**Filip Graliński*, Krzysztof Jassem*, Agnieszka Wagner**, Mikołaj Wypych*****

***Adam Mickiewicz University, Faculty of Mathematics and Computer Science, Poznań**

**** Adam Mickiewicz University, Faculty of Linguistics, Poznań**

*****Institute for Fundamental Technological Research, Polish Academy of Sciences, Warsaw**

# Linguistic Aspects of Text Normalization in a Polish Text-to-Speech System

## Abstract

The paper addresses linguistic problems of text normalization for the Polish language. Text normalization, which converts the written form of a text into the spoken form, is one of the preprocessing steps in text-to-speech systems. Normalization of texts in analytic languages like English does not necessarily require deep linguistic analysis. However, it is shown here that for synthetic languages, like Polish, linguistic analysis is crucial for the normalization process. Existing Polish text-to-speech systems, even though highly estimated for the naturalness of output, do not solve main normalization problems. The authors' team aims at developing a text-to-speech system that will include a strong text normalization module. The idea is to design the module using linguistic resources and mechanisms developed for a Machine Translation system, Translatica. Progress of research may be followed at www.poleng.pl, where the user may input a source Polish text in the written form and obtain its "translation" after normalization.

## 1. Architecture of the text-to-speech system

The main data stream in the system flows through four modules: 1) normalization, 2) phonetization, 3) prosody generation and 4) unit selection. The system accesses two databases: the lexicon (during normalization and phonetization) and the speech corpus (for unit selection).

The normalization module developed by the authors uses the engine designed for a machine translation system, Translatica (www.translatica.pl). Linguistic aspects of the normalization process are the main topic of this paper.

The remaining modules of the text-to-speech system have been designed by Wypych basing on his previous research and the Festival [1] speech synthesis system:

The phonetization module converts normalized text into a sequence of phonetic symbols, syllable boundaries and lexical stress marks. The module is based on the rule-based transcription algorithm presented in [2]. (The module extends the algorithm with new procedures for syllable division and lexical accent).

The prosody generation module provides three major suprasegmental parameters of speech segments: pitch, duration and loudness. The module uses trainable models of the Festival speech synthesis framework [1], [3] and an automatic intonation recognizer presented in [4].

Speech synthesis is based on the concatenative method: In order to synthesize a sentence, the unit selection module searches for the most appropriate speech segments in the speech corpus. The unit selection module uses the Festival Multisyn speech synthesizer. The automatic segmental labeling of the corpus has been done by means of the Sonic speech recognizer [5].

The key concept in the system development has been the maximal reuse of existing technologies. Notably, the first version of the system has been implemented in no more than six months. The system has been successfully integrated with the Translatica machine translation system, where a selected Polish text may be read by a computer. Translatica also supports an experimental function of speech translation: The user's English utterance is recognized by a Microsoft SAPI speech recognizer, translated into Polish and the spoken equivalent is generated by the text-to-speech function.

## 2. Text normalization in speech synthesis

Text normalization is an automated process, which converts the written form (orthographic form) of the text into the spoken form [6]. Various approaches have been taken towards text normalization [7, 8, 9, 10, 11]. Normalization is often the first phase of text preprocessing in a text-to-speech system [12, 13, 14]. It consists in expanding abbreviations, converting names, numbers, acronyms, dates etc. into their spoken form. For example, the string *$200* should be expanded in English into *two hundred dollars*.

Normalization of a Polish text is more complex and this is due to the inflected character of the language and the lack of efficient and widely available language analysis tools such as morphological or syntactic analyzers. As a result, so far the existing text-to-speech systems for Polish have not succeeded in solving many text normalization related problems.

We have tested some expressions which require normalization with Realspeak [15], Ivona [16] and Acapela [17] systems. The following problems have been identified:

a) Unrecognized abbreviations, e.g. *na os. Rzeczpospolitej m. 5* that should be expanded into *na osiedlu Rzeczpospolitej mieszkania pięć* (*at Rzeczpospolitej estate, apartament five*) or *por. rozdz. VII* that should be read: *porównaj rozdział siódmy* (compare chapter seven). The tested systems do not expand abbreviations.

b) Improper inflected forms, e.g. the phrase *19 listopada ub. roku*, which should be read: *dziewiętnastego listopada ubiegłego roku* (*on the nineteenth of November last year*). The tested TTS systems recognized the abbreviation *ub.* (*ubiegły – last*) but failed to inflect it correctly. The same problem occurred with the expression *na str. 30* (at page thirty) – *str.* was expanded into the nominative instead of the locative case (i.e. *strona* instead of *stronie*), or in the expression *we współpracy z (in collaboration with) prof. dr hab. Nowakiem*, where the title of Mr. Nowak should be put in the instrumental case and was not.

c) Unrecognized formats of dates and some numerical expressions, e.g. *6-Sty-80* (*6-Jan-80*) or *Czw Sty 26, 2006* (*Thur, Jan 26, 2006*), *semestr letni 97/98* (*summer semester 97/98*), wynik *5:3* (*the score five to three*).

Difficulties of text normalization in the Polish language may be illustrated by the following sentence:

**Example 1. Text normalization of a Polish sentence**

*(0) dla (1) p. (2) dr. (3) J. (4) Kowalskiego (5) leg.(6) się (7) dow. (8) osob. (9) BAC1234567, (10) zam. (11) na (12) os. (13) B. (14) Chrobrego (15) 10 (16) m (17) 7, (18) 61-100 (19) Poznań*

*In English: (0) for (1) Mr. (2) Phd. (3) J. (4) Kowalski (5) holding (7) I (8) D (9) BAC1234567, (10) living (11) at (12) est. (13) B. (14) Chrobrego (15) 10 (16) apt. (17) 7, (18) 61-100 (19) Poznań*

The sentence should be expanded into:

*(0) dla (1) pana (2) doktora (3) jot (4) kowalskiego (5) legitymującego (6) się (7) dowodem (8) osobistym (9) be a ce jeden dwa trzy cztery pięć sześć siedem, (10) zamieszkałego (11) na (12)*

*osiedlu (13) bolesława (14) chrobrego (15) dziesięć (16) mieszkania (17) siedem, (18) sześćdziesiąt jeden sto (19) Poznań*

*In English: (0) for (1) Mister (2) Philosophy Doctor (3) J (4) Kowalski (5) holding (7) identity (8) card (9) bee a cee one two three four five six seven, (10) living (11) at (12) estate (13) bolesława (14) chrobrego (15) ten (16) apartment (17) seven, (18) sixty one hundred (19) Poznań*

It is claimed here that the process of text normalization needs to include deep linguistic analysis. As it happens, normalization may be successfully treated by the same mechanisms and algorithms as transfer-based machine translation, with the exception that the target language is the spoken form of the input. The authors have applied the formalism and mechanisms developed for the MT system, Translatica, to the normalization of Polish texts. The effect may be traced at www.poleng.pl, where the user may input a Polish expression and choose the direction of translation into: English, Russian, or …normalized Polish. The last choice calls translation of the text into its spoken form.

The first and essential step of text normalization is the preparation of the lexicon. The lexicon is the integral part of the system, which must be prepared beforehand. Creation of the lexicon is described in Section 3.

For a given input, linguistic steps needed for proper normalization are the following:

- tokenization (i.e. division into words, numerals, punctuation marks, and other types of tokens) – Section 4
- sentence segmentation – Section 4
- lexical analysis (including look-up for lexical phrases) – Section 5
- syntactic analysis for each sentence in order to obtain a set of possible parses – Section 6
- syntactic disambiguation in order to choose the best parse – Section 7
- semantic disambiguation in order to select the best of possible meanings – Section 8
- transfer into spoken form – Section 9
- inflectional synthesis of the output – Section 10

Let us notice that the similar steps are usually taken in translation of a text from one natural language into another via the transfer method (see [18]).

## 3. Preparation of the lexicon

Let us clear our interpretation of basic lexicographical notions:

**Word (single word)** is an element of a natural language written without a space.

**Multiword** is a sequence of at least two words that in the linguistic analysis (lexical, syntactic, semantic) should be treated as an inseparable unit – in the same way as words. Examples of multiwords are *na przykład* (*for example*), *na pewno* (*for sure*).

**Lexical phrase** is a group of at least two words, which:

- does not convey the meaning that may be derived compositionally from its component words;
- may be assigned a syntactical role in a sentence;
- is not an inseparable unit in the computerized text processing.

An example of a lexical phrase is *brać kogoś żywcem* (*to take someone alive*).

**Lexical unit** is one of: word, multiword or lexical phrase.

**Lexeme** is a set of *lexical units* that undergo the same description in a lexicon. Lexical units of one lexeme are called **inflected forms** of the *lexeme*. A selected inflected form of the lexeme is called the **basic form** of the *lexeme*. Usually, the basic form is infinitive for verbs, nominative singular for nouns, etc. All the other forms of the lexemes may be listed in a lexicon either **explicitly** or mentioned **implicitly** by giving a method of deriving them from the basic form.

**Entry** is a structure that consists of:
1) *lexeme* – given *explicitly* or *implicitly*
2) information about the lexeme depending on the destination of a lexicon.
**Lexicon** is a set of *entries*.

If the destination is text normalization, the information stored in the lexicon should contain for each entry, apart from linguistic attributes, spoken equivalents of written forms. Linguistic attributes of entries are necessary for the proper linguistic analysis of the whole text – it is shown further that such analysis is crucial for the process of normalization.

A lexicon intended for text normalization is called here a **normalization lexicon.**

**Separating entries**

One of the decisions that should be made before creating a lexicon of any type is the criterion for separating entries represented by the same basic form. Various approaches have been taken in the leading traditional Polish dictionaries. The Oxford-PWN Polish-English dictionary [19] separates different meanings of the adjective *dobry* (good): literal and colloquial (*he waited for her a good hour*). The Great Universal Dictionary of Polish [20] tends to separate entries regarding to their etymology (e.g. *gazowy* - *gauze*, *gazowy* - *gaseous*).

Here, we assume that each entry is identified by the pair <basic form; grammatical category>. (We prefer the notion of a grammatical category to a more popular notion of a part-of-speech to emphasize that the lexicon entries come from written texts). Our classification of grammatical categories is strictly connected with our algorithms for the processing of Polish texts – for example we distinguish a few noun categories as regards to the gender.

The normalization lexicon for Polish includes some entries that differ only in the grammatical category, having the common basic form. Table 1 shows the most spectacular example: The abbreviation "*p.*" forms no less than 5 different entries in the normalization lexicon:

| Abbr. | Expansion | Meaning | Gram. Cat. | interpretation of Grammatical Cat. | Example of normalization |
|---|---|---|---|---|---|
| p. | pan | Mr | N:1 | noun, masculine, human | dla p. Kowalskiego → dla pana Kowalskiego |
| | pani | Mrs | N:4 | noun, feminine | dla p. Kowalskiej → dla pani Kowalskiej |
| | punkt,<br>pokój | point,<br>room | N:3 | noun, masculine, inanimate | zdobył 5 p. → zdobył pięć punktów<br>Hotel Imbiss, p. 5. → Hotel Imbiss, pokój pięć |
| | piętro, | floor | N:5 | noun, neutral | 3-e p. → trzecie piętro |
| | patrz | see | S | sentence | p. [1] → patrz jeden |

**Table 1. Lexical entries for the abbreviation *p.***

The abbreviation *p.* is separated into 5 entries because in different contexts the abbreviation should be treated as belonging to 5 different grammatical categories (moreover, two possible expansions: *punkt* and *pokój* belong to the same grammatical category).

**Various equivalents**

One lexical entry may have various "translations" into their spoken form (this is analogous to a situation in a bilingual dictionary). The situation takes place for the two

expansions of *p.* (*punkt, pokój*). Another example is the abbreviation *ks.* (belonging to the grammatical class N:1) that may be expanded to either *książe* or *ksiądz*.

**Inflected forms**

The normalization lexicon should include all inflected forms of the entry words in the explicit form. The lexeme represented by the basic form *dr* (abbreviation for *doctor*, English equivalent: *PhD*) should contain all inflected forms that may occur in real texts. There exist at least three ways of denoting the inflection forms of the lexeme *dr*: with a dot, with the ending preceded by the hyphen and with the ending without the hyphen. For example, the dative case of the expression: *dr Kowalski* may be written as: *dr Kowalskiego, dr. Kowalskiego, dr-a Kowalskiego, dra Kowalskiego* (all these forms should be translated in the normalization process into: *doktora*).

## 4. Tokenization and sentence segmentation

Tokenization is the division of a text into tokens – inseparable units of texts, such as words, numerals, punctuation marks or non-lexical strings (e.g. e-mail addresses). In a text-to-speech system proper tokenization is needed not only for word segmentation but also for sentence delimitation. In order to correctly split a text into sentences, the splitting algorithm should properly handle punctuation marks, particularly dots. A dot may end the sentence, may serve to denote the abbreviation (e.g. *p.*) or may have the both functions simultaneously (e.g. *itd.* – English: *etc.*).

Three types of abbreviations ending with a dot may be distinguished:

Type 1) abbreviations that do not end the sentence, e.g. *p., zam.* (*living*)

Type 2) abbreviations that usually end the sentence, e.g. *itd.* (*etc.*), *itp.* (*and alike*)

Type 3) the abbreviations that *may* end the sentence, e.g. *ub. r.* (*last year*), *r.* (*year*), *ub. m.* (*last month*), *m.* (*month*).

The normalization lexicon should include the information on the type of the abbreviation and the algorithm for sentence segmentation should take the information into account. One of the possible treatments of the abbreviations by the segmentation algorithm may be the following:

1) never divide the sentence after abbreviations of Type 1,
2) use standard segmentation rules for abbreviations of Type 2 (e.g. divide after a dot and before a space followed by a capital letter),
3) use standard segmentation rules after abbreviations of Type 3 unless they precede numerals (like e.g. in the expression *w r. 2006* (*in year 2006*)).

In text normalization it is crucial that the tokenization should distinguish between types of numerical expressions. Numerical expressions such as telephone numbers, dates, time, ZIP codes, vehicle license numbers should be identified in the text and handled in a special way because their pronunciation very much depends on what they represent. For example, the numerical string *0(48)12 628 24 30* most likely represents a telephone number and should be expanded into the Polish equivalent of *zero, forty-eight, twelve, six hundred and twenty eight, twenty-four, thirty* rather than be read digit by digit. The same applies to the string *22-02-2006*, which represents a date and should be pronounced *dwudziesty drugi lutego dwa tysiące sześć (twenty-second of February two thousand six)*. The distinction between types of numerical expressions may be made by providing a symbolic representation (e.g. in terms of regular expressions) [21] of each numerical expression in question. Table 2 shows examples of different types of expressions and their representation by regular expressions:

| expression type | examples | symbolic representation (regular expression) |
|---|---|---|

| | | |
|---|---|---|
| **telephone numbers** | (0-33) 82-82-826<br>0 801 555 555<br>0-(prefix)-52-32-60-725<br>(48-56) 660-71-77<br>0048/12/616-21-04<br>061 426 10 12 | `(tel\. ?)?\+?([0-9]{0,4}[-/ ])?`<br>`(\(?[0-9]{0,2}\)?[-/ ])?`<br>`(\(?prefi(x\|ks)\)?[-/ ])?`<br>`(\(?[0-9]{0,2}\)[-/ ])?`<br>`[0-9]{2,3}[-/ ][0-9]{2,3}[-/ ][0-9]{2,3}` |
| **ZIP codes** | 61-255 | `[0-9][0-9][- ][0-9][0-9][0-9]` |
| **car licence numbers** | WX 0025<br>PO 0223X<br>PZA 121Y | `[A-Z]{2,3} [0-9]{3,4}[A-Z]` |
| **dates** | 1 VII 2006 r.<br>12.XI.1999<br>9-VII-1999<br>dnia 24-07-1995 | `(dni(a\|u))?`<br>`([0-2]?[1-9]\|[3][0-2][.- ])?`<br>`(I{1,3}\|I[VX]\|[XV]I\|V\|VI{1,3}\|X\|XII\|`<br>`0?[1-9]\|1[012])[.- ]`<br>`[1-2][0-9]{3}(r(\.\|oku))?` |
| **PESEL**[1] | 49040501580 | `[0-9]{11}` |
| **NIP**[2] | 889-10-16-508 | `[0-9]{3}[- ][0-9]{2}[- ][0-9]{2}[- ][0-9]{3}` |
| **time** | 21:35:00<br>20.00 | `(godz\.?\|g\.)?`<br>`([01]?[0-9]\|[2][0-4])([:.][0-5][0-9]){1,2}` |

**Table 2. Regular expressions for numerals in Polish texts**

The expression for telephone numbers says that such a number consists of at least seven digits with spaces or hyphens in between. The preceding context may be helpful: telephone numbers are usually preceded by an abbreviation *tel*. written with or without a dot. A telephone number (without the directory prefix) should be read: triple, pair, pair, e.g. *426 10 12* is expanded into the Polish equivalent of *four hundred and twenty-six*, *ten*, *twelve*.

The disambiguation of Polish ZIP codes is easy. They always fit the pattern: two digits followed by a hyphen or space followed by three digits. They are read: pair – triple, thus the way of pronunciation is indicated by the delimiting punctuation.

The expansion of dates into text is somewhat more complicated and requires the list of month numbers (Roman and Arabic). Digits representing the day and the year should be expanded into ordinal numbers. The date may be preceded by the word *dnia* or *dniu* (*day*) and followed by *r.* or *roku* (*year*) (e.g. *w dniu 20 X 1863 r.* – pronounced as the equivalent of *on the twentieth of October eighteen hundred sixty-three*).

The PESEL numbers are made up of 11 digits with no spaces or delimiting punctuation marks in between. They are read: 4 pairs followed by a triple, e.g. 80052705300 is expanded into the Polish equivalent of *eighty, zero five, twenty-seven, zero five, three hundred*.

Pairs of digits separated by dots or colons, expressing time, are expanded into ordinal numbers and read pair by pair, e.g. *21:35* is expanded into *dwudziesta pierwsza trzydzieści pięć* (English literal translation: *twenty first thirty five*).

Let us notice that the proper conversion of "long numerals", such as shown in Table 2 rarely requires linguistic analysis: Recognizing the type of a numeral by means of regular expressions suffices for the generation of the correct spoken output. "Short numerals" however, which do not comply to regular expressions listed in Table 2, must be processed in the context of the surrounding text, as is shown in Section 9.

[1] Polish national identification number)

[2] tax-payer identification number

## 5. Lexical analysis

Lexical analysis searches for each recognized token of the text in the lexicon and then "memorizes" the found information in order to use it in further steps. (In case a token is not found, lexical analysis calls special procedures in order to pose hypotheses about the syntactic and semantic characteristics of the unknown token.)

For example, lexical analyzer finds the word *Kowalskiego* in the lexicon and memorizes the fact that it denotes a surname in genitive. The same applies to the abbreviations: the analyzer memorizes the fact that the first two interpretations of the basic form *p.* usually precede a name, one entry being of masculine gender, the other – feminine.

A normalization lexicon should include special cases of multiwords. Entries such as *art. mal.*, *pod kier.*, *pod red.* should be included in the normalization lexicon and then identified in the lexical analysis as multiwords, so that they can be expanded respectively into: *artysta malarz* (*painter artist*), *pod kierunkiem* (*managed by*) *pod redakcją* (*edited by*). This is important because the abbreviated words in default contexts are expanded differently: *art.* → *artykuł* (*article*), *kier.* → *kierownik* (*manager*), *red* → *redakcja* (*editors*).

A normalization lexicon should allow for the storage of lexical phrases. This may enable the conversion of the string *ul. B. Chrobrego* (*B. Chrobrego Street*) into *ulica Bolesława Chrobrego* (*Bolesława Chrobrego Street*) provided that the lexicon is supplied with full names of famous people.

## 6. Syntactic analysis

Syntactic analysis makes it possible to link words into phrases (sentence components). Often, a phrase requires the agreement of morphological features of its components (e.g. case agreement between noun and its modifying adjective). If a phrase contains an abbreviation (or a numeral), this requirement may help generate proper inflected forms of the abbreviation (numeral).

Let us follow the steps of the syntactic parser used to analyze examples given in Table 1.

1) *dla p. Kowalskiego*

The parser "knows" from the lexical analysis that *Kowalskiego* is a surname and that the abbreviation *p.* should be linked into one sentence component with a surname only if it represents the entry "translated" into *pan* or *pani*. The analyzer chooses the masculine entry *pan* because of the agreement of the genders between the abbreviation and the name.

2) *dla p. Kowalskiej* – the same procedure as in 1) chooses the entry translated into *pani*.

3) *zdobył 5 p.*

The abbreviation does not precede a name – it should not be handled in one of the two ways mentioned above. The parser attempts to link it with the preceding numeral (*5)* and selects two interpretations: either *punkt* or *piętro*.

The syntactic analyzer does not find a clue which interpretation to prefer – this is left for semantic disambiguation.

4) *Hotel Imbiss, p. 5.*

A numeral precedes the abbreviation. A syntactic rule says that the analyzer should link the abbreviation with a preceding numeral only if it represents the entry translated into *pokój*.

5) *3-e p.*

The ending *-e* defines the numeral 3-e as an ordinal numeral in the neutral gender. The only interpretation of the abbreviation that comes in the neutral gender is *piętro* and this interpretation will be chosen by the syntactic parser.

6) *p. [1]*

The parser looks for a grammar rule that will make it possible to form a phrase out of the string *p. [1]*. In the absence of such a rule the parser will assume *p.* to be interpreted as a separate component, expanded into *patrz*.

## 7. Syntactic disambiguation

Not all ambiguities are solved in the syntactic analysis: often the syntactic parser returns alternative parses. It is up to the syntactic disambiguation to choose the parse which is most likely to be meant in the context. Two approaches may be taken to solve the problem: statistical (based on the analysis of text corpora) or heuristic (e.g. scoring the alternative parse subtrees during syntactic analysis and then selecting the highest-scored parse tree in the disambiguation phase). The authors address the analogous problem for machine translation in [22].

Syntactic disambiguation may be helpful in finding the preference of one spoken form over the other, e.g. to choose the interpretation of *m 7* (see Example 1) as *mieszkania siedem* (Eng. lit. *apartment seven*) over *mieszkania siódmego* (Eng. lit. *apartment seventh*), although both interpretations are theoretically correct in Polish.

## 8. Semantic disambiguation

Semantic disambiguation allows for the determination of one of possible equivalents (spoken forms) of the word. This is usually done by examining the context, i.e. the text surrounding the word in question. If a context of the expression *5 p.* is connected with sport or games of any type, the semantic disambiguator will choose the interpretation *pięć punktów* (*five points*), whereas the "building" context will force the disambiguator to choose *piąte piętro* (*fifth floor*)..

## 9. Transfer into the spoken form

### Treatment of numerals

"Long numerals" that comply with patterns listed in Table 2 are assigned the appropriate type in the tokenization process and do not require deeper linguistic analysis to be properly pronounced. The case is different for "short numerals" (such as *7*, *1013*, *-5,07*[3], *XIX*), which may denote any inflected form of either ordinal numbers or cardinal numbers (ordinal numbers inflect for gender, case and number, cardinal numbers inflect for gender and case). Some examples of normalization for the number *24* are shown in Table 3.

| Polish text | normalization | English translation |
|---|---|---|
| *zobaczyłem 24 samochody* | *zobaczyłem dwadzieścia cztery samochody* | *I saw 24 cars* |
| *zobaczyłem 24 mężczyzn* | *zobaczyłem dwudziestu czterech mężczyzn* | *I saw 24 men* |
| *zobaczyłem 24 dzieci* | *zobaczyłem dwadzieścioro czworo dzieci* | *I saw 24 children* |
| *z 24 samochodami* | *z dwudziestoma czterema samochodami* | *with 24 cars* |
| *24 maja* | *dwudziesty czwarty maja* | *the 24th of May* |

[3]In Polish a traditional way to mark the radix point is to use a coma; however a dot "comes into fashion" nowadays. This dualism complicates the normalization of numbers.

| Polish text | normalization | English translation |
|---|---|---|
| *przed 24 maja* | *przed dwudziestym czwartym maja* | *before the 24th of May* |

**Table 3. Normalization of Polish numerals.**

"Short numerals" are in our module processed by a special (transductor-like) procedure which returns their basic spoken form (e.g. *dwadzieścia cztery* for the cardinal rendering of *24*) and a special "inflectional instruction", which specifies in a precise manner how the given numeral should be inflected. The type of the numeral (ordinal or cardinal) as well as the values of inflectional attributes (such as gender, case or number) are determined in syntactical analysis.

It is worth noting that the transfer phase returns basic forms of numerals. Inflection of numerals is left for inflectional synthesis (Section 10), where the "inflectional instruction" is applied.

**Treatment of other tokens**

The algorithm for conversion of other tokens in the normalization process is simple: Look for the equivalent of a token in the lexicon and replace the token with its equivalent. If the equivalent is missing, echo the input token.

Similar to the treatment of numerals, in the transfer phase the replaced tokens are converted to the basic forms of their expansions.

## 10. Inflectional synthesis

The inflectional synthesis of the output generates inflected forms of the "translated" expressions. This means that for each expanded word, the synthesis phase generates its inflected form according to the information gathered in previous steps. For example, the syntactic analysis determines the abbreviation *leg.* as the modifier of the name *Kowalskiego*. The expansion of the abbreviation should therefore occur in the same inflected form as the name: *genitive masculin*e. The transfer phase translates *leg.* into the basic form *legitymujący*. The inflectional synthesis merges both pieces of information and produces the inflected form *legitymującego*.

Let us notice that for the sake of proper inflection, syntactic analysis should be capable of linking "distant" words, like *Kowalskiego* and *zam.* (*living*) (see Example 1) and assure the abbreviation to be expanded to the same case and gender as the name, in this case: genitive masculine.

## 11. Other normalization problems

Our testing procedure consists in applying the normalization program to large text corpora selected from the Internet. The testing procedure returns only strings that are not echoed by the algorithm. Results of testing often reveal problems that have not been foreseen by linguists who have prepared normalization rules.

Here are some examples of revealed problems:

- sport results (e.g. *4:2* or *4-2*),
- specifications of resolution (e.g. *1024x768*)
- IP addresses (e.g. *127.0.0.1*)
- combinations of units of measurement (e.g. *7 kg/ha*).

A large variation of notation has been observed for expressions described in previous sections (such as numbers or dates). Some constructions are incorrect from a prescriptive

point of view, but are quite frequent (*5-u mężczyzn* or *5-ciu mężczyzn* instead of *5 mężczyzn*) in real texts.

Handling such cases is necessary and makes normalization even more complex.

## 12. Status of the system

We have tried to evaluate our methods in the following way:

We prepared four bunches of sentences, each containing between 200 and 400 sentences. Each sentence of a bunch contained a specific token that should be expanded in normalization, namely: **prof.** (a token ending with a dot that should be expanded into an inflected form of one word: *profesor*, *English: professor* ), **tzw.** (a token ending with a dot that should be expanded into an inflected form of a multiword: *tak zwany*, *English: so-called*), **km** (a token not ending with dot) and **4** (a numeral).

We then compared the effectiveness of our algorithm to the effectiveness of two naïve algorithm, used by most other text-to-speech systems: one expands the token into its canonical form, the other – into its most frequent form, both: irrespective of the context. The results are shown in Table 4.

Each row characterizes one bunch of texts. The second column stands for the number of sentences in the bunch. The third column shows the correctness of the first naïve algorithm, the fourth column – the correctness of the second naïve algorithm. The correctness of our method is shown in the last column.

| **Token** | **Number of sentences** | Algorithm 1 | Algorithm 2 | Our method |
|---|---|---|---|---|
| prof. | 387 | 62% (profesor) | 62% (profesor) | 84% |
| tzw. | 391 | 11% (tak zwany) | 19% (tak zwane) | 76% |
| km | 380 | 4% (kilometr) | 72% (kilometrów) | 73% |
| 4 | 220 | 58% (cztery) | 58% (cztery) | 79% |

**Table 4. Evaluation of a few algorithms for normalization**

Table 4. shows that the largest improvement takes place in the case of a multiword. Another conclusion is that there is still room for further improvement – presumably by the enhancement of linguistic analysis as well as more detailed description of lexicon entries.

Currently, our lexicon of abbreviations contains 546 entries. However, in order to fully analyze texts the system requires an exhaustive lexicon of words and phrases. Most lexicon entries are not expanded in the normalization process but are still necessary for the proper linguistic analysis of a text.

The lexicon used here – based on the Translatica MT dictionary – contains 108 737 word lexemes, and 141 868 multiwords and phrases.

## 13. Conclusions

The authors' team aims at developing a text-to-speech system for Polish. One of the important steps of text-to-speech conversion is the text normalization. The paper shows that proper normalization of texts written in a synthetic language, like Polish, requires deep linguistic analysis. The process of text normalization have a lot in common with the process

of transfer-based machine translation – the spoken form in normalization may be treated as the equivalent of the target language in MT. The authors have successfully applied formalisms and mechanisms used in the MT system Translatica, to the normalization of Polish texts.

## Literature


1. Black A. W. Taylor P., Caley R. (1999), The Festival Speech Synthesis System. System documentation. Edition 1.4
2. Wypych M, Demenko G., Baranowska E. (2003), A Grapheme-to-Phoneme Transcription Algorithm Based on the SAMPA Alphabet Extension for The Polish Language, in: Proceedings of International Congress of Phonetic Science, Barcelona
3. Black A. W., Lenzo K. A (2003), Building Synthetic Voices, LTI, Carnegie Mellon University
4. Wypych M (2005), An Automatic Intonation Recognizer for the Polish Language Based on Machine Learning and Expert Knowledge, in: Proceedings of Interspeech 2005, Lisbon 2005.
5. Pellom B., Hacioglu K. (2003), Recent Improvements in the CU SONIC ASR System for Noisy Speech: The SPINE Task, in Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Hong Kong, April 2003
6. Speech Synthesis Markup Language (SSML), (2004). Version 1.0. W3C Recommendation, 7 September 2004, http://www.w3.org/TR/speech-synthesis/
7. Sproat, R. (1996). Multilingual text analysis for text-to-speech synthesis. Journal of Natural Language Engineering, 2(4):369--380.
8. Xydas G., Karberis G., Kouroupetroglou G. (2004), Text Normalization for the Pronunciation of Non-standard Words in an Inflected Language, in: Proceedings of the 3rd Hellenic Conference on Artificial Intelligence (SETN04),. Samos, Greece, May 5-8, 2004.
9. Yarowsky D. (1993), Text normalization and ambiguity resolution in speech synthesis, in: The Journal of the Acoustical Society of America, Vol. 94, Issue 3, p. 1841
10. Reichel U.D., Pfitzinger H. R. (2006), Text Preprocessing for Speech Synthesis. TC-STAR Workshop on Speech-to-Speech Translation, June 19–21, 2006 Barcelona, Spain
11. Black A., Sproat R., Chen S. (2000), Text normalization tools for the Festival speech synthesis system. http://festvox.org/nsw
12. The IMS German Festival Manual(2001), Institute for Natural Language Processing, University of Stuttgart, available at http://www.ims.uni-tuttgart.de/phonetik/synthesis/
13. Oliver D. (1998), Polish Text-to-Speech Synthesis, Master Thesis, University of Edinburgh 1998
14. Marasek K. (2003), Synteza Mowy: przegląd technologii i zastosowań za szczególnym uwzględnieniem języka polskiego, Polsko-Japońska Wyższa Szkoła Technik Komputerowych, Warszawa
15. Realspeak, http://www.nuance.com/realspeak/
16. Ivona, http://www.ivo.pl/
17. Acapela, http://www.acapela-group.com/
18. Jassem K. Transfer w systemie POLENG3, in: G. Demenko, W. Jassem, K. Jassem, M. Karpiński (red.)*: "Speech and Language Technology, vol. 6"*, Poznań, Polskie Towarzystwo Fonetyczne, 2002
19. Linde-Usiekniewicz J. (red.) (2003), The Great English-Polish Dictionary, Wydawnictwo Naukowe PWN, Warszawa
20. Dubisz S. (red.) (2003), The Universal Dictionary of Polish, Wydawnictwo Naukowe PWN, Warszawa, 2003

21. A. Mikheev (2004) “Text segmentation”, in: R. Mitkov (eds.) The Oxford Handbook of Computational Linguistics, Oxford University Press
22. Jassem K., Graliński F., Wypych M. (2003), Statistical and Heuristic Approach to Meaning Disambiguation in POLENG MT System, in: Demenko G. (red.) Analiza, synteza i rozpoznawanie mowy w lingwistyce, technice i medycynie, SZCZYRK 2003, Polskie Towarzystwo Fonetyczne