously existing one already carries. Cases like these occur, when we stepwise fill in paths for right invariance. Π_3 takes care of those cases where there is no longer path we could connect the newly grown path to. This happens when for a specific sub part of the description right invariance is already fullfiled. In this case no variety is prescribed; thus we are free to choose one out of those stated in the appropriateness conditions. Π sums up these three parts and adds the old description.

After an application of Π to a description which was previously closed under transitivity, transitivity may no longer hold. Thus we must reestablish it. T obviously adds transitivity for one pair of literals where it wasn't given before. T^* constructs the closure under transitivity in a standard way making use of the T_i. Although we construct full transitive closure, it turns out, that this is not really needed. As a consequence of some of the lemmata in the next section, especially lemma 46, one could show that full transitivity is reached already after two applications of T; that is $\mathrm{T}_3(\Sigma) = \mathrm{T}_2(\Sigma)$, if Σ is normal. But this is just a side issue which we will make no use of.

Π^* connects the one step growing and the transitive closure. It will be one of the important results, that, if Σ is normal, then so is $\Pi^*(\Sigma)$. This means we grow the descriptions in a rather controlled way. Ξ is the closure of Π^*, we iterate the application of Π^* until we reach a fixed point. The final proposition of the next section states, that if a description Σ is normal, then $\Xi(\Sigma)$ is closed. Thus we can bridge the gap between the normal sets and the closed sets.

Although it will be stated explicitly each time, we want to point out here already that in all of the following lemmata and their proofs until the end of the next section we always assume Σ to be a normal set of literals.

We begin with a lemma which shows that Π_2 assigns a unique variety to each path if, it is applied to a normal set.

37. Lemma Let Σ be a normal set. Let $\tau \sim \varphi \in \Pi_2(\Sigma)$ (with $\tau \in \mathfrak{T}$ and $\varphi \in \mathfrak{V}$). Then there exists no $\varphi' \in \mathfrak{V}$ different from φ with $\tau \sim \varphi' \in \Pi_2(\Sigma)$.

Proof: Let $\tau \sim \varphi, \tau \sim \varphi' \in \Pi_2(\Sigma)$.

$$\begin{aligned}
\Longrightarrow\ & (\exists \tau_1 \in \textsc{Term}(\Sigma), \alpha \in \mathfrak{A}, \varphi_1 \in \mathfrak{V} : \tau = \tau_1\alpha, \tau_1 \sim \varphi_1 \in \Sigma,\\
&\quad \mathfrak{S}\langle \varphi_1, \alpha\rangle \neq \emptyset, \tau \approx \tau \notin \Sigma, \exists \kappa_1 \in \textsc{Term}(\Sigma) : \kappa_1 \approx \tau_1 \in \Sigma, \kappa_1\alpha \sim \varphi \in \Sigma)\\
&\quad \wedge (\exists \tau_2 \in \textsc{Term}(\Sigma), \beta \in \mathfrak{A}, \varphi_2 \in \mathfrak{V} : \tau = \tau_2\beta, \tau_2 \sim \varphi_2 \in \Sigma,\\
&\quad\quad \mathfrak{S}\langle \varphi_2, \beta\rangle \neq \emptyset, \tau \approx \tau \notin \Sigma, \exists \kappa_2 \in \textsc{Term}(\Sigma) : \kappa_2 \approx \tau_2 \in \Sigma, \kappa_2\beta \sim \varphi' \in \Sigma)\\
&&\text{(Def. of } \Pi_2)\\
\Longrightarrow\ & (\exists \tau_1 \in \textsc{Term}(\Sigma), \alpha \in \mathfrak{A}, \varphi_1 \in \mathfrak{V} : \tau = \tau_1\alpha, \tau_1 \sim \varphi_1 \in \Sigma,\\
&\quad \mathfrak{S}\langle \varphi_1, \alpha\rangle \neq \emptyset, \tau \approx \tau \notin \Sigma, \exists \kappa_1 \in \textsc{Term}(\Sigma) : \kappa_1 \approx \tau_1 \in \Sigma, \kappa_1\alpha \sim \varphi \in \Sigma\\
&\quad \wedge (\exists \tau_2 \in \textsc{Term}(\Sigma), \beta \in \mathfrak{A}, \varphi_2 \in \mathfrak{V} : \tau = \tau_2\beta, \tau_2 \sim \varphi_2 \in \Sigma,\\
&\quad\quad \mathfrak{S}\langle \varphi_2, \beta\rangle \neq \emptyset, \tau \approx \tau \notin \Sigma, \exists \kappa_2 \in \textsc{Term}(\Sigma) : \kappa_2 \approx \tau_2 \in \Sigma, \kappa_2\beta \sim \varphi' \in \Sigma\\
&\quad \wedge \tau_1 = \tau_2, \alpha = \beta)) && (\alpha, \beta \in \mathfrak{A})\\
\Longrightarrow\ & \exists \tau_1 \in \textsc{Term}(\Sigma), \alpha \in \mathfrak{A}, \varphi_1 \in \mathfrak{V} : \tau = \tau_1\alpha, \tau_1 \sim \varphi_1 \in \Sigma, \mathfrak{S}\langle \varphi_1, \alpha\rangle \neq \emptyset,\\
&\quad \exists \kappa_1 \in \textsc{Term}(\Sigma) : \kappa_1 \approx \tau_1 \in \Sigma, \kappa_1\alpha \sim \varphi \in \Sigma,\\
&\quad \exists \varphi_2 \in \mathfrak{V} : \tau_1 \sim \varphi_2 \in \Sigma, \mathfrak{S}\langle \varphi_2, \alpha\rangle \neq \emptyset, \exists \kappa_2 \in \textsc{Term}(\Sigma) :\\
&\quad\quad \kappa_2 \approx \tau_1 \in \Sigma, \kappa_2\alpha \sim \varphi' \in \Sigma\\
\Longrightarrow\ & \exists \tau_1 \in \textsc{Term}(\Sigma), \alpha \in \mathfrak{A}, \varphi_1 \in \mathfrak{V} : \tau = \tau_1\alpha, \tau_1 \sim \varphi_1 \in \Sigma, \mathfrak{S}\langle \varphi_1, \alpha\rangle \neq \emptyset,\\
&\quad \exists \kappa_1 \in \textsc{Term}(\Sigma) : \kappa_1 \approx \tau_1 \in \Sigma, \kappa_1\alpha \sim \varphi \in \Sigma,\\
&\quad \exists \varphi_2 \in \mathfrak{V} : \tau_1 \sim \varphi_2 \in \Sigma, \mathfrak{S}\langle \varphi_2, \alpha\rangle \neq \emptyset, \exists \kappa_2 \in \textsc{Term}(\Sigma) :\\
&\quad\quad \kappa_2 \approx \tau_1 \in \Sigma, \kappa_2\alpha \sim \varphi' \in \Sigma,\\
&\quad \varphi_1 = \varphi_2 && (\Sigma \text{ normal})
\end{aligned}$$

$$\begin{aligned}
\Longrightarrow\ & \exists \tau_1 \in \textsc{Term}(\Sigma), \alpha \in \mathfrak{A}, \varphi_1 \in \mathfrak{V} : \tau = \tau_1\alpha, \tau_1 \sim \varphi_1 \in \Sigma, \mathfrak{S}\langle \varphi_1, \alpha\rangle \neq \emptyset, \\
& \quad \exists \kappa_1 \in \textsc{Term}(\Sigma) : \kappa_1 \approx \tau_1 \in \Sigma, \kappa_1\alpha \sim \varphi \in \Sigma, \\
& \quad \exists \kappa_2 \in \textsc{Term}(\Sigma) : \kappa_2 \approx \tau_1 \in \Sigma, \kappa_2\alpha \sim \varphi' \in \Sigma \\
\Longrightarrow\ & \exists \tau_1 \in \textsc{Term}(\Sigma), \alpha \in \mathfrak{A}, \varphi_1 \in \mathfrak{V} : \tau = \tau_1\alpha, \tau_1 \sim \varphi_1 \in \Sigma, \mathfrak{S}\langle \varphi_1, \alpha\rangle \neq \emptyset, \\
& \quad \exists \kappa_1 \in \textsc{Term}(\Sigma) : \kappa_1 \approx \tau_1 \in \Sigma, \kappa_1\alpha \sim \varphi \in \Sigma, \\
& \quad \exists \kappa_2 \in \textsc{Term}(\Sigma) : \kappa_2 \approx \tau_1 \in \Sigma, \kappa_2\alpha \sim \varphi' \in \Sigma, \kappa_1 \approx \kappa_2 \in \Sigma \qquad (\Sigma \text{ normal}) \\
\Longrightarrow\ & \exists \tau_1 \in \textsc{Term}(\Sigma), \alpha \in \mathfrak{A}, \varphi_1 \in \mathfrak{V} : \tau = \tau_1\alpha, \tau_1 \sim \varphi_1 \in \Sigma, \mathfrak{S}\langle \varphi_1, \alpha\rangle \neq \emptyset, \\
& \quad \exists \kappa_1 \in \textsc{Term}(\Sigma) : \kappa_1 \approx \tau_1 \in \Sigma, \kappa_1\alpha \sim \varphi \in \Sigma, \\
& \quad \exists \kappa_2 \in \textsc{Term}(\Sigma) : \kappa_2 \approx \tau_1 \in \Sigma, \kappa_2\alpha \sim \varphi' \in \Sigma, \kappa_1 \approx \kappa_2 \in \Sigma, \\
& \qquad \kappa_1\alpha \approx \kappa_2\alpha \in \Sigma \qquad (\Sigma \text{ normal}) \\
\Longrightarrow\ & \varphi = \varphi' \qquad (\Sigma \text{ normal})
\end{aligned}$$

■

10 Embedding a Normal Set in a Closed Set

This section contains the central work of the thesis. It shows that a description in normal form can be embedded "naturally" into a description in closed form. That means, we can extend the description in normal form not only in a way which doesn't distort its properties but moreover in a way which strengthens them.

The section is divided into two sub parts. The first is a large collection of technical lemmata. Some of these lemmata are of extreme technical nature and give almost no insight into the main problems. But they are needed in proofs of other lemmata. Some of these technical lemmata though give some insights or appear as weaker variants of the lemmata in the second sub part. This second sub part contains the main flow of the argument. We will show stepwise that after an application of Π^* to a normal set, the result of this application is still normal.

Technical lemmata

The first lemma gives an alternative definition for a normal set. We change only a minor point: In condition (5) of a normal set, we refer to whole attribute paths $\sigma \in \mathfrak{A}^*$. We will show that it is sufficient to refer to single attributes $\alpha \in \mathfrak{A}$.

38. Lemma A set Σ is normal, if and only if it satisfies the following conditions

(1) $:\approx: \in \Sigma$;
(2) if $\tau_1 \approx \tau_2 \in \Sigma$ then $\tau_2 \approx \tau_1 \in \Sigma$; (symmetry)
(3) if $\tau_1 \approx \tau_2, \tau_2 \approx \tau_3 \in \Sigma$ then $\tau_1 \approx \tau_3 \in \Sigma$; (transitivity)
(4) if $\tau\sigma \approx \tau\sigma \in \Sigma$ then $\tau \approx \tau \in \Sigma$; (subterm reflexivity)
(5) if $\tau_1 \approx \tau_2, \tau_1\alpha \approx \tau_1\alpha, \tau_2\alpha \approx \tau_2\alpha \in \Sigma$ then $\tau_1\alpha \approx \tau_2\alpha \in \Sigma$;
(6) if $\tau \approx \tau \in \Sigma$ then for some $\varphi \in \mathfrak{V}, \tau \sim \varphi \in \Sigma$;
(7) if for some $\varphi \in \mathfrak{V}, \tau \sim \varphi \in \Sigma$ then $\tau \approx \tau \in \Sigma$;
(8) if $\tau_1 \approx \tau_2 \in \Sigma, \tau_1 \sim \varphi_1 \in \Sigma, \tau_2 \sim \varphi_2 \in \Sigma$ then $\varphi_1 = \varphi_2$;
(9) if $\tau \sim \varphi_1 \in \Sigma, \tau\alpha \sim \varphi_2 \in \Sigma$ then $\varphi_2 \in \mathfrak{S}\langle\varphi_1, \alpha\rangle$;
(10) if $\tau \sim \varphi \in \Sigma, \tau\alpha \in \mathrm{TERM}(\Sigma), \mathfrak{S}\langle\varphi, \alpha\rangle \neq \emptyset$ then $\tau\alpha \approx \tau\alpha \in \Sigma$;
(11) if $\neg\delta \in \Sigma$ then $\delta \notin \Sigma$.

Proof: We note that only condition (5) is altered. Now, the original condition

if $\tau_1 \approx \tau_2, \tau_1\sigma \approx \tau_1\sigma, \tau_2\sigma \approx \tau_2\sigma \in \Sigma$ then $\tau_1\sigma \approx \tau_2\sigma \in \Sigma$ $\quad (\sigma \in \mathfrak{A}^*)$

obviously entails the new condition

if $\tau_1 \approx \tau_2, \tau_1\alpha \approx \tau_1\alpha, \tau_2\alpha \approx \tau_2\alpha \in \Sigma$ then $\tau_1\alpha \approx \tau_2\alpha \in \Sigma$ $\quad (\alpha \in \mathfrak{A})$.

Thus if a set Σ is normal, it will satisfy the new conditions above. Therefore what needs to be shown here is that the new condition (5) entails the old one as well. This is done by an induction on the length of the path $\sigma \in \mathfrak{A}^*$. We claim: If the conditions (1) – (11) above hold for a set Σ, then the following condition holds, too

if $\tau_1 \approx \tau_2, \tau_1\sigma \approx \tau_1\sigma, \tau_2\sigma \approx \tau_2\sigma \in \Sigma$ then $\tau_1\sigma \approx \tau_2\sigma \in \Sigma$ $\quad (\sigma \in \mathfrak{A}^*)$

Proof of the claim:
Base case: For $\sigma = \epsilon$ the condition is vacuously satisfied.

Step case: Let $\sigma = \sigma'\alpha$ for some $\sigma' \in \mathfrak{A}^*$ and some $\alpha \in \mathfrak{A}$.
Let further $\tau_1 \approx \tau_2, \tau_1\sigma \approx \tau_1\sigma, \tau_2\sigma \approx \tau_2\sigma \in \Sigma$.

$\Longrightarrow \tau_1 \approx \tau_2, \tau_1\sigma'\alpha \approx \tau_1\sigma'\alpha, \tau_2\sigma'\alpha \approx \tau_2\sigma'\alpha \in \Sigma$

$\Longrightarrow \tau_1 \approx \tau_2, \tau_1\sigma'\alpha \approx \tau_1\sigma'\alpha, \tau_2\sigma'\alpha \approx \tau_2\sigma'\alpha, \tau_1\sigma' \approx \tau_1\sigma', \tau_2\sigma' \approx \tau_2\sigma' \in \Sigma$ (Cond. (4))

$\Longrightarrow \tau_1 \approx \tau_2, \tau_1\sigma'\alpha \approx \tau_1\sigma'\alpha, \tau_2\sigma'\alpha \approx \tau_2\sigma'\alpha, \tau_1\sigma' \approx \tau_1\sigma', \tau_2\sigma' \approx \tau_2\sigma',$
$\tau_1\sigma' \approx \tau_2\sigma' \in \Sigma$ (Ind. hypothesis)

$\Longrightarrow \tau_1\sigma'\alpha \approx \tau_2\sigma'\alpha \in \Sigma.$ (Cond. (5))

Since the other conditions are unchanged, this proves that a set is normal, iff it satisfies the conditions above. ■

The statement of the following two lemmata is: T does not create new terms. All terms in $\Pi^*(\Sigma)$ are already present in $\Pi(\Sigma)$. We split up this statement into two lemmata, since we want to use both in subsequent proofs.

39. Lemma Let Σ be a normal set. Let $\tau_1 \approx \tau_2 \in \mathrm{T}_i(\Pi(\Sigma))$ (with $\tau_1, \tau_2 \in \mathfrak{T}$). Then $\tau_1 \approx \tau_1, \tau_2 \approx \tau_2 \in \Pi(\Sigma)$.

Proof: We prove the claim only for the literal $\tau_1 \approx \tau_1$, since an analogous argument can be made for $\tau_2 \approx \tau_2$, as can easily be seen. The proof goes by induction on i.

Base case: $i = 0 \Longrightarrow \tau_1 \approx \tau_2 \in \Pi(\Sigma)$.
Let $\tau_1 \approx \tau_2 \in \Pi(\Sigma)$.

$\Longrightarrow \tau_1 \approx \tau_2 \in \Sigma$ or $\tau_1 \approx \tau_2 \in \Pi(\Sigma) \setminus \Sigma$ (Def. of Π)

$\Longrightarrow \tau_1 \approx \tau_1 \in \Sigma$ or $\tau_1 \approx \tau_2 \in \Pi(\Sigma) \setminus \Sigma$ (Σ normal)

$\Longrightarrow \tau_1 \approx \tau_1 \in \Sigma$ or
$\exists \alpha \in \mathfrak{A}, \tau_1', \tau_2' \in \textsc{Term}(\Sigma) : \tau_1 = \tau_1'\alpha, \tau_2 = \tau_2'\alpha, \tau_1' \approx \tau_2' \in \Sigma$ (Def. of Π_1)

$\Longrightarrow \tau_1 \approx \tau_1 \in \Sigma$ or $\exists \alpha \in \mathfrak{A}, \tau_1' \in \textsc{Term}(\Sigma) : \tau_1 = \tau_1'\alpha, \tau_1' \approx \tau_1' \in \Sigma$ (Σ normal)

$\Longrightarrow \tau_1 \approx \tau_1 \in \Sigma$ or $\tau_1'\alpha \approx \tau_1'\alpha \in \Pi(\Sigma)$ (Def. of Π)

$\Longrightarrow \tau_1 \approx \tau_1 \in \Pi(\Sigma)$.

Step case:
Let $\tau_1 \approx \tau_2 \in \mathrm{T}_{i+1}(\Pi(\Sigma))$.

$\Longrightarrow \tau_1 \approx \tau_2 \in \mathrm{T}_i(\Pi(\Sigma))$ or $\exists \tau_3 : \tau_1 \approx \tau_3, \tau_3 \approx \tau_2 \in \mathrm{T}_i(\Pi(\Sigma))$ (Def. of T)

$\Longrightarrow \tau_1 \approx \tau_1 \in \Pi(\Sigma)$ or $\exists \tau_3 : \tau_1 \approx \tau_3, \tau_3 \approx \tau_2 \in \mathrm{T}_i(\Pi(\Sigma))$ (Ind. hypothesis)

$\Longrightarrow \tau_1 \approx \tau_1 \in \Pi(\Sigma)$ or $\tau_1 \approx \tau_1 \in \Pi(\Sigma)$ (Ind. hypothesis)

$\Longrightarrow \tau_1 \approx \tau_1 \in \Pi(\Sigma)$.

■

40. Lemma Let Σ be a normal set. Let $\tau_1 \approx \tau_2 \in \Pi^*(\Sigma)$ (with $\tau_1, \tau_2 \in \mathfrak{T}$). Then $\tau_1 \approx \tau_1, \tau_2 \approx \tau_2 \in \Pi(\Sigma)$.

Proof: Let $\tau_1 \approx \tau_2 \in \Pi^*(\Sigma)$.

$\Longrightarrow \exists i : \tau_1 \approx \tau_2 \in \mathrm{T}_i(\Pi(\Sigma))$ (Def. of Π^*)

$\Longrightarrow \tau_1 \approx \tau_1, \tau_2 \approx \tau_2 \in \Pi(\Sigma)$ (the above lemma 39)

■

The next lemma is in some ways a consequence of the previous one. It states that if a term is defined in Π^*, it has a variety assigned to it already in Π.

41. Lemma Let Σ be a normal set and $\tau \approx \tau \in \Pi^*(\Sigma)$ ($\tau \in \mathfrak{T}$). Then there is a $\varphi \in \mathfrak{V}$ with $\tau \sim \varphi \in \Pi(\Sigma)$.

Proof: Let $\tau \approx \tau \in \Pi^*(\Sigma)$.

$\Longrightarrow \tau \approx \tau \in \Pi(\Sigma)$ (Lemma 40)

$\Longrightarrow \tau \approx \tau \in \Sigma$ or $\tau \approx \tau \in \Pi(\Sigma) \setminus \Sigma$ (Def. of Π)

$\Longrightarrow \exists \varphi \in \mathfrak{V} : \tau \sim \varphi \in \Sigma$ or $\tau \approx \tau \in \Pi(\Sigma) \setminus \Sigma$ (Σ normal)

$\Longrightarrow \exists \varphi \in \mathfrak{V} : \tau \sim \varphi \in \Sigma$ or

$\exists \tau' \in \textsc{Term}(\Sigma), \alpha \in \mathfrak{A}, \varphi' \in \mathfrak{V} : \tau = \tau'\alpha, \tau' \approx \tau' \in \Sigma, \tau' \sim \varphi' \in \Sigma,$

$\mathfrak{S}\langle \varphi', \alpha \rangle \neq \emptyset$ (Def. of Π_1)

$\Longrightarrow \exists \varphi \in \mathfrak{V} : \tau \sim \varphi \in \Sigma$ or $\exists \varphi \in \mathfrak{V} : \tau'\alpha \sim \varphi \in \Pi(\Sigma)$ (Π_2 or Π_3 applicable)

$\Longrightarrow \exists \varphi \in \mathfrak{V} : \tau \sim \varphi \in \Pi(\Sigma)$

■

The following lemma is needed in the proof, that condition (4) still holds after an application of Π^*. It states that Π^* can grow an attribute path by at most one attribute.

42. Lemma Let Σ be a normal set. Let $\tau \in \mathfrak{T}$ and $\alpha \in \mathfrak{A}$. If $\tau\alpha \approx \tau\alpha \in \Pi^*(\Sigma)$, then $\tau \approx \tau \in \Sigma$.

Proof: Let $\tau\alpha \approx \tau\alpha \in \Pi^*(\Sigma)$.

$\Longrightarrow \tau\alpha \approx \tau\alpha \in \Pi(\Sigma)$ (Lemma 40)

$\Longrightarrow \tau\alpha \approx \tau\alpha \in \Sigma$ or $\tau\alpha \approx \tau\alpha \in \Pi(\Sigma) \setminus \Sigma$ (Def. of Π)

$\Longrightarrow \tau \approx \tau \in \Sigma$ or $\tau\alpha \approx \tau\alpha \in \Pi(\Sigma) \setminus \Sigma$ (Σ normal)

$\Longrightarrow \tau \approx \tau \in \Sigma$ or $\tau \approx \tau \in \Sigma$ (Def. of Π_1)

$\Longrightarrow \tau \approx \tau \in \Sigma$

■

The following lemmata are needed in the proof that condition (5) still holds after an application of Π^*. The first three are of purely technical nature.

43. Lemma Let Σ be a normal set and $\tau_1, \tau_2 \in \mathfrak{T}$. If $\tau_1 \approx \tau_1 \in \Sigma, \tau_2 \approx \tau_2 \in \Sigma, \tau_1 \approx \tau_2 \in \Pi(\Sigma)$, then $\tau_1 \approx \tau_2 \in \Sigma$.

Proof: We prove this lemma by contradiction.
Let $\tau_1 \approx \tau_1 \in \Sigma, \tau_2 \approx \tau_2 \in \Sigma, \tau_1 \approx \tau_2 \in \Pi(\Sigma) \setminus \Sigma$.
$\Longrightarrow \tau_1 \approx \tau_1 \in \Sigma, \tau_2 \approx \tau_2 \in \Sigma, \tau_1 \approx \tau_2 \in \Pi_1(\Sigma)$ (Def. of Π)
$\Longrightarrow \tau_1 \approx \tau_1 \in \Sigma, \tau_2 \approx \tau_2 \in \Sigma, \exists \alpha \in \mathfrak{A},\ \tau_1', \tau_2' \in \textsc{Term}(\Sigma):$
$\tau_1 = \tau_1'\alpha, \tau_2 = \tau_2'\alpha, \tau_1' \approx \tau_2' \in \Sigma, \tau_1 \approx \tau_2 \notin \Sigma$ (Def. of Π_1)
$\Longrightarrow \tau_1 \approx \tau_2 \in \Sigma, \tau_1 \approx \tau_2 \notin \Sigma$ (Σ normal)
$\Longrightarrow$ Contradiction.

Therefore $\tau_1 \approx \tau_2 \in \Sigma$. ■

44. Lemma Let Σ be a normal set. Let $n \in \mathbb{N}$ and $\{\tau_i \mid i \leq n\} \subseteq \mathfrak{T}$. If $\tau_0 \approx \tau_0 \in \Sigma, \{\tau_i \approx \tau_{i+1} \mid i < n\} \subseteq \Pi(\Sigma)$ and $\tau_n \approx \tau_n \in \Sigma$, then $\tau_0 \approx \tau_n \in \Sigma$.

Proof: The proof of this lemma goes by induction on n.

Base case: $n = 0$.
$\Longrightarrow \tau_0 \approx \tau_0 \in \Sigma, \{\tau_i \approx \tau_{i+1} \mid i < 0\} \subseteq \Pi(\Sigma), \tau_0 \approx \tau_0 \in \Sigma$
$\Longrightarrow \tau_0 \approx \tau_0 \in \Sigma$

Step case:
Let $\tau_0 \approx \tau_0 \in \Sigma, \{\tau_i \approx \tau_{i+1} \mid i < n+1\} \subseteq \Pi(\Sigma), \tau_{n+1} \approx \tau_{n+1} \in \Sigma$.
We prove the step case by a simple case division.

Step Case 1: Let $\tau_0 \approx \tau_0 \in \Sigma, \{\tau_i \approx \tau_{i+1} \mid i < n+1\} \subseteq \Pi(\Sigma), \tau_{n+1} \approx \tau_{n+1} \in \Sigma$ and $\tau_n \approx \tau_n \in \Sigma$.
$\Longrightarrow \tau_0 \approx \tau_0 \in \Sigma, \{\tau_i \approx \tau_{i+1} \mid i < n\} \subseteq \Pi(\Sigma), \tau_n \approx \tau_n \in \Sigma,$
$\tau_n \approx \tau_{n+1} \in \Pi(\Sigma), \tau_{n+1} \approx \tau_{n+1} \in \Sigma$
$\Longrightarrow \tau_0 \approx \tau_0 \in \Sigma, \{\tau_i \approx \tau_{i+1} \mid i < n\} \subseteq \Pi(\Sigma), \tau_n \approx \tau_n \in \Sigma, \tau_n \approx \tau_{n+1} \in \Sigma$ (Lemma 43)
$\Longrightarrow \tau_0 \approx \tau_n \in \Sigma, \tau_n \approx \tau_{n+1} \in \Sigma$ (Ind. hypothesis)
$\Longrightarrow \tau_0 \approx \tau_{n+1} \in \Sigma$ (Σ normal)

End of proof of step case 1.

Step Case 2: Let $\tau_0 \approx \tau_0 \in \Sigma, \{\tau_i \approx \tau_{i+1} \mid i < n+1\} \subseteq \Pi(\Sigma), \tau_{n+1} \approx \tau_{n+1} \in \Sigma$ and $\tau_n \approx \tau_n \notin \Sigma$.
$\Longrightarrow n > 0, \tau_0 \approx \tau_0 \in \Sigma, \{\tau_i \approx \tau_{i+1} \mid i < n-1\} \subseteq \Pi(\Sigma),$
$\tau_{n-1} \approx \tau_n \in \Pi(\Sigma), \tau_n \approx \tau_n \notin \Sigma, \tau_n \approx \tau_{n+1} \in \Pi(\Sigma), \tau_{n+1} \approx \tau_{n+1} \in \Sigma$
$\Longrightarrow n > 0, \tau_0 \approx \tau_0 \in \Sigma, \{\tau_i \approx \tau_{i+1} \mid i < n-1\} \subseteq \Pi(\Sigma),$
$\tau_{n-1} \approx \tau_n \in \Pi(\Sigma) \setminus \Sigma, \tau_n \approx \tau_n \notin \Sigma, \tau_n \approx \tau_{n+1} \in \Pi(\Sigma) \setminus \Sigma, \tau_{n+1} \approx \tau_{n+1} \in \Sigma$ (Σ normal)
$\Longrightarrow n > 0, \tau_0 \approx \tau_0 \in \Sigma, \{\tau_i \approx \tau_{i+1} \mid i < n-1\} \subseteq \Pi(\Sigma),$
$\tau_{n-1} \approx \tau_n \in \Pi_1(\Sigma), \tau_n \approx \tau_n \notin \Sigma, \tau_n \approx \tau_{n+1} \in \Pi_1(\Sigma), \tau_{n+1} \approx \tau_{n+1} \in \Sigma$ (Def. of Π)

$$
\begin{aligned}
\Longrightarrow\ & n>0, \tau_0 \approx \tau_0 \in \Sigma, \{\tau_i \approx \tau_{i+1} \mid i < n-1\} \subseteq \Pi(\Sigma), \tau_{n+1} \approx \tau_{n+1} \in \Sigma, \\
& \exists \tau'_{n-1}, \tau'_n \in \mathfrak{T}, \alpha \in \mathfrak{A}, \varphi \in \mathfrak{V} : \tau_{n-1} = \tau'_{n-1}\alpha, \tau_n = \tau'_n\alpha, \\
& \quad \tau'_{n-1} \approx \tau'_n \in \Sigma, \tau'_{n-1} \sim \varphi \in \Sigma, \tau'_n \sim \varphi \in \Sigma, \mathfrak{S}\langle\varphi,\alpha\rangle \neq \emptyset, \\
& \exists \tau''_n, \tau'_{n+1} \in \mathfrak{T}, \beta \in \mathfrak{A}, \varphi' \in \mathfrak{V} : \tau_n = \tau''_n\beta, \tau_{n+1} = \tau'_{n+1}\beta, \\
& \quad \tau''_n \approx \tau'_{n+1} \in \Sigma, \tau''_n \sim \varphi' \in \Sigma, \tau'_{n+1} \sim \varphi' \in \Sigma, \mathfrak{S}\langle\varphi',\beta\rangle \neq \emptyset && \text{(Def. of } \Pi_1\text{)} \\
\Longrightarrow\ & n>0, \tau_0 \approx \tau_0 \in \Sigma, \{\tau_i \approx \tau_{i+1} \mid i < n-1\} \subseteq \Pi(\Sigma), \tau_{n+1} \approx \tau_{n+1} \in \Sigma, \\
& \exists \tau'_{n-1}, \tau'_n \in \mathfrak{T}, \alpha \in \mathfrak{A}, \varphi \in \mathfrak{V} : \tau_{n-1} = \tau'_{n-1}\alpha, \tau_n = \tau'_n\alpha, \\
& \quad \tau'_{n-1} \approx \tau'_n \in \Sigma, \tau'_{n-1} \sim \varphi \in \Sigma, \tau'_n \sim \varphi \in \Sigma, \mathfrak{S}\langle\varphi,\alpha\rangle \neq \emptyset, \\
& \exists \tau''_n, \tau'_{n+1} \in \mathfrak{T}, \beta \in \mathfrak{A}, \varphi' \in \mathfrak{V} : \tau_n = \tau''_n\beta, \tau_{n+1} = \tau'_{n+1}\beta, \\
& \quad \tau''_n \approx \tau'_{n+1} \in \Sigma, \tau''_n \sim \varphi' \in \Sigma, \tau'_{n+1} \sim \varphi' \in \Sigma, \mathfrak{S}\langle\varphi',\beta\rangle \neq \emptyset, \\
& \tau'_n = \tau''_n, \alpha = \beta, \varphi = \varphi' && (\Sigma \text{ normal}) \\
\Longrightarrow\ & n>0, \tau_0 \approx \tau_0 \in \Sigma, \{\tau_i \approx \tau_{i+1} \mid i < n-1\} \subseteq \Pi(\Sigma), \tau_{n+1} \approx \tau_{n+1} \in \Sigma, \\
& \exists \tau'_{n-1}, \tau'_n, \tau'_{n+1} \in \mathfrak{T}, \alpha \in \mathfrak{A}, \varphi \in \mathfrak{V} : \tau_{n-1} = \tau'_{n-1}\alpha, \tau_n = \tau'_n\alpha, \\
& \quad \tau_{n+1} = \tau'_{n+1}\alpha, \tau'_{n-1} \approx \tau'_n \in \Sigma, \tau'_n \approx \tau'_{n+1} \in \Sigma, \tau'_{n-1} \sim \varphi \in \Sigma, \\
& \quad \tau'_n \sim \varphi \in \Sigma, \tau'_{n+1} \sim \varphi \in \Sigma, \mathfrak{S}\langle\varphi,\alpha\rangle \neq \emptyset, \\
& \tau'_{n-1} \approx \tau'_{n+1} \in \Sigma && (\Sigma \text{ normal}) \\
\Longrightarrow\ & n>0, \tau_0 \approx \tau_0 \in \Sigma, \{\tau_i \approx \tau_{i+1} \mid i < n-1\} \subseteq \Pi(\Sigma), \tau_{n+1} \approx \tau_{n+1} \in \Sigma, \\
& (\tau_{n-1} \approx \tau_{n+1} \in \Sigma \ \text{ or } \ \tau_{n-1} \approx \tau_{n+1} \in \Pi_1(\Sigma)) && \text{(Def. of } \Pi_1\text{)} \\
\Longrightarrow\ & n>0, \tau_0 \approx \tau_0 \in \Sigma, \{\tau_i \approx \tau_{i+1} \mid i < n-1\} \subseteq \Pi(\Sigma), \\
& \tau_{n-1} \approx \tau_{n+1} \in \Pi(\Sigma), \tau_{n+1} \approx \tau_{n+1} \in \Sigma \\
\Longrightarrow\ & \tau_0 \approx \tau_{n+1} \in \Sigma && \text{(Ind. hypothesis)}
\end{aligned}
$$

And with this step the proof of the step case 2 and therefore the proof of the whole lemma is finished. ∎

45. Lemma Let Σ be a normal set and $\tau, \tau' \in \mathfrak{T}$. If $\tau \approx \tau' \in \Pi^*(\Sigma)$, then there exists an $n \in \mathbb{N}$ and $\{\tau_i \mid i \leq n\} \subseteq \mathfrak{T}$ such that $\tau = \tau_0, \{\tau_i \approx \tau_{i+1} \mid i < n\} \subseteq \Pi(\Sigma)$ and $\tau_n = \tau'$.

Proof: We first prove by induction on j the following
Claim: If $\tau \approx \tau' \in \mathrm{T}_j(\Pi(\Sigma))$, then there exists an $n \in \mathbb{N}$ and $\{\tau_i \mid i \leq n\} \subseteq \mathfrak{T}$ such that $\tau = \tau_0, \{\tau_i \approx \tau_{i+1} \mid i < n\} \subseteq \Pi(\Sigma)$ and $\tau_n = \tau'$.

Proof of the claim:
Base case: Let $\tau \approx \tau' \in \mathrm{T}_j(\Pi(\Sigma))$.

$\Longrightarrow \tau \approx \tau' \in \Pi(\Sigma)$

$\Longrightarrow$ with $n = 1$ the claim is trivially satisfied:

$\quad \tau = \tau_0, \tau_0 \approx \tau_1 \in \Pi(\Sigma), \tau_1 = \tau'$

Step case: Let $\tau \approx \tau' \in \mathrm{T}_{j+1}(\Pi(\Sigma))$.

$$
\begin{aligned}
\Longrightarrow\ & \tau \approx \tau' \in \mathrm{T}(\mathrm{T}_j(\Pi(\Sigma))) \\
\Longrightarrow\ & \tau \approx \tau' \in \mathrm{T}_j(\Pi(\Sigma)) \ \text{ or} \\
& \exists \tau'' \in \mathfrak{T} : \tau \approx \tau'' \in \mathrm{T}_j(\Pi(\Sigma)), \tau'' \approx \tau' \in \mathrm{T}_j(\Pi(\Sigma)) && \text{(Def. of T)} \\
\Longrightarrow\ & \exists n \in \mathbb{N}, \{\tau_i \mid i \leq n\} \subseteq \mathfrak{T} : \tau = \tau_0, \{\tau_i \approx \tau_{i+1} \mid i < n\}, \tau_n = \tau' \ \text{ or}
\end{aligned}
$$

(Ind. hypothesis)

$$\exists \tau'' \in \mathfrak{T},\ n, n' \in \mathbb{N}, \{\tau_i \mid i \leq n\} \subseteq \mathfrak{T}, \{\tau'_i \mid i \leq n'\} \subseteq \mathfrak{T} :$$
$$\tau \approx \tau_0, \{\tau_i \approx \tau_{i+1} \mid i < n\}, \tau_n = \tau'',$$
$$\tau'' = \tau'_0, \{\tau'_i \approx \tau'_{i+1} \mid i < n'\}, \tau'_{n'} = \tau' \quad \text{(Ind. hypothesis twice)}$$
$$\Longrightarrow \exists n \in \mathbb{N}, \{\tau_i \mid i \leq n\} \subseteq \mathfrak{T} : \tau = \tau_0, \{\tau_i \approx \tau_{i+1} \mid i < n\}, \tau_n = \tau'$$

And this concludes the proof of the sub claim. We can now show the main claim of the lemma.
Let $\tau \approx \tau' \in \Pi^*(\Sigma)$.

$$\Longrightarrow \exists j \in \mathbb{N} : \tau \approx \tau' \in \mathrm{T}_j(\Pi(\Sigma)) \quad \text{(Def. of } \Pi^*)$$
$$\Longrightarrow \exists n \in \mathbb{N}, \{\tau_i \mid i \leq n\} \subseteq \mathfrak{T} : \tau = \tau_0, \{\tau_i \approx \tau_{i+1} \mid i < n\}, \tau_n = \tau' \quad \text{(Claim above)}$$

■

The next lemma makes quite an interesting statement about terms in Π^*. It claims that when Π or T introduces a new literal of the type $\tau \approx \tau'$, one of the two terms τ or τ' must be a new term, that did not exist before.

46. Lemma Let Σ be a normal set and $\tau, \tau' \in \mathfrak{T}$. If $\tau \approx \tau \in \Sigma, \tau' \approx \tau' \in \Sigma, \tau \approx \tau' \in \Pi^*(\Sigma)$, then $\tau \approx \tau' \in \Sigma$ already.

Proof: Let $\tau \approx \tau \in \Sigma, \tau' \approx \tau' \in \Sigma, \tau \approx \tau' \in \Pi^*(\Sigma)$.

$$\Longrightarrow \exists n \in \mathbb{N}, \{\tau_i \mid i \leq n\} \subseteq \mathfrak{T} : \tau = \tau_0, \tau_0 \approx \tau_0 \in \Sigma,$$
$$\{\tau_i \approx \tau_{i+1} \mid i < n\} \subseteq \Pi(\Sigma), \tau_n \approx \tau_n \in \Sigma, \tau_n = \tau' \quad \text{(Lemma 45)}$$
$$\Longrightarrow \tau \approx \tau' \in \Sigma \quad \text{(Lemma 44)}$$

■

The next lemma is a corollary to the previous one, it is here to make later proofs easier.

47. Lemma Let Σ be a normal set. Let $\tau_1, \tau_2 \in \mathfrak{T}$ and $\alpha \in \mathfrak{A}$. If $\tau_1 \approx \tau_2 \in \Pi^*(\Sigma), \tau_1\alpha \approx \tau_1\alpha \in \Pi^*(\Sigma), \tau_2\alpha \approx \tau_2\alpha \in \Pi^*(\Sigma)$, then $\tau_1 \approx \tau_2 \in \Sigma$ already.

Proof: Let $\tau_1 \approx \tau_2 \in \Pi^*(\Sigma), \tau_1\alpha \approx \tau_1\alpha \in \Pi^*(\Sigma), \tau_2\alpha \approx \tau_2\alpha \in \Pi^*(\Sigma)$.

$$\Longrightarrow \tau_1 \approx \tau_2 \in \Pi^*(\Sigma), \tau_1 \approx \tau_1 \in \Sigma, \tau_2 \approx \tau_2 \in \Sigma \quad \text{(Lemma 42)}$$
$$\Longrightarrow \tau_1 \approx \tau_2 \in \Sigma \quad \text{(Lemma 46)}$$

■

The following lemma is a stronger version of condition (7) for normal sets when Π^* is applied.

48. Lemma Let Σ be a normal set and $\tau \in \mathfrak{T}$. If $\exists \varphi \in \mathfrak{V} : \tau \sim \varphi \in \Pi^*(\Sigma)$ then $\tau \approx \tau \in \Pi(\Sigma)$.

Proof: Let $\exists \varphi \in \mathfrak{V} : \tau \sim \varphi \in \Pi^*(\Sigma)$.

$\Longrightarrow \tau \sim \varphi \in \Pi(\Sigma)$ (Def. of T)

$\Longrightarrow \tau \sim \varphi \in \Sigma$ or $\tau \sim \varphi \in \Pi(\Sigma) \setminus \Sigma$ (Def. of Π)

$\Longrightarrow \tau \approx \tau \in \Sigma$ or $\tau \sim \varphi \in \Pi(\Sigma) \setminus \Sigma$ (Σ normal)

$\Longrightarrow \tau \approx \tau \in \Sigma$ or
$\exists \alpha \in \mathfrak{A}, \tau' \in \text{TERM}(\Sigma), \varphi' \in \mathfrak{V} : \tau = \tau'\alpha, \tau' \sim \varphi' \in \Sigma, \mathfrak{S}\langle \varphi', \alpha \rangle \neq \emptyset$
(Def. of Π_2 or Π_3)

$\Longrightarrow \tau \approx \tau \in \Sigma$ or
$\exists \alpha \in \mathfrak{A}, \tau' \in \text{TERM}(\Sigma), \varphi' \in \mathfrak{V} : \tau = \tau'\alpha, \tau' \sim \varphi' \in \Sigma, \mathfrak{S}\langle \varphi', \alpha \rangle \neq \emptyset,$
$\tau' \approx \tau' \in \Sigma$ (Σ normal)

$\Longrightarrow \tau \approx \tau \in \Sigma$ or
$\exists \alpha \in \mathfrak{A}, \tau' \in \text{TERM}(\Sigma) : \tau = \tau'\alpha, \tau'\alpha \approx \tau'\alpha \in \Pi(\Sigma)$ (Def. of Π_1)

$\Longrightarrow \tau \approx \tau \in \Pi(\Sigma)$

■

The next two lemmata prepare the proof that condition (8) still holds after applying Π^* to a normal set. The first lemma serves mainly technical purposes.

49. Lemma Let Σ be a normal set. Let $\tau \in \mathfrak{T}$ and $\varphi \in \mathfrak{V}$. If $\tau \sim \varphi \in \Pi(\Sigma)$, $\tau \in \text{TERM}(\Sigma)$, then $\tau \sim \varphi \in \Sigma$ already.

Proof: By contradiction.
Let $\tau \in \text{TERM}(\Sigma), \tau \sim \varphi \in \Pi(\Sigma) \setminus \Sigma$.

$\Longrightarrow \tau \in \text{TERM}(\Sigma), \exists \varphi' \in \mathfrak{V}, \beta \in \mathfrak{A}, \tau' \in \text{TERM}(\Sigma) :$
$\tau = \tau'\beta, \tau' \sim \varphi' \in \Sigma, \tau \approx \tau \notin \Sigma, \varphi \in \mathfrak{S}\langle \varphi', \beta \rangle$ (Def. of Π_2 and Π_3)

$\Longrightarrow \tau \in \text{TERM}(\Sigma), \exists \varphi' \in \mathfrak{V}, \beta \in \mathfrak{A}, \tau' \in \text{TERM}(\Sigma) :$
$\tau = \tau'\beta, \tau' \sim \varphi' \in \Sigma, \tau \approx \tau \notin \Sigma, \varphi \in \mathfrak{S}\langle \varphi', \beta \rangle, \tau' \approx \tau' \in \Sigma$ (Σ normal)

$\Longrightarrow \tau \approx \tau \notin \Sigma, \tau \approx \tau \in \Sigma$ (Σ normal)

$\Longrightarrow$ Contradiction.

Therefore: $\tau \sim \varphi \in \Sigma$. ■

This lemma states that condition (8) still holds after an application of Π to a normal set, thus it provides half of the way we have to go in the final proof on condition (8). Unfortunately, the proof is very long.

50. Lemma Let Σ be a normal set. Let $\tau_1, \tau_2 \in \mathfrak{T}$ and $\varphi_1, \varphi_2 \in \mathfrak{V}$. If $\tau_1 \approx \tau_2 \in \Pi(\Sigma), \tau_1 \sim \varphi_1 \in \Pi(\Sigma), \tau_2 \sim \varphi_2 \in \Pi(\Sigma)$, then $\varphi_1 = \varphi_2$.

Proof: In order to prove the main result, we have to show some sub claims none of which carries the status of an independent lemma.

Claim 1: If $\tau_1 \approx \tau_2 \in \Pi(\Sigma), \tau_1 \sim \varphi_1 \in \Sigma, \tau_2 \sim \varphi_2 \in \Sigma$, then $\varphi_1 = \varphi_2$.

Proof of claim 1:
Let $\tau_1 \approx \tau_2 \in \Pi(\Sigma), \tau_1 \sim \varphi_1 \in \Sigma, \tau_2 \sim \varphi_2 \in \Sigma$.

$$\begin{aligned}
\Longrightarrow\ & \tau_1 \approx \tau_2 \in \Pi(\Sigma), \tau_1 \sim \varphi_1 \in \Sigma, \tau_2 \sim \varphi_2 \in \Sigma, \\
& \tau_1 \approx \tau_1 \in \Sigma, \tau_2 \approx \tau_2 \in \Sigma && (\Sigma \text{ normal}) \\
\Longrightarrow\ & \tau_1 \approx \tau_2 \in \Sigma, \tau_1 \sim \varphi_1 \in \Sigma, \tau_2 \sim \varphi_2 \in \Sigma && (\text{Lemma 43}) \\
\Longrightarrow\ & \varphi_1 = \varphi_2 && (\Sigma \text{ normal})
\end{aligned}$$

End of proof of claim 1.

Claim 2: If $\tau_1 \approx \tau_2 \in \Pi(\Sigma), \tau_1 \sim \varphi_1 \in \Sigma, \tau_2 \sim \varphi_2 \in \Pi(\Sigma) \setminus \Sigma$, then $\varphi_1 = \varphi_2$.

Proof of claim 2:
Let $\tau_1 \approx \tau_2 \in \Pi(\Sigma), \tau_1 \sim \varphi_1 \in \Sigma, \tau_2 \sim \varphi_2 \in \Pi(\Sigma) \setminus \Sigma$.

$$\begin{aligned}
\Longrightarrow\ & \tau_1 \approx \tau_2 \in \Pi(\Sigma), \tau_1 \sim \varphi_1 \in \Sigma, \tau_2 \sim \varphi_2 \in \Pi(\Sigma) \setminus \Sigma, \\
& \exists \tau_2' \in \textsc{Term}(\Sigma), \alpha \in \mathfrak{A}, \varphi \in \mathfrak{V} : \tau_2 = \tau_2'\alpha, \tau_2' \sim \varphi \in \Sigma, \varphi_2 \in \mathfrak{S}\langle \varphi, \alpha \rangle, \\
& \tau_2 \approx \tau_2 \notin \Sigma && (\text{Def. of } \Pi_2 \text{ and } \Pi_3) \\
\Longrightarrow\ & \tau_1 \approx \tau_2 \in \Pi_1(\Sigma), \tau_1 \sim \varphi_1 \in \Sigma, \tau_2 \sim \varphi_2 \in \Pi(\Sigma) \setminus \Sigma, \\
& \exists \tau_2' \in \textsc{Term}(\Sigma), \alpha \in \mathfrak{A}, \varphi \in \mathfrak{V} : \tau_2 = \tau_2'\alpha, \tau_2' \sim \varphi \in \Sigma, \varphi_2 \in \mathfrak{S}\langle \varphi, \alpha \rangle, \\
& \tau_2 \approx \tau_2 \notin \Sigma && (\Sigma \text{ normal}) \\
\Longrightarrow\ & \tau_1 \sim \varphi_1 \in \Sigma, \tau_2 \sim \varphi_2 \in \Pi(\Sigma) \setminus \Sigma, \\
& \exists \tau_1', \tau_2' \in \textsc{Term}(\Sigma), \alpha \in \mathfrak{A}, \varphi \in \mathfrak{V} : \tau_2 = \tau_2'\alpha, \tau_1 = \tau_1'\alpha, \tau_2 \sim \varphi \in \Sigma, \\
& \varphi_2 \in \mathfrak{S}\langle \varphi, \alpha \rangle, \tau_1' \approx \tau_2' \in \Sigma, \tau_2 \approx \tau_2 \notin \Sigma && (\text{Def. of } \Pi_1) \\
\Longrightarrow\ & \tau_1 \sim \varphi_1 \in \Sigma, \tau_2 \sim \varphi_2 \in \Pi_2(\Sigma), \\
& \exists \tau_1', \tau_2' \in \textsc{Term}(\Sigma), \alpha \in \mathfrak{A}, \varphi \in \mathfrak{V} : \tau_2 = \tau_2'\alpha, \tau_1 = \tau_1'\alpha, \tau_2 \sim \varphi \in \Sigma, \\
& \varphi_2 \in \mathfrak{S}\langle \varphi, \alpha \rangle, \tau_1' \approx \tau_2' \in \Sigma, \tau_2 \approx \tau_2 \notin \Sigma && (\text{Def. of } \Pi_2) \\
\Longrightarrow\ & \tau_2 \sim \varphi_2 \in \Pi_2(\Sigma), \tau_2 \sim \varphi_1 \in \Pi_2(\Sigma) && (\text{Def. of } \Pi_2) \\
\Longrightarrow\ & \varphi_1 = \varphi_2 && (\text{Lemma 37})
\end{aligned}$$

End of proof of claim 2.

Claim 3: If $\tau_1 \approx \tau_2 \in \Pi(\Sigma), \tau_1 \sim \varphi_1 \in \Pi(\Sigma) \setminus \Sigma, \tau_2 \sim \varphi_2 \in \Sigma$, then $\varphi_1 = \varphi_2$.

The proof of claim 3 is totally analogous to the proof of claim 2 with the roles of $\tau_1 \leftrightarrow \tau_2$ and $\varphi_1 \leftrightarrow \varphi_2$ exchanged.

Claim 4: If $\tau_1 \approx \tau_2 \in \Pi(\Sigma), \tau_1 \sim \varphi_1 \in \Pi_2(\Sigma), \tau_2 \sim \varphi_2 \in \Pi_2(\Sigma)$, then $\varphi_1 = \varphi_2$.

Proof of claim 4:
Let $\tau_1 \approx \tau_2 \in \Pi(\Sigma), \tau_1 \sim \varphi_1 \in \Pi_2(\Sigma), \tau_2 \sim \varphi_2 \in \Pi_2(\Sigma)$.

$$\begin{aligned}
\Longrightarrow\ & \tau_1 \approx \tau_2 \in \Pi(\Sigma), \exists \tau_1', \kappa_1 \in \textsc{Term}(\Sigma), \alpha \in \mathfrak{A}, \varphi_1' \in \mathfrak{V} : \tau_1 = \tau_1'\alpha, \\
& \tau_1' \sim \varphi_1' \in \Sigma, \varphi_1 \in \mathfrak{S}\langle \varphi_1', \alpha \rangle, \tau_1 \approx \tau_1 \notin \Sigma, \kappa_1 \approx \tau_1' \in \Sigma, \kappa_1\alpha \sim \varphi_1 \in \Sigma, \\
& \exists \tau_2', \kappa_2 \in \textsc{Term}(\Sigma), \beta \in \mathfrak{A}, \varphi_2' \in \mathfrak{V} : \tau_2 = \tau_2'\beta, \tau_2' \sim \varphi_2' \in \Sigma, \\
& \varphi_2 \in \mathfrak{S}\langle \beta, \varphi_2' \rangle, \tau_2 \approx \tau_2 \notin \Sigma, \kappa_2 \approx \tau_2' \in \Sigma, \kappa_2\beta \sim \varphi_2 \in \Sigma && (\text{Def. of } \Pi_2) \\
\Longrightarrow\ & \tau_1 \approx \tau_2 \in \Pi_1(\Sigma), \exists \tau_1', \kappa_1 \in \textsc{Term}(\Sigma), \alpha \in \mathfrak{A}, \varphi_1' \in \mathfrak{V} : \tau_1 = \tau_1'\alpha, \\
& \tau_1' \sim \varphi_1' \in \Sigma, \varphi_1 \in \mathfrak{S}\langle \varphi_1', \alpha \rangle, \tau_1 \approx \tau_1 \notin \Sigma, \kappa_1 \approx \tau_1' \in \Sigma, \kappa_1\alpha \sim \varphi_1 \in \Sigma, \\
& \exists \tau_2', \kappa_2 \in \textsc{Term}(\Sigma), \beta \in \mathfrak{A}, \varphi_2' \in \mathfrak{V} : \tau_2 = \tau_2'\beta, \tau_2' \sim \varphi_2' \in \Sigma, \\
& \varphi_2 \in \mathfrak{S}\langle \beta, \varphi_2' \rangle, \tau_2 \approx \tau_2 \notin \Sigma, \kappa_2 \approx \tau_2' \in \Sigma, \kappa_2\beta \sim \varphi_2 \in \Sigma && (\Sigma \text{ normal})
\end{aligned}$$

$$
\begin{aligned}
\Longrightarrow\ & \tau_1 \approx \tau_2 \in \Pi_1(\Sigma), \exists \tau_1', \kappa_1 \in \textsc{Term}(\Sigma), \alpha \in \mathfrak{A}, \varphi_1' \in \mathfrak{V} : \tau_1 = \tau_1'\alpha, \\
& \tau_1' \sim \varphi_1' \in \Sigma, \varphi_1 \in \mathfrak{S}\langle \varphi_1', \alpha \rangle, \kappa_1 \approx \tau_1' \in \Sigma, \kappa_1\alpha \sim \varphi_1 \in \Sigma, \\
& \exists \tau_2', \kappa_2 \in \textsc{Term}(\Sigma), \beta \in \mathfrak{A}, \varphi_2' \in \mathfrak{V} : \tau_2 = \tau_2'\beta, \tau_2' \sim \varphi_2' \in \Sigma, \\
& \varphi_2 \in \mathfrak{S}\langle \beta, \varphi_2' \rangle, \kappa_2 \approx \tau_2' \in \Sigma, \kappa_2\beta \sim \varphi_2 \in \Sigma, \\
& \alpha = \beta, \tau_1' \approx \tau_2' \in \Sigma && \text{(Def. of } \Pi_1\text{)} \\
\Longrightarrow\ & \exists \tau_1', \kappa_1 \in \textsc{Term}(\Sigma), \alpha \in \mathfrak{A}, \varphi_1' \in \mathfrak{V} : \tau_1 = \tau_1'\alpha, \tau_1' \sim \varphi_1' \in \Sigma, \\
& \varphi_1 \in \mathfrak{S}\langle \varphi_1', \alpha \rangle, \kappa_1 \approx \tau_1' \in \Sigma, \kappa_1\alpha \sim \varphi_1 \in \Sigma, \\
& \exists \tau_2', \kappa_2 \in \textsc{Term}(\Sigma), \varphi_2' \in \mathfrak{V} : \tau_2 = \tau_2'\alpha, \tau_2' \sim \varphi_2' \in \Sigma, \\
& \varphi_2 \in \mathfrak{S}\langle \alpha, \varphi_2' \rangle, \kappa_2 \approx \tau_2' \in \Sigma, \kappa_2\alpha \sim \varphi_2 \in \Sigma, \\
& \tau_1' \approx \tau_2' \in \Sigma, \varphi_1' = \varphi_2' && \text{(}\Sigma\text{ normal)} \\
\Longrightarrow\ & \exists \tau_1', \kappa_1 \in \textsc{Term}(\Sigma), \alpha \in \mathfrak{A}, \varphi_1' \in \mathfrak{V} : \tau_1' \sim \varphi_1' \in \Sigma, \kappa_1 \approx \tau_1' \in \Sigma, \\
& \kappa_1\alpha \sim \varphi_1 \in \Sigma, \\
& \exists \tau_2', \kappa_2 \in \textsc{Term}(\Sigma) : \tau_2' \sim \varphi_1' \in \Sigma, \kappa_2 \approx \tau_2' \in \Sigma, \kappa_2\alpha \sim \varphi_2 \in \Sigma, \\
& \tau_1' \approx \tau_2' \in \Sigma, \kappa_1 \approx \kappa_2 \in \Sigma && \text{(}\Sigma\text{ normal)} \\
\Longrightarrow\ & \exists \kappa_1, \kappa_2 \in \textsc{Term}(\Sigma), \alpha \in \mathfrak{A} : \kappa_1\alpha \sim \varphi_1 \in \Sigma, \kappa_2\alpha \sim \varphi_2 \in \Sigma, \\
& \kappa_1\alpha \approx \kappa_2\alpha \in \Sigma && \text{(}\Sigma\text{ normal)} \\
\Longrightarrow\ & \varphi_1 = \varphi_2 && \text{(}\Sigma\text{ normal)}
\end{aligned}
$$

End of proof of claim 4.

Claim 5: If $\tau_1 \approx \tau_2 \in \Pi(\Sigma), \tau_1 \sim \varphi_1 \in \Pi_3(\Sigma), \tau_2 \sim \varphi_2 \in \Pi_3(\Sigma)$, then $\varphi_1 = \varphi_2$.

Proof of claim 5:
Let $\tau_1 \approx \tau_2 \in \Pi(\Sigma), \tau_1 \sim \varphi_1 \in \Pi_3(\Sigma), \tau_2 \sim \varphi_2 \in \Pi_3(\Sigma)$.

$$
\begin{aligned}
\Longrightarrow\ & \tau_1 \approx \tau_2 \in \Pi(\Sigma), \exists \tau_1' \in \textsc{Term}(\Sigma), \alpha \in \mathfrak{A}, \varphi_1' \in \mathfrak{V} : \tau_1 = \tau_1'\alpha, \\
& \tau_1' \sim \varphi_1' \in \Sigma, \varphi_1 = \textsc{Choose}(\mathfrak{S}\langle \varphi_1', \alpha \rangle), \tau_1 \approx \tau_1 \notin \Sigma, \\
& \exists \tau_2' \in \textsc{Term}(\Sigma), \beta \in \mathfrak{A}, \varphi_2' \in \mathfrak{V} : \tau_2 = \tau_2'\beta, \tau_2' \sim \varphi_2' \in \Sigma, \\
& \varphi_2 = \textsc{Choose}(\mathfrak{S}\langle \varphi_2', \beta \rangle), \tau_2 \approx \tau_2 \notin \Sigma && \text{(Def. of } \Pi_3\text{)} \\
\Longrightarrow\ & \tau_1 \approx \tau_2 \in \Pi_1(\Sigma), \exists \tau_1' \in \textsc{Term}(\Sigma), \alpha \in \mathfrak{A}, \varphi_1' \in \mathfrak{V} : \tau_1 = \tau_1'\alpha, \\
& \tau_1' \sim \varphi_1' \in \Sigma, \varphi_1 = \textsc{Choose}(\mathfrak{S}\langle \varphi_1', \alpha \rangle), \\
& \exists \tau_2' \in \textsc{Term}(\Sigma), \beta \in \mathfrak{A}, \varphi_2' \in \mathfrak{V} : \tau_2 = \tau_2'\beta, \tau_2' \sim \varphi_2' \in \Sigma, \\
& \varphi_2 = \textsc{Choose}(\mathfrak{S}\langle \varphi_2', \beta \rangle) && \text{(}\Sigma\text{ normal)} \\
\Longrightarrow\ & \tau_1 \approx \tau_2 \in \Pi_1(\Sigma), \exists \tau_1' \in \textsc{Term}(\Sigma), \alpha \in \mathfrak{A}, \varphi_1' \in \mathfrak{V} : \tau_1 = \tau_1'\alpha, \\
& \tau_1' \sim \varphi_1' \in \Sigma, \varphi_1 = \textsc{Choose}(\mathfrak{S}\langle \varphi_1', \alpha \rangle), \\
& \exists \tau_2' \in \textsc{Term}(\Sigma), \beta \in \mathfrak{A}, \varphi_2' \in \mathfrak{V} : \tau_2 = \tau_2'\beta, \tau_2' \sim \varphi_2' \in \Sigma, \\
& \varphi_2 = \textsc{Choose}(\mathfrak{S}\langle \varphi_2', \beta \rangle), \\
& \alpha = \beta, \tau_1' \approx \tau_2' \in \Sigma && \text{(Def. of } \Pi_1\text{)} \\
\Longrightarrow\ & \exists \tau_1' \in \textsc{Term}(\Sigma), \alpha \in \mathfrak{A}, \varphi_1' \in \mathfrak{V} : \tau_1 = \tau_1'\alpha, \tau_1' \sim \varphi_1' \in \Sigma, \\
& \varphi_1 = \textsc{Choose}(\mathfrak{S}\langle \varphi_1', \alpha \rangle), \\
& \exists \tau_2' \in \textsc{Term}(\Sigma), \varphi_2' \in \mathfrak{V} : \tau_2 = \tau_2'\alpha, \tau_2' \sim \varphi_2' \in \Sigma, \\
& \varphi_2 = \textsc{Choose}(\mathfrak{S}\langle \varphi_2', \alpha \rangle), \\
& \tau_1' \approx \tau_2' \in \Sigma, \varphi_1' = \varphi_2' && \text{(}\Sigma\text{ normal)}
\end{aligned}
$$

$\Longrightarrow \exists \tau_1' \in \textsc{Term}(\Sigma), \alpha \in \mathfrak{A}, \varphi_1' \in \mathfrak{V} : \tau_1' \sim \varphi_1' \in \Sigma, \varphi_1 = \textsc{Choose}(\mathfrak{S}\langle \varphi_1', \alpha\rangle),$
$\quad \exists \tau_2' \in \textsc{Term}(\Sigma) : \tau_2' \sim \varphi_1' \in \Sigma, \varphi_2 = \textsc{Choose}(\mathfrak{S}\langle \varphi_1', \alpha\rangle)$ (Σ normal)
$\Longrightarrow \varphi_1 = \varphi_2$

End of proof of claim 5.

Claim 6: If $\tau_1 \approx \tau_2 \in \Pi(\Sigma), \tau_1 \sim \varphi_1 \in \Pi(\Sigma) \setminus \Sigma, \tau_2 \sim \varphi_2 \in \Pi(\Sigma) \setminus \Sigma$, then $\varphi_1 = \varphi_2$.

Proof of claim 6:
Let $\tau_1 \approx \tau_2 \in \Pi(\Sigma), \tau_1 \sim \varphi_1 \in \Pi(\Sigma) \setminus \Sigma, \tau_2 \sim \varphi_2 \in \Pi(\Sigma) \setminus \Sigma$.
$\Longrightarrow \tau_1 \approx \tau_2 \in \Pi(\Sigma), \tau_1 \sim \varphi_1 \in \Pi(\Sigma) \setminus \Sigma \wedge ($
$\quad \tau_2 \sim \varphi_2 \in \Pi_2(\Sigma)$
$\quad \vee\, \tau_2 \sim \varphi_2 \in \Pi_3(\Sigma))$ (Def. of Π)
$\Longrightarrow \tau_1 \approx \tau_2 \in \Pi(\Sigma), \tau_1 \sim \varphi_1 \in \Pi(\Sigma) \setminus \Sigma \wedge ($
$\quad (\tau_2 \sim \varphi_2 \in \Pi_2(\Sigma), \exists \kappa, \tau_2' \in \textsc{Term}(\Sigma), \alpha \in \mathfrak{A}, \varphi_2' \in \mathfrak{V} : \tau_2 = \tau_2'\alpha,$
$\quad \tau_2' \sim \varphi_2' \in \Sigma, \tau_2 \approx \tau_2 \notin \Sigma, \varphi_2 \in \mathfrak{S}\langle \varphi_2', \alpha\rangle, \kappa \approx \tau_2' \in \Sigma, \kappa\alpha \sim \varphi_2 \in \Sigma)$
(Def. of Π_2)
$\quad \vee\, (\tau_2 \sim \varphi_2 \in \Pi_3(\Sigma)), \exists \tau_2' \in \textsc{Term}(\Sigma), \alpha \in \mathfrak{A}, \varphi_2' \in \mathfrak{V} : \tau_2 = \tau_2'\alpha,$
$\quad \tau_2' \sim \varphi_2' \in \Sigma, \tau_2 \approx \tau_2 \notin \Sigma, \varphi_2 = \textsc{Choose}(\mathfrak{S}\langle \varphi_2', \alpha\rangle),$
$\quad \neg(\exists \kappa \in \mathfrak{T}, \varphi' \in \mathfrak{V}(\kappa \approx \tau_2' \in \Sigma, \kappa\alpha \sim \varphi' \in \Sigma)))$ (Def. of Π_3)
$\Longrightarrow \tau_1 \approx \tau_2 \in \Pi_1(\Sigma), \tau_1 \sim \varphi_1 \in \Pi(\Sigma) \setminus \Sigma \wedge ($
$\quad (\tau_2 \sim \varphi_2 \in \Pi_2(\Sigma), \exists \kappa, \tau_2' \in \textsc{Term}(\Sigma), \alpha \in \mathfrak{A}, \varphi_2' \in \mathfrak{V} : \tau_2 = \tau_2'\alpha,$
$\quad \tau_2' \sim \varphi_2' \in \Sigma, \tau_2 \approx \tau_2 \notin \Sigma, \varphi_2 \in \mathfrak{S}\langle \varphi_2', \alpha\rangle, \kappa \approx \tau_2' \in \Sigma, \kappa\alpha \sim \varphi_2 \in \Sigma)$
$\quad \vee\, (\tau_2 \sim \varphi_2 \in \Pi_3(\Sigma)), \exists \tau_2' \in \textsc{Term}(\Sigma), \alpha \in \mathfrak{A}, \varphi_2' \in \mathfrak{V} : \tau_2 = \tau_2'\alpha,$
$\quad \tau_2' \sim \varphi_2' \in \Sigma, \tau_2 \approx \tau_2 \notin \Sigma, \varphi_2 = \textsc{Choose}(\mathfrak{S}\langle \varphi_2', \alpha\rangle),$
$\quad \neg(\exists \kappa \in \mathfrak{T}, \varphi' \in \mathfrak{V}(\kappa \approx \tau_2' \in \Sigma, \kappa\alpha \sim \varphi' \in \Sigma)))$ (Σ normal)
$\Longrightarrow \tau_1 \approx \tau_2 \in \Pi(\Sigma) \wedge$
$\quad (\tau_1 \sim \varphi_1 \in \Pi(\Sigma) \setminus \Sigma, \tau_2 \sim \varphi_2 \in \Pi_2(\Sigma), \exists \kappa, \tau_1', \tau_2' \in \textsc{Term}(\Sigma), \alpha \in \mathfrak{A},$
$\quad \varphi_2' \in \mathfrak{V} : \tau_1 = \tau_1'\alpha, \tau_2 = \tau_2'\alpha, \tau_1' \approx \tau_2' \in \Sigma, \tau_2' \sim \varphi_2' \in \Sigma,$
$\quad \tau_2 \approx \tau_2 \notin \Sigma, \varphi_2 \in \mathfrak{S}\langle \varphi_2', \alpha\rangle, \kappa \approx \tau_2' \in \Sigma, \kappa\alpha \sim \varphi_2 \in \Sigma)$ (Def. of Π_1)
$\quad \vee\, (\tau_1 \sim \varphi_1 \in \Pi(\Sigma) \setminus \Sigma, \tau_2 \sim \varphi_2 \in \Pi_3(\Sigma)), \exists \tau_1', \tau_2' \in \textsc{Term}(\Sigma), \alpha \in \mathfrak{A},$
$\quad \varphi_2' \in \mathfrak{V} : \tau_1 = \tau_1'\alpha, \tau_2 = \tau_2'\alpha, \tau_1' \approx \tau_2' \in \Sigma,$
$\quad \tau_2' \sim \varphi_2' \in \Sigma, \tau_2 \approx \tau_2 \notin \Sigma, \varphi_2 = \textsc{Choose}(\mathfrak{S}\langle \varphi_2', \alpha\rangle),$
$\quad \neg(\exists \kappa \in \mathfrak{T}, \varphi' \in \mathfrak{V}(\kappa \approx \tau_2' \in \Sigma, \kappa\alpha \sim \varphi' \in \Sigma)))$ (Def. of Π_1)
$\Longrightarrow \tau_1 \approx \tau_2 \in \Pi(\Sigma) \wedge$
$\quad (\tau_1 \sim \varphi_1 \in \Pi(\Sigma) \setminus \Sigma, \tau_2 \sim \varphi_2 \in \Pi_2(\Sigma), \exists \kappa, \tau_1', \tau_2' \in \textsc{Term}(\Sigma), \alpha \in \mathfrak{A},$
$\quad \varphi_2' \in \mathfrak{V} : \tau_1 = \tau_1'\alpha, \tau_2 = \tau_2'\alpha, \tau_1' \approx \tau_2' \in \Sigma, \tau_1' \sim \varphi_2' \in \Sigma,$
$\quad \tau_1 \approx \tau_1 \notin \Sigma, \mathfrak{S}\langle \varphi_2', \alpha\rangle \neq \emptyset, \kappa \approx \tau_1' \in \Sigma, \kappa\alpha \sim \varphi_2 \in \Sigma)$ (Σ normal.)
$\quad \vee\, (\tau_1 \sim \varphi_1 \in \Pi(\Sigma) \setminus \Sigma, \tau_2 \sim \varphi_2 \in \Pi_3(\Sigma)), \exists \tau_1', \tau_2' \in \textsc{Term}(\Sigma), \alpha \in \mathfrak{A},$
$\quad \varphi_2' \in \mathfrak{V} : \tau_1 = \tau_1'\alpha, \tau_2 = \tau_2'\alpha, \tau_1' \approx \tau_2' \in \Sigma,$
$\quad \tau_1' \sim \varphi_2' \in \Sigma, \tau_1 \approx \tau_1 \notin \Sigma, \mathfrak{S}\langle \varphi_2', \alpha\rangle \neq \emptyset,$
$\quad \neg(\exists \kappa \in \mathfrak{T}, \varphi' \in \mathfrak{V}(\kappa \approx \tau_1' \in \Sigma, \kappa\alpha \sim \varphi' \in \Sigma)))$ (Σ normal.)

$$\begin{aligned}
&\Longrightarrow (\tau_1 \approx \tau_2 \in \Pi(\Sigma), \tau_1 \sim \varphi_1 \in \Pi_2(\Sigma), \tau_2 \sim \varphi_2 \in \Pi_2(\Sigma)) && \text{(Def. of } \Pi_2\text{)}\\
&\qquad \vee (\tau_1 \approx \tau_2 \in \Pi(\Sigma), \tau_1 \sim \varphi_1 \in \Pi_3(\Sigma), \tau_2 \sim \varphi_2 \in \Pi_3(\Sigma)) && \text{(Def. of } \Pi_3\text{)}\\
&\Longrightarrow \varphi_1 = \varphi_2 && \text{(Claim 4 above)}\\
&\qquad \vee \varphi_1 = \varphi_2 && \text{(Claim 5 above)}\\
&\Longrightarrow \varphi_1 = \varphi_2
\end{aligned}$$

End of proof of claim 6.

Now we can finally proof the main claim of the lemma. As a reminder, the claim was: If $\tau_1 \approx \tau_2, \tau_1 \sim \varphi_1, \tau_2 \sim \varphi_2 \in \Pi(\Sigma)$, then $\varphi_1 = \varphi_2$.

Let $\tau_1 \approx \tau_2, \tau_1 \sim \varphi_1, \tau_2 \sim \varphi_2 \in \Pi(\Sigma)$.

$$\begin{aligned}
&\Longrightarrow \tau_1 \approx \tau_2 \in \Pi(\Sigma) \wedge (\\
&\qquad\quad (\tau_1 \sim \varphi_1 \in \Sigma, \tau_2 \sim \varphi_2 \in \Sigma)\\
&\qquad \vee (\tau_1 \sim \varphi_1 \in \Sigma, \tau_2 \sim \varphi_2 \in \Pi(\Sigma) \setminus \Sigma)\\
&\qquad \vee (\tau_1 \sim \varphi_1 \in \Pi(\Sigma) \setminus \Sigma, \tau_2 \sim \varphi_2 \in \Sigma)\\
&\qquad \vee (\tau_1 \sim \varphi_1 \in \Pi(\Sigma) \setminus \Sigma, \tau_2 \sim \varphi_2 \in \Pi(\Sigma) \setminus \Sigma)) && \text{(Def. of } \Pi\text{)}\\
&\Longrightarrow \tau_1 \approx \tau_2 \in \Pi(\Sigma) \wedge (\\
&\qquad\quad \varphi_1 = \varphi_2 && \text{(Claim 1 above)}\\
&\qquad \vee \varphi_1 = \varphi_2 && \text{(Claim 2 above)}\\
&\qquad \vee \varphi_1 = \varphi_2 && \text{(Claim 3 above)}\\
&\qquad \vee \varphi_1 = \varphi_2) && \text{(Claim 6 above)}\\
&\Longrightarrow \varphi_1 = \varphi_2
\end{aligned}$$

■

The next three lemmata will be used in the proof that condition (10) still holds after applying Π^* to a normal set.

51. Lemma Let Σ be a normal set. Let $\tau \in \mathfrak{T}, \alpha \in \mathfrak{A}$ and $\varphi \in \mathfrak{V}$. If $\tau \sim \varphi \in \Pi(\Sigma)$, $\mathfrak{S}\langle\varphi, \alpha\rangle \neq \emptyset$, $\tau\alpha \in \textsc{Term}(\Sigma)$, then $\tau\alpha \approx \tau\alpha \in \Sigma$.

Proof: Let $\tau \sim \varphi \in \Pi(\Sigma), \mathfrak{S}\langle\varphi, \alpha\rangle \neq \emptyset, \tau\alpha \in \textsc{Term}(\Sigma)$

$$\begin{aligned}
&\Longrightarrow \tau \sim \varphi \in \Pi(\Sigma), \mathfrak{S}\langle\varphi, \alpha\rangle \neq \emptyset, \tau, \tau\alpha \in \textsc{Term}(\Sigma) && \text{(Def. of Term)}\\
&\Longrightarrow \tau \sim \varphi \in \Sigma, \mathfrak{S}\langle\varphi, \alpha\rangle \neq \emptyset, \tau\alpha \in \textsc{Term}(\Sigma) && \text{(Lemma 49)}\\
&\Longrightarrow \tau\alpha \approx \tau\alpha \in \Sigma && (\Sigma \text{ normal})
\end{aligned}$$

■

52. Lemma Let Σ be normal. Let $\tau \in \mathfrak{T}$ and $\alpha \in \mathfrak{A}$. If $\tau\alpha \in \textsc{Term}(\Pi(\Sigma)) \setminus \textsc{Term}(\Sigma)$, then $\tau \in \textsc{Term}(\Sigma)$.

Proof: Let $\tau\alpha \in \textsc{Term}(\Pi(\Sigma)) \setminus \textsc{Term}(\Sigma)$.

$$\begin{aligned}
&\Longrightarrow \tau\alpha \in \textsc{Term}(\Pi(\Sigma)), \tau\alpha \notin \textsc{Term}(\Sigma)\\
&\Longrightarrow \exists \tau' \in \textsc{Term}(\Sigma) : \tau\alpha \approx \tau'\alpha \in \Pi(\Sigma) \setminus \Sigma, \tau \approx \tau' \in \Sigma
\end{aligned}$$

$$\text{or } \exists \varphi, \varphi' \in \mathfrak{V} : \tau\alpha \sim \varphi \in \Pi(\Sigma) \setminus \Sigma, \tau \sim \varphi' \in \Sigma \qquad \text{(Def. of } \Pi)$$
$$\Longrightarrow \tau \in \textsc{Term}(\Sigma)$$

■

53. Lemma Let Σ be a normal set. Let $\tau \in \mathfrak{T}, \alpha \in \mathfrak{A}$ and $\varphi \in \mathfrak{V}$. If $\tau\alpha \in \textsc{Term}(\Pi(\Sigma)) \setminus \textsc{Term}(\Sigma), \tau \sim \varphi \in \Pi(\Sigma), \mathfrak{S}\langle\varphi, \alpha\rangle \neq \emptyset$, then $\tau\alpha \approx \tau\alpha \in \Pi(\Sigma)$.

Proof: Let $\tau\alpha \in \textsc{Term}(\Pi(\Sigma)) \setminus \textsc{Term}(\Sigma), \tau \sim \varphi \in \Pi(\Sigma), \mathfrak{S}\langle\varphi, \alpha\rangle \neq \emptyset$

$$\Longrightarrow \tau\alpha \in \textsc{Term}(\Pi(\Sigma)) \setminus \textsc{Term}(\Sigma), \tau \sim \varphi \in \Pi(\Sigma), \mathfrak{S}\langle\varphi, \alpha\rangle \neq \emptyset, \tau\alpha \approx \tau\alpha \notin \Sigma \qquad \text{(Def. of Term)}$$
$$\Longrightarrow \tau \in \textsc{Term}(\Sigma), \tau \sim \varphi \in \Pi(\Sigma), \mathfrak{S}\langle\varphi, \alpha\rangle \neq \emptyset, \tau\alpha \approx \tau\alpha \notin \Sigma \qquad \text{(Lemma 52)}$$
$$\Longrightarrow \tau \sim \varphi \in \Sigma, \mathfrak{S}\langle\varphi, \alpha\rangle \neq \emptyset, \tau\alpha \approx \tau\alpha \notin \Sigma \qquad \text{(Lemma 49)}$$
$$\Longrightarrow \tau \sim \varphi \in \Sigma, \tau \approx \tau \in \Sigma, \mathfrak{S}\langle\varphi, \alpha\rangle \neq \emptyset, \tau\alpha \approx \tau\alpha \notin \Sigma \qquad (\Sigma \text{ normal})$$
$$\Longrightarrow \tau\alpha \approx \tau\alpha \in \Pi(\Sigma) \qquad \text{(Def. of } \Pi_1)$$

■

Finally the last two preparative lemmata follow. They deal with the two sub cases which arise in the proof that condition (11) still holds after applying Π^*. These sub cases reflect the fact that we have two types of positive literals.

54. Lemma Let Σ be a normal set. Let $\tau \in \mathfrak{T}$ and $\varphi \in \mathfrak{V}$. If $\neg\tau \sim \varphi \in \Pi^*(\Sigma)$, then $\tau \sim \varphi \notin \Pi^*(\Sigma)$.

Proof: We first prove a small claim which then helps us proving the main result.

Claim: $\tau \sim \varphi \notin \Sigma, \tau \in \textsc{Term}(\Sigma) \Longrightarrow \tau \sim \varphi \notin \Pi(\Sigma)$.

Proof of claim by contradiction:
Let $\tau \sim \varphi \notin \Sigma, \tau \in \textsc{Term}(\Sigma), \tau \sim \varphi \in \Pi(\Sigma)$.

$$\Longrightarrow \tau \sim \varphi \notin \Sigma, \tau \in \textsc{Term}(\Sigma), \tau \sim \varphi \in \Pi_2(\Sigma) \cup \Pi_3(\Sigma) \qquad \text{(Def. of } \Pi)$$
$$\Longrightarrow \tau \in \textsc{Term}(\Sigma), \exists \varphi' \in \mathfrak{V}, \alpha \in \mathfrak{A}, \tau' \in \textsc{Term}(\Sigma) :$$
$$\tau = \tau'\alpha, \tau' \sim \varphi' \in \Sigma, \varphi \in \mathfrak{S}\langle\varphi', \alpha\rangle, \tau \approx \tau \notin \Sigma \qquad \text{(Def. of } \Pi)$$
$$\Longrightarrow \tau \approx \tau \in \Sigma, \tau \approx \tau \notin \Sigma \qquad (\Sigma \text{ normal})$$
$$\Longrightarrow \text{Contradiction.}$$

Therefore $\tau \sim \varphi \notin \Pi(\Sigma)$.

The main result can now be shown.
Let $\neg\tau \sim \varphi \in \Pi^*(\Sigma)$.

$$\Longrightarrow \neg\tau \sim \varphi \in \Pi(\Sigma) \qquad \text{(Def. of } \Pi^*)$$
$$\Longrightarrow \neg\tau \sim \varphi \in \Sigma \qquad \text{(Def. of } \Pi)$$
$$\Longrightarrow \tau \sim \varphi \notin \Sigma, \tau \in \textsc{Term}(\Sigma) \qquad (\Sigma \text{ normal})$$
$$\Longrightarrow \tau \sim \varphi \notin \Pi(\Sigma) \qquad \text{(Claim above)}$$
$$\Longrightarrow \tau \sim \varphi \notin \Pi^*(\Sigma) \qquad \text{(Def. of } \Pi^*)$$

■

55. Lemma Let Σ be a normal set and let $\tau_1, \tau_2 \in \mathfrak{T}$. If $\neg\tau_1 \approx \tau_2 \in \Pi^*(\Sigma)$, then $\tau_1 \approx \tau_2 \notin \Pi^*(\Sigma)$.

Proof: Again, we prove an intermediate claim to smooth the main argument.

Claim: $\tau_1 \approx \tau_2 \notin \Sigma,\ \tau_1, \tau_2 \in \textsc{Term}(\Sigma) \Longrightarrow \tau_1 \approx \tau_2 \notin \Pi^*(\Sigma)$.

Proof of the claim by contradiction:
Let $\tau_1 \approx \tau_2 \notin \Sigma,\ \tau_1, \tau_2 \in \textsc{Term}(\Sigma), \tau_1 \approx \tau_2 \in \Pi^*(\Sigma)$.

$\Longrightarrow\ \tau_1 \approx \tau_2 \notin \Sigma,\ \tau_1, \tau_2 \in \textsc{Term}(\Sigma), \tau_1 \approx \tau_2 \in \Pi^*(\Sigma),$
$\quad \tau_1 \approx \tau_1 \in \Pi(\Sigma), \tau_2 \approx \tau_2 \in \Pi(\Sigma)$ (Lemma 40)

$\Longrightarrow\ \tau_1 \approx \tau_1 \in \Pi(\Sigma), \tau_2 \approx \tau_2 \in \Pi(\Sigma),\ \tau_1, \tau_2 \in \textsc{Term}(\Sigma),$
$\quad (\tau_1 \approx \tau_1 \notin \Sigma$ or $\tau_2 \approx \tau_2 \notin \Sigma)$ (Lemma 46)

Without loosing generality, we choose $\tau_1 \approx \tau_1 \notin \Sigma$.

$\Longrightarrow\ \exists \alpha \in \mathfrak{A}, \varphi \in \mathfrak{V}, \tau_1' \in \textsc{Term}(\Sigma) : \tau_1 = \tau_1'\alpha,\ \tau_1' \approx \tau_1' \in \Sigma,$
$\quad \tau_1' \sim \varphi \in \Sigma, \mathfrak{S}\langle\varphi, \alpha\rangle \neq \emptyset, \tau_1 \in \textsc{Term}(\Sigma), \tau_1 \approx \tau_1 \notin \Sigma$ (Def. of Π_1)

$\Longrightarrow\ \tau_1 \approx \tau_1 \in \Sigma, \tau_1 \approx \tau_1 \notin \Sigma$ (Σ normal)

$\Longrightarrow$ Contradiction.

Therefore $\tau_1 \approx \tau_2 \notin \Pi^*(\Sigma)$.

The statement of the lemma can now be shown.
Let $\neg\tau_1 \approx \tau_2 \in \Pi^*(\Sigma)$.

$\Longrightarrow\ \neg\tau_1 \approx \tau_2 \in \Pi(\Sigma)$ (Def. of Π^*)

$\Longrightarrow\ \neg\tau_1 \approx \tau_2 \in \Sigma$ (Def. of Π)

$\Longrightarrow\ \tau_1 \approx \tau_2 \notin \Sigma, \tau_1, \tau_2 \in \textsc{Term}(\Sigma)$ (Σ normal)

$\Longrightarrow\ \tau_1 \approx \tau_2 \notin \Pi^*(\Sigma)$ (Claim above)

$\blacksquare$

The main flow of the argument

The next sequence of lemmata goes through the conditions for a normal set, checking one by one that the result of an application of Π^* to a normal set still obeys to all the conditions for a normal set. We do not check the conditions of a normal set as given in the original definition for it. Rather we use the alternative characterisation as provided in lemma 38. For the most part, this doesn't make any difference at all, since it is only the condition (5) which is formulated differently in the characterisation by lemma 38.

After this is done we show that iterated applications of Π^* still give a normal set. The last step then is to show that the closure of Π^* applications is indeed not only normal, but closed.

We start with condition (2) which demands symmetry of the relation $\approx$ to hold in $\Pi^*(\Sigma)$.

56. Lemma Let Σ be a normal set and $\tau_1, \tau_2 \in \mathfrak{T}$. If $\tau_1 \approx \tau_2 \in \Pi^*(\Sigma)$. Then $\tau_2 \approx \tau_1 \in \Pi^*(\Sigma)$.

Proof: We first prove by induction on i that for all i if $\tau_1 \approx \tau_2 \in \mathrm{T}_i(\Pi(\Sigma))$ then $\tau_2 \approx \tau_1 \in \mathrm{T}_i(\Pi(\Sigma))$. The rest will follow naturally by the definition of Π^*.

Base case: $i = 0 \Longrightarrow \mathrm{T}_i(\Pi(\Sigma)) = \Pi(\Sigma)$.
Let $\tau_1 \approx \tau_2 \in \Pi(\Sigma)$.

$$\begin{array}{ll}
\Longrightarrow \tau_1 \approx \tau_2 \in \Sigma \text{ or } \tau_1 \approx \tau_2 \in \Pi(\Sigma) \setminus \Sigma & \text{(Def. of } \Pi) \\
\Longrightarrow \tau_2 \approx \tau_1 \in \Sigma \text{ or } \tau_1 \approx \tau_2 \in \Pi(\Sigma) \setminus \Sigma & (\Sigma \text{ normal}) \\
\Longrightarrow \tau_2 \approx \tau_1 \in \Sigma \text{ or} & \\
\quad \exists \alpha \in \mathfrak{A}, \tau_1', \tau_2' \in \textsc{Term}(\Sigma) : \tau_1 = \tau_1'\alpha, \tau_2 = \tau_2'\alpha, \tau_1' \approx \tau_2' \in \Sigma & \text{(Def. of } \Pi_1) \\
\Longrightarrow \tau_2 \approx \tau_1 \in \Sigma \text{ or} & \\
\quad \exists \alpha \in \mathfrak{A}, \tau_1', \tau_2' \in \textsc{Term}(\Sigma) : \tau_1 = \tau_1'\alpha, \tau_2 = \tau_2'\alpha, \tau_2' \approx \tau_1' \in \Sigma & (\Sigma \text{ normal}) \\
\Longrightarrow \tau_2 \approx \tau_1 \in \Sigma \text{ or } \exists \alpha \in \mathfrak{A}, \tau_1', \tau_2' \in \textsc{Term}(\Sigma) : \tau_1 = \tau_1'\alpha, \tau_2 = \tau_2'\alpha, & \\
\qquad \tau_2'\alpha \approx \tau_1'\alpha \in \Pi_1(\Sigma) & \text{(Def. of } \Pi_1) \\
\Longrightarrow \tau_2 \approx \tau_1 \in \Pi(\Sigma). & \text{(Def. of } \Pi)
\end{array}$$

Step case:
Let $\tau_1 \approx \tau_2 \in \mathrm{T}_{i+1}(\Pi(\Sigma))$.

$$\begin{array}{ll}
\Longrightarrow \tau_1 \approx \tau_2 \in \mathrm{T}_i(\Pi(\Sigma)) \text{ or } \exists \tau_3 : \tau_1 \approx \tau_3, \tau_3 = \tau_2 \in \mathrm{T}_i(\Pi(\Sigma)) & \text{(Def. of T)} \\
\Longrightarrow \tau_2 \approx \tau_1 \in \mathrm{T}_i(\Pi(\Sigma)) \text{ or } \exists \tau_3 : \tau_1 \approx \tau_3, \tau_3 = \tau_2 \in \mathrm{T}_i(\Pi(\Sigma)) & \text{(Ind. hypothesis)} \\
\Longrightarrow \tau_2 \approx \tau_1 \in \mathrm{T}_i(\Pi(\Sigma)) \text{ or } \exists \tau_3 : \tau_3 \approx \tau_1, \tau_2 = \tau_3 \in \mathrm{T}_i(\Pi(\Sigma)) & \text{(Ind. hypothesis)} \\
\Longrightarrow \tau_2 \approx \tau_1 \in \mathrm{T}_i(\Pi(\Sigma)) \text{ or } \tau_2 \approx \tau_1 \in \mathrm{T}_{i+1}(\Pi(\Sigma)) & \text{(Def. of T)} \\
\Longrightarrow \tau_2 \approx \tau_1 \in \mathrm{T}_{i+1}(\Pi(\Sigma)). &
\end{array}$$

Thus we proved that $\forall i : \tau_1 \approx \tau_2 \in \mathrm{T}_i(\Pi(\Sigma)) \Longrightarrow \tau_2 \approx \tau_1 \in \mathrm{T}_i(\Pi(\Sigma))$. Now, let $\tau_1 \approx \tau_2 \in \Pi^*(\Sigma)$.

$$\begin{array}{ll}
\Longrightarrow \exists i : \tau_1 \approx \tau_2 \in \mathrm{T}_i(\Pi(\Sigma)) & \text{(Def. of } \Pi^*) \\
\Longrightarrow \exists i : \tau_2 \approx \tau_1 \in \mathrm{T}_i(\Pi(\Sigma)) & \text{(shown above)} \\
\Longrightarrow \tau_2 \approx \tau_1 \in \Pi^*(\Sigma). & \text{(Def. of } \Pi^*)
\end{array}$$

■

Next follows transitivity.

57. Lemma Let Σ be normal and $\tau_1, \tau_2, \tau_3 \in \mathfrak{T}$. Let $\tau_1 \approx \tau_2 \in \Pi^*(\Sigma)$ and $\tau_2 \approx \tau_3 \in \Pi^*(\Sigma)$. Then $\tau_1 \approx \tau_3 \in \Pi^*(\Sigma)$.

Proof: This follows directly from Π^* being the transitive closure of Π.
Let $\tau_1 \approx \tau_2, \tau_2 \approx \tau_3 \in \Pi^*(\Sigma)$.
$\Longrightarrow \exists i \in \mathbb{N} : \tau_1 \approx \tau_2, \tau_2 \approx \tau_3 \in \mathrm{T}_i(\Pi(\Sigma))$
$\Longrightarrow \exists i \in \mathbb{N} : \tau_1 \approx \tau_3 \in \mathrm{T}(\mathrm{T}_i(\Pi(\Sigma))) = \mathrm{T}_{i+1}(\Pi(\Sigma))$. ■

This is subterm reflexivity.

58. Lemma Let Σ be a normal set. Let $\sigma \in \mathfrak{A}^*$ and $\tau \in \mathfrak{T}$. If $\tau\sigma \approx \tau\sigma \in \Pi^*(\Sigma)$ then $\tau \approx \tau \in \Pi^*(\Sigma)$.

Proof: Let $\sigma \in \mathfrak{A}^*$. We can prove the claim by a simple case division.

Case 1: $\sigma = \epsilon$ (the empty path). Then $\tau \approx \tau \in \Pi^*(\Sigma)$ by premise and nothing needs to be shown.

Case 2: $\sigma = \sigma'\alpha$ for some $\alpha \in \mathfrak{A}$ and $\sigma' \in \mathfrak{A}^*$.
Let $\tau\sigma'\alpha \approx \tau\sigma'\alpha \in \Pi^*(\Sigma)$.

$\Longrightarrow \tau\sigma' \approx \tau\sigma' \in \Sigma$ (Lemma 42)

$\Longrightarrow \tau \approx \tau \in \Sigma$ (Σ normal)

$\Longrightarrow \tau \approx \tau \in \Pi^*(\Sigma)$ (Def. of Π^*)

■

The next lemma shows that Π^* is closed under a weaker version of right invariance, the condition (5) of a normal set. We do not proof this condition directly. Rather we make use of the equally strong condition (5) in the alternative definition of a normal set as given in lemma 38.

59. Lemma Let Σ be a normal set. Let $\tau_1, \tau_2 \in \mathfrak{T}$ and $\alpha \in \mathfrak{A}$. If $\tau_1 \approx \tau_2 \in \Pi^*(\Sigma), \tau_1\alpha \approx \tau_1\alpha \in \Pi^*(\Sigma), \tau_2\alpha \approx \tau_2\alpha \in \Pi^*(\Sigma)$, then $\tau_1\alpha \approx \tau_2\alpha \in \Pi^*(\Sigma)$.

Proof: We first prove three claims concerning three sub cases of the lemma's claim.

Claim 1: If $\tau_1 \approx \tau_2 \in \Sigma, \tau_1\alpha \approx \tau_1\alpha \in \Sigma, \tau_2\alpha \approx \tau_2\alpha \in \Pi^*(\Sigma) \setminus \Sigma$, then $\tau_1\alpha \approx \tau_2\alpha \in \Pi(\Sigma)$.

Proof of claim 1:
Let $\tau_1 \approx \tau_2 \in \Sigma, \tau_1\alpha \approx \tau_1\alpha \in \Sigma, \tau_2\alpha \approx \tau_2\alpha \in \Pi^*(\Sigma) \setminus \Sigma$.

$\Longrightarrow \tau_1 \approx \tau_2 \in \Sigma, \tau_1\alpha \approx \tau_1\alpha \in \Sigma, \tau_2\alpha \approx \tau_2\alpha \in \Pi(\Sigma) \setminus \Sigma$ (Lemma 40)

$\Longrightarrow \tau_1 \approx \tau_2 \in \Sigma, \tau_1\alpha \approx \tau_1\alpha \in \Sigma, \tau_2\alpha \approx \tau_2\alpha \in \Pi_1(\Sigma)$ (Def. of Π)

$\Longrightarrow \tau_1 \approx \tau_2 \in \Sigma, \tau_1\alpha \approx \tau_1\alpha \in \Sigma, \tau_2\alpha \approx \tau_2\alpha \in \Pi_1(\Sigma)$,
$\exists \varphi \in \mathfrak{V} : \tau_2 \sim \varphi \in \Sigma, \mathfrak{S}\langle\varphi, \alpha\rangle \neq \emptyset$ (Def. of Π_1)

$\Longrightarrow \tau_1 \approx \tau_2 \in \Sigma, \tau_1\alpha \approx \tau_1\alpha \in \Sigma, \tau_2\alpha \approx \tau_2\alpha \in \Pi_1(\Sigma)$,
$\exists \varphi \in \mathfrak{V} : \tau_1 \sim \varphi \in \Sigma, \tau_2 \sim \varphi \in \Sigma, \mathfrak{S}\langle\varphi, \alpha\rangle \neq \emptyset$ (Σ normal)

$\Longrightarrow \tau_1\alpha \approx \tau_2\alpha \in \Pi(\Sigma)$ (Def. of Π_1)

End of proof of claim 1.

Claim 2: If $\tau_1 \approx \tau_2 \in \Sigma, \tau_1\alpha \approx \tau_1\alpha \in \Pi^*(\Sigma) \setminus \Sigma, \tau_2\alpha \approx \tau_2\alpha \in \Sigma$, then $\tau_1\alpha \approx \tau_2\alpha \in \Pi(\Sigma)$.

The proof of claim 2 is complete analogous to the proof of claim 1 with the roles of τ_1 and τ_2 exchanged.

Claim 3: If $\tau_1 \approx \tau_2 \in \Sigma, \tau_1\alpha \approx \tau_1\alpha \in \Pi^*(\Sigma) \setminus \Sigma, \tau_2\alpha \approx \tau_2\alpha \in \Pi^*(\Sigma) \setminus \Sigma$, then $\tau_1\alpha \approx \tau_2\alpha \in \Pi(\Sigma)$.

Proof of claim 3:
Let $\tau_1 \approx \tau_2 \in \Sigma, \tau_1\alpha \approx \tau_1\alpha \in \Pi^*(\Sigma) \setminus \Sigma, \tau_2\alpha \approx \tau_2\alpha \in \Pi^*(\Sigma) \setminus \Sigma$.

$$\Longrightarrow \tau_1 \approx \tau_2 \in \Sigma, \tau_1\alpha \approx \tau_1\alpha \in \Pi(\Sigma) \setminus \Sigma, \tau_2\alpha \approx \tau_2\alpha \in \Pi(\Sigma) \setminus \Sigma \qquad \text{(Lemma 40)}$$
$$\Longrightarrow \tau_1 \approx \tau_2 \in \Sigma, \tau_1\alpha \approx \tau_1\alpha \in \Pi_1(\Sigma), \tau_2\alpha \approx \tau_2\alpha \in \Pi_1(\Sigma) \qquad \text{(Def. of } \Pi\text{)}$$
$$\Longrightarrow \tau_1 \approx \tau_2 \in \Sigma, \tau_1\alpha \approx \tau_1\alpha \in \Pi_1(\Sigma), \tau_2\alpha \approx \tau_2\alpha \in \Pi_1(\Sigma),$$
$$\exists \varphi_1, \varphi_2 \in \mathfrak{V} : \tau_1 \sim \varphi_1 \in \Sigma, \mathfrak{S}\langle \varphi_1, \alpha\rangle \neq \emptyset, \tau_2 \sim \varphi_2 \in \Sigma, \mathfrak{S}\langle \varphi_2, \alpha\rangle \neq \emptyset \qquad \text{(Def. of } \Pi_1\text{)}$$
$$\Longrightarrow \tau_1 \approx \tau_2 \in \Sigma, \exists \varphi_1, \varphi_2 \in \mathfrak{V} :$$
$$\tau_1 \sim \varphi_1 \in \Sigma, \mathfrak{S}\langle \varphi_1, \alpha\rangle \neq \emptyset, \tau_2 \sim \varphi_2 \in \Sigma, \mathfrak{S}\langle \varphi_2, \alpha\rangle \neq \emptyset, \varphi_1 = \varphi_2 \qquad (\Sigma \text{ normal})$$
$$\Longrightarrow \tau_1 \approx \tau_2 \in \Sigma, \exists \varphi_1 \in \mathfrak{V} : \tau_1 \sim \varphi_1 \in \Sigma, \tau_2 \sim \varphi_1 \in \Sigma, \mathfrak{S}\langle \varphi_1, \alpha\rangle \neq \emptyset$$
$$\Longrightarrow \tau_1\alpha \approx \tau_2\alpha \in \Pi(\Sigma) \qquad \text{(Def. of } \Pi_1\text{)}$$

End of proof of claim 3.

And now we can proof the main claim of the lemma.
Let $\tau_1 \approx \tau_2 \in \Pi^*(\Sigma), \tau_1\alpha \approx \tau_1\alpha \in \Pi^*(\Sigma), \tau_2\alpha \approx \tau_2\alpha \in \Pi^*(\Sigma)$.

$$\Longrightarrow \tau_1 \approx \tau_2 \in \Sigma, \tau_1\alpha \approx \tau_1\alpha \in \Pi^*(\Sigma), \tau_2\alpha \approx \tau_2\alpha \in \Pi^*(\Sigma) \qquad \text{(Lemma 47)}$$
$$\Longrightarrow (\tau_1 \approx \tau_2 \in \Sigma, \tau_1\alpha \approx \tau_1\alpha \in \Sigma, \tau_2\alpha \approx \tau_2\alpha \in \Sigma)$$
$$\vee (\tau_1 \approx \tau_2 \in \Sigma, \tau_1\alpha \approx \tau_1\alpha \in \Sigma, \tau_2\alpha \approx \tau_2\alpha \in \Pi^*(\Sigma) \setminus \Sigma)$$
$$\vee (\tau_1 \approx \tau_2 \in \Sigma, \tau_1\alpha \approx \tau_1\alpha \in \Pi^*(\Sigma) \setminus \Sigma, \tau_2\alpha \approx \tau_2\alpha \in \Sigma)$$
$$\vee (\tau_1 \approx \tau_2 \in \Sigma, \tau_1\alpha \approx \tau_1\alpha \in \Pi^*(\Sigma) \setminus \Sigma, \tau_2\alpha \approx \tau_2\alpha \in \Pi^*(\Sigma) \setminus \Sigma)$$
$$\Longrightarrow \tau_1\alpha \approx \tau_2\alpha \in \Sigma \qquad (\Sigma \text{ normal})$$
$$\vee\, \tau_1\alpha \approx \tau_2\alpha \in \Pi(\Sigma) \qquad \text{(Claim 1)}$$
$$\vee\, \tau_1\alpha \approx \tau_2\alpha \in \Pi(\Sigma) \qquad \text{(Claim 2)}$$
$$\vee\, \tau_1\alpha \approx \tau_2\alpha \in \Pi(\Sigma) \qquad \text{(Claim 3)}$$
$$\Longrightarrow \tau_1\alpha \approx \tau_2\alpha \in \Pi^*(\Sigma) \qquad \text{(Def. of } \Pi^*\text{)}$$

■

Now follows the check of condition (6) which demands that if a term is defined in $\Pi^*(\Sigma)$ then it must receive a variety.

60. Lemma Let Σ be a normal set and $\tau \in \mathfrak{T}$. If $\tau \approx \tau \in \Pi^*(\Sigma)$, then there exists a $\varphi \in \mathfrak{V}$ with $\tau \sim \varphi \in \Pi^*(\Sigma)$.

Proof: This is an immediate consequence of lemma 41 and the fact that $\Pi(\Sigma) \subseteq \Pi^*(\Sigma)$. ■

The next lemma handles condition (7) of normal sets which demands that a term which receives a variety must be defined. This is the inverse of condition (6).

61. Lemma Let Σ be a normal set and $\tau \in \mathfrak{T}$. If $\exists \varphi \in \mathfrak{V} : \tau \sim \varphi \in \Pi^*(\Sigma)$ then $\tau \approx \tau \in \Pi^*(\Sigma)$.

Proof: This is an immediate consequence of lemma 48 and the fact that $\Pi(\Sigma) \subseteq \Pi^*(\Sigma)$. ■

It follows the check of condition (8) which requires that variety assignment be compatible with path equations.

62. Lemma Let Σ be a normal set. Let $\tau_1, \tau_2 \in \mathfrak{T}$ and $\varphi_1, \varphi_2 \in \mathfrak{V}$. If $\tau_1 \approx \tau_2 \in \Pi^*(\Sigma), \tau_1 \sim \varphi_1 \in \Pi^*(\Sigma), \tau_2 \sim \varphi_2 \in \Pi^*(\Sigma)$', then it must be the case that $\varphi_1 = \varphi_2$.

Proof: We prove first by induction a minor variant of the statement of the lemma.

Claim: If $\tau_1 \approx \tau_2 \in \mathrm{T}_i(\Pi(\Sigma)), \tau_1 \sim \varphi_1 \in \Pi^*(\Sigma), \tau_2 \sim \varphi_2 \in \Pi^*(\Sigma)$, then $\varphi_1 = \varphi_2$.

Proof of the claim:
Base case: $i = 0 \Longrightarrow \mathrm{T}_i(\Pi(\Sigma)) = \Pi(\Sigma)$.
Let $\tau_1 \approx \tau_2 \in \Pi(\Sigma), \tau_1 \sim \varphi_1 \in \Pi^*(\Sigma), \tau_2 \sim \varphi_2 \in \Pi^*(\Sigma)$.
$\Longrightarrow \tau_1 \approx \tau_2 \in \Pi(\Sigma), \tau_1 \sim \varphi_1 \in \Pi(\Sigma), \tau_2 \sim \varphi_2 \in \Pi(\Sigma)$ (Def. of Π^*)
$\Longrightarrow \varphi_1 = \varphi_2$ (Lemma 50)
Step case: Let $\tau_1 \approx \tau_2 \in \mathrm{T}_{i+1}(\Pi(\Sigma)), \tau_1 \sim \varphi_1 \in \Pi^*(\Sigma), \tau_2 \sim \varphi_2 \in \Pi^*(\Sigma)$.
$\Longrightarrow \tau_1 \approx \tau_2 \in \mathrm{T}_i(\Pi(\Sigma)), \tau_1 \sim \varphi_1 \in \Pi^*(\Sigma), \tau_2 \sim \varphi_2 \in \Pi^*(\Sigma)$ or
$\exists \tau_3 \in \mathrm{TERM}(\Pi(\Sigma)) : \tau_1 \approx \tau_3 \in \mathrm{T}_i(\Pi(\Sigma)), \tau_3 \approx \tau_2 \in \mathrm{T}_i(\Pi(\Sigma))$,
$\tau_1 \sim \varphi_1 \in \Pi^*(\Sigma), \tau_2 \sim \varphi_2 \in \Pi^*(\Sigma)$ (Def. of T)
$\Longrightarrow \varphi_1 = \varphi_2$ or (Ind. hypothesis)
$\exists \tau_3 \in \mathrm{TERM}(\Pi(\Sigma)) : \tau_1 \approx \tau_3 \in \mathrm{T}_i(\Pi(\Sigma)), \tau_3 \approx \tau_2 \in \mathrm{T}_i(\Pi(\Sigma))$,
$\tau_1 \sim \varphi_1 \in \Pi^*(\Sigma), \tau_2 \sim \varphi_2 \in \Pi^*(\Sigma), \tau_3 \approx \tau_3 \in \mathrm{T}_i(\Pi(\Sigma))$ (Lemma 39)
$\Longrightarrow \varphi_1 = \varphi_2$ or
$\exists \tau_3 \in \mathrm{TERM}(\Pi(\Sigma)), \varphi_3 \in \mathfrak{V} : \tau_1 \approx \tau_3 \in \mathrm{T}_i(\Pi(\Sigma)), \tau_3 \approx \tau_2 \in \mathrm{T}_i(\Pi(\Sigma))$,
$\tau_1 \sim \varphi_1 \in \Pi^*(\Sigma), \tau_2 \sim \varphi_2 \in \Pi^*(\Sigma), \tau_3 \approx \tau_3 \in \mathrm{T}_i(\Pi(\Sigma)), \tau_3 \sim \varphi_3 \in \Pi^*(\Sigma)$
(Lemma 41)
$\Longrightarrow \varphi_1 = \varphi_2$ or
$\varphi_1 = \varphi_3, \varphi_3 = \varphi_2$ (Induction hypothesis)
$\Longrightarrow \varphi_1 = \varphi_2$
End of the proof of the claim.

The main claim of the lemma is now a direct consequence of the claim above:
Let $\tau_1 \approx \tau_2 \in \Pi^*(\Sigma), \tau_1 \sim \varphi_1 \in \Pi^*(\Sigma), \tau_2 \sim \varphi_2 \in \Pi^*(\Sigma)$.
$\Longrightarrow \exists i : \tau_1 \approx \tau_2 \in \mathrm{T}_i(\Pi(\Sigma)), \tau_1 \sim \varphi_1 \in \Pi^*(\Sigma), \tau_2 \sim \varphi_2 \in \Pi^*(\Sigma)$ (Def. of Π^*)
$\Longrightarrow \varphi_1 = \varphi_2$ (Claim above)

■

This lemma deals with condition (9) which prescribes that the description be well-typed. This is another way of saying that the description obeys to the appropriateness restrictions.

63. Lemma Let Σ be a normal set. Let $\tau \in \mathfrak{T}, \alpha \in \mathfrak{A}$ and $\varphi_1, \varphi_2 \in \mathfrak{V}$. If $\tau \sim \varphi_1 \in \Pi^*(\Sigma)$ and $\tau\alpha \sim \varphi_2 \in \Pi^*(\Sigma)$, then $\varphi_2 \in \mathfrak{S}\langle\varphi_1, \alpha\rangle$.

Proof: Let $\tau \sim \varphi_1, \tau\alpha \sim \varphi_2 \in \Pi^*(\Sigma)$.

$\Longrightarrow \tau \sim \varphi_1, \tau\alpha \sim \varphi_2 \in \Pi(\Sigma)$ (Def. of T)

$\Longrightarrow \tau \sim \varphi_1, \tau\alpha \sim \varphi_2, \tau\alpha \approx \tau\alpha \in \Pi(\Sigma)$ (Lemma 48)

$\Longrightarrow \tau \sim \varphi_1, \tau\alpha \sim \varphi_2 \in \Pi(\Sigma), \tau \approx \tau \in \Sigma$ (Def. of Π_1)

$\Longrightarrow \tau \approx \tau, \tau \sim \varphi_1 \in \Sigma, \tau\alpha \sim \varphi_2 \in \Pi(\Sigma)$ (Σ normal)

Now, since the literal $\tau\alpha \sim \varphi_2$ can have three different origins according to the definition of Π, we must check these three cases separately.

Case 1: Let $\tau \approx \tau, \tau \sim \varphi_1 \in \Sigma, \tau\alpha \sim \varphi_2 \in \Sigma$.

$\Longrightarrow \varphi_2 \in \mathfrak{S}\langle\varphi_1, \alpha\rangle$ (Σ normal)

Case 2: Let $\tau \approx \tau, \tau \sim \varphi_1 \in \Sigma, \tau\alpha \sim \varphi_2 \in \Pi_2(\Sigma)$.

$\Longrightarrow \tau \sim \varphi_1 \in \Sigma, \tau\alpha \sim \varphi_2 \in \Pi_2(\Sigma), \exists\tau_3 : \tau_3 \approx \tau \in \Sigma, \tau_3\alpha \sim \varphi_2 \in \Sigma$ (Def. of Π_2)

$\Longrightarrow \exists\tau_3 : \tau_3\alpha \sim \varphi_2, \tau_3 \sim \varphi_1 \in \Sigma$ (Σ normal)

$\Longrightarrow \varphi_2 \in \mathfrak{S}\langle\varphi_1, \alpha\rangle$ (Σ normal)

Case 3: Let $\tau \approx \tau, \tau \sim \varphi_1 \in \Sigma, \tau\alpha \sim \varphi_2 \in \Pi_3(\Sigma)$.

$\Longrightarrow \tau \approx \tau, \tau \sim \varphi_1 \in \Sigma, \tau\alpha \sim \varphi_2 \in \Pi_3(\Sigma),$
$\quad \exists\varphi_3 : \tau \sim \varphi_3 \in \Sigma, \varphi_2 = \textsc{Choose}(\mathfrak{S}\langle\varphi_3, \alpha\rangle)$ (Def. of Π_3)

$\Longrightarrow \tau \approx \tau, \tau \sim \varphi_1 \in \Sigma, \exists\varphi_3 : \tau \sim \varphi_3 \in \Sigma, \varphi_2 \in \mathfrak{S}\langle\varphi_3, \alpha\rangle$ (Def. of Choose)

$\Longrightarrow \varphi_1 = \varphi_3, \varphi_2 \in \mathfrak{S}\langle\varphi_1, \alpha\rangle$ (Σ normal)

And this conclude the proof, since we showed that in all case $\varphi_2 \in \mathfrak{S}\langle\varphi_1, \alpha\rangle$ holds. ∎

Here we come to the check of condition (10). We explained earlier that this condition for normal sets is a weaker variant of totally well-typedness.

64. Lemma Let Σ be a normal set. Let $\tau \in \mathfrak{T}, \alpha \in \mathfrak{A}$ and $\varphi \in \mathfrak{V}$. If $\tau \sim \varphi \in \Pi^*(\Sigma), \tau\alpha \in \textsc{Term}(\Pi^*(\Sigma))$ and $\mathfrak{S}\langle\varphi, \alpha\rangle \neq \emptyset$, then $\tau\alpha \approx \tau\alpha \in \Pi^*(\Sigma)$.

Proof: Let $\tau \sim \varphi \in \Pi^*(\Sigma), \tau\alpha \in \textsc{Term}(\Pi^*(\Sigma)),\ \mathfrak{S}\langle\varphi, \alpha\rangle \neq \emptyset$

$\Longrightarrow \tau \sim \varphi \in \Pi(\Sigma), \tau\alpha \in \textsc{Term}(\Pi^*(\Sigma)), \mathfrak{S}\langle\varphi, \alpha\rangle \neq \emptyset$ (Def. of T)

$\Longrightarrow \tau \sim \varphi \in \Pi(\Sigma), \tau\alpha \in \textsc{Term}(\Pi(\Sigma)), \mathfrak{S}\langle\varphi, \alpha\rangle \neq \emptyset$ (Def. of T)

$\Longrightarrow (\tau\alpha \in \textsc{Term}(\Pi(\Sigma)) \setminus \textsc{Term}(\Sigma), \tau \sim \varphi \in \Pi(\Sigma), \mathfrak{S}\langle\varphi, \alpha\rangle \neq \emptyset)$
$\quad \vee (\tau\alpha \in \textsc{Term}(\Sigma), \tau \sim \varphi \in \Pi(\Sigma), \mathfrak{S}\langle\varphi, \alpha\rangle \neq \emptyset)$ (Def. of Π)

$\Longrightarrow \tau\alpha \approx \tau\alpha \in \Pi(\Sigma)$
$\quad \vee (\tau\alpha \in \textsc{Term}(\Sigma), \tau \sim \varphi \in \Pi(\Sigma), \mathfrak{S}\langle\varphi, \alpha\rangle \neq \emptyset)$ (Lemma 53)

$\Longrightarrow \tau\alpha \approx \tau\alpha \in \Pi(\Sigma) \vee \tau\alpha \approx \tau\alpha \in \Sigma$ (Lemma 51)

$\Longrightarrow \tau\alpha \approx \tau\alpha \in \Pi^*(\Sigma)$

■

The final condition (11) demands consistency of negation.

65. Lemma Let Σ be a normal set and δ be a literal. If $\neg\delta \in \Pi^*(\Sigma)$, then $\delta \notin \Pi^*(\Sigma)$.

Proof: This is an immediate consequence of the lemmata 54 and 55. ■

Now we can sum up what we have done so far and formulate a first important result.

66. Lemma If Σ is normal, then $\Pi^*(\Sigma)$ is normal.

Proof: We have to check conditions (1) – (11) for a normal set. Now, since $:\approx:\in \Sigma \subseteq \Pi^*(\Sigma)$, condition (1) obviously holds. We checked conditions (2) – (11) of the alternative definition of a normal set given in lemma 38. Lemmata 56 to 65 show that these conditions all hold for $\Pi^*(\Sigma)$. Therefore $\Pi^*(\Sigma)$ is normal by virtue of lemma 38. ■

67. Lemma If Σ is normal, then all $\Xi_i(\Sigma)$ are normal.

Proof: By simple induction on i. Let Σ be normal. Then $\Xi_0(\Sigma) = \Sigma$ is normal by premise. $\Xi_{i+1}(\Sigma) = \Pi^*(\Xi_i(\Sigma))$ is normal, because Π^* preserves being normal (lemma 66) while $\Xi_i(\Sigma)$ is normal by induction hypothesis. ■

68. Lemma If Σ is normal, then $\Xi(\Sigma)$ is normal.

Proof: The previous lemma shows that all $\Xi_i(\Sigma)$ are normal. Furthermore $\forall i, j : i \leq j \Longrightarrow \Xi_i(\Sigma) \subseteq \Xi_j(\Sigma)$. Thus holds for all n that $\bigcup_{i=0}^{n} \Xi_i(\Sigma) = \Xi_n(\Sigma)$ is normal. In order to see that condition (10) holds for $\Xi(\Sigma)$, we observe that $\textsc{Term}(\Xi(\Sigma)) \subseteq \bigcup_{i\in\omega} \textsc{Term}(\Xi_i(\Sigma))$ by definition of $\textsc{Term}$.

We show that condition (11) holds by a simple proof with contradiction. Suppose there is a literal δ with $\delta \in \Xi(\Sigma), \neg\delta \in \Xi(\Sigma)$.

$\Longrightarrow \exists n \in \omega : \delta \in \Xi_n(\Sigma), \exists m \in \omega : \neg\delta \in \Xi_m(\Sigma)$ (Def. of Ξ)

$\Longrightarrow \exists k_{nm} \in \omega : \delta \in \Xi_{k_{nm}}(\Sigma), \neg\delta \in \Xi_{k_{nm}}(\Sigma)$ (Def. of the Ξ_i)

$\Longrightarrow \exists k_{nm} \in \omega : \delta \in \Xi_{k_{nm}}(\Sigma), \delta \notin \Xi_{k_{nm}}(\Sigma)$ ($\Xi_{k_{nm}}(\Sigma)$ normal)

$\Longrightarrow$ Contradiction.

Thus $\neg\delta \in \Xi(\Sigma)$ implies $\delta \notin \Xi(\Sigma)$.
Therefore $\Xi(\Sigma) = \bigcup_{i\in\omega} \Xi_i(\Sigma)$ is normal, too. ■

Finally, we can proof the claim of the whole section which is that a normal set can be embedded in a closed set.

69. Proposition Let Σ be a normal set of literals. Then $\Xi(\Sigma)$ is a closed set of literals which contains Σ.

Proof: The last lemma shows that $\Sigma \subseteq \Xi(\Sigma)$ and that $\Xi(\Sigma)$ is normal. So what remains to be proven is $(\tau, \tau_1, \tau_2 \in \mathfrak{T}, \varphi \in \mathfrak{V}, \alpha \in \mathfrak{A}, \sigma \in \mathfrak{A}^*)$
(1) if $\tau \sim \varphi \in \Xi(\Sigma)$ and $\mathfrak{S}\langle\varphi, \alpha\rangle \neq \emptyset$, then $\tau\alpha \approx \tau\alpha \in \Xi(\Sigma)$
(2) if $\tau_1 \approx \tau_2, \tau_1\sigma \approx \tau_1\sigma \in \Xi(\Sigma)$, then $\tau_1\sigma \approx \tau_2\sigma \in \Xi(\Sigma)$.

First on (1). Let $\tau \sim \varphi \in \Xi(\Sigma), \mathfrak{S}\langle\varphi, \alpha\rangle \neq \emptyset$ and $\tau\alpha \approx \tau\alpha \notin \Xi(\Sigma)$.

$\Longrightarrow$ $\tau\alpha \approx \tau\alpha \notin \Xi(\Sigma), \exists i \in \mathbb{N} : \tau \sim \varphi \in \Xi_i(\Sigma), \mathfrak{S}\langle\varphi, \alpha\rangle \neq \emptyset, \tau\alpha \approx \tau\alpha \notin \Xi_i(\Sigma)$ (Def. of Ξ)

$\Longrightarrow$ $\tau\alpha \approx \tau\alpha \notin \Xi(\Sigma)$,
$\exists i \in \mathbb{N} : \tau \sim \varphi \in \Xi_i(\Sigma), \mathfrak{S}\langle\varphi, \alpha\rangle \neq \emptyset, \tau\alpha \approx \tau\alpha \notin \Xi_i(\Sigma), \tau \approx \tau \in \Xi_i(\Sigma)$ ($\Xi_i(\Sigma)$ normal)

$\Longrightarrow$ $\tau\alpha \approx \tau\alpha \notin \Xi(\Sigma), \exists i \in \mathbb{N} : \tau\alpha \approx \tau\alpha \in \Pi_1(\Xi_i(\Sigma))$ (Def. of Π_1)

$\Longrightarrow$ $\tau\alpha \approx \tau\alpha \notin \Xi(\Sigma), \tau\alpha \approx \tau\alpha \in \Xi(\Sigma)$ (Def. of Ξ)

$\Longrightarrow$ Contradiction.

Therefore $\tau\alpha \approx \tau\alpha \in \Xi(\Sigma)$.

Now on (2), the extended requirements on path equations. We prove (2) by induction on the length of the attribute path σ.

Base case: Let $\sigma = \epsilon$ the empty path. Then nothing needs to be shown.

Step case: $\sigma \in \mathfrak{A}^* \setminus \{\epsilon\}$.
Let $\tau_1 \approx \tau_2 \in \Xi(\Sigma), \tau_1\sigma \approx \tau_1\sigma \in \Xi(\Sigma)$.

$\Longrightarrow$ $\exists \sigma_1 \in \mathfrak{A}^*, \sigma_2 \in \mathfrak{A} : \sigma = \sigma_1\sigma_2,\ \tau_1 \approx \tau_2 \in \Xi(\Sigma), \tau_1\sigma_1\sigma_2 \approx \tau_1\sigma_1\sigma_2 \in \Xi(\Sigma)$

$\Longrightarrow$ $\exists \sigma_1 \in \mathfrak{A}^*, \sigma_2 \in \mathfrak{A} : \sigma = \sigma_1\sigma_2,\ \tau_1 \approx \tau_2 \in \Xi(\Sigma), \tau_1\sigma_1\sigma_2 \approx \tau_1\sigma_1\sigma_2 \in \Xi(\Sigma)$,
$\tau_1\sigma_1 \approx \tau_1\sigma_1 \in \Xi(\Sigma)$ ($\Xi(\Sigma)$ normal)

$\Longrightarrow$ $\exists \sigma_1 \in \mathfrak{A}^*, \sigma_2 \in \mathfrak{A} : \sigma = \sigma_1\sigma_2,\ \tau_1 \approx \tau_2 \in \Xi(\Sigma), \tau_1\sigma_1\sigma_2 \approx \tau_1\sigma_1\sigma_2 \in \Xi(\Sigma)$,
$\tau_1\sigma_1 \approx \tau_1\sigma_1 \in \Xi(\Sigma), \tau_1\sigma_1 \approx \tau_2\sigma_1 \in \Xi(\Sigma)$ (Ind. hypothesis)

$\Longrightarrow$ $\exists \sigma_1 \in \mathfrak{A}^*, \sigma_2 \in \mathfrak{A} : \sigma = \sigma_1\sigma_2,\ \tau_1\sigma_1\sigma_2 \approx \tau_1\sigma_1\sigma_2 \in \Xi(\Sigma)$,
$\tau_1\sigma_1 \approx \tau_1\sigma_1 \in \Xi(\Sigma), \tau_1\sigma_1 \approx \tau_2\sigma_1 \in \Xi(\Sigma), \tau_2\sigma_1 \approx \tau_2\sigma_1 \in \Xi(\Sigma)$ ($\Xi(\Sigma)$ normal)

$\Longrightarrow$ $\exists \sigma_1 \in \mathfrak{A}^*, \sigma_2 \in \mathfrak{A}, \varphi \in \mathfrak{V} : \sigma = \sigma_1\sigma_2,\ \tau_1\sigma_1\sigma_2 \approx \tau_1\sigma_1\sigma_2 \in \Xi(\Sigma)$,
$\tau_1\sigma_1 \approx \tau_1\sigma_1 \in \Xi(\Sigma), \tau_1\sigma_1 \approx \tau_2\sigma_1 \in \Xi(\Sigma), \tau_2\sigma_1 \approx \tau_2\sigma_1 \in \Xi(\Sigma)$,
$\tau_1\sigma_1 \sim \varphi \in \Xi(\Sigma), \tau_2\sigma_1 \sim \varphi \in \Xi(\Sigma), \mathfrak{S}\langle\varphi, \sigma_2\rangle \neq \emptyset$ ($\Xi(\Sigma)$ normal)

$\Longrightarrow$ $\exists \sigma_1 \in \mathfrak{A}^*, \sigma_2 \in \mathfrak{A} : \sigma = \sigma_1\sigma_2,\ \tau_1\sigma_1\sigma_2 \approx \tau_1\sigma_1\sigma_2 \in \Xi(\Sigma)$,
$\tau_1\sigma_1 \approx \tau_2\sigma_1 \in \Xi(\Sigma), \tau_2\sigma_1\sigma_2 \approx \tau_2\sigma_1\sigma_2 \in \Xi(\Sigma)$ ((1) above)

$\Longrightarrow$ $\exists \sigma_1 \in \mathfrak{A}^*, \sigma_2 \in \mathfrak{A} : \sigma = \sigma_1\sigma_2, \tau_1\sigma_1\sigma_2 \approx \tau_2\sigma_1\sigma_2 \in \Xi(\Sigma)$ ($\Xi(\Sigma)$ normal)

And this concludes our proof that $\Xi(\Sigma)$ is closed. ∎

11 Satisfiability Proof for Closed Sets

Our next task is to show that a closed set is satisfiable. We do this in two steps. First, we construct an appropriate $\mathfrak{S}$-structure, then we show that the denotation of the set using the newly constructed $\mathfrak{S}$-structure is non-empty.

70. Definition Given a closed set Σ of literals we construct an $\mathfrak{S}$-structure which satisfies Σ as follows. We define

$|\tau|_\Sigma := \{\tau' \mid \tau \approx \tau' \in \Sigma\}$ the equivalence class of τ;
$U_\Sigma := \{|\tau|_\Sigma \mid \tau \approx \tau \in \Sigma\}$ the universe of the equivalence classes in Σ;
$V_\Sigma := \{\langle|\tau|_\Sigma, \varphi\rangle \mid \tau \sim \varphi \in \Sigma\}$ the variety assignment;
$A_\Sigma(\alpha) := \{\langle|\tau|_\Sigma, |\tau\alpha|_\Sigma\rangle \mid \tau \approx \tau, \tau\alpha \approx \tau\alpha \in \Sigma\}$ the attribute interpretation; and
$S_\Sigma := \langle U_\Sigma, V_\Sigma, A_\Sigma\rangle$.

71. Lemma $S_\Sigma = \langle U_\Sigma, V_\Sigma, A_\Sigma\rangle$ as defined above is an $\mathfrak{S}$-structure.

Proof: Symmetry and transitivity are amongst the conditions of a closed set. Furthermore there is one specific variety that every member of a given equivalence class receives: Let $\tau' \in |\tau|_\Sigma$.

$\Longrightarrow \tau \approx \tau, \tau \approx \tau' \in \Sigma$
$\Longrightarrow \tau \approx \tau, \tau \approx \tau', \tau' \approx \tau' \in \Sigma$ (Σ closed)
$\Longrightarrow \tau \approx \tau, \tau \approx \tau', \tau' \approx \tau' \in \Sigma,$
$\exists \varphi, \varphi' \in \mathfrak{V} : \tau \sim \varphi \in \Sigma, \tau' \sim \varphi' \in \Sigma$ (Σ closed)
$\Longrightarrow \tau \approx \tau, \tau \approx \tau', \tau' \approx \tau' \in \Sigma,$
$\exists \varphi, \varphi' \in \mathfrak{V} : \tau \sim \varphi \in \Sigma, \tau' \sim \varphi' \in \Sigma,$
$\varphi_1 = \varphi_2$ (Σ closed)
$\Longrightarrow \tau \approx \tau, \tau \approx \tau', \tau' \approx \tau' \in \Sigma, \exists \varphi \in \mathfrak{V} : \tau \sim \varphi, \tau' \sim \varphi \in \Sigma$
$\Longrightarrow \exists \varphi \in V\ \forall \tau' \in |\tau|_\Sigma : \tau' \sim \varphi \in \Sigma.$

Therefore the equivalence class construction of U_Σ makes sense. $V_\Sigma : U_\Sigma \to \mathfrak{V}$ is a variety assignment function: Let $|\tau|_\Sigma \in U_\Sigma$. Then there is a $\varphi \in \mathfrak{V}$ with $\tau \sim \varphi \in \Sigma$. Therefore each object in the universe receives a variety, V_Σ is total. By the above it is also well-defined: Let $\tau_1, \tau_2 \in |\tau|_\Sigma$.

$\Longrightarrow \exists \varphi \in \mathfrak{V} : \tau \sim \varphi, \tau_1 \sim \varphi, \tau_2 \sim \varphi \in \Sigma$
$\Longrightarrow V_\Sigma(\tau_1) = V_\Sigma(\tau_2) = V_\Sigma(\tau) = \varphi.$

Next we check that A_Σ is a attribute interpretation function which obeys the two rules connected with it in the definition of an $\mathfrak{S}$-structure. As we can see from the definition of A_Σ, there can be attributes α for which $A_\Sigma(\alpha) = \emptyset$ since the attribute may be defined on no object of our universe. Let's suppose $A_\Sigma(\alpha) \neq \emptyset$. Let $\langle v, v'\rangle \in A_\Sigma(\alpha)$. Then by definition of A_Σ there exists a τ with $v = |\tau|_\Sigma, v' = |\tau\alpha|_\Sigma, \tau \approx \tau \in \Sigma$ and $\tau\alpha \approx \tau\alpha \in \Sigma$. Therefore are both v and $v' \in U_\Sigma$. Furthermore let both $\langle v, v_1\rangle$ and $\langle v, v_2\rangle \in A_\Sigma(\alpha)$. By definition of

A_Σ there is a τ such that $v = |\tau|_\Sigma$ and $v_1 = |\tau\alpha|_\Sigma$ as well as $v_2 = |\tau\alpha|_\Sigma$. Thus $v_1 = v_2$. This shows that $A_\Sigma(\alpha) : U \rightharpoonup U$ is indeed a partial function from U to U. Now we check conditions S1 and S2 of the definition of an $\mathfrak{S}$-structure. Firstly let $\langle v_1, v_2\rangle \in A_\Sigma(\alpha), V_\Sigma(v_1) = \varphi_1, V_\Sigma(v_2) = \varphi_2$.

$$\begin{aligned}
&\Longrightarrow \exists\tau \in \mathfrak{T} : v_1 = |\tau|_\Sigma, v_2 = |\tau\alpha|_\Sigma, \tau \approx \tau \in \Sigma, \tau\alpha \approx \tau\alpha \in \Sigma, \\
&\qquad V_\Sigma(v_1) = \varphi_1, V_\Sigma(v_2) = \varphi_2 && \text{(Def. of } A_\Sigma(\alpha)\text{)}\\
&\Longrightarrow \exists\tau \in \mathfrak{T} : \tau \approx \tau \in \Sigma, \tau\alpha \approx \tau\alpha \in \Sigma, V_\Sigma(|\tau|_\Sigma) = \varphi_1, V_\Sigma(|\tau\alpha|_\Sigma) = \varphi_2 \\
&\Longrightarrow \exists\tau \in \mathfrak{T} : \tau \approx \tau \in \Sigma, \tau\alpha \approx \tau\alpha \in \Sigma, \tau \sim \varphi_1 \in \Sigma, \tau\alpha \sim \varphi_2 \in \Sigma && \text{(Def. of } V_\Sigma\text{)}\\
&\Longrightarrow \varphi_2 \in \mathfrak{S}\langle\varphi_1, \alpha\rangle && (\Sigma \text{ closed})\\
&\Longrightarrow V_\Sigma(A_\Sigma(\alpha)(v_1)) \in \mathfrak{S}\langle V_\Sigma(v_1), \alpha\rangle.
\end{aligned}$$

Secondly, let $V_\Sigma(v) = \varphi$ and $\mathfrak{S}\langle\varphi, \alpha\rangle \neq \emptyset$.

$$\begin{aligned}
&\Longrightarrow \exists\tau \in \mathfrak{T} : v = |\tau|_\Sigma, \tau \sim \varphi \in \Sigma, \mathfrak{S}\langle\varphi, \alpha\rangle \neq \emptyset && \text{(Def. of } V_\Sigma\text{)}\\
&\Longrightarrow \exists\tau \in \mathfrak{T} : v = |\tau|_\Sigma, \tau \sim \varphi \in \Sigma, \mathfrak{S}\langle\varphi, \alpha\rangle \neq \emptyset, \tau\alpha \approx \tau\alpha \in \Sigma && (\Sigma \text{ closed})\\
&\Longrightarrow \langle|\tau|_\Sigma, |\tau\alpha|_\Sigma\rangle \in A_\Sigma(\alpha), v = |\tau|_\Sigma && \text{(Def. of } A_\Sigma(\alpha)\text{)}\\
&\Longrightarrow A_\Sigma(\alpha) \text{ is defined on } v.
\end{aligned}$$

We have shown that A_Σ is a suitable attribute interpretation, and this concludes the proof of the lemma. ∎

The $\mathfrak{S}$-structure we provide is very similar to the standard term model in a classical logic. A term is interpreted as the equivalence class of all the terms it is equal with. An attribute is thus taken as the partial function from the class of terms for which this attribute is appropriate to the class which is reached by suffixing the terms of the domain class by the attribute. This works because of the following. A unique variety is assigned to all of the terms in one class. Therefore an attribute is appropriate for a class, if it is appropriate for the variety the class receives. And since each appropriate attribute must be present, we know that there is a class which consists of all the terms in the domain class suffixed by the attribute. Indirectly it is already expressed, that the variety assignment function takes each term class to the variety it gets assigned in the description.

Before we go on to show that a closed Σ is satisfiable, we need a few lemmata.

72. Lemma Let Σ be a closed set and $\sigma \in \mathfrak{A}^*$ an attribute path. Then $[:\sigma] = \{\langle|\tau|_\Sigma, |\tau\sigma|_\Sigma\rangle \mid \tau \approx \tau \in \Sigma, \tau\sigma \approx \tau\sigma \in \Sigma\}$.

Proof: By induction on the length of the attribute path σ.
If $\sigma = \epsilon$ the empty path, then $[:] = \{\langle|\tau|_\Sigma, |\tau|_\Sigma\rangle \mid \tau \approx \tau \in \Sigma\}$ which is obviously the identity function on U as demanded by definition 4, the definition of term interpretation.

If $\sigma = \sigma'\alpha$ where $\sigma' \in \mathfrak{A}^*$ and $\alpha \in \mathfrak{A}$,
then $[:\sigma] = [:\sigma'\alpha]$

$$\begin{aligned}
&= A_\Sigma(\alpha) \circ [:\sigma'] && \text{(Def. 4)}\\
&= \{\langle|\tau|_\Sigma, |\tau\alpha|_\Sigma\rangle \mid \tau \approx \tau \in \Sigma, \tau\alpha \approx \tau\alpha \in \Sigma\} && \text{(Def. of } A_\Sigma\text{)}\\
&\qquad \circ \{\langle|\tau|_\Sigma, |\tau\sigma'|_\Sigma\rangle \mid \tau \approx \tau \in \Sigma, \tau\sigma' \approx \tau\sigma' \in \Sigma\} && \text{(Ind. hypothesis)}
\end{aligned}$$

$$= \{\langle|\tau|_\Sigma, |\tau\sigma'\alpha|_\Sigma\rangle \mid \tau \approx \tau \in \Sigma, \tau\sigma'\alpha \approx \tau\sigma'\alpha \in \Sigma\}$$
$$= \{\langle|\tau|_\Sigma, |\tau\sigma|_\Sigma\rangle \mid \tau \approx \tau \in \Sigma, \tau\sigma \approx \tau\sigma \in \Sigma\}$$

■

73. Lemma Let Σ be a closed set and $\tau \in \mathfrak{T}$. If $\tau \approx \tau \in \Sigma$, then $[\tau]_{S_\Sigma}(|{:}|_\Sigma) = |\tau|$.

Proof: Let $\tau \approx \tau \in \Sigma$. There is exactly one $\tau' \in \mathfrak{A}^*$ with $\tau = {:}\tau'$.
Now $[\tau]_{S_\Sigma}(|{:}|_\Sigma) = [{:}\tau']_{S_\Sigma}(|{:}|_\Sigma)$

$$= \{\langle|\kappa|_\Sigma, |\kappa\tau'|_\Sigma\rangle \mid \kappa \approx \kappa \in \Sigma, \kappa\tau' \approx \kappa\tau' \in \Sigma\}(|{:}|_\Sigma) \qquad \text{(Lemma 72)}$$
$$= (\{\langle|{:}|_\Sigma, |{:}\tau'|\rangle\} \uplus \{\langle|\kappa|_\Sigma, |\kappa\tau'|_\Sigma\rangle \mid \kappa \approx \kappa \in \Sigma, \kappa\tau' \approx \kappa\tau' \in \Sigma, \kappa \neq {:}\})(|{:}|_\Sigma)$$
$$= |{:}\tau'|_\Sigma$$
$$= |\tau|_\Sigma$$

■

After these technical lemmata we present two lemmata which do all the hard work for the proof of the following theorem.

74. Lemma Let Σ be a closed set of literals. Equivalent are ($\tau_1, \tau_2 \in \mathfrak{T}$):

i. $\tau_1 \approx \tau_2 \in \Sigma$

ii. $|{:}|_\Sigma \in [\![\tau_1 \approx \tau_2]\!]_{S\Sigma}$.

Proof: i. $\Longrightarrow$ ii.
Let $\tau_1 \approx \tau_2 \in \Sigma$.

$$\Longrightarrow \tau_1 \approx \tau_1, \tau_2 \approx \tau_2 \in \Sigma \qquad (\Sigma \text{ closed})$$

Now $[\tau_1]_{S_\Sigma}(|{:}|_\Sigma)$

$$= |\tau_1|_\Sigma \qquad \text{(Lemma 73)}$$
$$= |\tau_2|_\Sigma \qquad (\tau_1 \approx \tau_2 \in \Sigma)$$
$$= [\tau_2]_{S_\Sigma}(|{:}|_\Sigma). \qquad \text{(Lemma 73)}$$

Therefore $|{:}|_\Sigma \in [\![\tau_1 \approx \tau_2]\!]_{S_\Sigma} = \{u \in U_\Sigma \mid [\tau_1]_{S_\Sigma}(u) = [\tau_2]_{S_\Sigma}(u)\}$.

ii. $\Longrightarrow$ i.
Let $|{:}|_\Sigma \in [\![\tau_1 \approx \tau_2]\!]_{S_\Sigma} = \{u \in U_\Sigma \mid [\tau_1]_{S_\Sigma}$ is defined on u and $[\tau_2]_{S_\Sigma}$ is defined on u and $[\tau_1]_{S_\Sigma}(u) = [\tau_2]_{S_\Sigma}(u)\}$.

$$\Longrightarrow [\tau_1]_{S_\Sigma}(|{:}|_\Sigma) = [\tau_2]_{S_\Sigma}(|{:}|_\Sigma)$$
$$\Longrightarrow |\tau_1|_\Sigma = |\tau_2|_\Sigma \qquad \text{(Lemma 73)}$$
$$\Longrightarrow \tau_1 \approx \tau_2 \in \Sigma \qquad (\Sigma \text{ closed})$$

■

75. Lemma Let Σ be a closed set of literals. Equivalent are ($\tau \in \mathfrak{T}, \varphi \in \mathfrak{V}$):

i. $\tau \sim \varphi \in \Sigma$

ii. $|{:}|_\Sigma \in [\![\tau \sim \varphi]\!]_{S_\Sigma}$.

Proof: i. $\Longrightarrow$ ii.
Let $\tau \sim \varphi \in \Sigma$.
Now $V_\Sigma([\tau]_{S_\Sigma}(|{:}|_\Sigma))$

$$= V_\Sigma(|\tau|_\Sigma) \quad \text{(Lemma 73)}$$

$$= \varphi \quad \text{(Def. of } V_\Sigma\text{)}$$

Therefore $|{:}|_\Sigma \in [\![\tau \sim \varphi]\!]_{S_\Sigma}$

$$(= \{u \in U_\Sigma \mid [\tau]_{S_\Sigma} \text{ is defined on } u \text{ and } V_\Sigma([\tau]_{S_\Sigma}(u)) = \varphi\})$$

ii. $\Longrightarrow$ i.
Let $|{:}|_\Sigma \in [\![\tau \sim \varphi]\!]_{S_\Sigma}$.
$\Longrightarrow V_\Sigma([\tau]_{S_\Sigma}(|{:}|_\Sigma)) = \varphi$ and $|{:}|_\Sigma \in U_\Sigma$
$\Longrightarrow V_\Sigma(|\tau|_\Sigma) = \varphi$
$\Longrightarrow \{\langle |\kappa|_\Sigma, \rho\rangle \mid \kappa \sim \rho \in \Sigma\}(|\tau|_\Sigma) = \varphi$
$\Longrightarrow \tau \sim \varphi \in \Sigma$ (Σ closed)

■

76. Proposition Let Σ be a closed set. Then $S_\Sigma = \langle U_\Sigma, V_\Sigma, A_\Sigma\rangle$ is an $\mathfrak{S}$-structure which satisfies the Σ. More exactly $|{:}|_\Sigma \in [\![\Sigma]\!]_{S_\Sigma}$.

Proof: Lemma 71 shows that S_Σ is an $\mathfrak{S}$-structure for the closed set Σ. To prove that $|{:}|_\Sigma \in [\![\Sigma]\!]_{S_\Sigma}$ we have to show that for each $\delta \in \Sigma$ $|{:}|_\Sigma \in [\![\delta]\!]_{S_\Sigma}$. By definition of a closed set δ can have one of four different forms.
Case 1: $\delta = \tau_1 \approx \tau_2$ (with $\tau_1, \tau_2 \in \mathfrak{T}$).
Let $\tau_1 \approx \tau_2 \in \Sigma$. According to lemma 74 this is equivalent to $|{:}|_\Sigma \in [\![\tau_1 \approx \tau_2]\!]_{S_\Sigma}$.
Case 2: $\delta = \tau \sim \varphi$ (with $\tau \in \mathfrak{T}$ and $\varphi \in \mathfrak{V}$).
Let $\tau \sim \varphi \in \Sigma$. According to lemma 75 this is equivalent to $|{:}|_\Sigma \in [\![\tau \sim \varphi]\!]_{S_\Sigma}$.
Case 3: $\delta = \neg\tau_1 \approx \tau_2$ (with $\tau_1, \tau_2 \in \mathfrak{T}$).
Let $\neg\tau_1 \approx \tau_2 \in \Sigma$. Then $\tau_1 \approx \tau_2 \notin \Sigma$, because Σ is closed. According to lemma 74 this is equivalent to $|{:}|_\Sigma \notin [\![\tau_1 \approx \tau_2]\!]_{S_\Sigma}$. Furthermore $|{:}|_\Sigma \in U_\Sigma$, because $:\approx: \in \Sigma$. Therefore $|{:}|_\Sigma \in U_\Sigma \setminus [\![\tau_1 \approx \tau_2]\!]_{S_\Sigma} = [\![\neg\tau_1 \approx \tau_2]\!]_{S_\Sigma}$.
Case 4: $\delta = \neg\tau \sim \varphi$ (with $\tau \in \mathfrak{T}$ and $\varphi \in \mathfrak{V}$).
Let $\neg\tau \sim \varphi \in \Sigma$. Then $\tau \sim \varphi \notin \Sigma$. According to lemma 75 this is equivalent to $|{:}|_\Sigma \notin [\![\tau \sim \varphi]\!]_{S_\Sigma}$. Furthermore $|{:}|_\Sigma \in U_\Sigma$, because $:\approx: \in \Sigma$. Therefore $|{:}|_\Sigma \in U_\Sigma \setminus [\![\tau \sim \varphi]\!]_{S_\Sigma} = [\![\neg\tau \sim \varphi]\!]_{S_\Sigma}$. ■

77. Theorem Let Σ be a *normal* set of literals. Then Σ is satisfiable.

Proof: This is an immediate consequence of the propositions 69 and 76. ■

IV Conclusion

We have shown that King's typed feature logic is satisfiable. Therefore we have given a consistency check for feature structures à la HPSG, that is typed feature structures with appropriateness restrictions, full disjunction and full negation. This is an important result insofar as a formalism for which consitency cannot be checked is rather useless. Why formalise a grammar, if one can't tell whether this formalisation is contradictory or meaningful. The ability to use full negation is particularly important. HPSG grammar rules make heavy use of implications with complex antecedentia and therefore need full negation. But so far there was no justification for this use in terms of a formalism that could cope with it. We have also made the first necessary step towards grammaticality. Although much more needs to be done to reach that goal, satisfiability is a necessary premise to grammaticality. It was important to lay that ground. Without the positive result that we provide there would have been quite some shadow over the whole task of HPSG, at least with its current underlying formalism.

Further research needs to be made into two directions. Firstly, there are theoretical issues. One of them is already mentioned above: The question of whether grammaticality is decidable. As we saw, Pollard and Sag demand this. But it is far from clear, if that can be reached. There is another way of looking at this problem. One could regard grammar rules as a logic program or a rewrite system, instead of constraints for grammaticality. This view is taken in implementations like TROLL (Gerdemann forthcoming). Therefore it would be very nice to give a denotational semantics for rules. To put it differently, one would wish to show that typed feature structures as described by King's logic form an appropriate domain for computation in the domain theoretic sense.

Another direction of further research copes with implementational issues. As stated in the introduction, the algorithm we present was not designed to be efficient. Rather it was the task to show that one exists. Therefore the algorithm is far from being useful in an implementation. Another point arrises out of the statement that the first normal form we give, the normal sets, are the weakest possible constraints for satisfiable feature structures. Thus one may want to consider to swap from a graphs bound feature structure representation to a representation that uses descriptions and methods of logic programming. Whether this step is useful could be a topic of further research.

References

Carpenter, Bob (1992). *The logic of typed feature structures*, Vol. 32 of *Cambridge Tracts in Theoretical Computer Science*, Cambridge University Press.

Gerdemann, Dale (forthcoming). TROLL manual, *Technical Report*, Universität Tübingen.

Gerdemann, Dale and King, Paul John (to appear). Typed Feature Structures for Expressing and Computationally Implementing Feature Cooccurence Restrictions, *Proceedings of 4. Fachtagung der Sektion Computerlinguistik der Deutschen Gesellschaft für Sprachwissenschaft.*

Götz, Thilo (1993). A Normal Form Algorithm For King's Descriptive Formalisms, unpublished term paper, Universität Tübingen.

Götz, Thilo (forthcoming). Unique normal forms and subsumption for typed feature structures, *Arbeitspapiere des SFB 340*, Universität Tübingen.

Kasper, Robert T. (to appear). Typed feature structure constraint systems: structures and descriptions, *in* H. Trost (ed.), *Feature formalisms and linguistic ambiguity*, Ellis Horwood, Chichester, UK.

King, Paul John (1989). *A logical formalism for head-driven phrase structure grammar*, PhD thesis, University of Manchester.

King, Paul John (1993a). G-spaces and G-relations, unpublished manuscript, Universität Tübingen.

King, Paul John (1993b). Well-typed and sort-resolved feature structures are satisfiable, Unpublished ms., Universität Tübingen.

King, Paul John and Götz, Thilo (1993). Eliminating the feature introduction condition by modifying type inference, *Arbeitspapiere des SFB 340 Nr.31*, Universität Tübingen.

Pollard, Carl and Sag, Ivan (1987). *Information-Based Syntax and Semantics, Vol. 1: Fundamentals*, CSLI Lecture Notes 13, CSLI, Stanford, CA.

Pollard, Carl and Sag, Ivan (in press). *Head-Driven Phrase Structure Grammar*, To be published by University of Chicago Press and CSLI Publications.

Pollard, Carl J. (in press). Sorts in unification based grammar and what they mean, *in* M. Pinkal and B. Gregor (eds.), *Unification in natural language analysis*, MIT Press, Cambridge MA.

Wedekind, Jürgen (1992). Classical Logics for Attribute-value Languages, *EACL Proceedings, 5th Annual Meeting*, pp. 204–209.