MCHART: A FLEXIBLE, MODULAR CHART PARSING SYSTEM

HENRY THOMPSON
Department of Artificial Intelligence
University of Edinburgh
Hope Park Square, Meadow Lane
Edinburgh EH8 9NW

ABSTRACT

One of the most attractive properties of the active chart parsing methodology (Kay 1980; Thompson and Ritchie 1983) is the distinction it makes possible between essential bookkeeping mechanisms, scheduling issues, and details of grammatical formalisms. MCHART is a framework within which active chart parsing systems can be constructed. It provides the essential bookkeeping mechanisms, and carefully structured interfaces for the specification of scheduling and grammatical formalism. The resulting flexibility makes it useful both for pedagogical purposes and for quick prototyping. The system is available in UCILISP, FranzLisp, and Interlisp versions, together with a simple lexicon facility, example parsers and detailed documentation.

I. ACTIVE CHART PARSING: A BRIEF OVERVIEW

Constraints on space make it impossible to describe the active chart parsing methodology in any detail. This section merely presents a sketch, to outline the main points and establish salient terminology. For more detail see Kay or Thompson and Ritchie (op. cit.).

Active chart parsing takes the idea of a well-formed substring table, which is a passive lattice consisting of one edge per constituent (either initial or discovered), and makes it active by using edges to record partial or incomplete hypothesized constituents as well. Such active edges contain not only a description of the partial contents of the hypothesized constituents, but also some indication of what more is required to complete them. It follows that the central point in the active chart parsing process occurs when an active and inactive edge meet for the first time. In each such circumstance, one or more new edges may be added if the inactive edge (partially) satisfies the requirements for completion of the active edge. How this is determined will of course depend on the grammatical formalism being used. We call this whole operation <u>the fundamental rule</u>. Note that it speaks of <u>adding new edges</u>, not changing old ones - this is crucial, as will become clear later.

The fundamental rule alone is not of course sufficient to transform an initial chart of lexical edges into a fully analysed one. For this a source of empty active edges, that is, initial hypotheses about possible constituents, is also required. Such edges typically arise either in response to the addition of inactive edges to the chart (data-driven), or of active edges (hypothesis driven), based on the contents of the edges and the grammar. We call the choice a particular parser makes on this issue the <u>rule invocation strategy</u>, as it determines when and how rules in the grammar enter the chart as hypothesized constituents.

Both the fundamental rule and the rule invocation strategy give rise to non-determinism. The addition of a single edge to the chart, either active or inactive, may provoke a number of applications of the fundamental rule, each in turn possibly provoking new edges. Similarly, a new edge may provoke the rule invocation strategy to hypothesize a number of potential constituents. We call the question of how to order the pursuit of these various possible additions the <u>search strategy</u>.

II. INTERFACES TO MCHART

The core of MCHART simply provides a multi-level agenda mechanism, data structures for the vertices and edges which make up the chart, and a functional interface for submitting new edges for addition to it. We will first describe these, and then the mechanisms provided for specifying grammatical formalism, rule invocation strategy, and search strategy.

In understanding what follows, the reader must keep in mind that MCHART is not itself a parser - it is a framework for implementing parsers. When one sets out to build a particular parser using MCHART as a base, decisions must be made on each of the dimensions set out in the preceding section. Each subsection below specifies the interface whereby one such decision is communicated to MCHART.

A. The Agenda

MCHART is based on an agenda mechanism for dealing with the non-determinism inherent in its structure. This mechanism maintains an ordered set of ordered queues of function applications. It proceeds by removing and evaluating the first entry in the highest priority queue. Primitives exist for adding function applications to the beginning or end of particular queues, and for dynamically reordering a given queue. Modelled on the agenda mechanism of the GUS system (Bobrow et al, 1977a, this approach provides a simple yet powerful tool, whose use pervades MCHART and provides its overall architectural structure.

B. Data structures and the submission of edges

The chart is composed of vertices and edges. A vertex is a named collection of edges. The edges are held in four fields, distinguishing active from inactive and incoming from outgoing. An edge contains its left and right vertices (not by name, but directly, so the structures are circular), its name, and a label, which is not interpreted by MCHART, but is for use by each particular parser. Note that edges are not explicitly identified as active or inactive - only implicitly by where they are attached to their vertices.

A function named NewEdge exists for the construction and submission of new edges. The caller must specify left and right ends, label, and type (active or inactive). The edge is constructed immediately, but will not normally be added directly to the chart. Rather a function application to do this is scheduled on a queue determined by the type of edge.

When this application is evaluated, the edge is added to the chart. If the original caller provided a redundancy checking predicate, for instance to avoid indefinite left recursion in a top-down system, this will be evaluated at this point as well, and may forestall the addition of the edge.

C. Signals, the Fundamental Rule, and Rule Invocation

In constructing a framework intended to support a number of different styles of use, it proved useful to provide an interface less rigid than that of function call and return. At salient points in the basic operation of MCHART, the system raises named signals. If the particular parser involved has not declared a response to such a signal, processing continues. But if a response has been declared, in a global signal table, then the function named therein is called with appropriate arguments. Not only does this allow the parser writer to easily choose what signals to ignore and what to respond to, but it also allows a simple mechanism for the expression of defaults. There are in fact two signal tables - the public one, which is checked first, and one private one, which is checked only in the absence of a relevant entry in the public one.

This mechanism is used both for the fundamental rule and for rule invocation strategies. Whenever an edge is actually added to the chart, a signal called Fundamental is raised for each active-inactive edge pair which results. The parser may provide its own response, but the system provides a default, which simply schedules the application of a parser-provided function to each pair which satisfies a parser-provided predicate.

Similarly, a signal called AddingActiveEdge or AddingInactiveEdge is raised depending on the type of edge being added. There are no defaults for these, but the parser will typically respond to one or the other to implement a (top-down or bottom-up respectively) rule invocation strategy.

Other signals are raised at the creation of vertices and edges, and at various points during the opening and closing phases of the parsing process. Most of these have standard defaults, which provide simple names for the vertices and edges, construct the lexical edges, initiate top-down parsing, print out results, and so on.

D. Search Strategies

The agenda mechanism provides the necessary hooks for controlling search. The parser can specify the relative priority of the queues for new inactive edges (distinguishing lexical from complex if desired), new active edges, and the edge pairs resulting from the application of the fundamental rule. The queueing disciplines (LIFO, FIFO, or computed) can also be specified for each of these queues. In general, LIFO will produce a depth-first search, while FIFO will produce a breadth-first search, at least in conjunction with a top-down rule invocation strategy.

E. Functional interfaces and grammatical formalisms

At several points in the above description, reference has been made to the parser writer not only naming functions as response to signals, but also providing functions for specific tasks. At each point in the parsing process where something must be done to an edge or edges, but the way it is done depends on the internal structure of the edge label, that is, on the particular grammatical formalism involved, then MCHART calls a function which the user must provide. Including the functions called by the default response to the Fundamental signal, there are five such functions, two for the fundamental rule, two involved in displaying the contents of edges, and one for constructing lexical edges.

III. EXPERIENCE IN USING MCHART

The modular nature of MCHART as set out above was initially determined by a desire to use it in teaching students about parsing. In particular I was concerned to distinguish between the issues of search strategy, rule invocation strategy, and grammatical formalism. By using a distinct type of interface for the specification of each of these, their independence was demonstrated, and it became possible to vary them independently. Example parsers have been constructed within which top-down and bottom-up invocation strategies can be contrasted, and each considered with respect to depth-first or breadth-first search. As for grammaticl formalism, students at a 1981 Open University one week residential course were able to successfully convert a parser for context-free phrase structure grammars to recursive transition network, despite having no prior experience with either MCHART or Interlisp. Implementations also exist for various varieties of context-free grammars, including ones with optional or iterated constituents and with various feature systems, for RTNs, and for ATNs, with a number of different styles of tests, actions, and structures.

I have also developed a large parsing system for the GPSG grammatical formalism (Gazdar 1981c; Thompson 1981b- Thompson 1982b) using MCHART and have found it extremely useful to be able to focus on the issues of linguistic concern as the system has evolved, leaving MCHART to take care of the details. The flexibility it provides more than makes up for the price in efficiency which must be paid for it. I have found this particularly true in the area of prototyping, where a few days work has allowed a quick investigation of areas as diverse as lexical access using tree-structured lexicons and extended quasi-context-free formalisms for some crucial problems in linguistic theory (Bresnan Kaplan Peters and Zaenen 1982; Thompson 1983a).

AI has often been criticsied for failing to produce work which builds on other work. If this paper were just about another parser, it would be liable to that criticism. But as it is about a basis for producing parsers, I hope it can go a small way towards helping others avoid re-inventing the wheel. Enquiries are invited from those who would like to make use of MCHART as described herein - try it, you might like it.

REFERENCES

Bobrow, D. G., R. M. Kaplan, M. Kay, D. A. Norman, H. S. Thompson and T. Winograd. 1977a. "GUS-1, A frame-based dialog system". Artificial Intelligence 8(1).

Bresnan, J.W., R. Kaplan, S. Peters and A. Zaenen. 1982. "Cross-serial dependencies in Dutch". Linguistic Inquiry 13.

Gazdar, G. 1981c. "Phrase structure grammar". In P. Jacobson and G. Pullum, editors, The nature of syntactic representation. D. Reidel, Dordrecht.

Kay, M. 1980. "Algorithm Schemata and Data Structures in Syntactic Processing". In Proceedings of the Symposium on Text Processing. Nobel Academy. To appear. Also CSL-80-12, Xerox PARC, Palo Alto, CA.

Thompson, H. S. 1981b. "Chart Parsing and Rule Schemata in GPSG". In Proceedings of the Nineteenth Annual Meeting of the Association for Computational Linguistics. ACL, Stanford, CA. Also DAI Research Paper 165, Dept. of Artificial Ihtelligence, Univ. of Edinburgh.

Thompson, H. S. 1982b. "Handling Metarules in a Parser for GPSG". In Barlow, M., D. Flickinger and I. Sag, editors, Developments in Generalised Phrase Structure Grammar: Stanford Working Papers in Grammatical Theory, Volume 2, Indiana University Linguistics Club, Bloomington. Also DAI Research Paper 175, Dept. of Artificial Intelligence, Univ. of Edinburgh.

Thompson, H. S. 1983a. "Crossed Serial Dependencies: A low-power parseable extension to GPSG". In Proceedings of the Twenty-first Annual Meeting of the Association for Computational Linguistics. ACL, MIT, Cambridge, MA. To appear.

Thompson, H. S. and G. D. Ritchie. 1983. "Techniques for Parsing Natural Language: Two Examples". In Eisenstadt, M. and T. O'Shea, editors. Artificial Intelligence Skills. Harper and Row, London. Also DAI Research Paper 183, Dept. of Artificial Intelligence, Univ. of Edinburgh.