FORMAL LANGUAGES AND SYSTEMS

Authors:

Heinrich Herre
Universität Leipzig
Intitut für Informatik
Germany

Peter Schroeder-Heister
Universität Tübingen
Wilhelm-Schickard-Institut für Informatik
email: psh@informatik.uni-tuebingen.de

Draft (1995) of a paper which appeared 1998 in the *Routledge Encyclopedia of Philosophy*

FORMAL LANGUAGES AND SYSTEMS

Keywords:

Consequence relation
Deductive system
Formal grammar
Formal language
Formal system
Gentzen-style system
Hilbert-style system
Inference operation
Logical calculus
Rule-based system
Substructural logics

FORMAL LANGUAGES AND SYSTEMS

Formal languages and systems are concerned with symbolic structures considered under the aspect of generation by formal (syntactic) rules, i.e. irrespective of their or their components' meaning(s). In the most general sense, a formal language is a set of expressions (1). The most important way of describing this set is by means of grammars (2). Formal systems are formal languages equipped with a consequence operation yielding a deductive system (3). If one further specifies the means by which expressions are built up (connectives, quantifiers) and the rules from which inferences are generated, one obtains logical calculi (4) of various sorts, especially Hilbert-style and Gentzen-style systems.

1 Expressions

A *formal language* is a set $\mathcal{L}$ of expressions. *Expressions* are built up from a finite set Σ of atomic symbols or atoms (the *alphabet* of the language) by means of certain constructors. Normally one confines oneself to linear association (concatenation) $\circ$ as the only constructor, yielding *strings* of atoms, also called *words* (over Σ), starting with the empty word ϵ. Mathematically, the structure obtained is the semigroup freely

generated by the alphabet Σ. When $a_1 \circ a_2 \circ \ldots \circ a_n$ with atoms $a_1, a_2, \ldots, a_n$ is written in the usual way as $a_1 a_2 \ldots a_n$, it is assumed that the decomposition of this expression into atoms is unique. It should be noted, that languages based on more than one constructor may well be considered. An example would be Frege's two-dimensional *Begriffsschrift*, which may be viewed as a formal language based on constructors for horizontal and vertical alignment. However, since any constructor can be written linearly as a function symbol applied to arguments, the notion of formal languages as based on linear concatenation is sufficiently general. The atomic symbols which form the basis of a formal language are themselves abstract entities, in contrast to the tokens which constitute their individual realisations.

2 Grammars

A formal language $\mathcal{L}$ can be investigated from various points of view. One possibility is to specify a device $\mathcal{A}$ that recognizes exactly those words over Σ which belong to $\mathcal{L}$, i.e. which enters a certain state if and only if a word of $\mathcal{L}$ is given as an input to $\mathcal{A}$. When this occurs we say that $\mathcal{A}$ accepts $\mathcal{L}$. Such a device is called an *'automaton'* (in the mathematical, not in the physical sense). The relationship between languages and automata accepting them is studied in automata theory. A *formal grammar* $\mathcal{G}$ for $\mathcal{L}$ describes a way to *generate* the words of $\mathcal{L}$ by means of formal rules, beginning with a special start symbol S. A rule (also called 'production') has the form $u \rightarrow v$, which expresses that in any word of the form $w_1 u w_2$ the part u may be replaced by v yielding $w_1 v w_2$. In addition to the atoms of the alphabet Σ under consideration (which are called 'terminals' in formal language theory), rules may contain variables (called

'non-terminals'), among which S is distinguished. For example the grammar

$$S \rightarrow ASA$$

$$S \rightarrow a$$

$$A \rightarrow b$$

generates all words $b \ldots bab \ldots b$ over the alphabet $\{a, b\}$ with equally many b's on the left and right side of a. Formal languages may be classified by the types of grammars generating them. For example, a language $\mathcal{L}$ is called *context-free* if $\mathcal{L}$ can be generated by a context-free grammar, which is a grammar whose productions always have just a single variable as its left side (as in the example). Context-free grammars play a distinguished role in the construction of artificial languages (such as programming languages). However, for the study of (fragments of) natural languages more complicated ('context-sensitive') types of grammars have to be considered. Formal grammars have the same expressive power as computable functions: The computation by a Turing machine can be described by means of a formal grammar, and the generation of words in a formal grammar can be simulated by a Turing machine.

3 Deductive systems

A formal system is based on a formal language $\mathcal{L}$, endowing it with a *consequence operation* $\mathcal{C}$. This operation $\mathcal{C}$ can be specified at different levels of abstraction. In the most general sense $\mathcal{C}$ is just an arbitrary function transforming subsets of $\mathcal{L}$ into subsets of $\mathcal{L}$ ($2^{\mathcal{L}} \rightarrow 2^{\mathcal{L}}$). $\mathcal{C}$ is said to be an *inference operation* if the set of consequences of a set X comprises at least X:

$$X \subseteq \mathcal{C}(X) \quad \text{(inclusion)} .$$

The pair $\langle \mathcal{L}, \mathcal{C} \rangle$ is then called an *inferential system*. It is called a *closure system* if furthermore

$$\mathcal{C}(\mathcal{C}(X)) \subseteq \mathcal{C}(X) \quad \text{(idempotence)}$$

and

$$X \subseteq Y \Rightarrow \mathcal{C}(X) \subseteq \mathcal{C}(Y) \quad \text{(monotonicity)}$$

are fulfilled. It is called a *deductive system*, if the consequences of a set X can be obtained from a finite subset of X, i.e., if in addition to the three conditions mentioned,

$$\mathcal{C}(X) \subseteq \bigcup \{\mathcal{C}(Y) : \ Y \subseteq X, Y \textit{ finite}\} \quad \text{(compactness)}$$

holds. Equivalently, formal systems can be described by a *consequence relation* $X \vdash A$ between subsets of $\mathcal{L}$ and expressions of $\mathcal{L}$. The four conditions mentioned then become

$$X \cup \{A\} \vdash A$$

$$X \vdash A \ \Rightarrow \ X \cup Y \vdash A$$

$$(X \vdash A \textit{ for all } A \in Y \textit{ and } Y \cup Z \vdash B) \ \Rightarrow \ X \cup Z \vdash B$$

$$X \vdash A \ \Rightarrow \ Y \vdash A \textit{ for some finite } Y \subseteq X \ .$$

If we confine ourselves to consequences from finite sets X, Y, Z, which in the case of deductive systems is appropriate, these conditions are equivalent to

$$\{A\} \vdash A \quad \text{(identity)}$$

$$X \vdash A \Rightarrow X \cup \{B\} \vdash A \quad \text{(thinning)}$$

$$(X \vdash A \textit{ and } \{A\} \cup Z \vdash B) \ \Rightarrow \ X \cup Z \vdash B \quad \text{(cut)} \ ,$$

which are the basic structural (= logic-free) principles of *Gentzen-style sequent systems*.

Deductive systems can be given by means of a set Δ of inference rules R, where an inference rule is an $(n+1)$-ary relation $R \subseteq \mathcal{L}^n \times \mathcal{L}$. Any $(A_1, \ldots, A_n, B) \in R$ (also written as $A_1, \ldots, A_n \Rightarrow B$) is called an instance of R, and $A_1, \ldots, A_n$ are said to be the premisses and B the conclusion of that instance. If $n = 0$ then the rule $\Rightarrow A$ is called an axiom. If we define $\mathcal{C}_\Delta(X)$ to be the smallest set of expressions in $\mathcal{L}$ containing X and closed under the rules in Δ (i.e., if $\{A_1, \ldots, A_n\} \subseteq \mathcal{C}_\Delta(X)$, then $A \in \mathcal{C}_\Delta(X)$ for any instance $A_1, \ldots, A_n \Rightarrow B$ of a rule in Δ), then $\langle \mathcal{L}, \mathcal{C}_\Delta \rangle$ is a deductive system, called the *rule-based system* with respect to Δ. Conversely, the consequence relation of any deductive system defines a set of rules with respect to which this system can be viewed as a rule-based system. Like grammars, rule-based systems are as powerful as Turing machines, i.e. they can be used to express any algorithm.

4 Logical calculi

In logic one considers deductive systems over a language $\mathcal{L}$ whose expressions, called formulas, are built up from certain basic expressions by means of function symbols, predicate symbols and logical operators. In propositional logic, which for reasons of simplicity is considered in the following, formulas are built up from a subset $\mathcal{P}$ of $\mathcal{L}$ (the set of 'propositional letters' or 'propositional variables') and a finite set $\mathcal{S}$ of symbols of the alphabet Σ of $\mathcal{L}$ (the set of 'propositional connectives'). Prominent connectives are 'not' ($\sim$), 'and' ($\wedge$), 'or' ($\vee$) and 'implies' ($\supset$). The two principal ways of specifying a consequence operation $\mathcal{C}$ for such a system is (1) by presenting it as a rule-based system and (2) by formally describing its consequence relation $\vdash$. Other approaches modify or extend the notion of consequence by considering (3) multiple consequences, (4)

consequence relations with restricted structural principles ('substructural logics') and (5) the fine structure of proofs.

(1) For a *rule-based system* a set of axioms which one can start with and a set of inference rules transforming formulas into other formulas are given. Axioms for a fragment of propositional logic based on conjunction and implication (the so-called 'positive implication-conjunction logic') are, for example, all formulas of the form

$$A \supset (B \supset A)$$

$$(A \supset (B \supset C)) \supset ((A \supset B) \supset (A \supset C))$$

$$(A \supset (B \supset C)) \supset ((A \wedge B) \supset C)$$

$$((A \wedge B) \supset C) \supset (A \supset (B \supset C))$$

and inference rules all instances of the schema

$$A, A \supset B \;\Rightarrow\; B \quad \text{(modus ponens)} .$$

Rule-based systems of this kind are often called *Frege-Hilbert-style systems*. Typically, they contain a relatively long list of axioms and just a few inference principles (in ordinary propositional logic just modus ponens). The consequence relation '$X \vdash_{\mathcal{C}} A$' corresponding to the consequence operator $\mathcal{C}$ is called 'derivability from assumptions'.

(2) In contradistinction to this approach, *Gentzen-style sequent-systems* formulate the desired properties of a consequence relation '$X \vdash_{\mathcal{C}} A$' as formal rules of inference. In a sequent system of propositional logic based on the connectives $\wedge$ and $\supset$ they would typically postulate the rules above of identity, thinning and cut as logic-free structural

rules, and as logical rules the following ones:

$$X \cup \{A\} \vdash B \;\Rightarrow\; X \vdash A \supset B \quad (\supset - \text{introduction right})$$

$$X \vdash A \;,\; X \cup \{B\} \vdash C \;\Rightarrow\; X \cup \{A \supset B\} \vdash C \quad (\supset - \text{introduction left})$$

$$X \vdash A \;,\; X \vdash B \;\Rightarrow\; X \vdash A \wedge B \quad (\wedge - \text{introduction right})$$

$$X \cup \{A, B\} \vdash C \;\Rightarrow\; X \cup \{A \wedge B\} \vdash C \quad (\wedge - \text{introduction left}) \,.$$

Formally, this is just a specification of a rule-based system with respect to a more complicated language. Rather than a set of formulas $\mathcal{L}$ over $\mathcal{P}$ and $\mathcal{S}$ one considers a language $\mathcal{L}'$, whose expressions are 'sequents' of the form '$X \vdash A$', where $A \in \mathcal{L}$ with X finite. The deductive system $\mathcal{C}'$ over $\mathcal{L}'$ is then given by the rules stated. (Actually, to capture the idea of finite *sets* X in *expressions* '$X \vdash A$', one has to consider *lists* instead and to add special inference rules.) Intuitively, however, $\langle \mathcal{L}', \mathcal{C}' \rangle$ states features of the relation $\vdash_{\mathcal{C}}$ which are intended to describe properties of the connectives of $\mathcal{S}$. So a Gentzen-style sequent-system is a rule-based system with a consequence relation $\vdash_{\mathcal{C}'}$ that can be viewed as a description of a deductive system with a consequence relation $\vdash_{\mathcal{C}}$ such that $X \vdash_{\mathcal{C}} A$ if and only if $\emptyset \vdash_{\mathcal{C}'} (X \vdash A)$.

(3) However, in the Gentzen tradition more complicated types of consequence relations '$\vdash$' have been investigated that do not completely fit the pattern described. One such approach is to consider consequence relations with a set of formulas appearing on the right side of '$\vdash$' (*multiple-conclusion logics*, see MULTIPLE-CONCLUSION LOGICS, sometimes also called *Scott consequence relations*). The intended reading of such a set is disjunctive: '$X \vdash Y$' means that under the assumptions X at least one element of Y holds (although that cannot be formally derived!). Gentzen has proposed this idea to formulate sequent calculi for classical logic in which each connective obtains

a 'natural' meaning via introduction rules for this connective on the right or left side of the turnstile without having to consider special axioms or rules to guarantee the validity of e.g. $A \vee \sim A$ or $\sim\sim A \supset A$. Negation and disjunction rules in this system can be formulated as follows:

$$X \cup \{A\} \vdash Y \Rightarrow X \vdash Y \cup \{\sim A\} \quad (\sim - \text{introduction right})$$

$$X \vdash Y \cup \{A\} \Rightarrow X \cup \{\sim A\} \vdash Y \quad (\sim - \text{introduction left})$$

$$X \vdash Y \cup \{A, B\} \Rightarrow X \vdash Y \cup \{A \vee B\} \quad (\vee - \text{introduction right})$$

$$X \cup \{A\} \vdash Y \ , \ X \cup \{B\} \vdash Y \ \Rightarrow X \cup \{A \vee B\} \vdash Y \quad (\vee - \text{introduction left})$$

(Note the duality between these $\vee$-rules and the $\wedge$-rules formulated above, the right and left sides of the sequents being interchanged.) The *tertium non datur* $\{\vdash A \vee \sim A\}$ is then obtained from $\{A\} \vdash \{A\}$ via $\vdash \{A, \sim A\}$. It is obvious that this notion of consequence has strong elements of symmetry which are characteristic of classical logic.

(4) Another approach is to restrict the structural principles underlying deductive systems. If in $X \vdash A$ we consider X no longer as a set but as a list of formulas, we may discuss whether in all cases we want to assume that elements of X are *permutable* (i.e., $X, A, B, Y \vdash C \Rightarrow X, B, A, Y \vdash C$ holds), or whether two identical formulas in X can be *contracted* to a single one (i.e., $X, A, A, Y \vdash C \Rightarrow X, A, Y \vdash C$ holds), or whether we want to assume or reject the principle of *thinning*. Such considerations, which apply analogously to the multiple-conclusion case lead to so-called *substructural logics*, which have become important in Computer Science and Linguistics (but not only there), in particular *relevant logics* (logics without thinning), *contraction-free logics* (the multiplicity of formulas counts), *linear logics* (without thinning and contraction) and *Lambek logics* (without thinning, contraction and permutation). If at the level of general

consequence relations we consider logics without monotonicity, we enter the general field of *non-monotonic logics* which is of special interest in Artificial Intelligence, since it helps to frame many common forms of reasoning.

(5) In *proof theory* (see PROOF THEORY) one is often interested in the structure of the derivation that leads from a set of assumptions X to a conclusion A and in criteria when two proofs of the same 'derivability fact' $X \vdash A$ are to be regarded identical, rather than in whether $X \vdash A$ holds or not. Significant cases are calculi of natural deduction (see NATURAL DEDUCTION). With respect to derivability, they can be described as Gentzen sequent systems in the sense above. However, crucial features of the fine structure of derivations such as the fact that different occurrences of the same formula as an assumption may be treated differently, cannot be captured that way.

Nonetheless one should be aware that in cases (3) - (5) as well as in other cases, in which extensions or modifications of the usual concept of a deductive system are considered with respect to some 'object logic', the system in which this object logic is described has itself the structure of a deductive system in the ordinary sense. Sequent-calculi for a multiple conclusion consequence relation $\vdash$, perhaps with substructural features, are themselves rule-based systems. Even the structure of natural deduction proofs can be characterized by a rule-based system, viz. a system of type assignment for λ-terms. Therefore, it is not so much the specific type of consequence relation *studied* which makes logical systems formal systems, but rather the fact that the variety of logical systems can be described *in terms of formal systems* in the very specific sense of a rule-based system.

References and further reading

Došen, K., Schroeder-Heister, P. (eds.) (1993) Substructural Logics, Oxford: Clarendon Press. (Overview of logics with restricted structural rules.)

Frege, G. (1879) Begriffsschrift, eine der arithmetischen nachgebildete Formelsprache des reinen Denkens. English translation in From Frege to Gödel: A Source Book in Mathematical Logic (J. van Heijenoort, ed.), Harvard University Press 1967. (The first formulation of a complete axiom system for first-order logic.)

Gentzen, G. (1934/35) 'Untersuchungen über das logische Schließen', English translation in The Collected Papers of Gerhard Gentzen (M.E. Szabo, ed.), Amsterdam: North-Holland. (Gentzen's doctoral thesis, in which he created the notions of natural deduction and sequent calculi.)

Harrison, M.A. (1978) Introduction to Formal Language Theory, Reading Mass: Addison Wesley. (Like Salomaa 1973 an advanced textbook of formal language theory.)

Hopcroft, J.E., Ullman, J.D. (1979) Introduction to Automata Theory, Languages and Computation, Reading Mass: Addison-Wesley. (Elementary textbook of formal language theory.)

Kleene, S.C. (1952, 1974) Introduction to Metamathematics. Groningen etc.: Wolters-Noordhoff et al. (Classic textbook of mathematical logic with emphasis on logical calculi.)

Salomaa, A. (1973) Formal Languages, New York: Academic Press. (Like

Harrison 1978 an advanced textbook of formal language theory.)

Takeuti, G. (1975) Proof Theory. Amsterdam: North Holland. (Textbook of Gentzen-style proof theory.)

Tarski, A. (1983) Logic, Semantics, Metamathematics. Papers from1923 to 1938, Indianapolis: Hackett. (Contains Tarski's basic logical papers, particularly the one of 1930 'On some fundamental concepts of metamathematics', in which he defined the abstract notion of consequence for the first time.)

Wójcicki, R. (1988). Theory of Logical Calculi: Basic Theory of Consequence Operations, Dordrecht: Kluwer. (A comprehensive development of the theory of consequence relations.)

HEINRICH HERRE & PETER SCHROEDER-HEISTER