Deterministic Machine – report on the deterministic parsing module of the Grammar Checker

Jan Hric Alexandr Rosen Markéta Straňáková*

January 6, 1995

1 Introduction

There are some constructions in natural language, which are frequent, typical and relatively unambiguous. The 'Deterministic Machine' is supposed to handle such constructions. Motivations for including a preprocessing module of this type – the Deterministic Machine (henceforth DM) – as a component of the grammar checker are of the following two kinds:

- Decrease in computational costs (esp. time) is expected if DM succeeds in predicting correct analyses of some frequently occurring and locally identifiable grammar phenomena in a significant share of the input.
- The distinction between the processing of local and sentential contexts, which is inherent in the two-component design of the system, enables testing of linguistic hypotheses concerning locality of various grammar phenomena.

The former consideration has influenced one of the fundamental aspects of the DM design: only left modifications of heads are processed.

To ensure full compatibility of data (lexicon, grammar, input strings and output structures), the DM is an integral part of the whole system and shares as many functions as possible with the non-deterministic part.

---

*These are the names of authors who contributed directly to this part of the report. However, the authors also wish to thank other people whose contribution was indirect but all the more substantial, most notably Petr Jäger, Eva Hajičová and Karel Oliva. This does not mean that they are to be blamed for any defects.

2 Basic features of the deterministic machine

- DM uses the same lexicon as the non-deterministic module and a subset of its grammar. The form of lexical entries and grammar rules is identical. In fact, there is a single grammar and lexicon; grammar rules which describe phenomena to be covered by DM are identified as such in the grammar.

- Grammar rules used by DM should cover relatively frequent, locally detectable phenomena. Furthermore, the grammar rules are restricted to head-final structures, i.e., right modifiers (complements or adjuncts) are excluded. Typical examples consist of head-final adverbial, adjectival and nominal phrases (possibly including prepositional phrases). Simple cases of coordination are other likely candidates. Provisions are made for parsing right modifiers of such heads in the non-deterministic module. The rules can be ordered into classes, processed successively in more passes.

- The output of DM is written directly into the chart of the non-deterministic module as a string of partial parses. If the non-deterministic module fails to confirm the hypotheses made by DM, processing backtracks to the original input, which is now parsed by the non-deterministic module directly.

- DM accepts as input strings sentences processed by lexical lookup component, admitting lexical ambiguities. These ambiguities can be resolved in DM and no more ambiguities should be introduced by the rules.

- DM processes the input string, a sequence of edges, from left to right. After all possible rules are applied on all possible sequences of edges in a certain stretch of the string, delimited by the string beginning and the edge most recently read from the input, edges used by the rules are eliminated, and, subsequently, edges which do not form a path within that stretch. A new edge from the input string (to the right) is then read.

- DM performs only (partial) analysis of the input. No error checking as such is done here.

3 Linguistic issues

3.1 The extent and organization of grammar

The question of which grammar phenomena should be covered by DM is an empirical one: testing on large text samples with different grammar-rule setups is previewed. As the first approximation of potential candidates, the following phenomena have been selected (expressed in a quasi-rule notation, bar levels ignored, heads in capital letters):

1. adverb → adverb + ADVERB
2. coordination of adverbs[1]
3. adjective → adverb + ADJECTIVE
4. coordination of adjectives
5. noun → adjective + NOUN
6. coordination of nouns
7. 'prepositional phrase'[2] → preposition + NOUN
8. coordination of 'prepositional phrases'

The order of rules might correspond to the sequence of passes, or more rules can be processed in a single pass.

3.2 Preliminary testing of linguistic adequacy

3.2.1 The data

The function of DM was simulated by hand on samples taken from several texts – mainly newspaper texts (*Lidové noviny, MF Dnes*, the general interest magazine *Elán*) and texts from the literary and art magazine *Revolver Revue*. Eighty sentences have been selected, the average number of words in a sentence being 16.

3.2.2 The choice and the effect of grammar rules

At first only LEFT MODIFIERS represented by the following four rules were processed (heads in capitals; bar levels ignored; agreement beween noun and its modifying adjective is respected).

(i) Adv → Adv ADV

(ii) Adj → Adv ADJ

(iii) Noun → Adj NOUN

(iv) PP → Prep NOUN

---

[1] Same as all other structures, coordination is treated by binary rules, i.e., two rule applications are necessary to cover the coordination of two conjuncts.

[2] More precisely, *noun in prepositional case.*

ad (i) The rule (i) was applied six times correctly and six times incorrectly. The main problem was that the rule was incorrectly applied on pairs of adverbs which did not form a phrase[3].

(1) a. Premiér poté podrobně vysvětlil ...
Premier then in_detail explained ...

b. Pravidla českého pravopisu, dále jen Pravidla, ...
Rules [of] Czech orthography, below only Rules, ...

c. A možná také proto zůstaly nepovšimnuty.
And maybe also therefore remained unnoticed.

ad (ii) The second rule was applied 19 times correctly and 9 times incorrectly. There was one type of misapplication: again, the rule applied on adverbs which did not modify the head[4].

(2) a. jinde staré podlahy podpírají kůly
elsewhere old floors are_supported by_piles

b. Zákon by neovlivňoval negativně vytváření rodiny
the_law would not_influence negatively [the] creation/created of_family

Sometimes the cause was ambiguity between Prep and Adv[5].

(3) a. Podle jejich slov ...
According_to/vilely their words ...

b. kampaň proti jaderné elektrárně
campaign against [the] nuclear power_plant

ad (iii) The third rule was applied 165 times correctly. Except for the problems concerning right modifiers (which will not be handled by DM), parallel edges (see below) and coordination (see below) only one error occurred.

(4) chránit bezpečnostní zóny vyhlášené OSN
to_protect [the] safety zones proclaimed [by the] UN

[3]Also a problem with relative adverbs appeared – the rule was incorrectly applied on relative adverbs followed by a common adverb, but this was just a matter of too loosely contrained grammar used for this simulation, (e.g. *V lokalitách, kde již bylo poškozeno* ... 'in localities where already has_been damaged ...')

[4]Similarly as above, relative adverbs were incorrectly interpreted as modifiers of a following adjective *Případů, kdy zaběhlý pes napadne* ... 'Cases when [a] stray dog attacks'

[5]The ambiguity of *proti* 'against' is an arguable case and should probably be solved by eliminating the adverbial variant.

There were relatively many cases (13) where a noun was adjacent to an adjective which was not its modifier. Incongruent agreement features mostly prohibited incorrect application of the rule, but not always – the only incorrect application of this rule was of this type.

(5) a. rozhodnutí ministerstva ...
decision/decided-masc;pl:nom ministry-neut;sg:gen/pl:nom,acc,voc ...

b. rozšíření těžby vápence
extension/extended-masc,pl:nom production-fem; sg:gen/pl:nom,acc,voc of_limestone

ad (iv) The last rule was applied 90 times correctly and 22 times incorrectly. The problems concerned right modifiers again – 18 cases.

(6) a. ve většině případů ...
in [the] majority [of] cases ...

b. protestoval proti útoku Muslimů
protested-3sg against [the] attack [by the] Muslims

Such misinterpretations will be handled by the non-deterministic module in a routine fashion[6]. Other problems concerned parallel lexical structures (esp. homonymy of demonstrative pronouns, see below) and coordination (see below).

3.2.3 Ambiguities

The string of lexical representations of the words of an input sentence may contain parallel edges. In some cases DM accepts one reading, in other cases it preserves non-determinism from the input.

The proposed strategy was successful in 53 cases and failed in 20 other cases. Many of these errors (12) concerned the rule (iv) PP → Prep NOUN, mainly in connection with the ambiguity between Adj and N (8 occurences, esp. demonstrative pronouns – these may be removed by a modified classification of demonstrative pronouns, see bellow).

Example:

[6] All structures parsed by DM enter the non-deterministic module with the attribute 'constituent_type' set to 'lexical', which makes it possible to parse such 'right modifications of PPs"

Z tohoto důvodu ... ‘for this reason’

```
  Z   tohoto   duvodu  ...
        _Adj_
 Prep  /     \     N
*----*<       >*-------*
 \     \__N__//
  \___PP_____/
```

The fourth rule was applied on the edges Prep and N, correct reading was omitted.

This ambiguity caused also another type of errors:

Example:
Jsou tou svatou chvílí , kdy ... ‘[they] are that sacred/saint moment, when’

```
  Jsou  tou   svatou   chvili ,  kdy  ...
        _N_     Adj
  V    /   \   /   \     N
*----*<     >*<     >*-------*
      \\Adj/   \_N_//
       \_____N_____/
```

Two readings remained (N + Adj, N [Adj + N]), both incorrect.

Ambiguity between Prep and Adv may cause errors, too.

Example:
Podle jejich slov ... ‘according_to/vilely their words’

```
  Podle   jejich   slov  ...
   Prep
  /    \   Adj      N
*<      >*------*-------*
 \\Adv_/       /
  \_____Adj___/
```

Other interesting examples:

(Stěny a klenby) domů ničí voda. ‘Walls and vaults of_houses/home[ward] is_destroying/nobody’s water-nom.’

```
(Steny   a   klenby)   domu   nici   voda .
                       _N_     _V_
   N    Conj   N      /   \   /   \    N
*-----*-----*-------*<     >*<     >*-----*
                    \\\Adv/   \Adj//      /
                     \\____Adj____/      /
                      \________N________/
```

The rules (ii) and (iii) were applied, edges with correct reading (N and V) were deleted[7] as a cul-de-sac.[8]

Toto postižení lidé chápou … ‘This-neut,sg afflicted-masc.anim.,pl/affliction-neut,sg people-masc.anim.,pl understand.’

```
   Toto   postizeni   lide   chapou   ...
   _____N______
  /ADJ     __N__\
 //   \   /     \\     N       V
*<     >*<       >*-------*--------*
  \_N_/  \\_Adj_/         /
          \______N_______/
```

The rule (iii) was applied twice, then the first NP edge was deleted (as a cul-de-sac).

There are two readings of this sentence:

1. *(Toto postižení) lidé chápou* … (discarded by DM).
2. *Toto (postižení lidé) chápou* … (accepted by DM).

3.2.4 Coordination

Another type of linguistic phenomenon considered as a potential candidate to be treated by DM is coordination of nouns (NP’s) and adjectives (AdjP’s).

[7]However, the Adv Adj reading is very weird, which is again due to the loosely constrained specification of the grammar used for simulation. The word *ničí* in the adjectival reading is actually possessive pronoun, which can be modified only by a very restricted class of adverbs like *opravdu* ‘really’.

[8]Edges are in cul-de-sac if they do not connect both ends of the graph, i.e. if they do not represent a path from the leftmost to the rightmost node.

Problems occur in coordination of the type Prep (N + N) (e.g. *Slovník spisovné češtiny pro školu a veřejnost* 'Dictionary of literary Czech for school and [the] public'; *Při setkání s ministrem a předsedou řekl ...)* 'during [the] meeting with [the] minister and [the] chairman [he] said'.

Example:
pro školu a veřejnost 'for school and public'

```
   pro   skolu   a   verejnost

   Prep    N    Conj     N
*-----*-------*---*-----------*
 \_____PP____/ \   \_?Ncoor__//
                 \___?Ncoor'__/
```

DM applies the rule (iv) on the Prep and first N, this N is deleted and then the two members of coordination cannot be put together.

As for other type of errors – DM can connect lexical items which in fact do not belong together.

Example:
Viděl Marii a Kláru přehlédl. '[He] saw Marie-acc and Klára-acc [he] overlooked.'

```
   Videl   Marii   a   Klaru   prehledl.

     V      N    Conj     N        V
*-------*------*----*---------*---------*
         \      \    \?Ncoor///
          \      \__?Ncoor'_//
           \______N_________/
```

DM applies coordination rules on two N's and Conj incorrectly. In fact this sentence is composed by two clauses, not by one clause with coordination of N's.

The following interesting example illustrates the action of DM:

Za hlavní považoval zdraví a postižení lidé k němu nesměli.

‘As the_main_thing [he] considered health / healthy and affliction / afflicted people to him were_not_allowed.’

```
Za  hlavni povazoval zdravi a postizeni lide k nemu nesmeli.
                     ______N________
                    /_N_          _N_\
Prep  Adj      V   //   \  Conj /   \\  N   Prep N    V
*--*------*-------*<     >*---*<     >*----*---*---*-------*
                  \\\Adj/       \Adj//    /
                   \\______Adj______/    /
                    \__________N________/
```

First, both coordination rules for N’s and Adj’s are applied. The rule (iii) is used on the resulting Adj and the following N edges with an NP edge as an output. In the next step the edge corresponding to the coordination of N’s is deleted as a cul-de-sac. Thus N modified by coordination of Adjs remains as the valid reading. But the correct reading has to interpret *zdraví* as N and *nemocní* as Adj.

A small modification of the sentence (adding the left modifier své – reflexive possessive – to the N *zdraví* ‘health’) leads to a completely different structure:

Za hlavní považoval své zdraví a postižení lidé k němu nesměli.

```
Za hlavni povazoval sve zdravi a postizeni lide k nemu
                     _________N________           nesmeli.
                    /__N____           \
                   //    _N_\        _N \
Prep  Adj    V    //Adj /   \  Conj /   \\  N   Prep N   V
*--*-----*-------*---*<     >*---*<     >*----*---*---*---*
                        \Adj/       \Adj/
```

Now, the rule (iii) is applied on Adj *své* and N *zdraví*, the edge with Adj for *zdraví* is deleted as a cul-de-sac and the coordination rules for Ns are used (the edge with Adj for *postižení* is deleted). Thus the coordination of NP [své zdraví] and N [postižení] remains as the valid reading. However, the correct reading is discarded as in the previous sentence.

3.2.5 Conclusions from preliminary testing

Evaluation of the results has helped to tune the DM parsing algorithm, to form an estimate of linguistic phenomena and contexts which DM can handle and to make a preliminary assessment concerning the choice of grammar rules and their processing order (if any).

Suggestions:

(a) Some semantic constraints on adverbs in the rules (i) and (ii) would be useful.

(b) Demonstrative pronouns: It seems to be better (for the purposes of DM and perhaps also for the non-deterministic module) to classify demonstrative pronouns only as Adj (syntactic class), not both as Adj and N. This would remove some cases of parallel edges and, consequently, some problems mainly with the application of the rule (iv). This classification is corroborated by the fact that such situation is relatively frequent when Adj occurs without N (ellipsis) and demonstrative pronouns seem to behave similarly as these Adj's.

    Example:

    Líbí se mi žlutá barva, modrou ráda nemám.

    Líbí se mi tahle barva, tamtu ráda nemám.

(c) PP $\rightarrow$ Prep Noun

    The question is whether prepositional phrase is a suitable linguistic phenomenon to be treated by DM. The problems concerning the rule (iv) are connected with right modifiersth parallel edges and mainly with coordination. Testing on more material is needed to substantiate a decision.

(d) Coordination – similarly as with prepositional phrases, testing on more data is needed.

4 Description of the algorithm and its implementation

4.1 An informal description of the algorithm

In the following description the term "head" will be used for an edge which can 'fold' other edges during parsing of a sentence, provided that a grammar rule licenses the action. I.e., the other edges must be labelled by categories interpretable as being governed by the head.

DM starts at the leftmost edge, proceeds to the right and performs the following operations on each input string:

If a current edge has subordinate edges at the left-hand side, it folds the longest chain of subordinated edges, i.e. it creates a new edge, which will represent the whole folded phrase. If there are several longest chains, it creates more parallel edges. Then it prunes all used edges and all edges which do not belong to some continuous path from the start node to the end node. Processing then continues using the next right-hand edge.

4.2 Interface to the non-deterministic module

The whole processing of a single sentence is performed in three stages:

1. DM processes an input string (a sentence), possibly producing parses of its substrings as the output.
2. The non-deterministic module processes the output of DM. If the whole sentence is parsed successfully, then the non-deterministic module writes out a message "No errors found" and stops; else processing continues with the next stage.
3. The non-deterministic module processes the original input. If DM has created some edge(s) which are not justified in the broader (sentential) context, then – in the current design of the system – the non-deterministic module cannot undo such edges in the stage 2. Therefore it backtracks and attempts to parse the original input. At this stage, it also looks for errors.

4.3 Problems and solutions

4.3.1 Phases of processing

DM need not perform all its computation during one traversal through the input edges, but it can traverse the input string more times. It can use different rules for different linguistic phenomena in successive phases. The input of DM is the input into the first phase and each subsequent phase takes as its input the output of the previous phase. The output of the last phase is the output of the whole DM.

The implementation enables the user to define phases and correspondence of the rules of grammar to the phases.

DM produces correct parses if all left modifiers of a head are identified before the head itself is interpreted as a left modifier of another head.

Example:

Input: A + B + C

Rules: 1. B → A + B[9]

2. C → B + C

The rule (1) must be used before the rule (2) in this example to get a successful parse, i.e. (1) must be used in the same or in a preceding phase with respect to (2). Else, after application of the rule (2) we get "A + C", and then it is not possible to interpret the edge A as a modifier of B (without announcing an error). In this case the parsing of the sentence enters the stage 3 according to the description of the whole algorithm.

We can look at the whole computation of the non-deterministic module as a phase which follows all phases of DM. If we want to gain full advantage of DM and to exclude similar cases as in the preceding example then the deterministic module must

1. recognize and fold all edges to the left of a head which are interpretable as modifiers of that head;
2. have rules which can be recursively applied in the same phase.

The need to invoke the non-deterministic module on the original input sentence will depend on the quality of the output from DM, i.e. on the quality of decisions which phenomena (and corresponding rules) the deterministic module will work with. If the decisions will be wrong then some unnecessary work will be done during the processing by DM (stage 1) and the following processing of the misinterpreted input by the non-deterministic module (stage 2), but the behavior of the whole system will not degrade into a failure. The reason is that the final stage 3 remedies all wrong decisions of DM.

4.4 The treatment of structures not covered by a single application of a binary rule

Both modules, deterministic and non-deterministic, will have the same grammar. The non-deterministic module does not allow rules which have more than two items on the right-hand side (i.e., it accepts only 'unary' and binary rules). But some structures including those considered as candidates for the treatment by DM consist of more than two items. If the construction of new edges were irreversible then DM would fail to parse the following strings correctly:

(7) a. velký + a + malý + pes
big-adj + and-conj + small-adj + dog-n

b. student + a + malý + pes
student-n + and-conj + small-adj + dog-n

[9]Bar levels are ignored.

The edge ‘malý’ can be extended to the left with edge ‘a’ in both cases. But in (7b) there is no other edge with Adj before the edge ‘a’, and DM is not able to complete the whole structure Adj + Conj + Adj. The edge corresponding to the unfinished construction ‘a malý’ remains and it does not allow to parse the whole sentence by the non-deterministic module correctly. In the following a solution is decribed which blocks the creation of such incomplete edges.

Newly created edges are distinguished as ‘complete’ or ‘incomplete’. An edge is marked as ‘complete’ if it represents a complete linguistic structure. Otherwise it is marked as ‘incomplete’. An ‘incomplete’ edge corresponds to a part of some structure, which consists of at least three elements, where not all of these elements have so far been parsed. This type of edge has no lexical meaning and the corresponding string cannot occur in a sentence alone. All ‘incomplete’ edges are later pruned if they cannot be extended, i.e. they are never passed over to the following phase. The creation and pruning of edges allows some kind of nondeterminism, because it is possible to try to extend the actual head – the right-most read word – to the left side, but if the extension is impossible to complete, thus wrong, the algorithm is able to recover used edges in the following manner.

Each edge has the status ‘free’, ‘restorable’, ‘used’ or ‘pruned’. An edge is initially ‘free’, which means that it was not used for the construction of a new edge. The edge changes its status to ‘used’ if it was used for the construction of a ‘complete’ new edge. If it was used for the construction of some new edges, but all of them were ‘incomplete’, then it acquires the status ‘restorable’. If an ‘incomplete’ edge is used for the creation of a ‘complete’ one, then this information is propagated recursively to all ‘restorable’ edges, whose status is changed to ‘used’. As soon as all possible left modifiers of the currently read head are found (the head is projected as far to the left as possible), all incomplete edges are pruned – their status changes to ‘pruned’ – as well as all used edges. The ‘restorable’ edges are changed to ‘free’, because they were used only to construct an ‘incomplete’ edge.

4.5 Pseudocode of the deterministic module

Input:

nodes 1..N

edges, marked as ‘free’

Used internal structures:

- Each newly created edge can be marked in one of the following ways according to its completeness:

'complete' – can occur as a part of the output;
'incomplete' – must be processed in deterministic module

- Each edge has a status from the following list:

'free' – not used yet, also initial value
'restorable' – used to create only 'incomplete' offsprings
'used' – used to create at least one 'complete' offspring
'pruned' – pruned, cannot be used in following phases

Algorithm:

```
1. for all phases do
 2. for all nodes J := 3..N do  // read next word
  3. for all nodes I := J-2..1 do // looking for left expansions
   4. for all heads A from K to J, I<K<J do
    5. for all edges B from I to J, which are not pruned, do
     6. for all rules X -> Y + Z
      7. if Y and Z respectively, match with A and B
           respectively then
           create the new edge X from I to J. Mark the new edge
           as 'free' and 'complete' or 'incomplete' according
           to the type of the rule.
       9. if X is 'complete' then mark A and B as 'used',
           otherwise mark them as 'restorable'. If A or B
           should be marked as 'used' and simultaneously as
           'restorable', then they are marked as 'used'.
       10. if X is 'incomplete', store the ancestor edges
           A and B, which X was created from.
       11. if A or B were 'incomplete'
           edges, which changed its status from 'free' or
           'restorable' to 'used', then change for
           all ancestors (recursively for all levels)
           stored in step 10 their status to 'used'.
      end if (7)
     end for (6)
    end for (5)
   end for (4)
  end for (3)
  12. // pruning
   13. all 'incomplete' edges change to 'pruned'
```

```
   14. all ‘restorable’ edges mark as ‘free’
   15. all ‘used’ edges mark as ‘pruned’
   16. all edges, which do not belong to any continuous path
     from node 1 to node N, mark as ‘pruned’
  end (12) // pruning
 end for (2)
end for (1)
```

Output:

All edges, which are not ‘pruned’. (All of them are ‘free’).