Syntactic spreadsheets. In search for a human-readable representation of parse tree forests

Janusz S. Bień

Formal Linguistics Department, the University of Warsaw
Browarna 8/10, 00-311 Warszawa, Poland
`jsbien@uw.edu.pl`

Abstract. The paper presents an example of a representation of parse tree forests advocated by the author. Although motivated by the need to analyse the forests generated by Świdziński's grammar, the representation can be used for any grammar handled by Woliński's Birnam parser, and the basic ideas can be applied to any Immediate Constituent grammar. Syntactic spreadsheets can serve several purposes. They can be simply included in printed publications or dynamically displayed by an appropriate viewer. Unfortunately the implementation of the idea is not easy and therefore it is still in progress.

Key words: parsing, parse trees, tree forests, visualization, Birnam parser

1 Introduction

At present computationally useful syntactic description of Polish is limited to the surface level. Ambiguity is an intrinsic feature of surface grammars, so parse tree forests need to be handled in a convenient way. An idea to use 'syntactic spreadsheets' for this purpose has been proposed by the present author in a paper in Polish [2]. Similar diagrams have been used to represent single trees at least since Charles Hockett's *A Course in Modern Linguistics*, first published in 1964 (cf., for example, the Polish translation [5, pp. 175-184]), so the main novelty of the proposal lies in applying them to whole forests. To the best of my knowledge, existing tools with similar purpose operate only on single trees. An example of such a tool is the *Linguistic User Interface* (`http://wiki.delph-in.net/moin/LkbLui`) developed in the DELPH-IN project. Although the problem is recognised

> Grammars often produce numerous tree structures for any input parsing or generation request.

the user is just offered many windows with a single tree in each of them (`http://wiki.delph-in.net/moin/LuiTree`).

The ultimate goal is to create a forest browser which will allow to dynamically change the arrangement and granularity (the amount and kind of details) of the display. It should also allow to dynamically highlight interesting parts of the forest. As the forest doesn't need to be complete, such a browser can be used also as a debugging tool for new grammars.

For the time being, however, the goal is much more modest: to create tools for including syntactic spreadsheets in research publications typeset with the current version of LaTeX, i.e. PDFeLaTeX 2_ε and XeLaTeX. The tools should allow the user to introduce — by hand, if necessary — corrections, additions and modifications. As a side effect, electronic spreadsheets with hyperlinks can be created without the size limits of paper.

2 Preliminaries

As the primary motivation is the need to analyse forests generated by Świgra (*ŚWIdzińskiego GRAmatyka*), which is the implementation [11] of Świdziński's formal grammar of Polish [9], we will illustrate the idea of syntactic spreadsheets with a Świgra forest.

We will use the notable example designed by Marcin Woliński and discussed e.g. in [10, p. 40], as it demonstrates in particular the ambiguity of input segmentation:

(1) *Miałem miał.*

The example sentence is assigned with 4 parse trees. Its primary reading is

(2) *I had [some] coal dust.*

The interesting thing is that two perfectly legal parse trees are assigned to this reading. The trees differ only in the segmentation of words into morphological units:

(3) *Miałem miał.*
I had [some] coal dust.

(4) *Miał + em miał.*
[some] coal dust I had.

There is also a second elliptic reading which results in the third parse tree:

(5) *Miałem miał.*
[some] coal dust$_{Instr}$ *he had.*
[With some] coal dust he had.

The sentence is quite correct as an answer to an appropriate question, e.g.

(6) *Czym miał posypaną podłogę?*
What had he covered his floor with?

The fourth parse tree is just an artifact of Świdziński's grammar.

The current way of presenting the results of Świgra employs hyperlinked PDF documents described in [13]; this is just a modification of the tree representation designed by Woliński over 10 years ago for the AMOS parser [1]. As the full versions of trees use a lot of space but contain too many uninteresting details, by 1999 the compact form was introduced for use with the AS parser [3]. In the compact form every path without branches is represented as a dotted line and the intermediate nodes are omitted, while real arcs are represented by continuous lines.

To make the paper self-contained, we present in Figure 2 the compact form of all the trees for the sentence under consideration, and in Figure 1 the full form of one of them. More examples of compact trees can be found at Woliński's site (`http://nlp.ipipan.waw.pl/~wolinski`), while both full and compact trees are provided at `http://fleksem.klf.uw.edu.pl/~jsbien/synspread/`; as the address suggests, in due time the trees will be supplemented or replaced by the respective syntactic spreadsheets (depending on the availability of the server disk storage, trees and spreadsheets will be provided either as PDF files or their LaTeX source).

3 The basic structure of syntactic spreadsheets

When using trees, we have to choose between full and compact forms. A spreadsheet however can contain various types of cells and, if needed, it can contain data present in both forms of trees. Moreover, while modifying a grammar, for easy comparison we can mix parse forests from several grammars in a single spreadsheet.

The spreadsheet is obviously a table. The number of columns is the length of the longest path, measured in some segmentation units, from the beginning to the end of the sentence. In the full version of the spreadsheet the segmentation units are those of the morphological analyzer. As demonstrated by our example, morphological segmentation in Polish can be ambiguous.

In our sample spreadsheet on Figure 3 there are three kinds of cells:

1. auxiliary,
2. terminal,
3. main (nonterminal).

Auxiliary nodes are used in the sample only for row identifiers (T.1, M.1 etc.), but can be used also to provide headers and footers with column identification (by numbers or by appropriate substrings of the input).

The purpose of the terminal cells is obvious, as well as their primary content: form, lemma, tags.

All other cells in the sample are the cells of main nonterminal nodes; by main nonterminal node we understand the nodes which are present in Woliński's compact form of the trees. In general, the spreadsheet can contain all nonterminal nodes, instead of main nodes, or in addition to them.

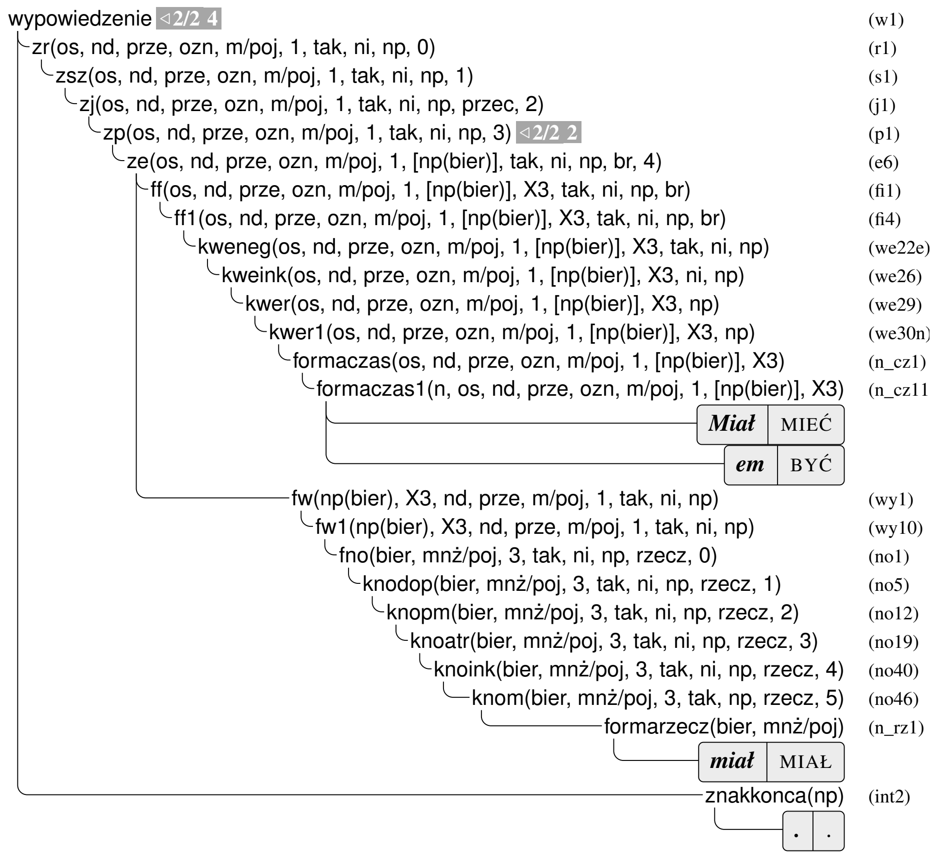
wypowiedzenie ◁2/2 4 (w1)
zr(os, nd, prze, ozn, m/poj, 1, tak, ni, np, 0) (r1)
zsz(os, nd, prze, ozn, m/poj, 1, tak, ni, np, 1) (s1)
zj(os, nd, prze, ozn, m/poj, 1, tak, ni, np, przec, 2) (j1)
zp(os, nd, prze, ozn, m/poj, 1, tak, ni, np, 3) ◁2/2 2 (p1)
ze(os, nd, prze, ozn, m/poj, 1, [np(bier)], tak, ni, np, br, 4) (e6)
ff(os, nd, prze, ozn, m/poj, 1, [np(bier)], X3, tak, ni, np, br) (fi1)
ff1(os, nd, prze, ozn, m/poj, 1, [np(bier)], X3, tak, ni, np, br) (fi4)
kweneg(os, nd, prze, ozn, m/poj, 1, [np(bier)], X3, tak, ni, np) (we22e)
kweink(os, nd, prze, ozn, m/poj, 1, [np(bier)], X3, ni, np) (we26)
kwer(os, nd, prze, ozn, m/poj, 1, [np(bier)], X3, np) (we29)
kwer1(os, nd, prze, ozn, m/poj, 1, [np(bier)], X3, np) (we30n)
formaczas(os, nd, prze, ozn, m/poj, 1, [np(bier)], X3) (n_cz1)
formaczas1(n, os, nd, prze, ozn, m/poj, 1, [np(bier)], X3) (n_cz11
Miał MIEĆ
em BYĆ
fw(np(bier), X3, nd, prze, m/poj, 1, tak, ni, np) (wy1)
fw1(np(bier), X3, nd, prze, m/poj, 1, tak, ni, np) (wy10)
fno(bier, mnż/poj, 3, tak, ni, np, rzecz, 0) (no1)
knodop(bier, mnż/poj, 3, tak, ni, np, rzecz, 1) (no5)
knopm(bier, mnż/poj, 3, tak, ni, np, rzecz, 2) (no12)
knoatr(bier, mnż/poj, 3, tak, ni, np, rzecz, 3) (no19)
knoink(bier, mnż/poj, 3, tak, ni, np, rzecz, 4) (no40)
knom(bier, mnż/poj, 3, tak, np, rzecz, 5) (no46)
formarzecz(bier, mnż/poj) (n_rz1)
miał MIAŁ
znakkonca(np) (int2)
. .

Fig. 1. One of 4 parsing trees in full form

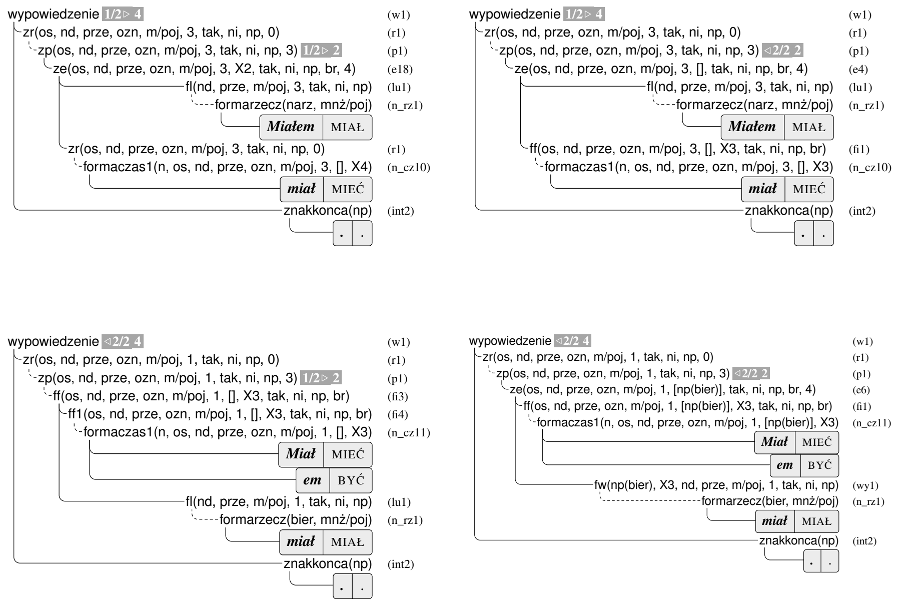
wypowiedzenie 1/2▷ 4 (w1)
zr(os, nd, prze, ozn, m/poj, 3, tak, ni, np, 0) (r1)
zp(os, nd, prze, ozn, m/poj, 3, tak, ni, np, 3) 1/2▷ 2 (p1)
ze(os, nd, prze, ozn, m/poj, 3, X2, tak, ni, np, br, 4) (e18)
fl(nd, prze, m/poj, 3, tak, ni, np) (lu1)
formarzecz(narz, mnż/poj) (n_rz1)
Miałem MIAŁ
zr(os, nd, prze, ozn, m/poj, 3, tak, ni, np, 0) (r1)
formaczas1(n, os, nd, prze, ozn, m/poj, 3, [], X4) (n_cz10)
miał MIEĆ
znakkonca(np) (int2)
. .
wypowiedzenie 1/2▷ 4 (w1)
zr(os, nd, prze, ozn, m/poj, 3, tak, ni, np, 0) (r1)
zp(os, nd, prze, ozn, m/poj, 3, tak, ni, np, 3) ◁2/2 2 (p1)
ze(os, nd, prze, ozn, m/poj, 3, [], tak, ni, np, br, 4) (e4)
fl(nd, prze, m/poj, 3, tak, ni, np) (lu1)
formarzecz(narz, mnż/poj) (n_rz1)
Miałem MIAŁ
ff(os, nd, prze, ozn, m/poj, 3, [], X3, tak, ni, np, br) (fi1)
formaczas1(n, os, nd, prze, ozn, m/poj, 3, [], X3) (n_cz10)
miał MIEĆ
znakkonca(np) (int2)
. .
wypowiedzenie ◁2/2 4 (w1)
zr(os, nd, prze, ozn, m/poj, 1, tak, ni, np, 0) (r1)
zp(os, nd, prze, ozn, m/poj, 1, tak, ni, np, 3) 1/2▷ 2 (p1)
ff(os, nd, prze, ozn, m/poj, 1, [], X3, tak, ni, np, br) (fi3)
ff1(os, nd, prze, ozn, m/poj, 1, [], X3, tak, ni, np, br) (fi4)
formaczas1(n, os, nd, prze, ozn, m/poj, 1, [], X3) (n_cz11)
Miał MIEĆ
em BYĆ
fl(nd, prze, m/poj, 1, tak, ni, np) (lu1)
formarzecz(bier, mnż/poj) (n_rz1)
miał MIAŁ
znakkonca(np) (int2)
. .
wypowiedzenie ◁2/2 4 (w1)
zr(os, nd, prze, ozn, m/poj, 1, tak, ni, np, 0) (r1)
zp(os, nd, prze, ozn, m/poj, 1, tak, ni, np, 3) ◁2/2 2 (p1)
ze(os, nd, prze, ozn, m/poj, 1, [np(bier)], tak, ni, np, br, 4) (e6)
ff(os, nd, prze, ozn, m/poj, 1, [np(bier)], X3, tak, ni, np, br) (fi1)
formaczas1(n, os, nd, prze, ozn, m/poj, 1, [np(bier)], X3) (n_cz11)
Miał MIEĆ
em BYĆ
fw(np(bier), X3, nd, prze, m/poj, 1, tak, ni, np) (wy1)
formarzecz(bier, mnż/poj) (n_rz1)
miał MIAŁ
znakkonca(np) (int2)
. .

Fig. 2. All 4 parsing trees in compact form

The top row of every non-auxiliary cell contains tree information: the cell identifier (e.g., T-1 or M-1), the number of trees in which the cell node occurs, and the total number of trees in the forest (redundant but convenient).

The crucial parts of the nonterminal cell are the component subrows. In the sample they contain in turn just 2 subsubrows: the component list and the list of relevant trees.

The component subsubrow may consist of a single (hyper)link to the appropriate cell, as in, e.g., M-1. In general, it consists of a list of (hyper)links to the appropriate cells, as in, e.g., M-3 and M-13. To save space, whenever possible such rows are collapsed into one. This is exemplified in rows M.11a and M.11b — each of them is to be interpreted as two subsubrows. Hence M-13 and M-14 are separate alternative components of M.11a.

At present the second subsubrow of the components subrow is just a list of the numbers of trees in which the nodes in question occur. It is planned that in the electronic version of the spreadsheet the numbers will be hyperlinks to the trees in Woliński's format (kept in separate files).

The components rows account for links downwards in the trees and the spreadsheet table. If needed, upwards links can be also provided. Upwards links can be provided also for terminal cells.

In the general case, the node label is actually a quite complicated Prolog term. In our sample the labels are represented only by their main functors. In

T.1	T-1 2/4 ***miał*** MIEĆ: praet sg:[m1, m2|m3]:imperf	T-2 2/4 ***em*** BYĆ: aglt sg:pri:imperf:wok	T-3 2/4 ***miał*** MIAŁ: subst sg:[nom|acc]:m3	T-4 4/4 **.** .: interp	T.1
T.2	T-5 2/4 ***miałem*** MIAŁ: subst sg:inst:m3		T-6 2/4 ***miał*** MIEĆ: praet sg:[m1, m2|m3]:imperf		T.2
M.1	M-1 2/4 ⇑ T-5 ⇑ *trees*: 1, 2 ***formarzecz***		M-2 2/4 ⇑ T-6 ⇑ *trees*: 1, 2 ***formaczas1***		M.1
M.2	M-3 2/4 ⇑ T-1 T-2 ⇑ *trees*: 3, 4 ***formaczas1***		M-4 2/4 ⇑ T-3 ⇑ *trees*: 3, 4 ***formarzecz***	M-5 4/4 ⇑ T-4 ⇑ *trees*: 1, 2, 3, 4 ***znakkonca***	M.2
M.3	M-6 2/4 ⇑ M-1 ⇑ *trees*: 1, 2 ***fl***		M-7 1/4 ⇑ M-2 ⇑ *trees*: 1 ***zr***		M.3
M.4			M-8 1/4 ⇑ M-2 ⇑ *trees*: 2 ***ff***		M.4
M.5	M-9 1/4 ⇑ M-3 ⇑ *trees*: 3 ***ff1***		M-10 1/4 ⇑ M-4 ⇑ *trees*: 3 ***fl***		M.5
M.6	M-11 1/4 ⇑ M-3 ⇑ *trees*: 4 ***ff***		M-12 1/4 ⇑ M-4 ⇑ *trees*: 4 ***fw***		M.6
M.7	M-13 1/4 ⇑ M-6 M-7 ⇑ *trees*: 1 ***ze***				M.7
M.8	M-14 1/4 ⇑ M-6 M-8 ⇑ *trees*: 2 ***ze***				M.8
M.9	M-15 1/4 ⇑ M-9 M-10 ⇑ *trees*: 3 ***ff***				M.9
M.10	M-16 1/4 ⇑ M-11 M-12 ⇑ *trees*: 4 ***ze***				M.10
M.11 a b	M-17 4/4 a: ⇑ M-13 ⇑ M-14 ⇑ *trees*: 1, 2 b: ⇑ M-15 ⇑ M-16 ⇑ *trees*: 3, 4 ***zp***				M.11 a b
M.12 a b	M-18 4/4 a: ⇑ M-17a ⇑ *trees*: 1, 2 b: ⇑ M-17b ⇑ *trees*: 3, 4 ***zr***				M.12 a b
M.13	M-19 4/4 ⇑ M-18 M-5 ⇑ *trees*: 1, 2, 3, 4 ***wypowiedzenie***				M.13

Fig. 3. Parse forest in compact form with the tree number 4 highlighted

the future the amount of displayed information about the label will be controlled by the user.

A specific tree can be highlighted by changing, e.g., the background of appropriate cells. In our sample spreadsheet we used this method to highlight tree number 4 (the same which is shown on Figure 1). As you can see, the tree is composed of all the cells of rows T1, M.2, M.6, M.10, subrows M.11b and M.12b, and the row M.13 (containin the single cell representing the root of the tree).

4 More examples

It should be stressed that the applications of syntactic spreadsheets are not limited to successful parses of single sentences. They can be used also to present the forests created during incomplete or unsuccessful parsing processes, so they can be used also as a debugging tool. Moreover, they can be used for units larger than a single sentence. Although the arrangement of the cells is important for clarity, the links between cells are specified explicitly, so in principle spreadsheets can show also the structure of sentences with non-standard word order and discontinuous constituents.

In the general case the sheet can be quite large and may require splitting into several pages. A technique analogical to that used for geographical maps and plans seems to be fully applicable also to syntactic spreadsheets.

For long sentences and large spreadsheets it seems useful to create partial spreadsheets representing only the top parts of the forest trees; in such a case the number of columns will be smaller as some columns will represent several consecutive morphological segments (words and punctuation marks).

We present now some sample spreadsheets used in [4] to ilustrate the parsing results for some computer messages. The spreadsheets has been more or less simplified for printing purposes. The more detailed versions, which are also more readable thanks to the use of color, are available at `http://fleksem.klf.uw.edu.pl/~jsbien/synspread/samples`.

Figure 4 demonstrates using a simplified fragment of a spreadsheet for texts larger than a single sentence. The text in the example consists of two sentences (*wypowiedzenie* literary means ‘utterance’), the segmentation has been done by the parser; the spreadsheet shows the morphological ambiguities, but the strictly syntactic parts contains only the tips of the forest.

Figure 5 shows rather a drastically simplified fragment of a spreadsheet for an unsuccessful parse result, which however provides useful information about recognized sub-sentence components; you can see that the culprit is the *mailman* placeholder for date, which has to be incorporated into the grammar.

Figure 6 demonstrates a case when parsing was bound to fail because the input string is not a complete sentence. The morphological analysis is highly ambiguous, so we see 4 possible syntactic interpretations of *lata*: genitive singular and nominative plural of LATO (‘summer’), a form of ROK (‘year’) and a form of LATAĆ (‘to fly’). Moreover the text contains also the *evolution* placeholder for a number. To the fragment *lata temu* (‘years ago’) 5 different syntactic structures

T-1 *trwa* TRWAĆ: fin	T-2 *pobieranie* POBIERANIE: subst	T-3 *.* .: interp	T-4 *czy* CZY: qub	T-5 *zapisać* ZAPISAĆ: inf	T-6 *zmiany* ZMIANA: subst	T-7 *?* ?: interp
	T-8 *pobieranie* POBIERAĆ: ger				T-9 *zmiany* ZMIANA: subst	
wypowiedzenie			***wypowiedzenie***			
trwa pobieranie.			***czy zapisać zmiany?***			

Fig. 4. Segmentation into sentences

T-1 *ostatni* OSTATNI: adj	T-2 *zwrot* ZWROT: subst	T-3 *otrzymano* OTRZYMAĆ: imps	T-4 *z* Z: prep	T-5 *twojego* TWÓJ: adj	T-6 *adresu* ADRES: subst	T-7 *dnia* DZIEŃ: subst	T-8 *%(date)s* %(DATE)S: [date]
zd							***mailman***
ostatni zwrot otrzymano z twojego adresu dnia							***%(date)s***

Fig. 5. Segmentation into sub-sentence units

are assigned represented by non-terminals *fl*, *fpt*, *fw*, *fzd*, *zd*; their meaning is specific to a variant of Świdziński's grammar (described in [8]), so there is no need to explain this here.

It should be noted that syntactic spreadsheets can incorporate also quite sophisticated methods of beautifying parse trees. As it was already pointed in [11, p. 36], in Świdziński's grammar the number of parse trees of sentences containing coordination of more than two components can be described as Catalan numbers, which depend in almost exponential way on the number of components. In consequence it would be very useful to collapse such segments of the tree forest into a single 'Catalan cell'. Similar ideas, but applied to single trees, has been advocated in [6].

5 Spreadsheet cells

In [2] I proposed to include in spreadsheet cells full information about their respective nodes, but it no longer seems practical to me. In particular, it seems there is no reason to print the node labels in exactly the same form as in the tree format. We plan, however, to include this information in a processed form. The idea is to display below the main functor only those of its arguments which have the same value for all component subrows, and in the subrows to highlight only the specific information. So cell M-12 of the spreadsheet presented in Figure 3 (highlighting the tree number 4) may look as in Figure 7. Please compare this notation with the second top row of the parsing trees in Figure 2 (the letter O is the abbreviation of *osoba* meaning *person*). The tree rows are labelled with the appropriate symbol of the grammar rule used (in this case r1), such information can be of course provided also in the spreadsheet cells.

T.1	T-1 ***%d*** %D: %d 0	T-2 ***lata*** LATAĆ: fin sg:ter:imperf	T-3 ***temu*** TEN: padj sg:dat:[m1, m2, m3, n1|n2]:pos	T.1
T.2		T-4 ***lata*** LATO: subst sg:gen:[n1|n2]		T.2
T.3		T-5 ***lata*** LATO: subst pl:[nom, acc|voc]:[n1|n2]		T.3
T.4		T-6 ***lata*** ROK: subst pl:[nom|acc]:m3		T.4
	evolution	***fl***		
		fpt		
		fw		
		fzd		
		zd		
	%d	***lata temu***		

Fig. 6. Parsing incomplete sentence

6 Technical considerations

The first approach assumed that the input to the spreadsheet generator is just the output of Woliński's Birnam parser [12]. It turned out, however, that recovering all the needed data from Birnam's output is quite difficult. As a consequence, a fork of Birnam is used now, which provides much more information in the output.

The additional information helps to determine the number of columns, which is not trivial because of a peculiarity of Świgra. The original grammar by Świdziński assumes the existence of the 'virtual comma' which has an empty realization. Because such rules are not allowed in bottom-up parsing, the commas are added to the input in appropriate places. As a consequence, counting the length of the longest path from the beginning to the end of a sentence is not sufficient, as unused virtual commas should not be included in the spreadsheet.

M.12	M-18 4/4
a	⇑ $\boxed{\text{M-17a}}$ ⇑
	trees: [1, 2]
	O=3
b	⇑ $\boxed{\text{M-17b}}$ ⇑
	trees: [3, 4]
	O=1
	zr
	os, nd, prze, ozn, m ,poj, ↑, tak, ni, np

Fig. 7. A sample cell

The spreadsheet is typeset as a longtable, using `\cellcolor` from the colortbl package for background and `\colorbox` from the color package for foreground elements [7]. Some details are still to be worked out (e.g., at present some vertical rules vanish when a cell is highlighted), but for the time being we consider the output to be satisfactory. For black and white printing the use of color is switched off by means of replacing the respective commands by dummy ones.

As every spreadsheet row consists of several lines, the data belonging to a single cell must be split into several parts, which makes the LaTeX source difficult to read and edit, even if supplemented by comments. Therefore, we introduced an intermediate forest representation in Prolog, which is used to generate the LaTeX source. The source of some sample spreadsheets described in the present paper is available for inspection at `http://fleksem.klf.uw.edu.pl/~jsbien/synspread/samples`.

It seems that the generated PDF files put a heavy strain on the previewers, as some of them display some spreadsheets in a distorted way. The matter is still to be investigated.

7 Closing remarks

A grammar and a parser are worth only as much as their results. In the case of language engineering tasks the output of the parser is processed further and is read by humans only for debugging purposes. However, if we aim at a linguistically sound description of a language, the grammar and the parser' output have to be easily readable. I hope to have demonstrated the great potential of syntactic spreadsheets in this respect.

Implementing the ideas appeared to be more difficult than expected. Therefore at the moment of writing (December 2008) there exists only a quick and dirty program to convert the output of Birnam's fork to the tentative intermediate representation. A program to convert the intermediate representation to LaTeX is still under development.

Work on the spreadsheet generation tool is going to be continued, hopefully with a help of a student, and in due time the results will be made available under the terms of the GPL, probably at `http://fleksem.klf.uw.edu.pl/~jsbien/synspread/`.

References

1. Bień, Janusz S., 1997. Komputerowa weryfikacja formalnej gramatyki Świdzińskiego. *Biuletyn PTJ*, LII:147–164.
2. Bień, Janusz S., 2006. Wizualizacja wyników analizy syntaktycznej. *Poradnik Językowy*, 9:24–29.
3. Bień, Janusz S., Krzysztof Szafran, and Marcin Woliński, 2000. Experimental parsers of Polish. In *3. Europäische Konferenz "Formale Beschreibung slavischer Sprachen, Leipzig 1999"*, volume 75 of *Linguistische ArbeitsBerichte*. Institut für Linguistik, Universität Leipzig.
4. Bień, Janusz S., Bilińska, Joanna A., Moszczyński, Radosław, 2008. Towards linguistic analysis of computer messages in Polish and English. In *Studia Kognitywne* 8, Slawistyczny Ośrodek Wydawniczy, Warszawa, pp. 288-302. `http://bc.klf.uw.edu.pl/32/`
5. Hockett, Charles F., 1968. *Kurs językoznawstwa współczesnego*. PWN.
6. Kovář, V., Horák A., 2007. Reducing the Number of Resulting Parsing Trees for the Czech Language Using the Beautified Chart Method. In *3rd Language & Technology Conference: Human Language Technologies as a Challenge for Computer Science and Linguistics*, October 5-7, 2007, Poznań, Poland, pp. 433-437.
7. Mittelbach, Frank, Michel Goosens, Johannes Braams, David Carliste, and Chris Rowley, 2004. *The LaTeX Companion*. Reading, MA: Addison-Wesley Publishing Company, 2nd edition.
8. Ogrodniczuk, Maciej, 2006. *Weryfikacja korpusu wypowiedników polskich (z wykorzystaniem gramatyki formalnej Świdzińskiego)*. PhD thesis, Uniwersytet Warszawski. `http://bc.klf.uw.edu.pl/30/`
9. Świdziński, Marek, 1992. *Gramatyka formalna języka polskiego*. Rozprawy Uniwersytetu Warszawskiego. Warszawa: Wydawnictwa Uniwersytetu Warszawskiego.
10. Woliński, Marcin, 2003. System znaczników morfosyntaktycznych w korpusie IPI PAN. *Polonica*, XXII–XXIII:39–55.
11. Woliński, Marcin, 2004. *Komputerowa weryfikacja gramatyki Świdzińskiego*. Ph.D. thesis, Instytut Podstaw Informatyki PAN, Warszawa. `http://www.ipipan.waw.pl/~wolinski/publ/mw-phd.pdf`
12. Woliński, Marcin, 2005. An efficient implementation of a large grammar of Polish. In Zygmunt Vetulani (ed.), *Human Language Technologies as a Challenge for Computer Science and Linguistics. 2nd Language & Technology Conference April 21–23, 2005*. Poznań.
13. Woliński, Marcin, 2006. Jak się nie zgubić w lesie, czyli o wynikach analizy składniowej według gramatyki Świdzińskiego. *Poradnik Językowy*, 9:102–114.