Meta
Journal des traducteurs
Translators' Journal

META

A Formal Language to Convey Linguistic Information. A Study in Practical Logic

Leonid L. Tsinman

Volume 40, numéro 4, décembre 1995

URI : https://id.erudit.org/iderudit/004514ar
DOI : https://doi.org/10.7202/004514ar

Aller au sommaire du numéro

Éditeur(s)

Les Presses de l'Université de Montréal

ISSN

0026-0452 (imprimé)
1492-1421 (numérique)

Découvrir la revue

Citer cet article

Tsinman, L. L. (1995). A Formal Language to Convey Linguistic Information. A Study in Practical Logic. *Meta*, *40*(4), 614–617. https://doi.org/10.7202/004514ar

Résumé de l'article

L'auteur discute de problèmes relatifs à la création de langages formels pour représenter les données linguistiques dans des systèmes de traduction automatique ou dans des interfaces linguistiques pour la communication homme-machine.

A FORMAL LANGUAGE TO CONVEY LINGUISTIC INFORMATION. A STUDY IN PRACTICAL LOGIC

LEONID L. TSINMAN
Institute for Information Transmission Problems, Russian Academy of Sciences, Moscow, Russia

Résumé

L'auteur discute de problèmes relatifs à la création de langages formels pour représenter les données linguistiques dans des systèmes de traduction automatique ou dans des interfaces linguistiques pour la communication homme-machine.

Abstract

The author discusses problems of creating formal languages to present linguistic data in machine translation systems or linguistic interfaces for man-computer communication.

For the last few years, the Laboratory of Computational Linguistics at the Institute for Information Transmission Problems, Russian Academy of Sciences, which is headed by Prof. Jurij Apresjan, has been developing various systems of natural language processing, including machine translation systems, linguistic interfaces for natural language man-computer communication, etc. (see Apresjan *et al.* 1989; Apresjan *et al.* 1992).

As such systems must tackle very large masses of versatile and intricately organized linguistic data, a fundamental problem is to create an adequate formal language in which these data could be presented. One of the most important requirements is that the language should be equally friendly to both the linguists who supply "primary" information and to the programmers who implement the system concerned on the computer. The linguists need a sufficiently free formal language which could provide them with a good choice of means of expression and which would be close enough to a natural language. As far as the programmers are concerned, however, the formal language must ensure that the algorithmic treatment of statements written in it could be as simple as possible. As these two sets of requirements obviously contradict one another, the success or failure of the whole enterprise largely depends on whether a reasonable compromise between the approaches could be reached.

In the systems developed by our laboratory, which are based on the "Meaning <—> Text" linguistic theory (Mel'čuk 1974), a very large proportion of linguistic data is naturally presented in the form of rules which consist of two parts: 1) list of conditions to be checked, which describe a certain linguistic reality, and 2) list of actions to be done once the above conditions are satisfied.

In practice, the major difficulties in developing the formal language have been associated with the formalization of the conditions to be checked. Generally speaking, these conditions could be viewed as arbitrary logical expressions, which made it clear from the start that the formal language would need to be of logical nature.

When the work was to be launched, the developer had access to a large corpus of linguistic descriptions (rules) for English, Russian, and French, written by linguists with some mathematical and computer science background. Thanks to the latter fact, the rules looked rather formal but their level of formality was obviously insufficient to allow

immediate implementation. An attentive analysis of the material made it possible to start work by specifying the signature of the formal logical language. The signature comprised around 600 linguistic terms (undefined notions), such as morphological characteristics, syntactic and semantic features, names of syntactic relations, etc., as well as over 100 predicates representing various types of statements to be made about the terms. In addition to the terms (object constants) and the predicate constants, the signature was made to include all necessary sorts of variables which have been widely used in formulating the conditions of the rules.

Generally speaking, a signature, when specified, defines the formal language. Any condition of a linguistic rule will in this case need to be represented as a logical expression over the given signature. Such a language has many advantages and could easily satisfy the linguists who work with the system. However, such a language would have a very serious drawback: the verification of logical expressions of arbitrary kinds is an extremely complicated task. As is known, even very poor logics which make wide use of variables may bring about the situation when the computer fails to exhaust the acceptable values of the variables. This is known to be a major drawback shared by all logical languages which makes them virtually unfit for use.

Meanwhile, it seems reasonable to ask the following question: must any formal language be made to handle all kinds of logical expressions, however cumbersome and complicated? It is well known that a person who makes a logical utterance and hopes to be understood is most likely to avoid statements with a complex logical structure, trying whenever possible to break them into several simpler statements and specifying salience points which will help the hearer to grasp the logical structure.

This implies that a human being who is reasoning within a logical language which implicitly underlies his statements uses in fact a very limited subclass of expressions feasible in this language. Unfortunately, little is known yet about this subclass of logical expressions, which could be referred to as "practical logic."

It goes without saying that good understanding of "practical logic," which is doubtless an important feature of human thinking, is of great theoretical importance. Such an understanding is, however, simply unvaluable for implementational issues with which we are primarily concerned: the algorithm of natural language processing should be tuned to process only those logical expressions which stay within the "practical logic" fragment, rather than to handle any logical expression possible. It is not unwise to expect that such an algorithm will be considerably simpler. Of course, when developing the formal language it should be borne in mind that any restrictions imposed on it can only be introduced in a way natural to the user rather than formulated in special logical science terms. It is also important to retain all the advantages of the formal language while imposing the restrictions.

A very important measure of the complexity of a logical utterance is the form in which the quantifier prefix for its variables is composed. A human will find it difficult to grasp the meaning of, or generate, an utterance whose quantifier prefix has more than two groups of quantifiers (the number of quantifiers within each group is less important). Another complexity parameter of a logical utterance is the use of alternate logico-algebraic operations. Even not too cumbersome an expression is difficult to grasp if logical operations appear in it in an irregular way. Finally, it must be noted that the arrangement of logical utterances stated in a natural language is rather consistent, too. By way of illustration, it could be recalled that when describing a phenomenon or an event the speaker typically proceeds from a certain number of objects and facts known beforehand and introduces new objects, or variables, in a gradual, "cautious" way. This is why correctly built logical judgements look as if they were gradually "unfolding." If one tries to trans-

pose elementary constituents of such a judgement, even in a equivalent way, the result may become unnatural and incomprehensible. So, the "practical logic" has its own regulations: even the conjunction may prove non-communicative in it.

These peculiar features of "practical logic" (as well as a number of other similar features) were taken in account when the formal logical language to convey linguistic information was designed. One of the ideas was to develop a convenient structure of rules. In the language, the structure allows to decompose, in a clear and natural way, any complex logical descriptions.

A rule may consist of a number of alternate subrules. Any subrule may include conditions of two sorts: 1) necessary conditions describing linguistically relevant text fragments which must be present for the rule to be applicable, and 2) excluding conditions which describe fragments whose presence renders the rule inapplicable.

The conditions of the first sort state that "a set of values of variables exists such that..." whereas the second type conditions assert that "it is not true that a set of values of variables exists such that..." Such a detailed decomposition allows to present any condition of a rule (both necessary or excluding) as a logical expression in which the quantifier prefix for all variables occuring in this condition is composed of existence quantifiers only (in fact, the universal quantifiers in the excluding conditions are transformed into the existence quantifiers with the help of negation). Due to this fact, quantifiers may be omitted in the formal language (in much the same way they are omitted in informal descriptions), while their interpretation and scope is done by the algorithm.

Any condition is written in a disjunctive normal form (DNF), *i.e.* it is presentable as a disjunction of conjunctions of predicates or negations of predicates. A DNF logical expression has the most transparent logical structure possible (as all options are explicitly listed).

Finally, our formal language requires that all new variables in logical descriptions be introduced in a way that excludes the appearance of an expression in which the description does not "unfold," in the sense discussed above.

So far, we have considered the logical expressions from the viewpoint of "practical logic." It is also necessary to classify such expressions with respect to the complexity of truth value verification. It is reasonable to define the algorithmic complexity of a logical expression as the total number of checks necessary to verify the truth values of all predicates contained in the expression. In this case, it is easy to see that logically equivalent expressions may have different algorithmic complexities. Even a transposition of elements of a simple conjunction may drastically increase the complexity due to an uneconomical lookover of variable values (provided the direction of conjunction processing remains the same).

A very interesting, if expected, fact deserves a special mention: the more transparent and comprehensible are logical descriptions, the less complex is their truth value verification (as compared to other logically equivalent descriptions). So, when developing formal language features described above (restrictions on the use of quantifiers, DNF, the "unfolding" character of the description), as well as other features not mentioned here (see Tsinman 1986 for details) our aim was twofold: to make logical descriptions natural and easy to understand, and to reduce, to the maximum extent possible, the algorithmic complexity of processing these descriptions.

To conclude, I would like to note that hundreds of rules have by now been written in this formal language, including the rules fully describing the syntax of English and Russian, and have been succesfully tested in systems operation. It should also be added that our formal language is not only a working language for the linguists, but also a language in which linguistic data are published (see Apresjan *et al.* 1989; Apresjan *et al.* 1992).

REFERENCES

APRESJAN, JU.D., BOGUSLAVSKIJ, I.M., IOMDIN, L.L., LAZURSKIJ, A.V., PERTSOV, N.V., SANNIKOV, V.Z. and L.L. TSINMAN (1989): *Lingvističeskoe obespečenie sistemy ÈTAP-2* (The linguistics of the ETAP-2 machine translation system), Moscow, Nauka, 295 p.

APRESJAN, JU.D., BOGUSLAVSKIJ, I.M., IOMDIN, L.L., LAZURSKIJ, A.V., MITJUŠIN, L.G., SANNIKOV, V.Z. and L.L. TSINMAN (1992): *Lingvističeskij processor dlja složnyx informacionnyx sistem* (A linguistic processor for advanced information systems), Moscow, Nauka, 256 p.

APRESJAN, JU.D., BOGUSLAVSKY, I.M., IOMDIN, L.L., LAZURSKY, A.V., SANNIKOV, V.Z. and L.L. TSINMAN (1992): «ETAP-2: The Linguistics of a Machine Translation System», *Meta*, 37-1, pp. 97-112

MEL'ČUK, I.A. (1974): *Opyt teorii lingvističeskix modelej "Smysl <—> Tekst"* (A study in linguistic models of the Meaning <—> Text type), Moscow, Nauka, 314 p.

TSINMAN, L.L. (1986): «Jazyk dlja zapisi lingvističeskoj informacii v sisteme avtomatičeskogo perevoda ETAP-2. Opyt "praktičeskoj logiki"» (A language to convey linguistic information in the ETAP-2 machine translation system. A study in "practical Logic"), *Semiotika i informatika*, issue 27, pp. 82-120