Carnegie-Mellon University

upon recommendation of its Faculty,
hereby confers upon

HIDETO TOMABECHI

the degree of

DOCTOR OF PHILOSOPHY

in recognition of
the completion of the course of study
prescribed for this degree
in the field(s) of

COMPUTATIONAL LINGUISTICS

Given under the seal of
the corporation at Pittsburgh
in the Commonwealth of Pennsylvania
on May 16, 1993

Robert A. Chapie
CHAIRMAN OF THE BOARD OF TRUSTEES

Robert Mehrabian
PRESIDENT

Efficient Unification for Natural Language

by

HIDETO TOMABECHI

Submitted to the Program in Computational Linguistics
in partial fulfillment of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

at the

CARNEGIE MELLON UNIVERSITY

May 1993

Efficient Unification for Natural Language

by

Hideto Tomabechi

Submitted to the Program in Computational Linguistics
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

CARNEGIE MELLON UNIVERSITY

April 1993

Signature of Author.....
Program in Computational Linguistics
May 26, 1993

Certified by.....
Masaru Tomita
Associate Professor, Computer Science
Thesis Supervisor

Certified by.....
Jaime G. Carbonell
Professor, Computer Science
Thesis Supervisor

Certified by.....
David A. Evans
Associate Professor, Linguistics and Computer Science
Thesis Supervisor

Certified by.....
Alex H. Waibel
Senior Research Scientist, Computer Science
Thesis Supervisor

Certified by.....
Jun-ichi Tsujii
Professor, Centre for Computational Linguistics, UMIST
Thesis Supervisor

Contents

1 Foundations 9

1.1 Introduction 9

1.2 Feature Structures 10

1.3 Four Types of Feature Structures 14

1.4 Generalization and Unification 15

1.5 Some Formal Properties of Feature Structures 18

1.6 Reentrancy 21

1.7 Extention and Subsumption 25

1.8 Unification and Generalization Revisited 28

2 Graph Unification in Natural Language 32

2.1 Feature Structure Graphs 32

2.2 The Nature of Graph Unification 36

2.3 Unification and Parsing 39

3 Past Representative Methods 45

3.1 Pereira's method 45

3.2 Karttunen's method 48

3.3 Wroblewski's method 49
3.4 Kogure's Method 55
3.5 Emele's Method 58

4 Quasi-Destructive Graph Unification 64

4.1 Introduction 64
4.2 The Quasi-Destructive Graph Unification Algorithm 67
4.3 Discussion 84

5 Quasi-Destructive Graph Unification with Structure-Sharing 88

5.1 Introduction 88
5.2 Quasi-Destructive Graph Unification with Structure-Sharing 90
5.3 Discussion 96

6 Empirical Results 105

6.1 Comparison using actual grammar 105
6.2 Comparison using a simulated grammar 110

7 Concluding Remarks 122

Preface

From theoretical linguistics to computational models of natural language, unification-based processing has become a central methodology in many research efforts. In theoretical linguistics, unification-based formalism has become one standard form of representation; many theories such as LFG ([Bresnan and Kaplan, 1982]), HPSG ([Pollard and Sag, 1987]), and JPSG ([Gunji, 1987]) use feature structure and unification as the base of constraint postulations. In computational linguistics, unification is used as the central constraint processing mechanism during parsing based upon the unification-based grammar analyses. In artificial intelligence, unification-based natural language is often used as an integral part of inference and learning mechanisms. Recent efforts in massively parallel artificial intelligence have also demonstrated the strength of graph unification as a uniform constraint processing mechanism for natural language in a massively parallel environment.

Despite the popularity of unification-based processing, graph unification, which is the computational method of unification-based processing, has remained a bottleneck of the unification-based systems. For example, in unification-based grammar parsing using parsing algorithms such as Earley's algorithm and Tomita's algorithm, unification operations often consume 85 to 95 percent of the total cpu time devoted to a parse. In one large-scale unification-based spoken language parser[1], sometimes 98 percent of the elapsed time is calculated to be devoted to unification operation alone ([Kogure, 1990]). Furthermore, the number of unification operations tends to grow as the grammar gets larger and more complicated. Thus, an unavoidable paradox is that when the natural language system gets larger and the coverage of linguistic phenomena is increased as an attempt to bring performance to a practical level, the number of unification

[1] ATR's HPSG-based spoken Japanese analysis system.

operations increases rapidly and the performance of the systems degrades to an impractical level. Thus an availability of efficient graph unification is of paramount importance both to theoretical natural language research as well as to practical natural language systems.

Overall parsing efficiency is crucial when building or experimenting with both practical and experimental natural language systems. For realtime practical systems, parsing speed is a prerequisite. For theoretical experimentation, the efficiency of hypothesis testing depends on the speed of constraint processing. In the modern linguistic framework, most parsing systems consist of two basic constraint processing mechanisms: 1) context-free parsing algorithms and 2) graph-unification algorithms. This thesis focuses on the efficiency gain of graph-unification algorithms. Given that most of the parsing time is consumed by graph-unification in large-scale NL systems, the benefit of improving graph-unification algorithms seems apparent.

The Center for Machine Translation of Carnegie Mellon University provided me with both the environment and the funding for pursuing my Ph.D. research at Carnegie Mellon University. The Laboratory for Computational Linguistics of Carnegie Mellon University was the base of the theoretical exploration of unification-based linguistic processing. Masaru Tomita was the chairman of my committee and provided me with everything I needed for pursuing my goals at Carnegie Mellon University, including the thesis topic. Jaime Carbonell, director of Center for Machine Translation and also one of my committee members, deserves many thanks for his support, advice and encouragement throughout my days at Carnegie Mellon University. Without his strong support, I would not have been able to continue my research at Carnegie Mellon. Also, without Tommy and Jaime, I would not have joined Carnegie Mellon University in the first place. David Evans, my advisor and also the director of the Laboratory for Computational Linguistics and of the Ph.D. program in Computational Linguistics, supported me throughout my graduate student years and also provided me with linguistic and philosophical

insights into natural language. Alex Waibel, also my advisor, provided me with an excellent opportunity to work on spoken language input for the parsing systems. He is the head of the JANUS project which is the CMU side of the Japan/US/German trilateral video-interpreting-telephony project which uses the unification algorithm described in this thesis. He has also contributed greatly to my research in the neural net frameworks which is not covered in this thesis. Junichi Tsujii, professor at UMIST (University of Manchester Institute of Science and Technology, U.K.) and the only non-CMU member of my committee, has been providing me with new insights for natural language and machine translation for almost ten years. Carl Pollard, a former faculty member in the Computational Linguistics Program, my teacher of HPSG, and my advisor for my master's level thesis, originally introduced me to unification-based linguistics. Bob Carpenter and Rich Thomason were among other faculty members who gave me useful comments and support when I needed them. I wish also to thank Sergei Nirenburg, Lori Levin, Eric Nyberg, Teruko Mitamura, and Todd Kaufmann at the Center for Machine Translation for all the help and advice they gave me while I was at the Center. Radha Rao, the Business Manager of the Center, always handled my last minute requests with generosity, as did current and the past secretaries of the Center, namely, Cerise Josephs, Joan Maddama, and Barbara Moore. The former and the current members of the Laboratory for Computational Linguistics and the Department of Philosophy also contributed greatly to the research contained in this thesis. Among them are Ted Gibson, Alex Franz, Margalit Zabludowski, Sondra Ahlen, Marion Kee, and Renee Schafer. Important parts of the thesis research were conducted during the two periods when I was a Visiting Research Scientist at ATR (Advanced Telecommunication Research) Interpreting Telephony Research Laboratories in Kyoto, Japan twice (10 and 8 month stays in 1990 and 1991). Akira Kurematsu, Tsuyoshi Morimoto, Hitoshi Iida, Kiyoshi Kogure, Osamu Furuse, Susumu Kato, Masaaki Nagata, Toshiyuki Takezawa, Kenji Kita, Genichiro

Kikui, Toshihisa Tashiro, Kazumi Ohkura are among the researchers at ATR who contributed significantly to this work.

Makoto Takahashi, Hidehiko Matsuo, and Kyoko Sagi, of Toyo Information Systems worked with researchers at ATR and did their fully separate and independent implementations of my algorithms for ATR's large scale speech-to-speech translation systems (SLTRANS and ASURA). They have provided me with invaluable feedback for developing the later versions of my algorithms. Among their contributions were extremely detailed experimental results on the algorithms' behaviour under different environments (memory management, data structures, GC methods/timings, number of function calls, different parsers, different rule granularities, etc.) and comparisons with other unification algorithms which were conducted for almost two years using ATR's grammar (probably the largest Japanese grammar ever developed). Their test results confirmed my smaller scale experiments based on my implementations which are reported in this thesis. Marie Boyle of the University of Tuebingen, Peter Neuhaus of Universität Karlsruhe, and graduate students at Tokushima University are among other researchers who independently implemented the early versions of my algorithms and provided me with many useful and important suggestions. I am also indebted to the members of ICOT (Institute for New Generation Computer Technology) Natural Language Group for useful discussions, especially in the framework of unification and constraint-based natural language processing. Satoshi Tojo of Mitsubishi Research Institute, Hiroaki Kitano of NEC, Koiti Takeda of IBM, Akihiro Hirai, formerly with Hitachi, Hidefumi Sawai and Toru Matsuda of Ricoh, and Tsuyosi Kitani of NTT Data Communications Systems are among the members of the industrial affiliate program of the Center for Machine Translation who contributed greatly to the research contained in this thesis. Finally, special thanks to the following professors and researchers in Japan for their suggestions and help: Makoto Nagao, Hozumi Tanaka, Hidetoshi Shirai, Yuji Matsumoto,

Jun-ichi Nakamura, Koiti Hasida, Jun-ichi Aoe, Takako Fujioka, Keiichi Sudo, and Katashi Nagao. Mario Tokoro, Eiichi Osawa, and Katashi Nagao of Sony Computer Science Laboratories provided me with the environment for preparing the later versions of this thesis as well as an opportunity to test my algorithms on their multimodal systems. I originally came to the United States as a Fulbright Scholar in 1985. Thanks are due to the members of the Fulbright Committee and Senator Fulbright for giving me the opportunity to pursue my graduate research in the United States. (first at Yale and then at CMU). Roger Schank, Christopher Riesbeck, and Lawrence Birnbaum were among my first teachers of AI/NLP. Finally, David Rockefeller, Jr., Richard Vowell, Jotaro Takagi, Koyata Hosokawa, and Kiyoaki Hara are among the people outside of the natural language community who contributed greatly to my stay in the United States. Some parts of this thesis were written while I was visiting the Department of Environmental Information of Keio University and while I was a member of the faculty in the Department of Information Science and Intelligent Systems of Tokushima University. Kazuhiko Tsuda, Alfredo Maeda, Hideki Mima, Koiti Iriguchi, and Akiko Kita of Tokushima University and Lonnie Bartusis, Yukiko Uehara and Imari Karasawa of Carnegie Mellon University were extremely helpful in preparing the final version of this thesis.

Chapter 1

Foundations

1.1 Introduction

A variety of grammatical formalisms have been proposed historically in computational linguistics, natural language processing, and artificial intelligence to capture the phenomena called 'language'. Kay proposed Functional Grammar and Functional Unification Grammar (FUG, [Kay, 1984]) motivated by the notion of *functional description* of language. Bresnan and Kaplan developed the Lexical Functional Grammar (LFG, [Bresnan and Kaplan, 1982]) based on the framework of lexically-oriented linguistics. In the aritificial intelligence community, Definite Clause Grammar (DCG, [Pereira and Warren, 1980]) was developed by Pereira and Warren in the logic programming framework. Logic programming and DCG later became the base of natural language research efforts in the Japanese fifth generation computer research (ICOT). Gazdar developed Generalized Phrase Structure Grammar (GPSG, [Gazdar, *et al*, 1985]) in the nontransformational model of linguistic analysis. Pollard and Sag developed Head-driven Phrase Structure Grammar (HPSG, [Pollard and Sag, 1987]) in the similar nontransformational framework centered around the notion of *the linguistic head of a phrase.* Gunji developed the

Japanese Phrase Structure Grammar (JPSG, [Gunji, 1987]), which is a Japanese cousin of HPSG. JPSG later became the central linguistic processing framework in the Japanese interpreting telephony research efforts (ATR).

In the more computational and implementational aspects, PATR-II ([Shieber, *et al*, 1983]) was developed at the SRI AI Center as a theory-neutral *simple* and *mathematically well-founded* tool for natural language processing. At Carnegie Mellon University, to address the inefficiency of unification algorithms, pseudo unification and Pseudo Unification Grammar were developed as a part of machine translation research ([Tomita and Knight, 1987]).

All these grammatical formalisms (at least the modern versions of them) use feature structure as objects for capturing linguistic objects and use unification as the central constraint processing mechanism. In this chapter we would like to review both basic and formal properties of feature structures and unification.

1.2 Feature Structures

Despite the variety of analysis captured in modern theoretical and computational models of language, the so-called *feature structure* has been accepted as the common object for representation. Pollard and Sag explain, "Instead of the NASA Physicists' Euclidean spaces and differential equations, though, the formal object of choice in information-based linguistics are things known as *feature structures*". A feature structure is a structured object that represents informational content by specifying a set of *features* and their *values* pairs. Feature structures provide partial information about the information-bearing entities such as linguistic objects. In other words, feature structures are partial descriptions of things that are captured in different theories of language. Formally, feature structures can be understood as partial functions from

features to their *values* where the underlying domain[1] of the partial functions is provided recursively. Conventionally, feature structures are represented using the matrices of feature value pairs. For example, a feature structure representing the linguistic object for a female professor named *Madoka* may be represented as below.

$$\begin{bmatrix} \textit{name} & \textit{Madoka} \\ \textit{sex} & \textit{female} \\ \textit{occupation} & \textit{professor} \end{bmatrix}$$

For the sake of economy of type-setting as well as of consistency with the sample feature structures in the appendix taken from the actual computer outputs, we also represent the same feature structure as below in this thesis.

```
[[name madoka]
 [sex female]
 [occupation professor]]
```

which can also be represented graphically as:

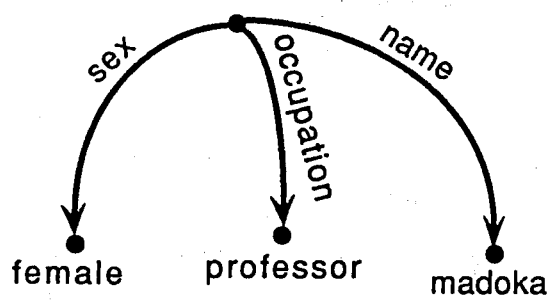

Figure 1-1: Graphical Representation of a Feature Structure

Since we will be representing feature structures in any of the above three ways in this thesis,

[1] I.e., As originally defined by Russell for domain of relations.

we will be using the terms *features*, *labels*, and *labels on the arcs* interchangeably.

A fundamental property of feature structures is their potential for *hierarchiality* ([Pollard and Sag, 1987]). Thus, the value of a feature itself may be another feature structure embedded within. For example, below is a feature structure providing a partial description of a linguistic entity for a third person singular feminine noun:

$$\begin{bmatrix} \textit{category} & N \\ \textit{agreement} & \begin{bmatrix} \textit{number} & \textit{singular} \\ \textit{person} & \textit{third} \\ \textit{gender} & \textit{feminine} \end{bmatrix} \end{bmatrix}$$

Or in our alternate notation:

```
[[category N]
 [agreement [[number singular]
             [person third]
             [gender feminine]]]]
```

Which can be graphically represented as Figure 1-2:

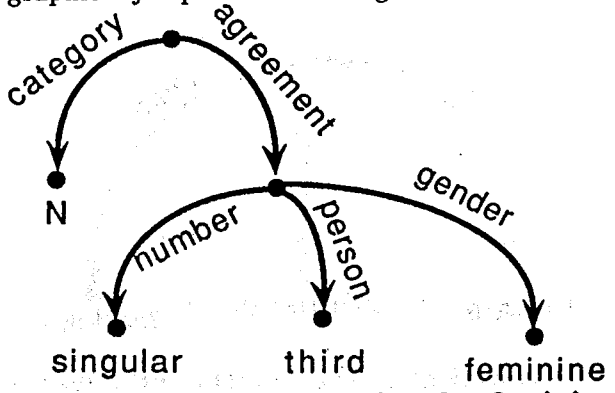

Figure 1-2: **A third person singular feminin noun entity**

Feature structures can be understood as partial functions mapping features to values. For example, in the example for *Madoka* provided above, the feature structure defines the mapping

of the feature *name* to the value *madoka*, of *sex* to the value *female*, and of *occupation* to *professor*. The partial descriptions by feature structures can be understood as "descriptions participating in a relationship of partiality with respect to each other" ([Pereira and Shieber, 1984]). Formally, feature structures can be defined as below:

Definition 1.2.1 (Feature Structures) *Let $\mathcal{F}$ be a possibly infinite set of features and C a possibly infinite set of atomic values. We first define special feature structures $\{\top, \bot\}$. $\top$ represents "no information" and $\bot$ represents "inconsistent information". The set Γ of feature structures can be defined by:*

$$\Gamma = \bigcup_{i=0}^{\infty} \Gamma_i \cup \{\bot\}$$

The set Γ_i is defined recursively such that

$\Gamma_0 = C \cup \{\top\}$

For $i \geq 1$, $\Gamma_i = \{\gamma | \gamma : \mathcal{F} \xrightarrow{\rho} \bigcup_{k=0}^{i-1} \Gamma_k\}$ where $Dom(\gamma) \in \mathcal{P}'(\mathcal{F})$.

Here $Dom(\gamma)$ denotes domain of the partial function γ[2] and $\mathcal{P}'(\mathcal{F})$ denotes a non-empty power set[3] of $\mathcal{F}$ and $\xrightarrow{\rho}$ notates partial mappings.

This way, the features $(l_1, ..., l_m)$ of a feature structure γ are mapped to the values $(\gamma(l_1), ..., \gamma(l_m))$ for which the feature structure, as a partial function, is defined.

NOTATION. Following Pereira's notation ([Pereira, 1985]), we represent the value of the feature l_i of a feature structure γ, i.e., $\gamma(l_i)$ by γ/l_i.

For example, $\gamma/category = N$ represents the feature structure provided above for the partial description of a third person singular feminine noun.

---

[2]Recall that in set theory, domain of a partial function f:A → B is the set $\{a | a \in A$ and $f(a) \in B\}$. Also image of f, written Im(f) is the set $\{f(a) | a \in Dom(f)\}$.

[3]Recall that power set of A is the set of all subsets of A. For example, if A={1,2,3}, power set of A is {1,2,3},{1,2},{2,3},{1,3},{1},{2},{3}, and ∅.

NOTATION. Following the standard notation of set theory, we shall write $Dom(\gamma)$ to denote the domain of the mappings from features to values for a feature structure γ. In other words $Dom(\gamma)$ represents the set of features on the feature structure γ. These features are not recursive, i.e., only the highest features are elements of $Dom(\gamma)$. For example, $Dom(\gamma)$ of the feature structure provided previously for the description of a third person singular feminine noun is $\{category, agreement\}$.

For embedded feature structures, we generalize the notion of features and introduce the notion of *path.* A path is a sequence of features from the outermost feature structure of the embedding to the feature of the innermost feature structure. In our vocabulary, it is a sequence of features from the highest to the lowest in the feature structure hierarchy. For instance, $< agreement, number >$ is a path for the sample feature structure provided above for a third person singular feminine noun.

NOTATION. We generalize the notation γ/l representing the value of the feature l of feature structure γ to apply to the path of features, written γ/p. Therefore, given the path $p \in \mathcal{F}^*$, which is $p =< l_1, ..., l_n >$ embedded in $\gamma \in \Gamma$, then $\gamma/p = (...((\gamma(l_1))(l_2))...)(l_n)$. For instance, with the above feature structure for a third person singular feminine noun,
$\gamma/ < agreement, number >= singular.$

1.3 Four Types of Feature Structures

We have four types of feature structures: *atomic*, *complex*, *variables* and *inconsistency.* Atomic feature structures are feature structures with constant atomic values. Complex feature structures are feature structures which contain feature structures within. Thus, to be precise, only complex feature structures can be viewed as partial functions from features to values. The values of complex feature structures themselves are always feature structures (complex, atomic,

or variable). Variables are the special feature structures with an empty domain. Variables are also called *Top* in this thesis[4]. Variables are the least informative feature structures indicating no information at all. *Inconsistency* are those that indicate inconsistent information. The idea behind *inconsistency* is that such feature structures represent more information than any feature structure possible. It indicates too much information to the extent that it is inconsistent. *Inconsistency* is also called *Bottom* in this thesis. Care needs to be taken in reading *Top* and *Bottom*, since due to historical reasons for looking at the hierarchy of information content, *Top* are somtimes called *Bottom* in the literature (and vice versa).

NOTATION. We shall denote *variables* by either [] or $\top$ and *inconsistency* by $\bot$.

1.4 Generalization and Unification

We can define two classical operations on feature structures: generalization and unification. Generalization is the operation to find a feature structure that contains only the information that is common in two feature structures. When common paths contain conflicting information, generalization introduces a variable and makes an abstraction. Unification is the operation to find a feature structure that contains the information in both feature structures but no additional information. If inconsistency of information is found at any depth of the paths, unification immediately returns *inconsistency* for the highest feature structure (the root feature structure). When a unification returns *inconsistency*, we say that the unification *failed*. This way, unification is a operation to determine the consistency of information between two feature structures. Informally, if γ is a feature structure ($\gamma \in \Gamma$), generalization of γ and $\top$ always returns $\top$ since by definition no information is common between the two. Alternatively, unification of γ and

[4]Note that [Tomabechi, 1991a] and [Tomabechi, 1992] called it *bottom* instead of *top*.

$\top$ always returns γ since by definition, γ contains the information of both and nothing more. Generalization of two atomic feature structures is $\top$ if they are not the same. That is, it has the property of making an abstraction by returning a variable for inconsistent information. If they are the same then the generalization is also the same. Unification of two atomic feature structures are $\bot$ if they are not the same. If they are the same then unification is also the same. Generalization of two complex feature structures is the feature structure only with paths that are common to two feature structures and unification of two complex feature structures is the one that contains both the unique paths and the common paths.

NOTATION. We shall denote generalization operation by $\coprod$ (or $\sqcup$) and unification operation by $\prod$ (or $\sqcap$) in this thesis.

More formally, generalization $\coprod$ and unification $\prod$ operations on feature structures Γ are defined below: But first, it is useful to define the two operations *Complementarcs* and *Intersectarcs* between feature structures. These operations were originally provided in [Pereira, 1985] and are central to unification-based algorithms including the one we are proposing in this thesis.

Definition 1.4.1 (Complementarcs and Intersectarcs) *For two feature structures $\gamma_1, \gamma_2 \in \Gamma$, the following two operations are defined corresponding to the set-difference and set-intersection operations on domains of the partial functions.*

$Complementarcs(\gamma_1, \gamma_2) = \{(l, \gamma) \in \gamma_1 \mid l \notin Dom(\gamma_2)\}$

$Intersectarcs(\gamma_1, \gamma_2) = \{(l, \gamma) \in \gamma_1 \mid l \in Dom(\gamma_2)\}$

Notation. We shall denote $Complementarcs(\gamma_1, \gamma_2)$ by $\gamma_1 \ominus \gamma_2$ and $Intersectarcs(\gamma_1, \gamma_2)$ by $\gamma_1 \lhd \gamma_2$.

$Complementarcs(\gamma_1, \gamma_2)$ is the set of mappings of γ_1 from features to values with features that exist in γ_1 but not in γ_2. Since mappings are often represented by arcs, they are called

Complementarcs. *Intersectarcs*(γ_1, γ_2) is the set of mappings of γ_1 from features to values with features that exist in both γ_1 and in γ_2. Note that values of Complementarcs and Intersectarcs are sets of mappings (i.e., partial functions) and not domains.

Definition 1.4.2 (Generalization) *Let* Γ *be the set of feature structures as defined previously, then below is the definition of the generalization operation (* $\bigsqcup$ *):*

1) $\forall \gamma \in \Gamma, \gamma \bigsqcup \top = \top$

2) $\forall \gamma_1 \in C \; \forall \gamma_2 \in C \setminus \{\gamma_1\}, \gamma_1 \bigsqcup \gamma_2 = \top$

3) $\forall \gamma \in \Gamma, \gamma \bigsqcup \gamma = \gamma$

4) $\forall \gamma_1, \gamma_2 \in \Gamma \setminus \{\top, \bot, \} \setminus C, \forall l_1 \in Dom(\gamma_1 \lhd \gamma_2), (\gamma_1 \bigsqcup \gamma_2) l_1 = \gamma_1(l) \bigsqcup \gamma_2(l)$

Definition 1.4.3 (Unification) *Let* Γ *be the set of feature structures as defined previously, then below is the definition of the unification operation (* $\prod$ *):*

1) $\forall \gamma \in \Gamma, \gamma \prod \top = \gamma$

2) $\forall \gamma_1 \in \Gamma \; \forall \gamma_2 \in C \setminus \{\gamma_1\}, \gamma_1 \prod \gamma_2 = \bot$

3) $\forall \gamma \in \Gamma, \gamma \prod \gamma = \gamma$

4) $\forall \gamma_1, \gamma_2 \in \Gamma \setminus \{\top, \bot, \} \setminus C,$

if $\exists l \in Dom(\gamma_1 \lhd \gamma_2), \gamma_1(l) \prod \gamma_2(l) = \bot$

then $\gamma_1 \prod \gamma_2 = \bot$

else $\forall l_1 \in Dom(\gamma_1 \lhd \gamma_2), \forall l_2 \in Dom(\gamma_2 \ominus \gamma_1)$

$(\gamma_1 \prod \gamma_2) l_1 = \gamma_1(l_1) \prod \gamma_2(l_1)$ *and* $(\gamma_1 \prod \gamma_2) l_2 = \gamma_2(l_2)$

Below is the example of unification and generalization:

```
1.
[[category N]
 [agreement [[number singular]
```

```
            [person third]]]]

2.
[[category N]
 [agreement [[number singular]
            [gender feminine]]]]

3. Unification of 1,2:
[[category N]
 [agreement [[number singular]
            [person third]
            [gender feminine]]]]

4. Generalization of 1,2:
[[category N]
 [agreement [[number singular]]]]


5.
[[category N]
 [agreement [[number plural]
            [person third]]]]

6. Unification of 3,5
 Inconsistency

7. Generalization of 3,5
[[category N]
 [agreement [[number []]
            [person third]]]]
```

1.5 Some Formal Properties of Feature Structures

From the definition of generalization and unification, we can easily see that unification and generalization satisfy the usual formal laws of idempotency, commutativity, associativity and absorption. However, distributivity is not satisfied.