Logics for Linguistic Structures

Trends in Linguistics
Studies and Monographs 201

Mouton de Gruyter
Berlin · New York

Logics for Linguistic Structures

Edited by

Fritz Hamm
Stephan Kepser

Mouton de Gruyter
Berlin · New York

Mouton de Gruyter (formerly Mouton, The Hague)
is a Division of Walter de Gruyter GmbH & Co. KG, Berlin.

Library of Congress Cataloging-in-Publication Data

Logics for linguistic structures / edited by Fritz Hamm and Stephan Kepser.
p. cm. − (Trends in linguistics ; 201)
Includes bibliographical references and index.
ISBN 978-3-11-020469-8 (hardcover : alk. paper)
1. Language and logic. 2. Computational linguistics. I. Hamm, Fritz, 1953− II. Kepser, Stephan, 1967−
P39.L5995 2008
401−dc22
2008032760

ISBN 978-3-11-020469-8
ISSN 1861-4302

Bibliographic information published by the Deutsche Nationalbibliothek

The Deutsche Nationalbibliothek lists this publication in the Deutsche Nationalbibliografie;
detailed bibliographic data are available in the Internet at http://dnb.d-nb.de.

Cover design: Christopher Schneider, Berlin.
Printed in Germany.

Contents

Introduction

Fritz Hamm and Stephan Kepser

Logic has long been playing a major role in the formalization of linguistic structures and linguistic theories. This is certainly particularly true for the area of semantics, where formal logic has been the major tool ever since the Fregean program. In the area of syntax it was the rising of principles based theories with the focus shifting away from the generation process of structures to defining general well-formedness conditions of structures that opened the way for logic. The naturalness by which many types of well-formedness conditions can be expressed in some logic or other led to different logics being proposed and used in diverse formalizations of syntactic theories in general and the field of model theoretic syntax in particular.

The contributions collected in this volume address central topics in theoretical and computational linguistics, such as quantification, types of context dependence and aspects concerning the formalisation of major grammatical frameworks, among others GB, DRT and HPSG. All contributions have in common a strong preference for logic as the major tool of analysis. Two of them are devoted to formal syntax, three to aspects of logical semantics. The paper by Robin Cooper contributes both to syntax and semantics. We therefore grouped the description of the papers in this preface in a syntactic and a semantic section with Cooper's paper as a natural interface between these two fields.

The contribution by Hristo Ganchev, Stoyan Mihov, and Klaus U. Schulz belongs to the field of finite state automata theory and provides a method how to reduce multi tape automata to single tape automata. Multi tape finite state automata have many applications in computer science, they are particularly frequently used in many areas of natural language processing. A disadvantage for the usability of multi tape automata is that certain automata constructions, namely composition, projection, and cartesian product, are a lot more complicated than for single tape automata. A reduction of multi tape automata to single tape automata is thus desirable.

The key construction by Ganchev, Mihov, and Schulz in the reduction is the definition of an automaton type that bridges between multi and single tape automata. The authors introduces so-called one-letter automata. These

are multi tape automata with a strong restriction. There is only one type of transitions permitted and that is where only a single letter in the k-tuple of signature symbols in the transition differs from the empty word. In other words, all components of the tuple are the empty word with the exception of one component. Ganchev, Mihov, and Schulz show that one-letter automata are equivalent to multi tape automata. Interestingly, one-letter automata can be regarded as single tape automata over an extended alphabet which consists of complex symbols each of which is a k-tuple with exactly one letter differing from the empty word.

One of the known differences between multi dimensional regular relations and one dimensional regular relations is that the latter are closed under intersection while the former are not. To cope with this difference, Ganchev, Mihov, and Schulz define a criterion on essentiality of a component in a k-dimensional regular relation and show that the intersection of two k-dimensional regular relations is regular if the two relations share at most one essential component. This result can be extended to essential tapes of one-letter automata and the intersection of these automata.

On the basis of this result Ganchev, Mihov, and Schulz present automata constructions for insertion, deletion, and projection of tapes as well as composition and cartesian product of regular relations all of which are based on the corresponding constructions for single tape automata. This way the effectiveness of one-letter automata is shown. It should hence be expected that this type of automata will be very useful for practical applications.

The contribution by Marcus Kracht provides a logical characterisation of the datastructures underlying the linguistic frameworks *Government and Binding* and *Minimalism*. Kracht identifies so-called multi-dominance structures as the datastructures underlying these theories. A multi-dominance structure is a binary tree with additional immediate dominance relations which have a restricted distribution in the following way. All parents of additional dominance relations must be found on the path from the root to the parent of the base dominance relation. Additional dominance relations provide a way to represent movement of some component from a lower part in a tree to a position higher up.

The logic chosen by Kracht to formalize multi-dominance structures is propositional dynamic logic (PDL), a variant of modal logic that has been used before on many occasions by Kracht and other authors to formalize linguistic theories. In this paper, Kracht shows that PDL can be used to axiomatize multi-dominance structures. This has the important and highly desirable

consequence that the dynamic logic of multi-dominance structures is decidable. The satisfiability of a formula can be decided in in 2EXPTIME.

In order to formalize a linguistic framework it is not enough to provide an axiomatisation of the underlying datastructures only. The second contribution of this paper is therefore a formalisation of important grammatical concepts and notions in the logic PDL. This formalisation is provided for movement and its domains, single movement, adjunction, and cross-serial dependencies. In all of these, care is taken to ensure that the decidability result of multi-dominance structures carries over to grammatical notions defined on these structures. It is thereby shown that large parts of the linguistic framework *Government and Binding* can be formalized in PDL and that this formalisation is decidable.

The contribution by Robin Cooper shows how to render unification based grammar formalisms with type theory using record structures. The paper is part of a broader project which aims at providing a coherent unified approach to natural language dialog semantics. The type theory underlying this work is based on set theory and follows Montague's style of recursively defining semantic domains. There are functions and function types available in this type theory providing a version of the typed λ-calculus. To this base records are added. A record is a finite set of fields, i.e., ordered pairs of a label and an object. A record type is accordingly a finite set of ordered pairs of a label and a type. Records and record types may be nested. The notions of dependent types and subtype relations are systematically extended to be applicable to record types.

The main contribution of this paper is a type theoretical approach to unification phenomena. Feature structures of some type play an important role in almost all modern linguistic frameworks. Some frameworks like LFG and HPSG make this rather explicit. They also provide a systematic way to combine two feature structures partially describing some linguistic object. This combination is based on ideas of unification even though this notion need no longer be explicitely present in the linguistic frameworks. In a type theoretical approach, records and their types render feature structures in a rather direct and natural way. The type theoretical tools to describe unification are meet types and equality. Cooper assumes the existence of a meet type for each pair of types in his theory including record types. He provides a function that recursively simplifies a record type. This function is particularly applicable to record types which are the result of the construction of a meet record type and should be interpreted as the counterpart of unification in type

theory with records. There are though important differences to feature structure unification. One of them is that type simplification never fails. If the meet of incompatible types was constructed, the simplification will return a distinguished empty type.

The main advantage of this approach is that it provides a kind of intensionality which is not available for feature structures. This intensionality can be used, e.g., to distinguish equivalent types such as the source of grammatical information. It can also be used to assign different empty types with different ungrammatical phrases. This may provide a way to support robust parsing in that ungrammatical phrases can be processed and the the consistent parts of their record types may contain useful informations for further processing. Type theory with records also offers a very natural way to integrate sematic analyses into syntactic analyses based on feature structures.

The paper by Eleni Kalyvianaki and Yiannis Moschovakis contains a sophisticated application of the theory of referential intensions developed by Moschovakis in a series of papers (see for instance (Moschovakis 1989a,b, 1993, 1998)), and applied to linguistics in (Moschovakis 2006). Based on the theory of referential intensions the paper introduces two notion of context dependent meaning, factual content and local meaning, and shows that these notions solve puzzles in philosophy of language and linguistics, especially those concerning the logic of indexicals.

Referential intension theory allows to define three notions of synonymy, namely referential synonymy, local synonymy, and factual synonymy. Referential synonymy, the strongest concept, holds between two terms A and B iff there referential intensions are the same; i.e., $int(A) = int(B)$. Here the referential intension of an expression A, $int(A)$ is to be understood as the natural algorithm (represented as a set–theoretical object) which computes the denotation of A with respect to a given model. Thus referential synonymy is a situation independent notion of synonymy. This contrasts with the other two notions of synonymy which are dependent on a given state a. Local synonymy is synonymy with regard to local meaning where the local meaning of an expression A is computed from the referential intension of A applied to a given state a. It is important to note that for the constitution of the local meaning of A the full meanings of the parts of A have to be computed. In this respect the concept of local meaning differs significantly from the notion *factual content* and for this reason from the associated notion of synonymy as well. This is best explained by way of an example.

If in a given state a $her(a) = Mary(a)$ then the factual content of the sentence *John loves her* is the same as the factual content of *John loves Mary*. The two sentences are therefore synonymous with regard to factual content. But they are not locally synonymous since the meaning of *her* in states other than a my well be different from the meaning of *Mary*.

The paper applies these precisely defined notions to Kaplan's treatment of indexicals and argues for local meanings as the most promising candidates for belief carriers. The paper ends with a brief remark on what aspects of meaning should be preserved under translation.

The paper by Edward L. Keenan tries to identify inference patterns which are specific for proportionality quantifiers. For instance, given the premisses (1-a), (1-b) in (1) we may conclude (1-c).

(1) a. More than three tenths of the students are athletes.
b. At least seven tenths of the students are vegetarians.
c. At least one student is both an athlete and a vegetarian.

This is an instance of the following inference pattern:

(2) a. More than n/m of the *As* are *Bs*.
b. At least $1 - n/m$ of the *As* are *Cs*.
c. Ergo: Some *A* is both a *B* and a *C*.

Although proportionality quantifiers satisfy inference pattern (2), other quantifiers do so as well, as observed by Dag Westerståhl. Building on (Keenan 2004) the paper provides an important further contribution to the question whether there are inference patterns specific to proportionality quantifiers.

The central result of Keenan's paper is the Mid-Point Theorem and a generalization thereof.

The Mid-Point Theorem *Let p,q be fractions with $0 \leq p \leq q \leq 1$ and $p + q = 1$. Then the quantifiers*

$$(BETWEEN\ p\ AND\ q)\ and\ (MORE\ THAN\ p\ AND\ LESS\ THAN\ q)$$

are fixed by the postcomplement operation.

The postcomplement of a generalized quantifier Q is that generalized quantifier which maps a set B to $Q(\neg B)$. The following pair of sentences illustrates this operation:

(3) a. Exactly half the students got an A on the exam.
b. Exactly half the students didn't get an A on the exam.

The mid-point theorem therefore guarantees the equivalence of sentences (4-a) and (4-b); and analogously the equivalence of sentences formed with $MORE\ THAN\ p\ AND\ LESS\ THAN\ q$).

(4) a. Between one sixth and five sixth of the students are happy.
b. Between one sixth and five sixth of the students are not happy.

However, this and the generalization of the mid-point theorem are still only partial answers to the question concerning specific inference patterns for proportionality quantifiers, since non-proportional determiner exist which still satisfy the conditions of the generalized mid-point-theorem.

The paper by Lawrence S. Moss studies syllogistic systems of increasing strength from the point of view of natural logic (for a discussion of this notion, see Purdy (1991)). Moss proves highly interesting new completeness results for these systems. More specifically, after proving soundness for all systems considered in the paper the first result states the completeness of the following two axioms for $\mathcal{L}(all)$ a syllogistic fragment containing only expressions of the form *All X are Y*:

$$\frac{}{\textit{All X are X}} \qquad \frac{\textit{All X are Z}\ \ \textit{All Z are Y}}{\textit{All X are Y}}$$

In addition to completness the paper studies a further related but stronger property, the *canonical model property*. A system which has the canonical model property is also complete, but this does not hold vice versa. Roughly, a model $\mathcal{M}$ is canonical for a fragment $\mathcal{F}$, a set Γ of sentences in $\mathcal{F}$ and a logical system for $\mathcal{F}$ if for all $S \in \mathcal{F}$, $\mathcal{M} \models S$ iff $\Gamma \vdash S$. A fragment $\mathcal{F}$ has the canonical model if every set $\Gamma \subseteq \mathcal{F}$ has a canonical model. The canonical model property is a rather strong property. Classical propositional logic, for instance, does not have this property, but the fragment $\mathcal{L}(all)$ has it. Some but not all of the systems in the paper have the canonical model property.

Other system studied in Moss' paper include *Some X are Y*, combinations of this system with $\mathcal{L}(all)$ and sentences involving proper names, systems with Boolean combinations, a combination of $\mathcal{L}(all)$ with *There are at least s many X as Y*, logical theories for *Most* and *Most + Some*. The largest logical system for which completeness is proved adds $\exists^{\geq}$ to the theory

$\mathcal{L}(all, some, no, names)$ with Boolean operations, where $\exists^{\geq}(X,Y)$ is considered true in case X contains more elements than Y.

Moss' paper contains two interesting digressions as well. The first is concerned with sentences of the form *All X which are Y are Z*, the second with *most*. For instance, Moss proves that the following two axioms are complete for *most*.

$$\frac{\textit{Most X are Y}}{\textit{Most X are X}} \qquad \frac{\textit{Most X are Y}}{\textit{Most Y are Y}}$$

Moreover, if *Most X are Y* does not follow from a set of sentences Γ then there exists a model of Γ with cardinality ≤ 5 which falsifies *Most X are Y*.

All papers collected in this volume grew out of a conference in honour of Uwe Mönnich which was held in Freudenstadt in November 2004. Since this event four years elapsed. But another important date is now imminent, Uwe's birthday. Hence we are in the lucky position to present this volume as a Festschrift for Uwe Mönnich on the occasion of his 70$^{\text{th}}$ birthday.

Tübingen, July 2008 Fritz Hamm and Stephan Kepser

References

Keenan, Edward L.
2004 Excursions in natural logic. In Claudia Casadio, Philip J. Scott, and Robert A.G. Seely, (eds.), *Language and Grammar: Studies in Mathematical Linguistics and Natural Language*. Stanford: CSLI.

Moschovakis, Yiannis
1989a The formal language of recursion. *The Journal of Symbolic Logic* 54: 1216–1252.

1989b A mathematical modeling of pure recursive algorithms. In Albert R. Meyer and Michael Taitslin, (eds.), *Logic at Botik '89*, LNCS 363. Berlin: Springer.

1993 Sense and denotation as algorithm and value. In Juha Oikkonen and Jouko Väänänen, (eds.), *Logic Colloquium '90*. Natick, USA: Association for Symbolic Logic, A.K. Peters, Ltd.

1998 On founding the theory of algorithms. In Harold Dales and Gianluigi Oliveri, (eds.), *Truth in Mathematics*. Oxford University Press.

2006 A logical calculus of meaning and synonymy. *Linguistics and Philosophy* 29: 27–89.

Purdy, William C.

1991 A logic for natural language. *Notre Dame Journal of Formal Logic* 32: 409–425.

Type Theory with Records and unification-based grammar

Robin Cooper

Abstract

We suggest a way of bringing together type theory and unification-based grammar formalisms by using records in type theory. The work is part of a broader project whose aim is to present a coherent unified approach to natural language dialogue semantics using tools from type theory.

1. Introduction

Uwe Mönnich has worked both on the use of type theory in semantics and on formal aspects of grammar formalisms. This paper suggests a way of bringing together type theory and unification as found in unification-based grammar formalisms like HPSG by using records in type theory which provide us with feature structure like objects. It represents a small offering to Uwe to thank him for many kindnesses over the years sprinkled with insights and rigorous comments.

This work is part of a broader project whose aim is to present a coherent unified approach to natural language dialogue semantics using tools from type theory. We are seeking to do this by bringing together Head Driven Phrase Structure Grammar (HPSG) (Sag et al. 2003), Montague semantics (Montague 1974), Discourse Representation Theory (DRT) (Kamp and Reyle 1993; van Eijck and Kamp 1997, and much other literature), situation semantics (Barwise and Perry 1983) and issue-based dialogue management (Larsson 2002) into a single type-theoretic formalism. A survey of our approach to the semantic theories (i.e., Montague semantics, DRT and situation semantics) and HPSG can be found in (Cooper 2005b). Other work in progress can be found on `http://www.ling.gu.se/~cooper/records`. We give a brief summary here: Record types can be used as discourse representation structures (DRSs). Truth of a DRS corresponds to there being an object of the appropriate record type and this gives us the effect of simultaneous binding of discourse referents (corresponding to labels in records) familiar from the

semantics of DRSs in (Kamp and Reyle 1993). Dependent function types provide us with the classical treatment of donkey anaphora from DRT in a way corresponding to the type theoretic treatment proposed by Mönnich (1985), Sundholm (1986) and Ranta (1994). At the same time record types can be used as feature structures of the kind found in HPSG since they have recursive structure and induce a kind of subtyping which can be used to mimic unification. Because we are using a general type theory which includes records we have functions available and a version of the λ-calculus. This means that we can use Montague's λ-calculus based techniques for compositional interpretation. From the HPSG perspective this gives us the advantage of being able to use "real" variable binding which can only be approximately simulated in pure unification based systems. From the DRT perspective this use of compositional techniques gives us an approach similar to that of Muskens (1996) and work on λ-DRT (Kohlhase et al. 1996).

In this paper we will look at the notion of unification as used in unification-based grammar formalisms like HPSG from the perspective of the type theoretical framework. This work has been greatly influenced by work of Jonathan Ginzburg (for example, Ginzburg in prep, Chap. 3). In Section 2 we will give a brief informal introduction to our view of type theory with records. The version of type theory that we discuss has been made more precise in (Cooper 2005a) and in an implementation called TTR (Type Theory with Records) which is under development in the Oz programming language. In Section 3 we will discuss the notion of subtype which records introduce (corresponding to the notion of subsumption in the unification literature). We will then, in Section 4, propose that linguistic objects are to be regarded as records whereas feature structures are to be regarded as corresponding to record *types*. Type theory is "function-based" rather than "unification-based". However, the addition of records to type theory allows us to get the advantages of unification without having to leave the "function-based" approach. We show how to do this in Section 5 treating some classical simple examples which have been used to motivate the use of unification. Section 6 deals with the way in which unification analyses are used to allow the extraction of linguistic generalizations as principles in the style of HPSG. The conclusion (Section 7) is that by using record types within a type theory we can have the advantages of unification-based approaches together with an additional intensionality not present in classical unification approaches and without the disadvantage of leaving the "function-based" approach which is necessary in order to deal adequately with semantics (at least).

2. Records in type theory

In this section[1] we give a very brief intuitive introduction to the kind of type theory we are employing. A more detailed and formal account can be found in (Cooper 2005a) and work in progress on the project can be found on `http://www.ling.gu.se/~cooper/records`. While the type theoretical machinery is based on work carried out in the Martin-Löf approach (Coquand et al. 2004; Betarte 1998; Betarte and Tasistro 1998; Tasistro 1997) we are making a serious attempt to give it a foundation in standard set theory using Montague style recursive definitions of semantic domains. There are two main reasons for this. The first is that we think it important to show the relationship between the Montague model theoretic tradition which has been developed for natural language semantics and the proof-theoretic tradition associated with type theory. We believe that the aspects of this kind of type theory that we need can be seen as an enrichment of Montague's original programme. The second reason is that we are interested in exploring to what extent intuitionistic and constructive approaches are appropriate or necessary for natural language. For example, we make important use of the notion "propositions as types" which is normally associated with an intuitionistic approach. However, we suspect that our Montague-like approach to defining the type theory to some extent decouples the notion from intuitionism. We would like to see type theory as providing us with a powerful collection of tools for natural language analysis which ultimately do not commit one way or the other to philosophical notions associated with intuitionism.

The central idea of records and record types can be expressed informally as follows, where $T(a_1,\ldots,a_n)$ represents a type T which depends on the objects $a_1,\ldots,a_n$.

If $a_1 : T_1, a_2 : T_2(a_1),\ldots,a_n : T_n(a_1,a_2,\ldots,a_{n-1})$, a record:

$$\begin{bmatrix} l_1 & = & a_1 \\ l_2 & = & a_2 \\ \ldots & & \\ l_n & = & a_n \\ \ldots & & \end{bmatrix}$$

is of type:

$$\begin{bmatrix} l_1 & : & T_1 \\ l_2 & : & T_2(l_1) \\ \ldots & & \\ l_n & : & T_n(l_1,l_2,\ldots,l_{n-1}) \end{bmatrix}$$

A record is to be regarded as a finite set of fields $\langle \ell, a \rangle$, which are ordered pairs of a label and an object. A record type is to be regarded as a finite set of fields $\langle \ell, T \rangle$ which are ordered pairs of a label and a type. The informal notation above suggests that the fields are ordered with types being dependent on previous fields in the order. This is misleading in that we regard record types as sets of fields on which a partial order is induced by the dependency relation. Dependent types give us the possibility of relating the values in fields to each other and play a crucial role in our treatment of both feature structures and semantic objects. Both records and record types are required to be the graphs of functions, that is, if $\langle \ell, \alpha \rangle$ and $\langle \ell', \beta \rangle$ are members of a given record or record type then $\ell \neq \ell'$. A record r is of record type R just in case for each field $\langle \ell, T \rangle$ in R there is a field $\langle \ell, a \rangle$ in r (i.e., with the same label) and a is of type T. Notice that the record may have additional fields not mentioned in the type. Thus a record will generally belong to several record types and any record will belong to the empty record type. This gives us a notion of subtyping which we will explore further in Section 3.

Let us see how this can be applied to a simple linguistic example. We will take the content of a sentence to be modelled by a record type. The sentence

a man owns a donkey

corresponds to a record type:

$$\begin{bmatrix} \mathrm{x} & : & \mathit{Ind} \\ \mathrm{c}_1 & : & \mathrm{man(x)} \\ \mathrm{y} & : & \mathit{Ind} \\ \mathrm{c}_2 & : & \mathrm{donkey(y)} \\ \mathrm{c}_3 & : & \mathrm{own(x,y)} \end{bmatrix}$$

A record of this type will be:

$$\begin{bmatrix} \mathrm{x} & = & a \\ \mathrm{c}_1 & = & p_1 \\ \mathrm{y} & = & b \\ \mathrm{c}_2 & = & p_2 \\ \mathrm{c}_3 & = & p_3 \end{bmatrix}$$

where

a, b are of type *Ind*, individuals
p_1 is a proof of man(a)
p_2 is a proof of donkey(b)
p_3 is a proof of own(a, b).

Note that the record may have had additional fields and still be of this type. The types 'man(x)', 'donkey(y)', 'own(x,y)' are dependent types of proofs (in a convenient but not quite exact abbreviatory notation – we will give a more precise account of dependencies within record types in Section 3). The use of types of proofs for what in other theories would be called propositions is often referred to as the notion of "propositions as types". Exactly what type 'man(x)' is depends on which individual you choose in your record to be labelled by 'x'. If the individual a is chosen then the type is the type of proofs that a is a man. If another individual d is chosen then the type is the type of proofs that d is a man, and so on. What is a proof? Martin-Löf considers proofs to be objects rather than arguments or texts. For non-mathematical propositions proofs can be regarded as situations or events. For useful discussion of this see (Ranta 1994, p. 53ff). We discuss it in more detail in (Cooper 2005a).

There is an obvious correspondence between this record type and a discourse representation structure (DRS) as characterised in (Kamp and Reyle 1993). The characterisation of what it means for a record to be of this type corresponds in an obvious way to the standard embedding semantics for such a DRS which Kamp and Reyle provide.

Records (and record types) are recursive in the sense that the value corresponding to a label in a field can be a record (or record type)[2]. For example,

$$r = \begin{bmatrix} \mathrm{f} & = & \begin{bmatrix} \mathrm{f} & = & \begin{bmatrix} \mathrm{ff} & = & a \\ \mathrm{gg} & = & b \end{bmatrix} \\ \mathrm{g} & = & c \end{bmatrix} \\ \mathrm{g} & = & \begin{bmatrix} \mathrm{h} & = & \begin{bmatrix} \mathrm{g} & = & a \\ \mathrm{h} & = & d \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

is of type

$$R = \begin{bmatrix} \mathrm{f} & : & \begin{bmatrix} \mathrm{f} & : & \begin{bmatrix} \mathrm{ff} & : & T_1 \\ \mathrm{gg} & : & T_2 \end{bmatrix} \\ \mathrm{g} & : & T_3 \end{bmatrix} \\ \mathrm{g} & : & \begin{bmatrix} \mathrm{h} & : & \begin{bmatrix} \mathrm{g} & : & T_1 \\ \mathrm{h} & : & T_4 \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

given that $a : T_1$, $b : T_2$, $c : T_3$ and $d : T_4$. We can use *path-names* in records and record types to designate values in particular fields, e.g.

$$r.\mathrm{f} = \begin{bmatrix} \mathrm{f} & = & \begin{bmatrix} \mathrm{ff} & = & a \\ \mathrm{gg} & = & b \end{bmatrix} \\ \mathrm{g} & = & c \end{bmatrix}$$

$$R.\mathrm{f.f.ff} = T_1$$