Domes as a Prodigal Shape in Synthesis-Enhanced Parsers [1]

Ife Adebara [a], Veronica Dahl [a]
[a] *Department of Computer Science, Simon Fraser University, 8888 University Drive, Burnaby, Canada*

Abstract.
Research on logic based bottom-up parsing - in particular, around Constraint Handling Rule Grammars [3]- is uncovering shape as an untapped fertile ground for natural language processing in general, and for bottom-up parsing and grammar induction in particular [1]. For instance, commonalities between shapes implicit in molecular biology strings and in natural language strings motivated an award winning paper that, based on this commonality of shapes, uncovered a dual processing scheme for both human and biological languages [5] illustrated around CHRG. In this article we investigate more generally useful ways of visually enhancing spoken human language analysis with static or interactive ways of expressing some of the parsing processes in terms of shape. In particular we examine a) shape replication as a visual aid for determining implicit elements in bottom-up parsing, and b) visual reformulation of parsing results as an economical approach to user-interactive disambiguation and correction. Methodologically, we use CHRG as the computational backbone.

Keywords. language analysis and synthesis, dome shapes for parsing, Constraint-Handling Rule Grammars (CHRG), constraint-based parsing, long-distance dependencies, pronominal references, implicit meanings, disambiguation.

1. Introduction

Usually, natural language processing schemes focus on a single processing mode: either analysis (i.e., producing meaning representations from surface form such as a sentence), or synthesis (generating surface form from meaning representation). There are very few exceptions to this view of language processing, unless the application targeted is one of translation. However even this task is usually not performed as simultaneous analysis and synthesis, but rather, it is tackled by analyzing a sentence in one language into an interlingua meaning representation of it, and then synthesizing, from this interlingua meaning obtained, the same sentence expressed in another language. The analyzing and synthesizing modules of a translator do not generally communicate or intermingle.

[1]This research was supported by NSERC Discovery grant 31611024

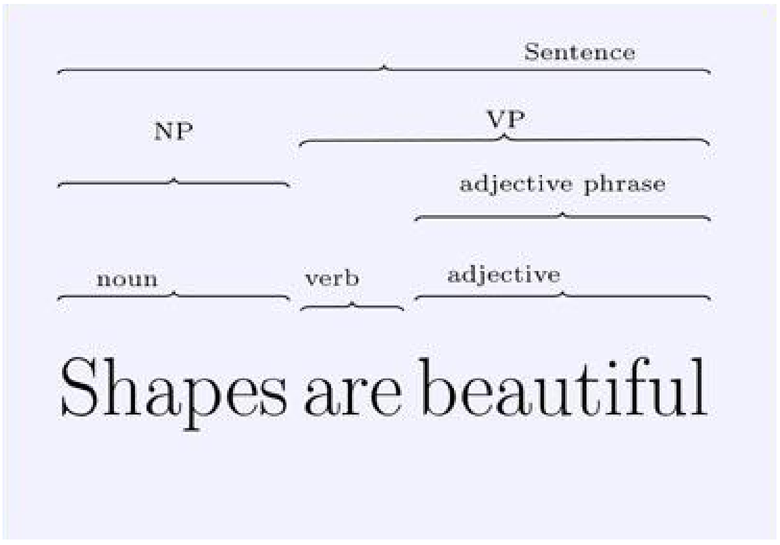

Figure 1: Sample Sentence Analysis

A beautiful trait of Constraint Handling Rule Grammars, or CHRG [3], is that its rules can be used for simultaneous analysis and synthesis. This is possible due to the fact that its processing scheme closely mimics the way we used to do sentence analysis in elementary school: by drawing labelled arcs around subtext, where the labelled groupings correspond to incrementally obtained subsets of our analysis. For instance, given the sentence “Shapes are beautiful”, we would line the words in sequence on a sheet of paper, draw arcs above them labeled by their lexical categories, then draw further arcs for each next level of analysis, and so on, ending up with something like Figure 1.

Admittedly, all we have so far is a graphic variant of a parse tree; however if we can also synthesize text as we go along in a bottom-up analysis, this allows us to superimpose all kinds of information in the same figure (including contextual information, which parse trees do not in general allow), by repeatedly using the basic shape of a dome, defined as a labelled arc between two sentence boundaries. This opens up interesting possibilities: for instance, what if we were able to take an input discourse containing pronouns, and not only analyze these as pronouns, but also superimpose the candidate antecedents they stand for in the same spot they occupy within the sentence? This superimposition would amount to synthesizing alternative words for the substring in question, as exemplified in the dome-decorated rendition of a discourse shown in Figure 2.

Such dome-decorated descriptions, by allowing us to replicate a potential antecedent into the position a pronoun occupies, highlight the available choices for pronominal reference, thus providing a visual formulation of intermediate results (namely the potential antecedents) which the user can then interactively aid in choosing from. For instance, if we had more than one candidate antecedent, as in: “My mother likes colours, I like shapes. They make me think of open spaces.”, the system will superimpose the two potential candidates (“colours” and “shapes”) as possible referents for “They”. For a human being presented with this option, I love shapes. $\overbrace{\underbrace{They}_{Shapes}}^{Pronoun}$ make me think of open spaces.

Figure 2: Synthesizing Alternate Words

it is very easy to choose the correct one and communicate it interactively to the system. On the other hand, leaving the choice to the system would require very sophisticated programming.

Of course, any human could arrive at the same conclusion without the annotations, but drawing one's attention to the possible antecedents in this very direct, visual way makes it much easier, particularly in cases where there are several possible antecedents, or there is more intervening material . For instance, there could be a long tirade, irrelevant to the pronoun reference task, between the first sentence and the second, which would not need to be even read.

Also of course, any parser that can recognize potential referents of a pronoun can present them to the user in any convenient way (including visually), given an appropriate interface. Our point in this paper is that by its very nature, CHRG rules lend themselves very directly to dome-annotation of phrases, and that in turn, phrases thus decorated offer great potential for dynamic fine-tunings during the parsing process- including interactions with human informants- owing to CHRG's ability to synthesize information at the same time as analyzing information, all in the same rule.

Another observation we exploit in this paper is that CHRG's doming constructs allow us to superimpose labels other than constituent names over the subtexts of the initial text. Thus, our dome-shaped representations can also serve to replicate the results of partial analyses in some useful form.

In fact, CHRGs allow us to draw domes even over empty strings, by drawing an arc between a start point and itself, looking like a loop (cf. Figure 5). This allows us to synthesize linguistic information of interest at the point where overt sentence elements are missing, which has interesting applications for reconstructing either these overt sentence elements or their meaning, at the points where they are implicit.

Of course, many different shapes can fulfill the functions the dome shape is chosen for in this paper, but they would be equivalent visual variants. We have chosen it because it is the most natural one, we get it for free with CHR, and it is the one usually chosen in paper and pencil parse traces, from elementary school onwards.

The remainder of this paper is organized as follows: Section 1 reviews our coding tool- CHRG; Section 2 shows that CHRG rules serve to both analyze and synthesize linguistic information; Section 3 shows that its constructs are naturally related to dome shaped visualizations; Section 4 investigates dome shape replication as a tool for synthesis-aided analysis, and Section 5 examines two more uses of shape replication: for helping relate constituents which involve implicit elements, and for human-supervised disambiguation. Section 6 discusses related work and Section 7 presents our concluding remarks.

2. Background: CHRG

CHRGs, or Constraint handling Rule Grammars [3], are a grammatical extension of CHR [14], providing it what DCGs provide to Prolog- namely, they invisibly handle input and output strings for the user. In addition, they include constructs to access those strings dynamically, and the possibility of reasoning in non-classical ways, with abduction or with resource-based assumptions. For the purposes of this paper, we only use two types of CHRG rules, which parallel the CHR rules of propagation and simplification, and are respectively defined as follows:

A propagation rule is of the form

$$\alpha \ \texttt{-\textbackslash} \ \beta \ \texttt{/-} \ \gamma \ \texttt{::>} \ G \ \texttt{|} \ \delta\texttt{.}$$

The part of the rule preceding the arrow `::>` is called the head, G the guard, and δ the body; $\alpha, \beta, \gamma, \delta$ are sequences of grammar symbols and constraints so that β contains at least one grammar symbol, and δ contains exactly one grammar symbol which is a nonterminal (and perhaps constraints); α (γ) is called *left (right) context* and β the *core* of the head; G is a conjunction of built-in constraints as in CHR and no variable in G can occur in δ. If left or right context is empty, the corresponding marker is left out and if G is empty (interpreted as `true`), the vertical bar is left out. The convention from DCG is adopted that constraints (i.e., non-grammatical stuff) in head and body of a rule are enclosed by curly brackets. Gaps and parallel match are not allowed in rule bodies. A gap in the rule heads is noted "...". Gaps are used to establish references between two long distance elements.

A *simplification (grammar) rule* is similar to a propagation rule except that the arrow is replaced by `<:>`.

Operationally, initial grammar symbols in a user's query are kept in a constraint store whose contents evolve through rule application. All rules applicable (i.e., whose left hand side symbols can be matched with symbols in the store and whose guard succeeds) will apply; in the case of propagation rules by adding instances of their right hand side that follow the same matching; in the case of simplification rules, by not only adding those, but deleting as well the symbols that matched the left-hand side ones. Internally, grammar symbols translate into constraints, as we shall explain next.

3. CHRGs as Natural Dome-Shape Inducers

The way CHRGs operate is by invisibly augmenting each grammar symbol of a parse with start and end points. [2] This is done by first adding to each word in the input sentence, word boundaries to reflect their sequential order, as exemplified in Fig. 3.

[2]This in itself is a classic notation, which has percolated into many formalisms, as discussed in our Related Work section

```
Shapes are  beautiful They make me dream of spaciousness
0    1   2          3    4    5  6     7  8           9
```

Figure 3: Word Boundaries

These word boundaries then percolate to every level of analysis, also invisibly to the user.

Specifically, when a user issues a CHRG command such as:

```
?- parse([shapes,are,beautiful]).
```

the following constraints are entered into CHRG's "bag of constraints"- a store in which the rules will operate :

```
word(0,1,shapes)
word(1,2,are)
word(2,3,beautiful)
```

Running that query with respect to the following propagation CHRG rules:

```
 word(shapes) ::> noun(shapes,plural).
 word(are) ::> verb(be,present,plural).
 word(beautiful) ::> adjective(beautiful).
```

will result in the following constraints being added into the store:

```
noun(0,1,shapes,plural).
verb(1,2,be,present,plural).
adjective(2,3,beautiful).
```

A natural correspondence between the effects of CHRG parsing and dome shape construction should now be quite apparent to the reader: any new constraint that enters the store delimits (by its two first arguments) an input string's substring on which to draw a new arc; this arc will be labelled by the name of the constraint plus its remaining arguments. In our example, once the three CHRG rules operate we are left with a store that implies the dome-enhanced string shown in Figure 4 (which is essentially a subfigure of Figure 1, augmented with word boundaries and with linguistic features -a word's overt form, its number- that were collected in the process):

In theory we should also draw domes for each word, but since they would not provide much new clarity (it is already clear from the picture which are words and what are their boundaries), we will not bother doing so.

4. Synthesis-aided Analysis through Dome Shape Replication

We are now in a position to describe how CHRG rules also lend themselves to synthesizing substrings at the same time as they analyze others. Going back to our

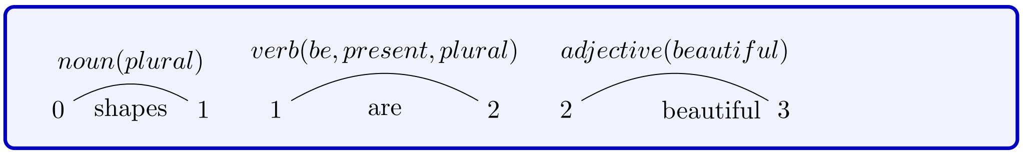

Figure 4: CHRG Parsing and Dome Shape Construction

pronoun reference example, “Shapes are beautiful. They make me think of open spaces”, all our parser needs to do in order to synthesize the potential antecedent at the same position occupied by the pronoun is to add CHR rules which replicate the noun’s dome at the position occupied by the pronoun (recall that internally, CHRG rules compile into equivalent CHR rules in which the word boundaries show explicitly; and that “...” stands for a gap):

```
word(Start,End,they) ==> pronoun(Start,End,they,plural).


noun(Start,End,N,Number) ... pronoun(S1,E1,P,Number) ==>
 noun(S1,E1,N,Number).
```

Note that this time we use *CHR* rules instead of *CHRG* rules as we did in Section 3. This is because we now need to have explicit access to the relevant start and end points in order to reconstruct the missing now at its appropriate place in the string.

This replication of the candidate antecedent’s dome at the position already occupied by the pronoun involves two cooperating processes: the *analysis* of the pronoun and of the antecedent it refers to, followed by the *synthesis*, from both of them (which involves matching the number in both), of another copy of N, between S1 and E1.

We are of course simplifying for explanatory purposes: further conditions need in fact to be tested to ensure the proposed noun phrase is the one the pronoun refers to [3]. But regardless of which is our chosen algorithm and which tests one could add, our point is that the parsing process is now treated as a dual process: the left hand side of our second CHRG rule *analyses* two substrings of interest, as its right-hand side *synthesizes* a copy of one of them by superimposing it where its meaning is needed (i.e., around the pronoun’s boundaries). We coin the term synalysis for this dual process of simultaneous synthesis and analysis.

Alternatively, we could just keep to CHRG rule format (knowing that it will compile into CHR anyway) and still have access to the word boundaries of interest, by using a CHRG construct (noted :(S,E)) that allows us to identify the input (S) and output (E) strings selectively. In our case, we would write:

```
noun(N,Number):(Start,End)...pronoun(P,Number):(S1,E1)::>
noun(N,Number):(S1,E1).
```

[3]Note that our aim here is not to provide a good algorithm for anaphora resolution- this is a very difficult problem on which volumes have been written-, but to facilitate the transfer of information between related constituents once a specific algorithm is chosen

5. Other Uses of Shape Replication

We can now use our shape visualization methodology for solving other language processing problems: how to reconstruct the meaning of elided elements by replicating the overt forms of missing substrings, or even their meaning rather than their form, and how to facilitate quick disambiguation by keeping alternative readings of ambiguous phrases, represented as superimposed domes.

5.1. Treating Long Distance Dependencies with Implicit Elements

5.1.1. Through Syntactic Shape Replication via Analysis and Synthesis

Let's now take the example of relative clauses, where we must relate a relative pronoun with the relative clause's antecedent, but reconstruct this antecedent *at some other point* where it is missing, rather than simply superimpose it with any other substring. For instance, in "This is the paper that Ife submitted", "the paper" is implicit at the position after "submitted" [4] (and its meaning should be identified with that of the missing direct object of "submitted"- more on this, later).

Here again our dome-based model can be used to our advantage. The following rough grammar fragment and the dome shaped figure it elicits (Figure 5) exemplify.

```
    word(the) ::> det(the).
    word(paper) ::> noun(paper).
    word(that) ::> relative pronoun(that).
    word(ife) ::> name(ife).
    word(submitted) ::> verb(submitted).

    name(R) ::> noun_phrase(R).

(2)  noun_phrase(N):(P0,P1), verb(V):(P1,P2)  ::>
     missing_noun_phrase:(P2,P2).

(3)  det(D):(P0,P1), noun(N):(P1,P2), ...,
     missing_noun_phrase([D|N]):(P4,P4) ::>
 P5=P4+1, P6=P5+1|
       det(P4,P5,D), noun(P5,P6,N).
```

The last two rules relate two long distant constituents through analysis plus synthesis: rule (2) synthesizes a new constituent, "missing noun phrase", to indicate a missing, or non-overt, noun phrase right after the verb (since it both starts and ends at P2, which is the end point of the verb); rule (3) synthesizes the implicit string at the point where it belongs (i.e., from point P4 onwards), after analyzing it from its overt position right before the relative pronoun.

[4] Note that while the relative pronoun "that" does indeed represent "the paper", within the relative clause itself what it represents is missing, so either the words or their meaning must be reconstructed from the antecedent.

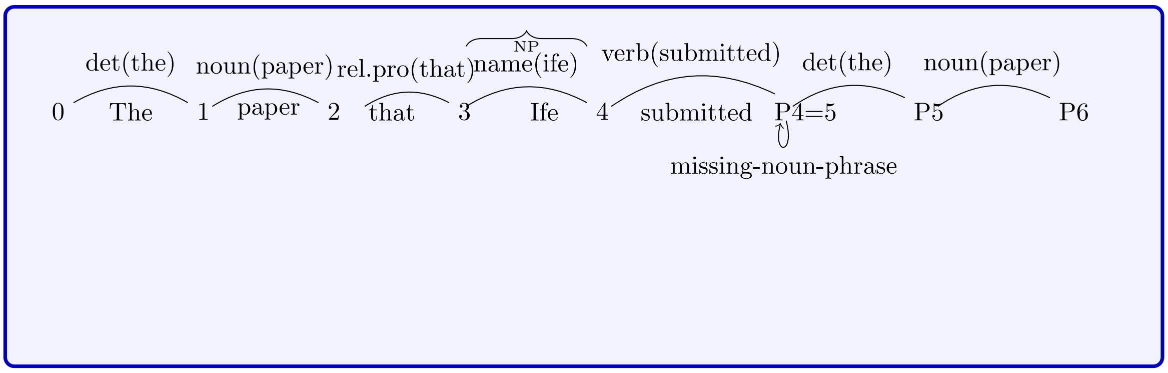

Figure 5: Replicating syntactic shapes for long distance dependencies

5.1.2. Synthesizing Meaning into the Thin Air of Missing Constituents

This first approximation described above does show our methodology in action, but is only concerned with *syntactic form* replication. More interesting is the case where we need to replicate the correct *meaning representation* at the point where the overt string that would give rise to it is missing. The following grammar fragment exemplifies this case. Note that we no longer need to synthesize the surface form of the missing string: we merely synthesize the meaning representation resulting from our analysis of a string, rather than the literal string itself, and fit it into the appropriate place in the overall meaning representation.

```
    word(the) ::> det(the).
    word(paper) ::> noun(paper).
    word(that) ::> relative pro(that).
    word(ife) ::> name(ife).
    word(submitted) ::> verb(X,Y,submitted(X,Y)).

    name(Meaning) ::> noun_phrase(Meaning).

(2')  noun_phrase(X):(P0,P1), verb(X,Y,M):(P1,P2)  ::>
 missing_noun_phrase(Y):(P2,P2).

(3')  det(_):(P0,P1), noun(N):(P1,P2),
      rel_pronoun:(P2,P3),...,
      missing_noun_phrase(N):(P4,P4) ::> true.
```

Rule (2') now synthesizes a place-holder Y for the meaning representation of the missing noun phrase, which is replicated by the verb rule into its appropriate place inside the skeleton representation induced by the verb (in this case, referenced(X,Y)). Rule (3') disregards the determiner's meaning (a simplistic choice) and identifies (unifies) the missing noun phrase's meaning (by calling it also N) with that of the noun itself, namely N (another simplistic choice, just for expla-

$$\text{Mary loves Math while Jill is a painter.}\overbrace{\underbrace{\underbrace{\text{She's}}_{\text{Mary}}}_{\text{Jill}}}^{\text{Pronoun}}\text{ always been good with numbers.}$$

Figure 6: Superimposing candidate referent shapes to aid in disambiguation

nation purposes), so that the final representation obtained by parsing the relative clause will be "submitted(ife,paper)".

Of course, we have left out for presentation and clarity purposes many small technical details, but this implies no loss of generality since they are easy, albeit tedious, to incorportate [5].

5.2. Shape replication for Disambiguation

In cases of ambiguity, given that all possibilities are simultaneously kept in the constraint store, we can readily see what candidates compete with each other, and perhaps use a human expert's opinion interactively in order to disambiguate on the basis of pragmatic or contextual information not available to the system, but apparent to the human. For instance, in "Mary likes Math while Jill is a painter. She's always been good with numbers", a parser would have to have access to pragmatic information from which to perform complex inferences in order to determine that "she" refers to "Mary" rather than to "Jill". However any ordinary human would be able to readily tell who "she" is most likely to reference, simply by looking at the dome-enhanced description, which proposed both candidates (and considering his or her world knowledge, which would tell that being good with numbers is more indicative of liking Math than painting).

6. Related Work

Grammar reversibility (the potential to use largely the same grammar or system for either analysis or synthesis) has been studied in the framework of logic grammars since their inception. Early work focusing on this subject includes [12]. Constraint-based grammars have also received attention from the reversibility point of view, e.g. [10] presents a typed constraint-based system where reversibility is achieved through abstract machines. In our approach we do not need extraneous apparatus, we simply use CHR grammar rules to synthesize whatever terminals or non-terminals we might need to superimpose in our boundary-annotated input string. Admittedly we do not deal with types, but these are not strictly

[5]For instance, rigorously speaking, these rules are an abuse of notation, since CHR does not allow the unification of variables inside constraints already in the constraint store, whereas our two CHR rules unify them freely. Other technicalities would need to be addressed as well in a full solution, e.g. rules (2) and (2') must be made to apply only when there is no overt noun phrase following the verb.

needed for the work proposed here. Combinatorial Categorial Grammars [11] are another attempt to make analysis and synthesis both equally accessible for the processing tool.

Dome shapes as materialized in this paper (i.e., through arcs that span word boundaries) have also appeared earlier in the literature. Relevant examples are Datalog Grammars [7,8] and XSB [13], where arcs are also-at least implicitly-present between the boundaries of phrases, and a chart data structure fills a similar role as the constraint store in CHR grammars. This approach is generally known as chart parsing, which generated a lot of interest in the 1970s (e.g [9], [17]). However the use we give to the arcs in this paper- such as the copying over already found phrases to make it evident for instance what the candidate pronoun antecedents in an ambiguous sentence or discourse might be- has not been given, to the best of our knowledge, in any previous work except [6].

An interesting use of domes was proposed in [16], namely to highlight repeated sections of a string with translucent arcs, which results in a diagram that illustrates the structure of a string. This has been used not only for spoken language strings, but also for musical compositions. Intricate diagrams result from such complex compositions, though, and there does not seem to be a way to manipulate them from the system itself, like we can in CHR rules.

7. Conclusion

We have shown how CHRG constructs can naturally relate to dome-shaped figures along a string to be analyzed, how CHRG rules lend themselves to a mixed (analyzing while synthesizing) mode of processing, and how these two features can be exploited to directly produce visual aids for solving some of the crucial problems facing computational linguists: pronoun reference, long distance dependencies, implicit meanings, disambiguation.

It is interesting to note that the resulting dome shapes can become multidimensional. While in this paper we have restricted ourselves to just "above" and "below" due to the dimensional limitations of the plane constituted by a page, we can imagine domes spreading in multiple directions if we were working in space (and if informationally important, we can use colour to simulate spatial dimensions).

All of these problems we have addressed are non trivial: long-distance dependencies involve, as we have seen, relating constituents that have been displaced from their canonical ordering and are an arbitrary distance apart. While in this paper we use the so-called filler-gap approach (which identifies a gap at an extraction site, and fills the gap with a filler), it is interesting to note that other proposals exist; in particular cognitive-functional alternatives have been proposed in which long-distance dependencies spontaneously emerge as a side effect of how grammatical constructions interact with each other for expressing different conceptualizations [15]. In forthcoming work we shall investigate how a different set of grammatical constructions- constraints in the sense of modular properties between daughter constituents of a phrase [2,4] might likewise allow us to automate the emergence of dependencies organically, from the interaction of these properties. We shall then investigate the possibilities of extending our companion work on shapes [1], which is also property-based, for automating a visual picture of how these dependencies emerge.

With this work we hope to stimulate further research into ways in which visual enhancements of the dome flavour presented here can promote interaction with humans, so that parsers have a straightforward way of becoming less secretive, less black-boxy.

References

[1] Adebara, I., Dahl, V.: Shape analysis as an aid for grammar induction. Submitted to: Shapes.3

[2] Blache, P.: Property grammars: A fully constraint-based theory. In: Proceedings of the First International Conference on Constraint Solving and Language Processing. pp. 1–16. CSLP'04, Springer-Verlag, Berlin, Heidelberg (2005), `http://dx.doi.org/10.1007/11424574_1`

[3] Christiansen, H.: CHR grammars. TPLP 5(4-5), 467–501 (2005)

[4] Dahl, V., Blache, P.: Directly executable constraint based grammars. In: Proc. Journees Francophones de Programmation en Logique avec Contraintes, JFPLC 2004 (2004)

[5] Dahl, V., Maharshak, E.: DNA replication as a model for computational linguistics. In J. Mira et al. (Eds.): IWINAC09 (Best Paper Award) LNCS 5601, 346–355 (2009)

[6] Dahl, V. and Maharshak, E.: Synalysis: A dual processing scheme for both human and biological languages. In: Bio-Inspired Models for Natural and Formal Languages, Cambridge Scholars Publishing, pp. 259, (2011)

[7] Dahl, V., Tarau, P. and Huang, Y-N.: Datalog Grammars. Joint Conference on Declarative Programming, Peniscola, Spain, (1994)

[8] J. Balsa and V. Dahl and J. G. Pereira Lopes.: Datalog Grammars for Abductive Syntactic Error Diagnosis and Repair. IN: PROC. NATURAL LANGUAGE UNDERSTANDING AND LOGIC PROGRAMMING WORKSHOP. pp. 111125, SpringerVerlag, (1995)

[9] Kay, Martin.: The MIND System. In: Randal] Rustin, ed., Natural Language Processing. AIgorithmics Press, New York, New York: pp.155-188. (1973)

[10] Marrafa, P. and Saint-Dizier, P.: A reversible constraint-based logic grammar. Springer Science + Business Media, B.V., (2012)

[11] Steedman M. and Baldridge J. and Borsley, R.D., Borjars, K.(eds.): Combinatory Categorial Grammar. Non-Transformational Syntax. Blackwell Publishing, pp. 181224, (2011)

[12] T. Strzalkowski, ed.: Reversible Grammar in Natural Language Processing Kluwer Academic Dordrecht 1994.

[13] T. Swift and D.S. Warren: XSB: Extending the Power of Prolog using Tabling. Cambridge University Press, (2011)

[14] Thom W. Frühwirth, T.W.: Theory and practice of constraint handling rules. J. Log. Program. 37(1-3), 95–138 (1998)

[15] Van Trijp, R.: Long-distance dependencies without fillergaps: a cognitive-functional alternative in fluid construction grammar. Language and Cognition 6, 242–270 (6 2014), `http://journals.cambridge.org/article_S1866980814000088`

[16] Wattenberg, M.: Arc diagrams: Visualizing structure in strings. In: Proceedings of the IEEE Symposium on Information Visualization. pp. 110–116 (2002)

[17] Wiren, Mats.: Interactive Incremental Chart Parsing. In:Proceedings of the Fourth Conference on European Chapter of the Association for Computational Linguistics, EACL 89, Manchester, England, pp 241248, http://dx.doi.org/10.3115/976815.976848, 10.3115/976815.976848, Association for Computational Linguistics, Stroudsburg, PA, USA (1989)