

Research Program

The core of my research involves application of concepts and tools of Mathematical Logic to the study of grammar formalisms, particularly those intended to capture natural languages. The fundamental goal is to employ these tools to characterize, in a highly abstract way, the types of properties these formalisms can enforce. Such characterizations not only help in developing accounts of syntactic phenomena that are appropriate for a given formalism, but provide, as well, a means of comparing accounts of those phenomena as expressed in different, even highly disparate, formalisms. Similarly, they can establish relationships between grammatical formalisms and automata-theoretic mechanisms which potentially provide feasible approaches to processing them. In exploring the practical application of this work I have begun to develop a system for producing and maintaining large, but computationally tractable, grammars purely in terms of the basic linguistic principles they are meant to express—a sort of logic programming for natural language grammars.

More recently, I have begun working on a problem in Software Engineering and Digital Circuit Design that lies within the intersection of Natural Language Processing and Formal Logic: the problem of reliably developing formal specifications from requirements expressed in natural language.

In the following sections I discuss each of these threads in more detail.

Model-Theoretic Syntax

This work is the continuation of my Doctoral work. It focuses on two broad types of issues: Model-Theoretic properties of the classes of sets of structures licensed by grammar formalisms and descriptive characterizations of those classes of sets of structures, that is, characterizations in terms of definability by logical languages with varying complexity over restricted classes of models. Studies of this type have come to be grouped under the rubric *Model-Theoretic Syntax*. This is an emerging area in the study of the formal properties of language and my work has played a significant role in its development.

These studies have their roots in characterizations by Büchi and Elgot of the class of regular languages by definability in the weak monadic second-order theory of the natural numbers with successor (wS1S) and the extension of those results by Doner and Thatcher and Wright to characterize the context-free languages by definability in the weak MSO theory of multiple successors (wSnS). In my Doctoral research I adapted these characterizations to a more linguistically oriented framework and employed the result to explore the complexity of a more-or-less standard Government and Binding Theory (GB) account of English D- and S-structure. I obtained both definability results and non-definability results: establishing both that this language is, in essence, context-free and obtaining a characterization, in terms natural to GB, of the kinds of principles that separate CF from non-CF accounts.

The practical value of this work is the fact that it potentially provides a mechanism for converting principles of GB into a certain class of finite-state automata, thus providing an approach to processing GB grammars efficiently. This idea is being pursued by a group under Uwe Mönnich at the University of Tübingen.

In post-doctoral work, I applied this approach to Generalized Phrase Structure Grammar (GPSG) a representative of one of the major families of alternatives to GB. This work clarifies a number of fundamental issues within GPSG and, by interpreting both GB and GPSG within the same logical framework, provides a foundation on which to compare these fundamentally dissimilar formalisms. From this perspective, many of the issues which have been topics of debate between the approaches can be seen to have little substantive consequence. The key difference between them seems to be the type of the Universal principles they employ, with GB employing first-order principles (properties of trees) and GPSG employing second-order principles (properties of sets of trees).

The primary limitation to this approach has been the relative weakness of the CFLs in capturing the syntax of natural languages. Consequently, there has been a great deal of interest in extending these results to larger language-theoretic complexity classes—a task complicated by the fact that SnS is generally understood to stand very near the boundary of decidable theories. (The collection *Mathematics of Syntactic Structure*, edited by Mönnich and Kolb, provides a snapshot of the work in this area as of 1997.) I have obtained a natural generalization by placing the results in a hierarchy of classes of models characterized by concatenation of structures of increasing dimensionality: At

level 0 there is no concatenation and we obtain the finite languages; at level 1 we concatenate strings (one dimensional objects) and obtain the regular languages; at level 2 we concatenate trees (two-dimensional) and obtain the CFLs; at level 3 we concatenate three-dimensional analogs of trees and obtain the class of *Tree-Adjoining Languages* (TALs); and we proceed in this manner through an infinite hierarchy. In current work, I have established that the classes of sets of strings accepted by finite-state automata operating on the structures in this hierarchy coincide with Wier's version of the *Control Language Hierarchy*.

From the perspective of model-theoretic syntax, the significance of these results is that the proofs employed in characterizing the regular and context-sensitive languages in terms of definability in $wS1S$ and $wSnS$ carry over uniformly to all levels of the hierarchy. Thus we have obtained descriptive characterizations not only of the TALs, but of an infinite hierarchy of mildly context-free languages, characterizations which provide means of transforming logical constraints of increasing complexity into automata-theoretic mechanisms for processing them. From the model-theoretic perspective, the significance of these results is that they promise, in their generalization to infinite structures, to establish an infinite hierarchy of decidable monadic second-order theories beyond SnS .

Towards “Logic Programming” for Tree-Adjoining Grammars

At the three-dimensional level, this work provides a mechanism for defining Tree-Adjoining Grammars purely declaratively in a concise and linguistically transparent way. I have shown that there are a number of issues arising in current TAG accounts of natural language that can be clarified using this approach. Moreover, as with GB and GPSG, this common abstract framework provides a way of relating TAG accounts to those expressed in other formalisms. This allows, for instance, one to import a number of attractive aspects of GPSG directly into TAGs.

As with the work on GB there are potential practical benefits from implementing these mechanisms, albeit of a complementary sort: the GB work allows existing abstract theory to be reduced to concrete mechanisms, the TAG work promises to allow theoretical issues to be abstracted away from their concrete realization. Currently I am undertaking, in collaboration with the Tübingen group, a program to develop a practical system for compiling systems of logical constraints into TAGs and for verifying that existing TAGs satisfy such constraints.

Interactive Formalization of English Language Constraints

The final component of my current research addresses what may be the last fundamental gap between the potential of formal methods for developing verified systems and the practical difficulties of applying them. The last ten years have seen the development of interactive mechanisms for synthesizing systems (realized either as software or circuits) from formal specifications with a high degree of confidence that the resulting system satisfies the specifications. Such mechanisms couple the design and verification processes in a way that makes it possible for an engineer, working in a framework familiar from existing design methodologies, to produce both the system and its proof of correctness at the same time. In this way one obtains the benefits of formal verification while avoiding the difficult and highly error-prone process of formally verifying a design after the fact.

This leaves the difficult and error-prone process of producing the formal specification to begin with. The difficulty of this process can be traced to substantive cognitive factors: the intuitive level on which requirements are understood involves a fundamentally different mode of thought than is required to produce and understand their formal realization. I have undertaken, along with a graduate student, to address this problem by providing interactive tools for translating specifications expressed in natural language into equivalent specifications expressed in Higher-Order Logic (HOL). We view this as a problem of obtaining an HOL representation of the intended semantics of the natural language input.

There are several key issues involved. The formal semantics of natural language is far from being fully understood. However, the fact that we are working with a highly restricted and relatively formal domain allows us to focus our attention on a relatively small fragment of language with reasonably simple semantics. Nonetheless, natural language is highly ambiguous and often imprecise, even when restricted to a formal register. This manifests itself in a multiplicity of readings for a given input. In formalizing that input, one must resolve these not just into a single reading but into one that is, with a high degree of confidence, the intended reading. We propose to resolve this issue by making the system incremental, bi-directional and interactive. As ambiguity arises in the translation process, the system will present natural language paraphrases of the alternative readings and require the engineer to choose the intended

meaning. This has the effect both of giving the highest possible likelihood of obtaining the intended meaning, and of helping the engineer to adopt a relatively unambiguous style in the input. Once the translation is complete, we can use the same mechanism to provide a natural language paraphrase of the full specification. Thus, as with the formal synthesis systems, while the engineer guides the process, they work only in a familiar domain.

Since the systems for reducing the formal specifications to software or circuits are necessarily interactive, the system we envisage should fit together with them seamlessly as a front-end. The plan is to implement this using the mechanism of Synchronous Tree-Adjoining Grammars, partly because it is inherently bi-directional and partly because there is an existing wide coverage TAG for English.

This is a long term project comprised of many relatively small components spanning a wide range of difficulties. It provides opportunities for research by students at a variety of levels. Currently, I am working with a graduate student on a system for translating Statecharts into an English language expression of the semantics of their transitions. This will provide a foundation for the same student's Doctoral research in which he will be developing a system for testing whether a given Statechart satisfies a given English language constraint (of a restricted type) by translating both into HOL and then using existing HOL theorem provers. Put together with the current work, this will provide a system that not only can verify that a Statechart correctly implements a high-level specification, but can also provide an English language description of the circumstances under which it fails to do so.