Chart Parsing of Robust Grammars *

Sebastian Goeser

gsr@dhdibm1.bitnet

IBM Deutschland GmbH - GADL
Hans-Klemm-Str. 45
D-7030 Böblingen

1 Introduction

Robustness is a formal behaviour of natural language grammars to assign a best partial description to linguistic events whose strong description is inconsistent or cannot be constructed. Events of this sort may be called defective with respect to a grammar fragment. Defectiveness arises from the performance use that human beings make of language. Since defectiveness can be seen as failure of linguistic description, the principal way to robustness is a method to weaken these descriptions.

Robust parsing, then, is parsing of robust grammars: a parser is robust iff it has the capability to interpret weak grammar fragments correctly. In this paper, I shall try to substantiate this claim by motivating a grammar dependent approach to robust parsing and then describing a chart parsing algorithm for robust grammars. Though only c(ontext) f(ree) grammars will be adressed, there is an obvious extension of the algorithm to annotated (unification-) grammars (WACSG formalism, see Goeser 1990) along the lines of (Shieber 1985).

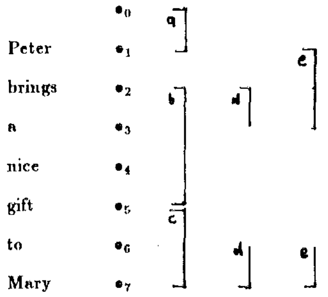
Grammar based robustness tools have been explored in a variety of formalisms, e.g. the metarule device within the ATN formalism (Weischedel and Sondheimer 1893), entity data structures in a case frame approach (Hayes 1984) or the weak description approach in unification based grammars (Kudo et al. 1988, Goeser 1990). Parsing cf grammars with robustness features competes with algorithmic approaches to robustness where parsing algorithms, (usually chart parsers except in Tomabechi and Tomita (1988) where LR(k) parsing is advocated) are extended to include robustness features (Mellish 1989, Lang 1988) and/or heuristics to handle defect cases (Langer 1990, Stock et al. 1988).

Maybe the most critical issue in robust parsing is ambiguity, which emerges when constituency is loosened to some cf substring analysis. E.g. Mellish (1989) parses for a cfg G the (cf) set $PAR(G)$ which is the set of all strings containing a sequence of nonempty substrings which is in the cf language $L(G)$ [1]. In the worst case scenario where *all* these sequences are in $L(G)$, we get for a $w \in L(G)$ with an ambiguity k (in G) an exponential ambiguity of $k \times 2^{|w|}$ as an upper bound. Even in a non-worst case, which should be the case of realistic cfgs, local ambiguities from substring analysis massively increase parsing time. E.g. in the (non-defective) example 1, the arcs a, b, c are empirically valid while the arcs d,e are artefacts of an algorithm parsing $PAR(G)$.

[0] The work reported has been done while the author received an LGF grant at the University of Stuttgart.

[1] See Goeser (1990) for a more formal discussion of $PAR(G)$.

(1)



Reflecting syntactic defectiveness in a cfg means to assign it a configurational regularity. Obviously, there is syntactic defectivity which is syntactically nonregular, such as corrupted output from a speech recognition device (Tomabechi and Tomita 1988) [2] or global constituent breaks (Goeser 1991), which can be subjected to syntactic prefix analysis only. On the other hand, there are spoken language constructions (Lindgren 1987, Goeser 1991, Langer 1990) and various kinds of "fragmentary utterances" (Carbonell and Hayes 1983) that definitively show configurational properties.

Let us look at a frequent spoken language construction called restart, as in the German corpus example (2) [3]. Restarts follow a pattern $< \alpha\beta \quad \mathcal{A} \quad \beta'\gamma >$ where the strings α and γ but not β and β' may be empty. The restart marker $\mathcal{A}$ is optional: in 67 from 96 restart samples β, which mostly ends in a constituent break, and β' were separated phonologically by tone constancy, a short pause or without any marking at all [4]. Restarts are a kind of constituent coordination not allowing for ellipsis phenomena such as gapping, left deletion, split coordination or sluicing. The β substring is usually defective and may indeed contain arbitrary noise (see e.g. example (3)) [5].

(2) da [is es dann noch ein $\mathcal{A}$
there [is it then still a $\mathcal{A}$
α β

kommt noch ein anderes Problem hinzu]
comes yet another problem to-that]
β' γ

(3) der Peter [hat konnte das dieses deshalb
the Peter [has could the this therefore

ehemaligen Lieferwagen
former truck

$\mathcal{A}$ hat das gekauft]
$\mathcal{A}$ has it bought]

2 Recursive partial string grammars

Recursive partial string grammars (RPSGs) are cfgs with a set of start symbols and with rules whose left hand side may be indexed with the keyword SET, SUB, or PAR. The SET index on a rule's LHS licenses the adjunction of any start symbol to the right or left of its RHS string. The SUB index licenses arbitrary terminal strings to the right or left of the indexed symbol's lexical projection. The PAR index includes SUB and additionally licenses any terminal strings within this lexical projection. (Left and right sided indices $SETL$, $SUBL$ and $SETR$, $SUBR$, respectively, are also in use). In a derivation relation $\Longrightarrow$ for RPSGs an indexed symbol A_η unifies with category A to give A_η. Formally, SET adjunction participates in the cf derivation relation, while SUB and PAR are interpreted by a recursive generation function gen operating on derivations:

$gen_\omega : t \times (Cat_{ind} \cup Lex)^+ \rightarrow \{0, 1\}$

where ω is a derivation, t its tree structure, Cat_{ind} the set of indexed or non-indexed nonterminals and Lex the set of terminals. The example derivation tree (4) shows SET adjunction (dotted lines) and areas where arbitrary

[2] This material may show phonological regularities, of course

[3] All corpus evidence reported here is psychotherapeutic discourse from the ULMER TEXTBANK

[4] Therefore, Langer's (1990) restart heuristics seems empirically inadequate insofar as it postulates a syntactic restart marker.

[5] For a more thorough discussion of restart syntax, see Goeser (1991).

substrings are licensed by an indexed node. Generally, local arbitrariness within a string may be eaily modelled with an RPSG. Though finite cfls are turned into infinite ones through RPSG indexing, the syntactic description with RPSG is still configurational up to certain local adjunctions.

3 Basic algorithm

As a parsing algorithm to start from, Earley's (1971) chart parser has been chosen, which has a top-down component adaptable to the top-down percolation of index information, and which guarantees a worst case complexity of $O(n^3)$ even for maximal ambiguity. We use the declarative Earley variant in Dörre (1987) . For a cfg $G = < Cat, Lex, P, Sset >$, where Cat is a set of non-terminals, Lex a set of terminals, P a set of rules and $Sset$ a set of start symbols, it is characterized by the following predictor concept:

- the predictor is a relation $D(i, A) \subseteq n^+ \times Cat$ between a vertex $i \leq n$ and a non-terminal A. It is integrated into the completer and scanner components (see below), This has the advantage that no cyclic items i.e. items with an empty string of parsed symbols, have to be asserted to the chart.
- initialization is the special predictor case $D(0, S)$ where S is a start symbol.

Let $V = Cat \cup Lex$, $A \longrightarrow \alpha\beta \in P$ and $0 \leq i < j \leq n$. $Chart[i,j]$ be the set of arcs between vertices i and j and $\stackrel{*}{\Longrightarrow}$ be the transitive cover of the derivation relation. Then every item in the chart may be characterized by the following membership condition [6] which respects both top-down (TD) and bottom-up (BU) information. Remark that for the (basis variant of the) Earley algorithm, while item membership depends on top-down predictor information, the acceptance of input strings is independent of the predictor (Kilbury 1985).

$A \longrightarrow \alpha.\beta \in Chart[i,j]$ iff

[6] see Dörre 1987

[TD] $\exists S \in Sset\ S \stackrel{*}{\Longrightarrow} w^{0,i} A\delta \wedge$

[BU] $\alpha \stackrel{*}{\Longrightarrow} w^{i,j}$

where $\delta \in V^*$

4 The RPSG variant

4.1 Item Concept

In the RPSG variant, items are represented as PROLOG facts

```
item( Number, Lind, Rind, LHS,
      Parsed, To_Parse, RefList)
```

where item number, the -possibly indexed- left hand symbol, the list of parsed symbols and the list of symbols yet to parse are well-known item parts. The variables *Lind* and *Rind* represent the status of substring generation to the left and to the right of the *Parsed* string, respectively. $Lind \neq Rind$ is possible even for the SUB index, since items represent prefix information on a constituent, whereas a PAR index always effects $Lind = Rind$. Partial string information from higher nodes, which is justified only within the appropriate derivation, must be distinguished from SUB or PAR indexing of an item's LHS symbol, which *always* licences arbitrary substrings. To allow reconstruction of a derivation, RefList records the pairs of items (or pairs of rule and item, see below) an item is completed from, or it equals lex for lexical items [7]. To state the chart membership condition of the RPSG variant, we generalize the function *gen* to an argument pair of strings of terminals and possibly indexed non-terminals:

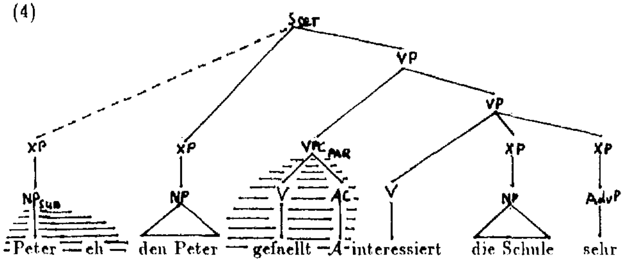
$gen^* : (V^+_{ind})^2 \rightarrow \{0,1\}$
where
$gen^*(\alpha, \beta) = 1$ iff β can be generated from α
$(\ \alpha, \beta \in V^+_{ind}\ )$

The RPSG membership condition, then, is:

$A_\eta \longrightarrow \alpha.\beta \in Chart[i,j]$ iff

[7] The *RefList* is also used for parse forest construction, see e.g. Doerre (1987) for a discussion

(4)



[TD] | $\exists S \in Sset_{ind}\ gen^*(S, w^{0,i} A_\eta \delta) = 1 \wedge$

[BU] | $gen^*(\alpha, w^{i,j}) = 1$ or $\alpha = \epsilon$

where $\alpha, \beta, \delta \in (\dot{V}_{ind})^*$

4.2 The Predictor

The predictor of the RPSG variant [8] is, again, a relation over vertices and non-terminals. In contrast to the basis variant, however, a null predictor would be incorrect for the RPSG variant, since the acceptance of a string now depends on the substring information percolated by the predictor. The first predictor clause allows an "initialisation" for every vertex. The second clause formulates the expectation of a non-terminal A_η by an active item i.e. an item with a nonempty list To-Parse, and the third the expectation by passive items with a SET index. Clause 4 expects a start symbol on the basis of left adjunction to a SET indexed symbol. The following proposition, a proof of which is available from the author, states the correctness of this predictor formalization.

$gen^*(S, w^{0,i} A_\eta \delta) = 1$ iff $D(i, A_\eta)$
for a $S \in Sset_{ind}$

4.3 The Completer

The completer component integrates the predictor relation and the substring generation function and has two rules for rightside and leftside adjunction under a set-indexed symbol. Given that the conditions in the if-clause (and the lookahead condition, see below) yield, the completer adds new items to the chart [9]. Clause 1 of the RPSG completer, is, up to the generation function instead of derivation, equivalent to the completer of the basis variant: Given a rightside passive item, it adds a new item both for a matching active item and for the prediction of an appropriate rules's LHS symbol. Thus, no cyclic items have to be created. Furthermore, since RPSGs do not have ϵ productions, there is no need to handle cyclic items at all. Clause 2 does rightside adjunction of a start symbol item to a passive SET indexed item. In left adjunction according to clause 3, the adjoined (passive) item can again be licensed both by another (active or passive) SET indexed item or by the predictor relation.

4.4 Scanner and Lookahead

Since the scanner component may be seen as a lexical case of the completer, the RPSG algorithm could be reduced to a single active completer component and the controlling relation D (Kilbury 1985). Remark that the scanner allows for RPSG rules with RHS strings of terminals and non-terminals. A partial lookahead of 1, being applied to active items only, has proven advantageous in the basic variant (Dörre 1987). In the RPSG variant, the length of the lookahead must be conditioned to the fact that zero or more non-derived but generated words may follow a given vertex. The lookahead fails if, for the first To-Parse sym-

[8] see Appendix A for a complete formal characterisation of the RPSG chart parser

[9] The relation F includes the operation $\mapsto$ which procedurally asserts new items to the chart

bol, there is no first *derivable* lexical item, that is accessible given the actual substring information.

Unfortunately, the scanner is not independent from this lookahead, since, in many cases, the item licensed by a lookahead operation onto a lexical item i is exactly the item licensing i within the predictor relation. That is, from a procedural viewpoint of entering items into the chart, the lookahead condition and the predictor block each other for certain lexical items. In this situation we decided to have a scanner without a predictor relation, thus paying for lookahead with an increased local lexical ambiguity.

5 Status and Conclusion

The algorithm described has been implemented and tested as part of the WACSG system that is based on the Stuttgart LFG system (Eisele 1987).

Chart parsing of robust cf grammars is a powerful method to cope with the configurational aspects of defectiveness. It is part of a major enterprise to re-analyze robustness not as a parsing problem but as a problem of weak linguistic description. Therefore, any formal work on the linguistics of defectiveness can be expected to improve our methods of robust parsing.

6 Bibliography

[1] Carbonell, J. and Hayes, P.: Recovery Strategies for Parsing Extragrammatical Language, in: AJCL 9, 3-4, 1983

[2] Dörre, J.: Weiterentwicklung des Earley-Algorithmus für kontextfreie und ID/LP-Grammatiken, LiLog-Report 28, IBM Deutschland 1987

[3] Earley, J.: An Efficient Context-free Parsing Algorithm, in: CACM 13, 2, 1970

[4] Goeser, S.: A linguistic Theory of Robustness, in: Proc. of COLING-13, Helsinki 1990

[5] Goeser, S.: Eine linguistische Theorie der Robustheit, Konstanz 1991

[6] Hayes, P.J.: Entity-Oriented Parsing, in: COLING-10, Stanford 1984

[7] Kilbury, J.: Chart Parsing and the Earley algorithm, in: Klenk, U. (ed.): Kontextfreie Syntaxen und verwandte Systeme, Max Niemeyer, Tübingen 1985

[8] Kwasny, S.C. and and Sondheimer, N.K.: Relaxation Techniques for Parsing Grammatically Ill-Formed Input, in: AJCL 7,2, 1981

[9] Lang, B.: Parsing Incomplete Sentences, in: Proc. COLING-12, Budapest 1988

[10] Langer, H.: Parsing Spoken Language, in: Proc. COLING-13, Helsinki 1990

[11] Mellish, C.S.: Some Chart-Based Techniques for parsing Ill-formed Input, in: Proc. ACL 27, Vancouver 1987

[12] Shieber, S.M.: Using Restriction to Extend Parsing Algorithms for Complex Feature Based Formalisms, in: Proc. ACL 25, 1985

[13] Stock, O., Falcone, R., Insimnamo, P.: Island Parsing and Bidirectional Charts, in: Proc. COLING 12, Budapest 1988

[14] Tomabechi, H. and Tomita,M.: The Integration of Unification-Based Pragmatics for Real-Time Understanding of Noisy Continuous Speech Input, in: Proc. AAAI 7, Saint Paul 1988.

[15] ULMER TEXTBANK: A machine-readable corpus of spoken language from psychotherapeutic discourse, University of Ulm

[16] Weischedel, R.M. and Sondheimer, N.K.: Metarules as a Basis for Processing Ill-Formed Input, in: AJCL 9, 3-4, 1983

Appendix

Algorithm: An RPSG Chart Parser

Input:

1. RPSG $G =< Cat_{ind}, Lex, P, Sset_{ind}$
2. string $w = w_1 \ldots w_n$

Output:

"accepted", if $S \longrightarrow \alpha. \in Chart[i,j]$ where
$S \in Sset_{ind}$ and $gen^*(\alpha, w^{0,n}) = 1$

condition (predictor) :

Let $D(i, A_\eta) \subseteq n^+ \times Cat_{ind}$

$D(i, A_\eta)$ iff

1. $\exists S_\zeta \in Sset_{ind}\ gen^*(S_\zeta, g^i A_\eta \delta) = 1$ or
2. $\exists\ C_\zeta \longrightarrow \alpha.B_\lambda\beta \in Chart[j,k] \quad k \leq i \quad \wedge$
   $gen^*(B_\lambda, g^{i-k} A_\eta \delta) = 1$ or
3. $\exists\ C_{SBT} \longrightarrow \alpha. \in Chart[j,k] \quad k \leq i \quad \wedge$
   $\exists D_\zeta \in Sset_{ind}\ gen^*(D_\zeta, g^{i-k} A_\eta \delta) = 1$ or
4. $\exists S_\eta \in Sset_{ind}\ gen^*(S_\eta, w^{0,i} C_\zeta \delta) = 1 \wedge A_\eta \in Sset_{ind} \wedge \exists C_{SBT} \longrightarrow \beta \in P$

condition (lookahead) :

Let $F \subseteq P^\circ \times n^2$.
$F(C_\eta \longrightarrow \alpha.\beta',\ i,\ j)$ iff

1. ($\beta' = \epsilon$ or
   $\beta' = B\beta$ and $gen^*(B, g^{k-j} w^{k,k+1} \delta) = 1$
   for $B \in Cat_{ind}$, $j \leq k \leq n$) and
2. $C_\eta \longrightarrow \alpha.\beta' \mapsto Chart[i,j]$

method:

- *scanner:* For $0 \leq i < j \leq n$:

  if $B_\zeta \longrightarrow w^{i,i+1} w' w^{j-1,j} \in P$ (where $w' \in PP_{w^{i,j}}$ oder $w' = \epsilon$) and $gen^*(B_\zeta, w^{i,j}) = 1$,

  then F($B_\zeta \longrightarrow w^{i,i+1} w' w^{j-1,j}.$,$i$,$j$)

- *completer:* For $0 \leq i < j \leq l \leq n$:

  1. **if** ($A_\eta \longrightarrow \alpha.B\beta \in$ Chart[i,j] or $D(j, A_\eta)$ and $A_\eta \longrightarrow B\beta \in$ P and $\alpha = \epsilon$) and $B_\zeta \longrightarrow \gamma. \in$ Chart[k,l] and $gen^*(\alpha B_\zeta, w^{i,l}) = 1$,

     then F($A_\eta \longrightarrow \alpha B_\zeta.\beta$,i,l)

  2. **if** $B_\zeta \longrightarrow \gamma. \in$ Chart[k,l] and $B \in Sset$ and $A_{SET} \longrightarrow \alpha. \in$ Chart[i,j] and $gen^*(\alpha B_\zeta, w^{i,l})$,

     then F($A_{SET} \longrightarrow \alpha B_\zeta.$,$i$,$j$)

  3. **if** $A_\eta \longrightarrow \alpha. \in$ Chart[i,j] and $A_\eta \in Sset$ and ($B_{SET} \longrightarrow \beta.\gamma \in$ Chart[k,l] or $D(l, B_{SET})$ and $\beta = \epsilon$ and $B_{SET} \longrightarrow .\gamma \in P$) and $gen^*(A_\eta \beta, w^{i,l} = 1)$,

     then F($B_{SET} \longrightarrow A_\eta \beta.\gamma$,$i$,l)