Nonstandard usage of Liang's algorithm: automatic insertion of the stress marks in Polish texts

Krzysztof Caputa, Agnieszka Kendzia and Przemysław Scherwentke

Instytut Matematyki i Informatyki Politechniki Wrocławskiej, Wybrzeże Wyspiańskiego 27, 50–370 Wrocław

Introduction I

In the language of modern Poles the ability of proper accenting is in decline. It is probably connected with the fact that word stress in the Polish language is basically paroxytonic, i.e. it falls on the syllable before the last (the penult). It may also fall on the last syllable or the third, fourth, fifth, sixth, or even seventh syllable, counting from the end of the word.

Introduction II

			7	6	5	4	3	2	1
							a	ku	**rat**
							e	tu	**i**
								my	li
							my	li	‿się
							my	li	śmy
						my	li	śmy	‿się
						my	li	by	śmy
					my	li	by	śmy	‿się
				my	li	by	li	by	śmy
			my	li	by	li	by	śmy	‿się
					za	sta	no	**wi**	li
za	sta	no	**wi**	li	by	li	by	śmy	‿się

Introduction III

Accenting on the penult in these cases is regarded as a gross error by properly educated people.

Why?

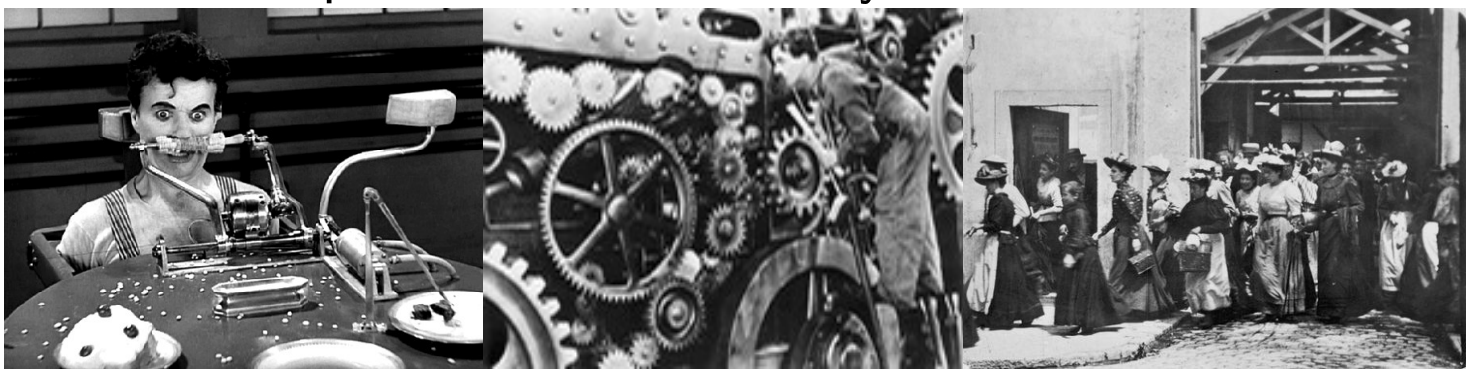
robi**li**ście $\neq$ ro**bi**liście

robi**li**ście $\approx$ robi liście (sb makes leaves)

Leaves?

Introduction IV

Makes? I.e. produces? In a factory?



Or even worse meaning of 'robi' (Warning! Offensive):

Introduction V

Introduction VI

myśmy zrobili = my zrobiliśmy
byśmy zrobili = zrobilibyśmy
bylibyśmy zrobili = zrobilibylibyśmy

Introduction VII

There has arisen an idea, then, to invent a tool helping in proper accenting by marking the proper part of a word. Liang's hyphenation algorithm has been our first choice. Because TEX has a tool for marking potential divisions of words, an adequate modification of the set of hyphenation patterns should result in indicating the place of word stress.

Hyphenation in TEX I

Each language has a separate system of hyphenation rules. The author of TEX was looking for an algorithm which would be independent from hyphenation rules; hyphenation rules were to be specified through data for that algorithm. An algorithm fulfilling these requirements has been invented by Frank M. Liang [2]. He has even managed to achieve more: the data are relatively easy to construct, even without the knowledge of information technology. The algorithm of hyphenation is very simple, provided we have got proper data, of course.

Hyphenation in TEX II

Example (Bogusław Jackowski, Marek Ryćko, Tam gdzie minus oznacza dzielenie, Biuletyn GUST **2**, 1993).
Let's assume we have got these data. They comprise a set of sequences of alternate letters and digits, e.g. *2s0z1z0, 2s0z0l0n0, 2t1ć0* itp. The extreme digit may by replaced by a dot, e.g. *.w0e3s2, 8r0s0z.* itp.
Such sequences are called patterns.
Let us say we have got a word, e.g. *odkaszlnąć* (to cough up).

Hyphenation in TEX III

First we transform it into the same form the patterns have, inserting the digit *0* between letters and a dot in each of the extreme positions, obtaining *.o0d0k0a0s0z0l0n0ą0ć.* (the dot marks the word's beginning or end). Then, in our file of data we look for all the patterns which may be covered with this word in such a way that the positions of the letters and the dots are the same, and the positions of the digits — not necessarily. Note that a digit may correspond to a dot but not to a letter.
Let us say that we have found the following patterns *.o2d2, .o0dk2, 0ą1, 2d1k0, 2l1n0, 2s0z1l0, 2s0z0l0n0, 8ć., 0a1, 0o1, 0s4z0.* If we cover the word *.o0d0k0a0s0z0l0n0ą0ć.* with them and take the the greatest number out of the corresponding numbers, we will obtain: *.o2d3k2a2s4z2l1n0ą8ć.*

Hyphenation in TEX IV

According to Liang's algorithm we are only allowed to divide the word in a place with an odd number. It means that the algorithm in this case permits two points of division: od-kaszl-nąć. (The example comes from [7]). The patterns are kept in the memory of the computer in a form of the tree structure. It allows for a dense data compression, quick access to the data, and easy implementation of the structure of the data and the algorithm itself.

It is worth noting that thanks to the effectiveness of the used method, TEX may retain in its memory several different sets of patterns and hyphenate words in a few languages (the programme does use this possibility) inside one document (and even a paragraph).

Hyphenation in TEX V

The main part of Liang's work concerned a method of creating patterns on the base of a dictionary containing correctly hyphenated words. The simplified recording of Liang's algorithm looks like this:

Hyphenation in TEX VI

m := The maximum length of a pattern
for c := 1 **to** 9 **do**
begin
Establish the criteria for pattern acceptability
for l := 1 **to** m **do**
begin
For every word in the dictionary check whether it provides information about the possibility of inserting the number *c* in the patterns of the length l with established criteria of acceptability.
end
end

The programme completes the set of patterns gradually for increasing numbers and increasing patterns, with a possibility of

Hyphenation in TEX VII

the user's intervention at any stage. The criteria of acceptability are established through giving three numbers:

- w_p — weight for correct hyphenations
- w_n — weight for incorrect hyphenations
- p — the level of acceptability

The pattern is accepted when
$\{k_p w_p - k_n w_n >= p\}$
where:

- k_p — The number of correct hyphenations generated by a given pattern
- k_n — The number of incorrect hyphenations generated by a given pattern

Hyphenation in TEX VIII

In some cases the program may not find a set of patterns hyphenating all the words correctly. The reason may lie in faulty data (e.g. the same word albeit with different hyphenation may be found in the dictionary twice) or in the incorrect selection of numbers w_p, w_n and p in successive iterations.

Our set of hyphenation patterns I

The main goal of the present work was to establish a set of patterns (on the base of a set of patterns for the Polish language) which would hyphenate a word in a way that would enable marking a vowel (or vowels) stressed in Polish. Because word stress in Polish is not always put in the same place, we need a hyphenation that would enable correct marking of an accented vowel (accented vowels). A "new Polish language" with new hyphenation would be created in such a way.
On the base of the dictionary [1] the words were grouped in three sets:

- The first set comprises words with the stress on the penult and monosyllables.

Our set of hyphenation patterns II

- The second set comprises words which have certain ending patterns, namely words with stress on the third syllable before the last (antepenult) and the fourth syllable before the last.
- The third set comprises accenting exceptions and words not belonging to either of the two other sets.

On the base of the second set particular endings were selected, e.g. *liśmy, libyśmy, yka, ika*. Then they were added to the file with hyphenating patterns (described in more detail later in the chapter), creating new hyphenation in the TEX system, e.g.:

Our set of hyphenation patterns III

before	after
by-li-śmy	by-liśmy
ku-pi-li-by-śmy	ku-pi-libyśmy
ma-te-ma-ty-ka	ma-te-ma-t-yka

Out of the third set some words were selected and placed in the file of hyphenation patterns in the space occupied by exceptions. Therefore a new set of hyphenation patters `akcent.tex` has been created on the base of the existing set.
The file has been created for the purpose of proper functioning of our programme. It is a formal hyphenation differing from standard

Our set of hyphenation patterns IV

rules of hyphenations in Polish. The following shows the way of adding our pattern of hyphenation to already existing patterns:
Using the standard TeX tree TDS:
In the implementation TeXlive 7

- In the directory
  `texmf\tex\generic\hyphen`
  a file `akcent.tex` with new patterns of hyphenation has been placed.
- In the directory
  `texmf\tex\platex\config` an addition has been placed in the file `language.dat` in the following form
  `akcent  akcent.tex`

Our set of hyphenation patterns V

- Then in the command line we call
  `mktexlsr.exe`
  Or for TeXlive 7:
  `Start` → `Programy` → `TeXlive` →
  → `Maintenance` →
  → `Rebuild ls-R filenames databases`
  In this way our new file has been added.
- Then in the command line we write
  `fmtutil.exe --all --dolinks`
  or
  `Start` → `Programy` → `TeXlive` →
  → `Maintenance` →
  → `Create all formats files`

Our set of hyphenation patterns VI

Then our hyphenation pattern will be included with the use of the command
`\selecthyphenation{akcent}`.

To create data enabling new hyphenation, on the basis of the dictionary [1] the words were divided into three groups

1. The first set comprises words with the stress on the penult and monosyllables. It is the biggest group. The hyphenation of the words from this group has not changed. TEX still hyphenates these words in the following way:

pe-sy-mizm	ko-niec	płacz
fi-zy-ko-te-ra-pia	sie-dzieć	mia-sto

Our set of hyphenation patterns VII

2 On the base of the dictionary [1], a file consisting of about 10,000 words in which the stress does not fall on the penult has been created. Creating this file proved to be long and hard work. However, it enabled us to notice that some endings, e.g. *liśmy, libyśmy, ika, yka*, appear repeatedly and thus we were able to isolate a group of about 60 endings. The second set comprises words with stress on the third syllable before the last (antepenult) and the fourth syllable before the last. Word hyphenation in this group has been altered. Our data (the endings) had to be modified in such a way as to fit the TEX pattern of Polish hyphenation. The pattern, created by H. Kołodziejska [4] (and modified by

Our set of hyphenation patterns VIII

B. Jackowski i M. Ryćko) on the basis of Liang's algorithm [2] looks like this:

⋮
.o2t3ch/l
.o3b4/l/a
.o3b4/l/e
.o3b4/loc
.o3b4luzg
.o3b4ra/c
⋮

Our set of hyphenation patterns IX

To add our file of endings to the existing hyphenation patterns in Polish we have to use Liang's algorithm. Below there is an example of how to transform our data in such a way that they have the form identical with the data in the set of hyphenation patterns.

liśmy
libyśmy
ika
yka

At the ends of these strings we insert dots signifying the end of the word.

Our set of hyphenation patterns X

li/smy.
liby/smy.
ika.
yka.

Then at the beginning of this string we insert the odd number 5 and between the letters the even number 6. Numbers from the range 0–9 describe the degree of word hyphenation. For our programme to function correctly it is enough to choose the numbers from the middle of the range.

Our set of hyphenation patterns XI

5l6i6/s6m6y.
5l6i6b6y6/s6m6y.
5i6k6a.
5y6k6a.

According to Liang's algorithm the word may be hyphenated only in the place occupied by an odd number. It means that the algorithm permits hyphenation only at the beginning of the sequence.

-liśmy
-libyśmy
-ika
-yka

Our set of hyphenation patterns XII

Therefore exemplary words will be hyphenated in the following way:

zro-bi-liśmy
ku-pi-libyśmy
ma-te-ma-t-yka

3 The third set comprises accenting exceptions and words not belonging to either of the two other sets, for example words accented on the ultima. The set of exceptions consists of about 40 words. These words were added to the new pattern of hyphenation in TEX. The words were correctly hyphenated and the accented vowel is written as a capital letter. Out of these exceptions we can enumerate:

Our set of hyphenation patterns XIII

aku-rAt
bAd-min-ton
na-Uka

That is how the data for a hyphenation pattern were obtained, creating for our purposes a hyphenation different from the rules of the Polish language.
On the basis of the hyphenation in the “new Polish language” we can easily establish where an accented vowel is (or where accented vowels are).

Our set of hyphenation patterns XIV

1. In the Polish language the stressed syllable is usually the one before the last. Such words belong to the first set in our classification. In order to mark word stress in this group, it is enough to find the syllable before the last one. The last hyphen shows us where the stressed syllable ends. The first encountered vowel (or vowels) before this hyphen and after a consonant signifies word stress, e.g.:

ko-ry-tarz cho-dnik
no-to-wać po-kre-wień-stwo
sie-dzieć za-się-gać

In monosyllabic words all the vowels are marked.

Our set of hyphenation patterns XV

krzyk śmiech
płacz jęk

2. As we already know from previous chapters, there are exceptions to the rules of accenting the penult. The second set in our classification comprises words:
   - with word stress on the third syllable before the last one.
   - with word stress on the fourth syllable before the last one.

   Using the new hyphenation, the one different from standard Polish hyphenation, we can mark the stress in this group. The words have been hyphenated as follows:

zro-bi-liśmy chcie-libyśmy
ana-li-t-yka bo-ta-n-ika

Our set of hyphenation patterns XVI

In order to mark the stress we have to find the last hyphen. The first encountered vowel (or vowels) before the hyphen and after a consonant constitutes the word stress.

zro- bi- liśmy chcie- libyśmy

ana- li-t- yka bo- ta-n- ika

3 The third set comprises the so called accenting exceptions. In this group there are words with the stress on the first or the last syllable and words not fitting the rules of either the first or the second set. As already mentioned, these words have been added to the set of hyphenating patterns in the part devoted to the hyphenating exceptions.

Our set of hyphenation patterns XVII

eks-mĄż aku-rAt
na-Uka A-plauz

Marking word stress in this group is very simple. A capital letter signifies the place of stress. It is enough to find this letter in a word, then to mark the stress and then transform the letter into a lower-case letter.

eks-mą̲ż aku-ra̲t
na-u̲ka a̲-plauz

How the programme works I

Marking word stress has been achieved by means of the hyphenating procedure in the TEX system. To obtain information about the place of hyphenation we use the command `\showhyphens`. Its standard outcome is adding to the file '.log'. The following is the result of using the command `\showhyphens` in the latin2 encoding:

```
\showhyphens {Źrebak}
[ ] \OT4/cmr/m/n/10 ^^99re-bak

\showhyphens {Żaba}
[ ] \OT4/cmr/m/n/10 ^^9Ba-ba
```

How the programme works II

```
\showhyphens {Ślimak}
[ ] \OT4/cmr/m/n/10 ^^91li-mak

\showhyphens {Ćma}
[ ] \OT4/cmr/m/n/10 ^^82ma
```

Some signs possess different representations, as demonstrated by the examples below:

Ź	—	^^99
Ż	—	^^9B
Ś	—	^^91
Ć	—	^^82

How the programme works III

It is enough to read the information from the file '.log', and then send it to the function which carries out marking word stress (function \PLakcent). Because of that, it was essential to correctly merge the information obtained from the file '.log' and information obtained from the file '.tex'.
To regain data from the file '.log' we used a programme written in GAWK (the implementation of AWK) because AWK has an extended mechanism of searching patterns. AWK is a script language so its programmes are clear and readable. A characteristic feature of AWK programmes is their amazingly small size in comparison with the number of possible tasks they can manage. Unfortunately, the programme worked properly only in

How the programme works IV

the Linux system (otherwise the command `system ( )` was not interpreted correctly).
Therefore, through the command:

```
a2p plik.awk > plik.pl
```

the programmes written in the AWK language were converted into programmes in the Perl language, because in Perl "a well-written installation programme will probably work in almost every operation system — Perl is available for almost all the popular platforms" [12].
All the TEX commands are read in by means of the package `akcent.sty`. To obtain the package you need to create the file `trans.tex`. This file should be created in your favourite editor

How the programme works V

with your own preamble and the content similar to the one below (the lines necessary in the file have been italicized)

```
\documentclass[12pt]{article}
\usepackage{polski}
\usepackage{\jobname}
\begin{document}
\noindent
aAcCeElLnNoOsSxXzZ
\input litery.tex
\end{document}
```

How the programme works VI

In the file`trans.tex` there is the command initiating `litery.tex`. Therefore, you need to create the file `litery.tex`. In this file you should place calling the function `Litery`, and its argument should constitute Polish diacritical signs in the following form:

```
\Litery {ąĄćĆęĘłŁńŃóÓśŚźŹżŻ}
```

Note that the order is essential here. Then we save and compile the file `trans.tex`:

platex trans

How the programme works VII

In the compilation process in '.dvi' we should obtain the desired effect — Polish diacritical signs. Note that the signs should be written in the single-bit form. The next step is activating `dostosuj.bat`. In the command line write:

dostosuj.bat

The effect of its work will be obtaining TEX files adjusted to the editor and the compilator. The following files will also be created:

- `akcent.sty` — the file of TEX macrodefinitions and commands resulting from recoding signs from the file `akc-tex.1`.

How the programme works VIII

- `ldb.pl` — Perl script obtained through the merge of files `ldb1.pl, ldb3.pl, znaki.pl` in which our association table used for recoding information from the '.log' file has been contained.

In an exemplary file `proba` we write calling out the function `\AkcentWywolanie`, and our argument should be the word on which we want to mark the stress. We compile our newly-created file with the use of platex.

platex proba

We activate the system script `kompiluj.bat`. In the command line we write:

How the programme works IX

kompiluj.bat proba

As a result we obtain the files `*.akc (e.g. 1.akc, 2.akc)`, which contain calling the function `\PLakcent` with arguments:

\PLakcent{the usser's argument}{the answer if the command \showhyphens}

The answer is read in from the '.log' file. As an additional result of the action `kompiluj.bat` a report is produced. In three columns there appear:

- the user's argument;
- The answer of the command \showhyphens;

How the programme works X

- the name of the file in which the calling of the function \Placent is placed.

Then we compile again:

platex proba

And in the result file we obtain the word with a marked stress.
If we want to delete the created files we should write the following in the command line:

czysc-al.bat proba

How the programme works XI

And then all the results of particular compilations will be deleted. The user may make changes, for example change the way of marking word stress. The following function carries this out in the standard way:

```
\def \Zaznacz#1{\underline{#1}}
```

This function marks the stress by underlining The user may change the way of marking the stress through redefining the function \Zaznacz (e.g. the change of the shape or colour of the character, etc.)

The description of the file akcent.sty I

In the file `akcent.sty` there is a definition for reading the file \newread.
In this case it is:

```
\newread\AkcentStream
```

The function which is called in the user's file in order to obtain a word with the stress marked is called by means of:

```
\def\AkcentWywolanie#1
```

with the given word as the parametre. Inside the function there is the following command:

The description of the file akcent.sty II

```
\let\temp=\s
```

There follows an assignment to \temp of whatever is meant by \s, therefore after \s 4 spaces are inserted. We send the spaces to the file '.log'. with the command \message. In such a way we obtain a more comprehensible content sent to the file '.log'.
The next stage of this definition is enlarging the counter of the calls of this function. Then the marker consisting of digits 12345 is sent to the file '.log' with the command \typeout.

```
\typeout{12345}
```

The description of the file akcent.sty III

The command \showhyphens{wyraz} shows in the file '.log' a message about the way TEX hyphenates given words. The argument is the word obtained from the called function. Command \MakeLowercase writes the word with lower-case letters.
\showhyphens{\MakeLowercase{#1}}
In order to hyphenate a word with our pattern of hyphenation akcent.tex the following command has been used:
{\selecthyphenation{akcent}\showhyphens{\MakeLowercase{#1}}}
and the counter of calls is placed after a space.
Next there is the procedure of checking whether the file is open. You try to open the file with the command

The description of the file akcent.sty IV

\openin\AkcentStream, while the name of the file is the value of the counter called with the extension .akc.
Then the formerly opened file is tested:

\ifeof\AkcentStream

There follows checking whether the stream \AkcentStream is opened.
If you have not managed to open the file, an information that the file *.akc (e.g. 1.akc, 2.akc) does not exist yet is sent to the file '.log'. If the file has been opened, we are given the information Akcent PL and the command \closein\AkcentStream which closes the file for reading.

The description of the file akcent.sty V

Checking the assignment of the value \s=\temp finishes the procedure \AkcentWywolanie.
The function \AkcentWywolanie is placed in the file by the user, and the exemplary file 1.akc is created in the process of compilation by an external programme.
The content of the exemplary file comes from the first call of the function \AkcentWywolanie e.g. 1.akc. The function \PLakcent will not be called until the file, e.g. 1.akc, appears. The function being called is the main function \PLakcent with two arguments:

\def\PLakcent #1#2

The description of the file akcent.sty VI

The first argument is the argument coming from the TEX file written by the user, and the second argument is the result of using the command \showhyphens used in \AkcentWywolanie Below there is the algorithm of the function \PLakcent.
Carrying out the function \PLakcent Proceeds as follows:

I. Reservation.
   There follows a reservation of the following block: definitions of conditions (ifs) and definitions of counters.

II. Recognition
   In this place the following functions are carried out:

   - The function \PoczDusz{#1} with one argument. The argument is the word given by the user. The function checks whether the word has been written with an upper-case letter.

The description of the file akcent.sty VII

- The function \TestMyslnik{#2} with one argument. The argument is a word coming from the file '.log'. The function checks if there is a hyphen in the word coming from the file '.log'.
- The function \TestDuzej{#2} with one argument which is the result of the use of the command \showhyphens. The function checks if there is a capital letter in a word coming from the file '.log', which would classify the word as an exception.

III. Marking the stress.
The next step is checking the argument:

* \ifCzyTeMy (whether there is a hyphen). If there is:

The description of the file akcent.sty VIII

false (there is no hyphen). If there is no hyphen, the word belongs to monosyllabic words and the following function is carried out: `\Jednosylabowe`.

– The function `\Jednosylabowe#1` is called with one argument. The function implements marking the stress in a monosyllabic word.

true If there is a hyphen, checking the argument follows:

* `\ifCzyDuza` (if there is an upper-case letter in the word coming from the file '.log')

The description of the file akcent.sty IX

true (there is an upper-case letter). This word is an exception and the function `\ZaznaczWyjatki` is implemented:

– The function `\ZaznaczWyjatki#1` with one argument. The function marks the stress in a word classified as an exception.

false (is not an exception).
These are words belonging to group 1 and 2 (described earlier). The stress in these words appears on the penult. To find the stress, we have to establish the area of marking - the beginning and the end of marking:
The following functions are implemented:

The description of the file akcent.sty X

- The function \Myslnik#1 with one argument. The function establishes the position of the last hyphen in a word coming from the file '.log'.
- The function \OkrMiej#1 with one argument. The function \OkrMiej#1 establishes the place in which marking the stress begins.
  Then, when the area of marking has been established, the following function is carried out:
- Function \ZazAkc#1 with one argument. The function marks the stress in a word in a place established by the functions in the area of marking.

```
\newread\AkcentStream %
\
\newcounter{LiczWywolan}%z
\setcounter{LiczWywolan}{0}
%
\newcounter{Faza}%
\newcounter{nrMysl}%
\newcounter{nrSam}%
\newcounter{nrLitra}%
\newif\ifCzyPD%
\newif\ifCzyTeMy%
\newif\ifCzyDuza%
\newif\ifCoJest%
\newif\ifCoBylo%
\newif\ifCzyMysl%
```

```
\def\Zaznacz#1{\underline{#1}}
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

\def\AkcentWywolanie#1{%
  \let\temp=\s%
  \def\s{ }%
  \def\SpCz{\message{\s\s\s\s}}%
  \stepcounter{LiczWywolan}%
  \typeout{12345}%
  {\selecthyphenation{akcent}
  \showhyphens{\MakeLowercase{#1}
  \theLiczWywolan}}
  \openin\AkcentStream=\theLiczWywolan.akc
  \ifeof\AkcentStream{
  \typeout{}
  \SpCz\SpCz\message{*** Akcent PL *** }
```

```
  \typeout{\s Plik '\theLiczWywolan.akc'
  dla wyrazu '#1' jeszcze nie istnieje}

  \SpCz\SpCz\message{*** Akcent PL *** }
  \typeout{}}
  \else\closein\AkcentStream
  \input{\theLiczWywolan.akc}%
  \fi%
  \let\s=\temp%
}
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
\def\PLakcent #1#2{%
  \CzyPDfalse  %
  \PoczDusz{#1}%
  %% \ifCzyPD{Jest du/z/a}
```

```
\else{Jest ma/l/a}\fi %
\CzyTeMyfalse%
\TestMyslnik{#2}%
%%
\CzyDuzafalse%
\TestDuzej{#2}%
\ifCzyTeMy{%
  \ifCzyDuza {%
    \setcounter{nrLitra}{0}% %
    \ZaznaczWyjatki{#2}%
  }%
  \else{%
    \setcounter{nrMysl}{0}%
    \setcounter{nrLitra}{0}% %
    \CoJestfalse%
    \CoBylofalse%
```

```
      \Myslnik{#2}%

      \setcounter{nrSam}{0}%
      \setcounter{nrLitra}{0}%
      \CoJestfalse%
      \CoBylofalse%
      \OkrMiej{#2}%
      \setcounter{nrLitra}{1}%u
      \stepcounter{nrMysl}%
      \ZazAkc{#2}}\fi%
}%
\else{%
  \CoJestfalse%
  \setcounter{nrLitra}{0}%
  \Jednosylabowe{#2}%
}%
```

```
  \fi%
}%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%


\def\Jednosylabowe#1{
\setcounter{Faza}{7}\fifo #1\ofif}%

\def\procJedoSy#1{%  %
  \CoJest#1{a/ae/eiouy/o}%
     \ifnum\thenrLitra<1%
       \ifCzyPD%
         \ifCoJest\MakeUppercase
         {\Zaznacz{#1}}
         \else\MakeUppercase{#1}\fi%
       \else%
```

```
        \ifCoJest\Zaznacz{#1}\else#1\fi%
      \fi%
    \else%
      \ifCoJest\Zaznacz{#1}\else#1\fi%
    \fi%
  \stepcounter{nrLitra}%
}%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
\def\ZazAkc #1{
\setcounter{Faza}{6}\fifo #1\ofif}%

\def\procZA#1{%
  \CzyMysl#1{-}%
  \CoJest#1{aeio/ouy/a/eAEIO/OU}%
  \stepcounter{nrLitra}%
```

```
  \ifCzyMysl% %
    \ifnum \thenrSam<\thenrLitra%
     \ifCoJest%
       \ifnum \thenrLitra<\thenrMysl%
        \JakZaznaczyc{#1}%
       \else#1\fi%
     \else\ifnum \thenrLitra=2
     \MakeUppercase{#1}
     \else#1\fi\fi%
  \else\ifCzyPD\ifnum \thenrLitra=2
  \MakeUppercase{#1}
  \else#1\fi
  \else#1\fi\fi%
\fi%
}%
```

```
\def\CzyMysl #1#2{%
  \def\rozpoznajB ##1#1##2\END{%
    \ifx\empty##2%
    \empty\CzyMysltrue%
    \else\CzyMyslfalse\fi}%
  \rozpoznajB#2#1\END}%

\def\JakZaznaczyc#1{%
  \Zaznacz{\ifnum \thenrLitra=2 %%
    \ifCzyPD{\MakeUppercase{#1}}%
    \else#1\fi%
    \else#1\fi%
  }%
}%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
\def\OkrMiej#1{
\setcounter{Faza}{5}\fifo #1\ofif}%

\def\procOM#1{%
  \CoJest#1{aeio/ouy/a/eAEIO/OU}%
  \stepcounter{nrLitra}%
  \ifnum \thenrLitra<\thenrMysl
  \ifCoJest{\ifCoBylo{}%
    \else{\setcounter{nrSam}{\thenrLitra}}
    \fi}
    \fi\fi%
    \ifCoJest\CoBylotrue\else\CoBylofalse
    \fi%
}%

\def\CoJest #1#2{%
```

```
  \def\rozpoznajB ##1#1##2\END{%
    \ifx\empty##2%
    \empty\CoJestfalse%
    \else\CoJesttrue\fi}%
  \rozpoznajB#2#1\END}%

%%%%%%%%%%%%%%%%%%%%%%%%%%

\def\Myslnik #1{
\setcounter{Faza}{4}\fifo #1\ofif}%

\def\procMysl#1{%
  \CoJest#1{-}%
  \stepcounter{nrLitra}%
  \ifCoJest{\ifCoBylo{}
  \else{\setcounter{nrMysl}{\thenrLitra}}
```

```
  \fi}\fi%
}%
%%%%%%%%%%%%%%%%%%%%%%%%%%

\def\ZaznaczWyjatki#1{
\setcounter{Faza}{3}\fifo #1\ofif}%

\def\procZaWyj#1{%  %
  \CzyMysl#1{-}%
  \ifCzyMysl %%
     \CoJest#1{AEIOUY/O/A/E}%
     \ifnum\thenrLitra<1%
       \ifCzyPD%
         \ifCoJest\Zaznacz{#1}
         \else\MakeUppercase{#1}\fi%
       \else%
```

```
          \ifCoJest\Zaznacz{\lowercase{#1}}
          \else#1\fi%
        \fi%
      \else%
        \ifCoJest\Zaznacz{\lowercase{#1}}
        \else#1\fi%
      \fi%
  \fi%%%\ifCzyMysl%
  \stepcounter{nrLitra}%
}%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%

\def\TestDuzej#1{
\setcounter{Faza}{2}\fifo #1\ofif}%

\def\procTD#1{%
```

```
  \CzyDuza#1{AEIOUY/O/A/E}%
}%

\def\CzyDuza #1#2{%
  \def\rozpoznajB ##1#1##2\END{%
    \ifx\empty##2%
    \empty%
    \else\CzyDuzatrue\fi}%
  \rozpoznajB#2#1\END}%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

\def\TestMyslnik#1{
\setcounter{Faza}{1}\fifo #1\ofif}%

\def\procTeMy#1{%
```

```
  \CoJest#1{-}%
  \ifCoJest\CzyTeMytrue\fi%
}%
%%%%%%%%%%%%%%%%%%%%%%%%%%%


\def\PoczDusz#1{
\setcounter{Faza}{0}\fifo #1\ofif}%

\def\procPoD#1{\CzyPD{#1}
{ABCDEFGHIJKL/LMNO/OPRSTUWZ/Z/X}}%

\def\CzyPD #1#2{%
  \def\rozpoznajB ##1#1##2\END{%
    \ifx\empty##2%
    \empty%
```

```
    \else\CzyPDtrue\fi}%
  \rozpoznajB#2#1\END}%
%%%%%%%%%%%%%%%%%%%%%%%%%

\def\fifo#1{\ifx\ofif#1\ofif\fi%
  \ifcase \theFaza%
  \procPoD{#1}% 0%
  \or \procTeMy{#1}% 1%
  \or \procTD{#1}% 2%
  \or \procZaWyj{#1}% 3%
  \or \procMysl{#1}% 4%
  \or \procOM{#1}% 5 %
  \or \procZA{#1}% 6%
  \or \procJedoSy{#1}% 7%
  \fi%
  \fifo}%
```

```
\def\ofif #1\fifo{\fi}  %
```



W. Doroszewski, Słownik poprawnej polszczyzny PWN,


F. M. Liang, Word Hy-phen-a-tion by Com-put-er, Ph.D. thesis, Stanford University, 1983.


J. Désarménien, The use of TEX in French: hyphenation and typography, [in:] D.Lucarella (ed.), "TEX for Scientific Documentation", Proc. of the First European Conference, Addison–Wesley, 1984.


H. Kołodziejska, Dzielenie wyrazów w systemie TEX, 1987.


W. Puza, Analiza porównawcza wybranych programów Desktop Publishing (DTP), M.Sc. thesis, Instytut Poligrafii Politechniki Warszawskiej.


Kazimierz M. Borkowski, LATEX. Profesjonalny skład publikacji, Toruń 1992


Bogusław Jackowski, Marek Ryćko, Tam gdzie minus oznacza dzielenie, Biuletyn GUST **2**, 1993.


Kees van der Laan, FIFO and LIFO sing the BLUes, Biuletyn GUST **4**, 1994.


Marek Ryćko, Podstawy systemu TEX, Wirtualna Akademia GUST.


Bogusław Lichoński, Tomasz Przechlewski, AWK — opis języka z przykładami, Biuletyn GUST **7**, 1996.

 Tomasz Przechlewski, Opis języka AWK, 2000

 Adam Dawidziuk, Piotr Bolek, Perl w przetwarzaniu tekstów, Biuletyn GUST **7**, 1996.

 Larry Wall, Tom Christiansen, Jon Orwant, Perl, O'Reilly, 2001.