LOCAL GRAMMAR BASED AUTO-PREFIXING MODEL FOR AUTOMATIC EXTRACTION IN INDONESIAN CORPUS[i] (Focus on Prefix *meN-)*

PRIHANTORO
Universitas Diponegoro (Undip) Semarang

Abstract

Many concerns have been given to morphophonemic phenomena in Indonesian from the perspective of morphology-phonology intersection or even semantics. This paper studies the same phenomenon from Natural Language Processing (NLP) application, and proposes an automatic prefixing model which can later be used to perform NLP tasks such as pattern matching and automatic extraction. The focus of this paper is the phenomena of *meN-* prefixing in Indonesian. The machine readable linguistic resources built in this paper is expected to transform the linguistic description pertaining to the morphophonemic constraints to an applicable mean for NLP. To accomplish this goal, Local Grammar (LG) approach is employed since this approach is very powerful to describe and formalize a linguistic phenomenon. A corpus processing software, UNITEX, is also employed to apply LG in carrying NLP tasks. It demonstrates how morphophonemic prefixing in Indonesian can be carried out automatically to generate an inflected form machine readable dictionaries as lexical resources and how the resources are used for pattern matching and automatic extraction. The existing linguistic resources from the experiment (machine readable dictionaries, regular expressions and LGGs) are maintainable, open for development and application in a larger corpus, and potential to be used to improve search engines performance for on-line documents in Indonesian.

Keywords: Morphophonemic Constraints, Inflection, Natural Language Processing, Local Grammar

1. INTRODUCTION

Indonesian is the official language in Indonesia. Indonesian language is typologically classified into the Austronesian language family (Lewis 2009). According to a survey performed by Summer Institute of Linguistics (SIL), the number of Indonesian speakers reaches 222,699,476 (Ethnologue 2009 edition), but most likely to grow now. There are various interesting topics to discuss in Indonesian linguistics, but the 'morphophonemic' phenomenon of Indonesian prefix is one of the most frequently discussed ones. The phenomenon is referred as 'morphophonemic' since the constraints that involve morphology-phonology intersection. Consider examples (1) and (2):

(1) *meN-ambil*
meN-take
*me*nambil*
mengambil
'to take'

(2) *meN-* sapu:
meN-broom
*me*ngsapu*
menyapu
'to sweep floor'

Examples (1) and (2) illustrate how prefix *meN*[ii]- is attached to two canonic forms from different parts of speech (POS): *ambil* 'to take' [V] and *sapu* 'a broom' [N]. Note that *sapu* might also be verb in some particular syntactic contexts. When they are prefixed with *meN-*, the outcomes are all verbs.

Some canonic forms in Indonesian do not allow direct concatenation with particular prefixes. Instead, it must go through several processes. Prefix *me*N- can be realized as *meng-* or *men*-depending on the initial sound of the canonic word that it inflects. Thus, it is called morphophonemic (Alwi, 1998:109-113). As it is described in example (1), the verb *ambil* prefers 'ng' (phonetically represented as /ŋ/). In the two examples, besides a word formation process (morphology), a nasal assimilation process (phonology) also takes place.

There are various perspectives to study the morphophonemic process in Indonesian. So far, studies concerning prefix *meN-* most commonly performed from the perspectives of phonology or morphology (Alwi 1988, Kager 1993, Ermanto 2008). However, very little concerns are given to how morphophonemic constraints apply in Natural Language Processing (NLP) by computer. Studying this phenomenon from NLP is also important since some computer applications rely heavily on language processing, such as: Question-Answer system, Text Mining, Information Extraction and etc.

The way a computer understands human language is different from what a human does. Linguistic resources (auditory, orthography, dictionary, grammar etc) for computer must be made machine readable for the input processing. Thus, it requires a tool that has both expressive power to describe the linguistic phenomena, and ability to carry out NLP tasks. To handle prefixing constraints on Indonesian morphophonemic, the tool must conceive a morphological analyzer.

There have been some efforts to build a morphological analyzer for Indonesian language, like studies performed by Hartono (2002) that employed PC-KIMMO. Pisceldo et al (2008) has also built another morphological analyzer and tested the analyzer to KBBI (Indonesian Standard Dictionary).

The morphological analyzer proposed in this paper will be applied to a text corpus (not dictionary) where language is actually in use. For this experiment, it requires a corpus and a processing software as a tool. The tool that fits the approach used by the writer in this paper is UNITEX, that has been tested in some languages like English, French, Korean, Chinese, Thai etc[iii]. This software has a user friendly interface (even for non computational linguists/ computer scientists) and fits for linguists.

UNITEX is designed under the frame of Local Grammar (Gross 1993), that are very expressive in describing a local linguistic phenomenon. This corpus processing software can transform Local Grammar (LG) into LG Graph (LGG), a linguistic resource that is not only visual friendly but can also carry out NLP tasks such as pattern matching, automatic inflection, automatic extraction etc.

This paper is organized as follow. Chapter 1 introduces the necessity of discussing morphophonemic constraints from computational perspective. Chapter 2 highlights LG formalism and its application in NLP with UNITEX. Chapter 3 accounts for description of morphophonemic constraints under LG frame. Chapter 4 demonstrates how LGG can carry out automatic inflection of prefix *meN-*, generate a machine

readable dictionary, and the application of the dictionary in a research corpus. Chapter 5 provides the conclusion of this paper that contains the summary and some perspectives for further research.

2. LOCAL GRAMMAR FORMALISM

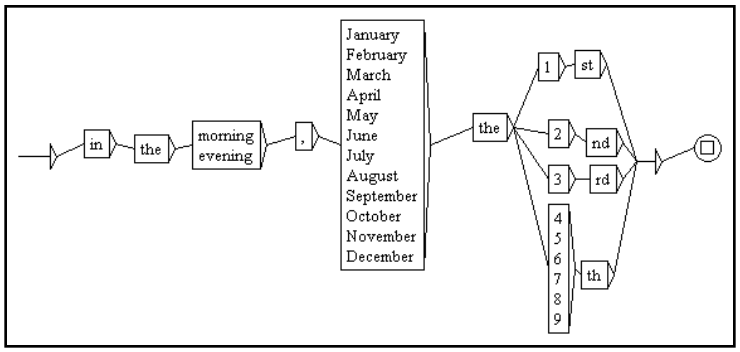
2.1 Local Grammar

Local grammar (LG) is a concept proposed by Maurice Gross (1993) to describe sequences that cannot be recognized by using previously existing rules; phrase structure rules and transformational grammar (e.g time and date expression). The basic idea is, LG can be constructed to capture recurring grammar and the resource can be use for description of larger linguistic construction; basically as bottom-up approach towards comprehensive description of language (Mason 2004:167). To apply this concept in NLP, a visual tool called Local Grammar Graph (LGG) is created (Zilberstein 1993). LGG has a very friendly interface even for a non-computer scientist.

2.2 The Architecture of Local Grammar Graph (LGG)

The architecture of LGG resembles the notion of Formal Grammar (Chomsky 1956), and has the power equal to Context Sensitive Grammar or Algebraic Grammar (Paumier 2003), which will further be explored in section four. LGG supports the four components of Formal Grammar (N,Σ,P,S), where the first three are finite sets: Terminal, Non-Terminal, Production rule and Start Symbol. LGG is composed of start and end state, box (lexicon repository, either terminal or non-terminal) which is connected one and each other by lines. This section describes how LGG fits those components with some computational improvements so that it can be used to perform NLP tasks as it is shown in figure 1.

Figure 1. LGG Sample (Date Expression)



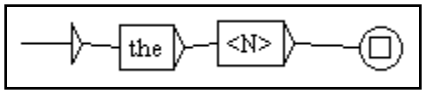
Boxes in the LGG in figure 1 are composed of some terminal lexicons and a punctuation symbol [,]. One box might be composed of one or more string. After the expression 'in the', there are some obligatory-optional features like to choose one from 'morning/evening', name one of one of the months, and a specific date. With the date, this LGG recognizes only correct expression like '1st, 2nd, 3rd, 4th, and does not recognize incorrect date expressions. Consider the examples of some possible PPs (Prepositional Phrases) in examples (3) to (7) that might or might not be recognized by the LGG sample shown in figure 1.

(3) In the morning, January the 1st
(4) In the evening, February the 2nd
(5) In the morning, March the 3rd
(6) In the evening, April the 4th
(7) *In the morning, March the 5nd (not recognized)

2.2.1 Reference Graph

To write all the expressions manually in a graph is a laborious work. For example, when we want to extract all NPs (Noun Phrases) strings composed of a determiner 'the' plus any noun, it is quite impossible to write all the nouns in one graph. For this kind of problem, a reference graph can be one solution. This graph allows direct reference to an applied machine readable dictionary. Note that to use a reference graph, the corpus must have already annotated in preprocessing stage by a machine readable dictionary. Consider illustration 2:

Figure 2. Reference Graph Sample (NP)

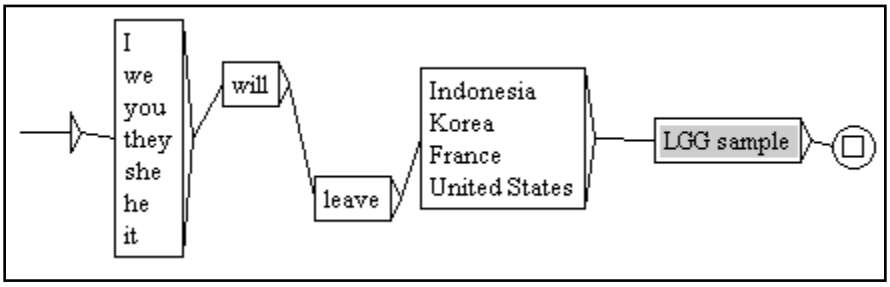


The LGG sample in figure 2 shows a reference graph to nouns <N>. By using this reference graph, all NPs in the corpus consisting of [the + N], for instances: the man, the teacher, the computer etc, will be extracted easily.

2.2.2 Sub-Graph

Sub-graph is a method of referencing to another graph, instead of a referencing to a machine readable dictionary. The graph might be composed of terminal or non-terminal. In section 2.2.1, reference is made to the applied machine readable dictionary. But a sub graph can be applied directly with or without the application of a dictionary. In this way, a graph can be simplified and made systematic. Please refer to figure 3:

Figure 3. Sub Graph Sample



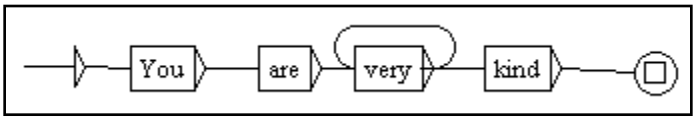
The LGG in figure 3 has a reference to graph named 'LGG sample', which is actually the date expression variants expressed in LGG in figure 1. This LGG will recognize a string composed of the

followings = pronoun + 'will leave' + a country name + date expression (sub graph). By using this sub-graph mechanism, it is also possible to create a production rule like [S→ NP + VP] in an LGG.

2.2.3 Recursiveness

It is possible to have a string where one part of the whole expression is repeated. For instance, we might have a sentence like, 'you are very very kind'. In this sentence, the adverb 'very' is repeated twice. Even though not completely grammatical, in a conversation, the adverb 'very' is sometimes repeated to emphasize the modified constituent. To recognize this kind of expression, an LGG illustrated in figure 4 can be considered:

Figure 4. Recursiveness with a Loop



The loop attached on lexicon box 'very' is eomployed to recognize the repetition of the lexicon inside the box. It will recognize one, two, three, four or more repetition of 'very' in a sentence.

2.2.4 Transducer

Transducer is a mechanism where an LGG can be used to generate output. It is sometimes referred as LGG transducer or Finite State Transducer for it generates a finite set of output. For instance, in figure 2, the expression recognized by the graph is a determiner 'the' plus any noun, which constitute an NP. If required for the users, this information can be attached in the concordance display (result). Consider figure 4 and 5:

Figure 4. Transducer Sample

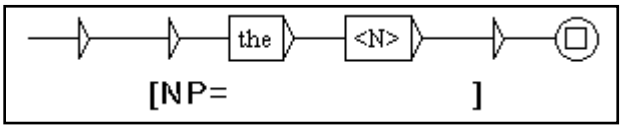


Figure 5. Concordance Sample

own situation, and [NP=the appearance] which he ma
e slumbered, under [NP=the appearance] of sullen c
take a turn round [NP=the back] o' the hill to ga
y, by the event of [NP=the battle] of Hastings, an
have mentioned in [NP=the beginning] of the chapt
rt of the cap that [NP=the bells] were attached; w
yet more close to [NP=the body], it was gathered

The graph in figure 4 is a transducer type. It assigns NP = to every string composed of [the + N]. The result of automatic extraction of NPs in an English Corpus is the concordance shown in figure 5. All the tokens after 'the' is tagged with <N> reference code in the dictionary.

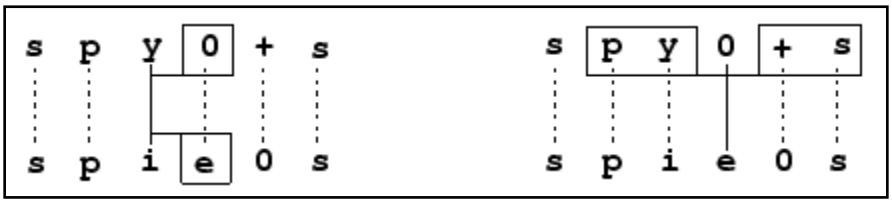
2.2.5 Inflection

Chomsky and Halle (1968) in SPE (Sound Pattern of English), describe rules that alter abstract phonological representations into surface forms through a series of intermediate representations and constraints. For instance the plural form of 'apple' is 'apples' but the plural form of 'spy' is not 'spys' or spyes', but 'spies. This morpho-phonological process which does not allow direct concatenation and involves contextual information, adopts context sensitive rewriting rules (Chomsky 1956, Partee et al 1993). It can be represented by the following mathematical notation:

Grammar Sample : A → γ/ α_β
String : α*A*β → αγβ

When this rule applies, a string 'α*A*β [iv]' will change to 'αγβ' because of the α_β context. This happens to some particular English singular-plural form like spy-spies. The context sensitive grammar ranges from phonology to syntax. 'Two-level morphology' is one of the linguistic phenomena where this context sensitive rule applies (Koskienniemi 1983). Two level morphology later be improved by Karttunen (1983), Gajek et al (1983), Dalrymple et al (1983), and implemented as morphological analyzer PC KIMMO (Antworth 1990) downloadable in the repository of SIL (Summer Institute of Linguistics). Figure 6 shows a two-level morphology processing for English pluralization of spy to spies, where [+] shows a morpheme boundary:

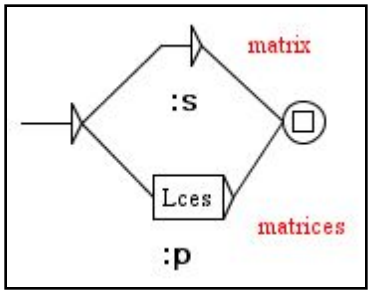
Figure 6. Pluralization of 'spy' to 'spies'



Pluralization of 'spy' shown in figure 6 can be described as two operations performed on a target string: [1] alternation of *y* -> *i* and [2] insertion of an epenthetic *e*.

In order to allow affixation on simple or multiword unit, the LGG must be designed to perform both direct concatenative and non-concatenative morphology. The mechanism to handle these phenomena in UNITEX is called 'inflection graph'. Inflection graph allows these two operations to take place. Consider figure 7 from Paumier (2003):

Figure 7. Inflection Graph



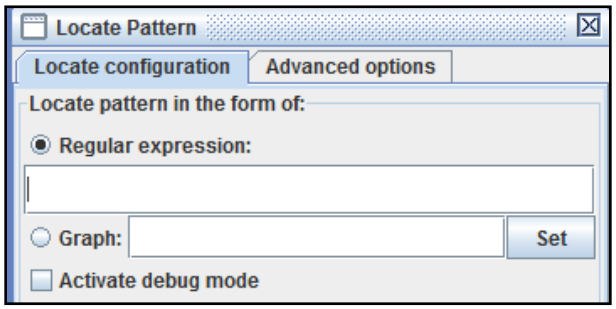
The inflection graph, in figure 7, handles pluralization for words like 'matrix' to 'matrices', 'radix' to 'radices' etc. There are two paths for the output: Singular word (:s) and plural words (:p). Singular words will not be given any treatment (upper path), but for pluralization (:p) the words are treated with formula (Lces). This formula deletes one character from the rightmost, and added (ces). Therefore, 'matrix' will be 'matrices' and 'radix' will be 'radices'. This inflection graph can be used to treat morphophonemic constraints in Indonesia. The constraints will be described further in section 3 of this paper.

2.3 Pattern Matching with UNITEX

Pattern matching is a basic function of NLP which is also used in search engines like Google, Yahoo, Bing etc. UNITEX can comply with plain text written under Unicode font and HTML format. There are two options for pattern matching operation, which is by either regular expression or LGG.

The first operation is via regular expression. A pattern matching with regular expression is quite similar to the query in search engines, where target strings are typed into a query box. Besides dictionary reference, it is possible also to generate output by using regular expression. However, pattern matching is much more effective by using an LGG search, especially for multi word expressions. This leads us to the second operation, a pattern matching via an LGG. Figure 8 illustrates a pattern matching window where there is an option to apply regular expression or LGG:

Figure 6. Pattern Matching Window



There are two options in the window shown in figure 6. Users might perform a pattern matching via regular expression, or graph (LGG). A pattern matching via regular expression requires users to type the query. While a pattern matching via graph requires users to set a previously drawn LGG. LGG search is recommended for multiword extraction. We will look into the practice of this extraction in section 4, which focus on a corpus on Indonesian language.

3. MORPHOPHONEMIC RULES FOR PREFIX *meN-*

Direct concatenation might apply for some prefixes in Indonesian: for instances, prefixes to mark passive (PASS) on transitive verbs e.g. *di-* (8) and, *ter-* (9). The first prefix applies when an action is done intentionally, and the second prefix is used when something happened accidentally. However, this direct concatenation rule does not apply for all prefixes. Prefix *meN-*, for instance, does not comply with direct concatenation rule, as it is illustrated by example (10):

(8) *di-bakar*
PASS[v]-burn
'be burnt with intention'

(9) *ter-bakar*
PASS-burn
'be burnt accidentally'

(10) **me-bakar| mem-bakar*
ACT-burn
'to burn'

Example (10) illustrates the representation of the underlying form *–N* as /m/. The preference of nasal sound /m/ over other nasal sounds agrees to the initial sound of the word that it inflects (also known as assimilation). This phenomenon can also be referred as an example of context sensitive grammar (*N* → *m*/ *meN_b*). Nasal sounds in Indonesian are orthographically represented as the followings: /m/= m, /n/= n, /ŋ/= ng or /ɳ/= ny. When the word in canonic form is attached by prefix *meN-*, a newly generated POS is commonly verb. A lexicon goes through the following schema (Pisceldo, 2004:5) before it gets prefixed by *meN-*. Consider figure 7:

Figure 7. Lexicon Path

Lexicon → Morphotactic Rules → Morphophonemic Rule → Surface representation

The above lexicon path illustrates how lexicons are prefixed. Morphotactic rules of Indonesian notes that prefixing of *me-*[vi] to the stem results on a transitive verb. But before it is represented on the surface, the stem words must undergo the following morphophonemic constraints (adapted from Alwi 1998 in Pisceldo 2004)[vii] as it is represented in table 1:

Table 1. Morphophonemic Constraints for Indonesian Prefix *meN-*

	Morphophonemic Process of *meN*			**Orthographical Representation of the initial phoneme of the word[viii]**
	1	2	3	
a	*me* insertion	concatenation	-	*m,l,n,r,w,y,ny,ng*
b	'*meng'* insertion	concatenation	-	all vowels, *g, h*
c	'*mem'* insertion	concatenation	-	*b,f,v*
d	'*men'* insertion	concatenation	-	*d,c,j,z*
e	delete first letter of stem	'*mem'* insertion	concatenation	*p*
f	delete first letter of stem	'*meny'* insertion	concatenation	*s*
g	delete first letter of stem	'*n'* insertion	concatenation	*t*
h	delete first letter of stem	'*meng'* insertion	concatenation	*k*
i	*menge-* insertion	concatenation	-	one syllable word

There are nine types of prefixing listed on table 1, but they can actually be narrowed down into three different categories depending on the canonic forms: [1] Insertion and [2] Substitution (Deletion + Insertion). Type 'a', 'b', 'c' and'd' fall to [1] Insertion category. Type 'e', 'f', 'g', 'h' are must undergo removal of first letter of the word. After that, prefix *meN-* can be inserted. These two processes in [2] might refer to a substitution.

Table 2. Prefixed Forms of *meN-*

Canonic Form	Type	Surface Representation	POS	Gloss
Larang	a	*melarang*	V	to forbid
Gali	b	*menggali*	V	to dig
Bakar	c	*membakar*	V	to burn
Duga	d	*menduga*	V	to guess
Peras	e	*memeras*	V	to squeeze
Sapu	f	*menyapu*	V	to sweep
Tulis	g	*menulis*	V	to write
Kemas	h	*mengemas*	V	to pack
bor	i	*mengebor*	V	to drill

The constraints are not always clear cut. There are some irregularities, which involves other linguistic aspects too, like semantic. As an example, the canonic forms *kaji* might have two different inflected forms. The first one *mengaji* follows the prefixing rule by deleting the first letter *k.* Another one, *mengkaji* maintains the *k.* Alwi et al (1988) believes that these two forms are used for meaning distinction. *Mengaji* is often associated with the activity of reading *quran* (the holy book for muslims), while *mengkaji* is associated to the activity of examining something thoroughly.

Some words might undergo POS shift when they are prefixed. As an example, *bor* is a noun, but when this word is prefixed with *meN-*, the POS shifts to a verb. But in some cases, we might also find the use of bare *bor* as a verb. In this case, the word does not undergo POS shift. Consider the examples (11) to (13):

(11) *Bor itu sudah rusak* (N)
(12) *Kami akan mengebor tanah* (V)
(13) *Bor tanah itu!* (V)

Bor in example 12 can easily be identified as a verb, since *bor* is already prefixed with *meN-*. But for examples 11 and 13, the identification of *bor* as two different POS requires larger context interpretation and hence, demands the word *bor* to be written as both noun and verb in the machine readable dictionary used for language processing. In other word, they are ambiguous for the computer. In the later section, I will show how LGGs can be used to treat these problems.

4. LOCAL GRAMMAR GRAPHS AND AUTOMATIC EXTRACTION FROM RESEARCH CORPUS

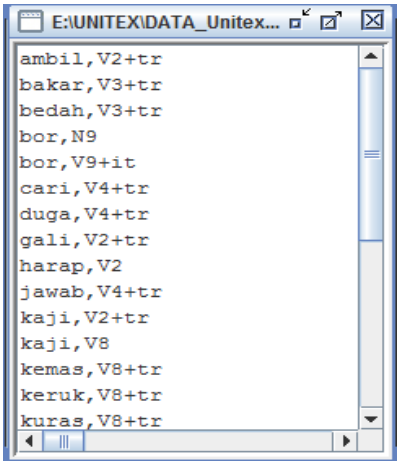
This section describes the procedures undergone for the experiment in this research, and proposes a model for machine readable dictionary building and automatic extraction NLP task for Indonesian language.

Some canonic forms (also known as dictionary form) with different types of *meN-* prefixes were collected randomly. The list was used to build an uninflected word machine readable dictionary. This uninflected word dictionary was then inflected with specifically designed inflection LGGs to comply with constraints pertaining to the morphophonemic phenomena of each *meN-* prefixing types. The inflection, in turns, automatically created an inflected word machine readable dictionary. Then, the existing inflected word dictionary was applied to the text corpus in preprocessing stage for the corpus annotation. After the corpus was annotated, pattern matching and extraction tasks were carried out by either a regular expression, or a graph. These process and the essential components that support it are described in the following sub sections.

4.1 Canonic Form Dictionary

There are 28 canonic forms selected to represent the nine types of morphophonemic constraints described in section 3 of this paper. The words are collected and written in UNITEX uninflected dictionary format (entry line formalism). The format comprises at least a POS and an inflection code. Consider figure 8:

Figure 8. Canonic Form Dictionary



Canonic form dictionary is a very essential linguistic resource for UNITEX before proceeding further to another linguistic resource, which is the inflected form dictionary. The inflected form dictionary is not manually written. Instead, it is automatically generated by inflection LGGs. The format is different from canonic form dictionary. The canonic form dictionary format is simpler. Main entry is written with lowercases. The uppercase indicates POS of each word. Right after the uppercase, there is a number, and a/some code/s. If necessary, some additional grammar or semantic code might be added. However, the most essential component is the inflection number attached after POS. The number is used to call the

correct inflection LGG in inflectional LGGs repository and automatically create inflected words dictionary. The procedures are described in section 4.2.

4.2 Inflectional LGGs and Inflected Words Dictionary

4.2.1 LGGs for Insertion and Substitution

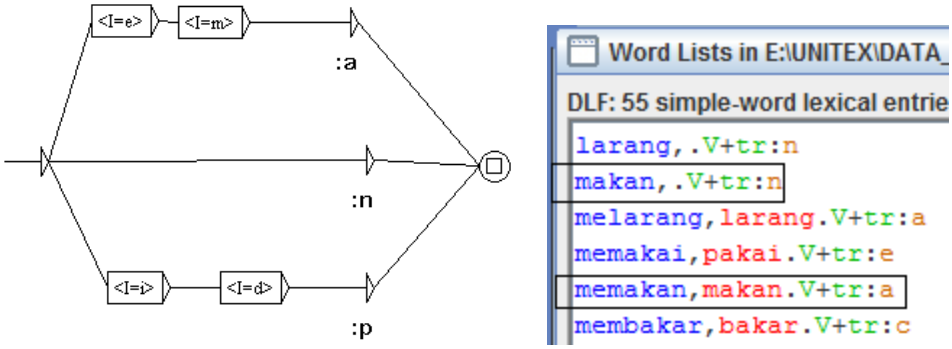
There might be a terminology problem considering the term, 'inflectional' LGGs as in morphology the term 'inflection' and 'derivation' is distinguished. However, in UNITEX, all affixes are handled by inflectional LGGs, regardless of the resulting word forms as an inflection or derivation.

As commented in the section three, there are two categories of prefixing. One is insertion; another one is substitution, which involves deletion. In inflectional LGGs, UNITEX use the following codes for the two constraints:

<I=?^{ix}> inserts the letter ? before the initial letter of the original word.
<X=n^{x}> removes the first n letters of the original word.

The codes must be written in the LGG boxes. Consider figure 8 which consists of an LGG on the right, and the resulting inflected form dictionary on the right:

Figure 8. Insertion

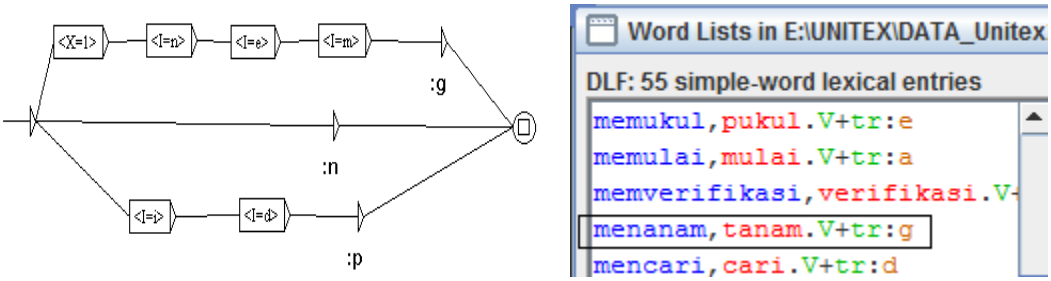


There are three lines for the LGG presented in figure 8. The upper line is used to insert *me* to target word, like *larang* to *melarang* (to forbid), *makan* to *memakan* (to eat), *mulai* to *memulai* (to start). Note that the underlying form *N-* is not represented in the surface form so that the prefix *me-* (as insert) can be concatenated directly to target stem. This kind of prefixing is assigned with *a* code. This code will appear in the inflected word dictionary, by the end of the entry line (Figure 8 right). Since there are nine types of constraints, then eight different inflection LGGs are created (*a* to *i*).

The middle line is used to indicate zero inflection. In a sentence like *makan buah itu!*, without prefix, the word *makan* is identified as a verb. This will give two possibilities in the processing to identify *makan* and *memakan* from the same basic form. This zero prefix is assigned with *n* code (figure 8 right).

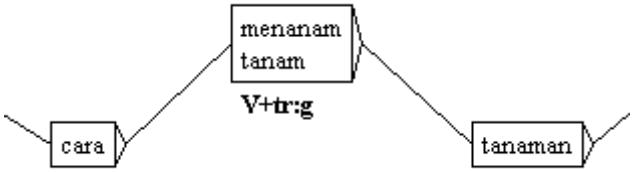
Note that UNITEX allows different type of prefixing in the same graph so that users do not have to write another graph. Consider the lower line in figure x, another type of prefixing, which is passive (PASS). The type of prefixing is an insertion. Consider also figure 9, where insertion and substitution (deletion) can coexist in the same graph.

Figure 9. Zero Prefix, Insertion and Deletion



Unlike the LGG represented in figure 7, the LGG represented in figure 8 comprises of different prefixing operations. The upper line handles a prefixing constraint that includes deletion and insertion. As an example, this LGG inflects *tanam* to *menanam* by removing the first *t* in *tanam*, and insert *men*. This operation applies for stems that require deletion, which are type e, f, g, h. Consider figure 10, which is a text automaton from the sentence containing *menanam*:

Figure 10. Text Automaton for *menanam*



If UNITEX recognizes a particular surface form that comes from basic form (inflected from, uppercased string), it will display both surface and basic form. Surface form is the form that is available on text, and basic form is a form that is available in the dictionary. As an example, figure 10 shows *menanam*, an inflected (surface) form of *tanam*. Below the automaton box, it also displays the codes (POS and inflection codes). Therefore, from the codes we might know that *menanam* is a verb, and it has a basic form *tanam*, and when prefixed with *meN-* it undergoes *t* deletion, and *m* insertion (prefixing constraint type *g*).

4.2.2 One Syllable Word Prefixing

Unlike prefix *di-*, there are some exceptions apply to the canonical forms prefixed by *me-*. One of exceptions is the prefixing to stem with only one syllable. As an example, the canonic form *bor* does not take *mem* insertion even though it begins with *b*. The underlying form *N* is represented as *nge-* instead of *m-*. This rule applies to all one syllable word. Therefore, the grammatically correct form is *mengebor* instead of **membor*. It cannot take same LGG applied to stems with two syllables (or more) like *bagi, bayar, beri* etc. This kind of stems must be classified to a distinct type from the others, and in turn, the prefixing is operated with different LGG, as it is represented in upper line, figure 11. Figure x shows the text automaton for *bor* where it is inflected as *mengebor*, followed by the text automaton in figure 12:

Figure 11. One syllable Prefixing LGG

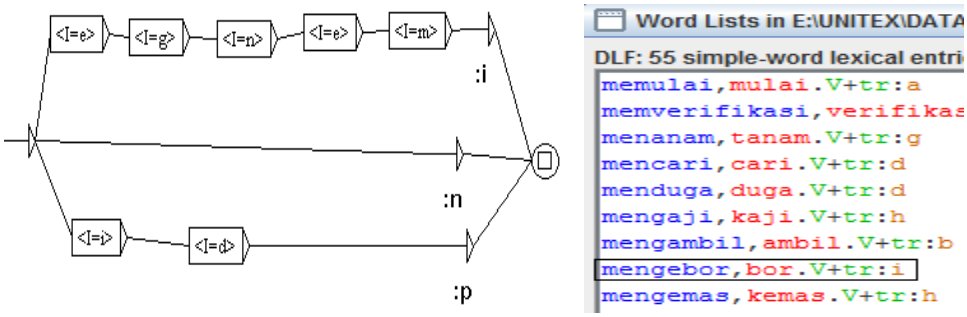
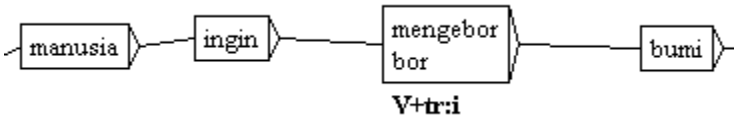


Figure 12. Text Automaton for *mengebor*



4.2.3 Different Prefixing Methods for one Canonic Form

One uninflected form can sometimes be prefixed with more than one LGGs since the stems might potentially take different prefixes. For example, when *meN-* is used to inflect *kaji*. There might be two correct forms: *mengaji* (deletion applies) and *mengkaji* (deletion does not apply). The first prefixed form (*k* deleted form) refers specifically to the activity of reading *quran* (the holy book for Muslims), while the second prefixed form refers to the activity of examining something thoroughly. Therefore, in the machine readable dictionary applied for the corpus processing in this research, these two forms are generated via the automatic prefixing of two different LGGs. Consider LGGs represented in figure 13, and the inflected form dictionary for *mengaji* and *mengkaji* in figure 14:

Figure 13. LGG with Deletion (left) and without Deletion (Right)

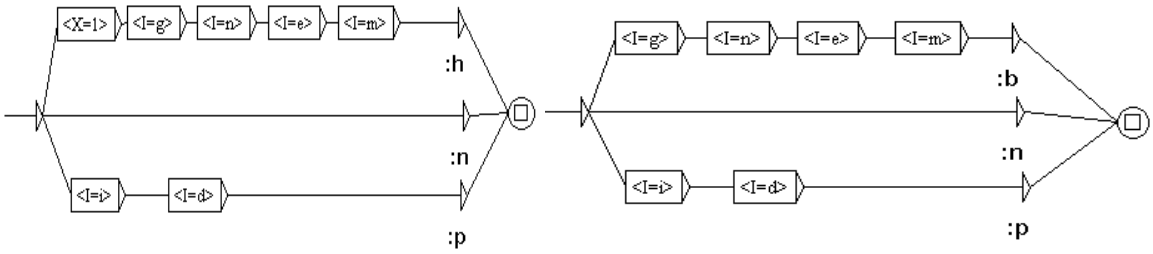
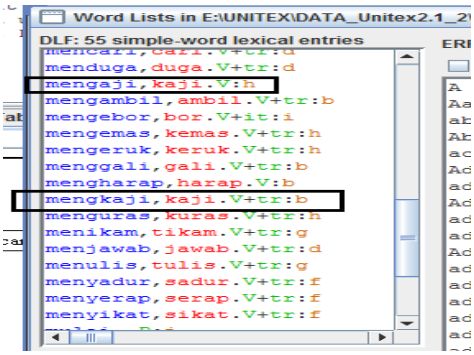


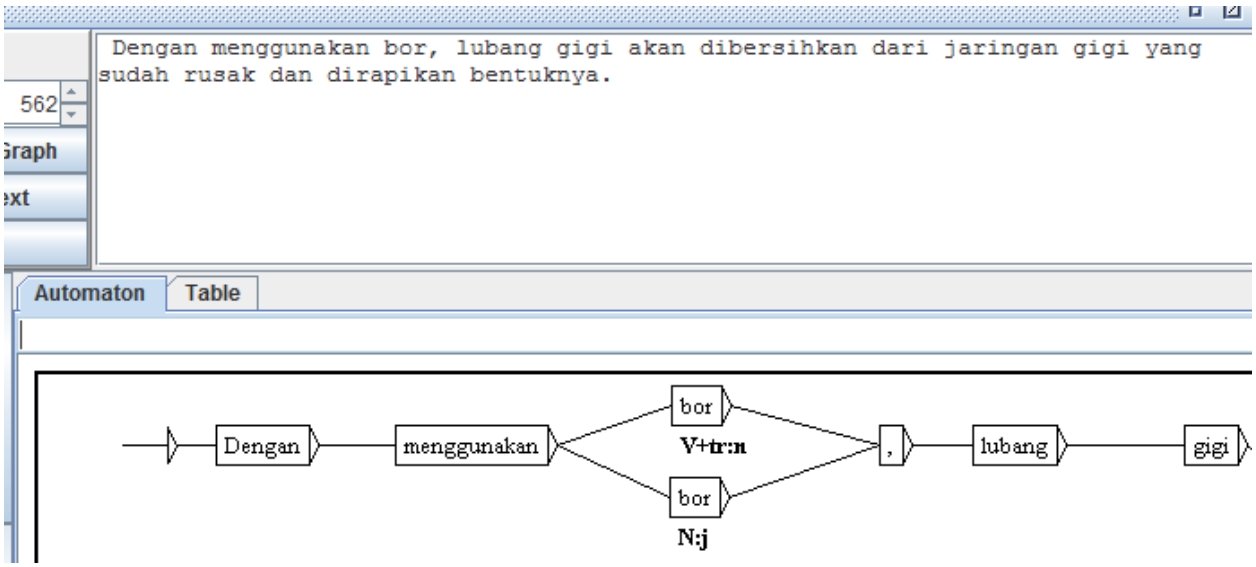
Figure 14. *mengaji* and *mengkaji* in the Inflected word Dictionary



4.2.4 Different POS for one Canonic Form

One word with different POS should be tagged with different tags. In UNITEX machine readable dictionary, this kind of word must be written by multiple entries comply to the POS tags. For example, *mulai* has two different POS tag, one as a verb as in *pertemuan itu dimulai dengan pemukulan gong* or as a preposition *mulai sekarang, kita adalah teman.* The text automaton resulted from the preprocessing can display all possible tags (ambiguity), as it is represented in figure 15 for *bor*:

Figure 15. Ambiguity



In the text automaton represented in figure 15, the word *bor* has two different POS tags: one as a verb, another as a noun. This is because the use of *bor* (without any prefix) can possibly be analyzed as verb as in the following structure '*bor gigi itu!*'. Within that syntactic structure, *bor* is analyzed as a verb instead of a noun. With prefix *meN-*, *bor* is commonly identified as a verb. However, it might also be analyzed as a noun in a sentence like *mengebor adalah kegiatan yang membosankan*. Here, we notice that even though the stem is prefixed with *meN-*, the prefixed form does not need necessarily to be a verb. In other words, the morphological context itself is not enough. Syntactical context is also required to perform an accurate processing. Still, the text automation is quite expressive in displaying ambiguity[xi].

4.3 Research Corpus, Pattern Matching and Automatic Extraction

Previous sections have discussed types of LGGs used in this research. These LGGs are used to inflect a canonic form dictionary to generate an inflected form dictionary. When this inflected form dictionary is ready, it must be applied to a corpus via preprocessing for the corpus annotation. Figure 16 shows the text automaton where the corpus is already preprocessed. The corpus is collected from various websites; but the contents can generally be divided into two. One is from newspaper websites[xii], another is from blogs[xiii]. Note that the stylistic of newspaper websites is commonly formal, while the stylistic of blogs may vary.

The corpus is preprocessed with both simple and inflected form dictionaries. The inflected forms dictionary is automatically generated by the LGGs described in the preceding sections. Consider the preprocessing result represented in figure 16:

Figure 16. Preprocessed Text

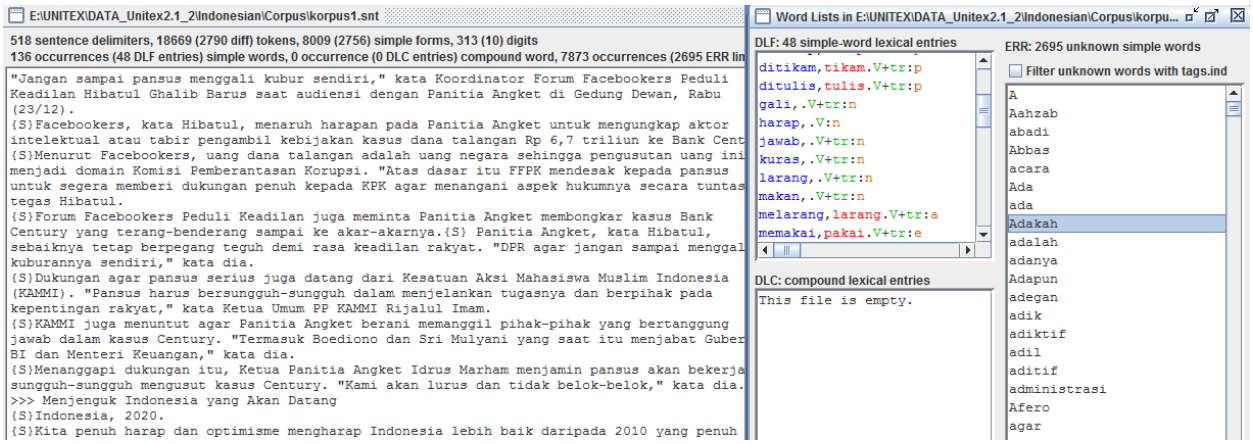


Figure 16 is divided into two parts. The left part displays a corpus that has been preprocessed with simple and inflected form dictionaries. It also shows the statistics generated from UNITEX computation. It detects 518 sentence delimiters, 22739 tokens and 3296 types.

The right part displays a word list from tokens recognized by the dictionaries. It also display a list of unknown words; words that are not in the dictionary. If the dictionary is accurate enough, the list of unknown words usually contains proper names.

This paper focus on extraction of target strings. Target strings may cover a single word, or multiword expressions (phrase). There are three type of extractions described in this paper; word form-based extraction and code based extraction (performed with regular expressions) to extract words, and complex extraction (performed with graphs) to extract multi word expressions.

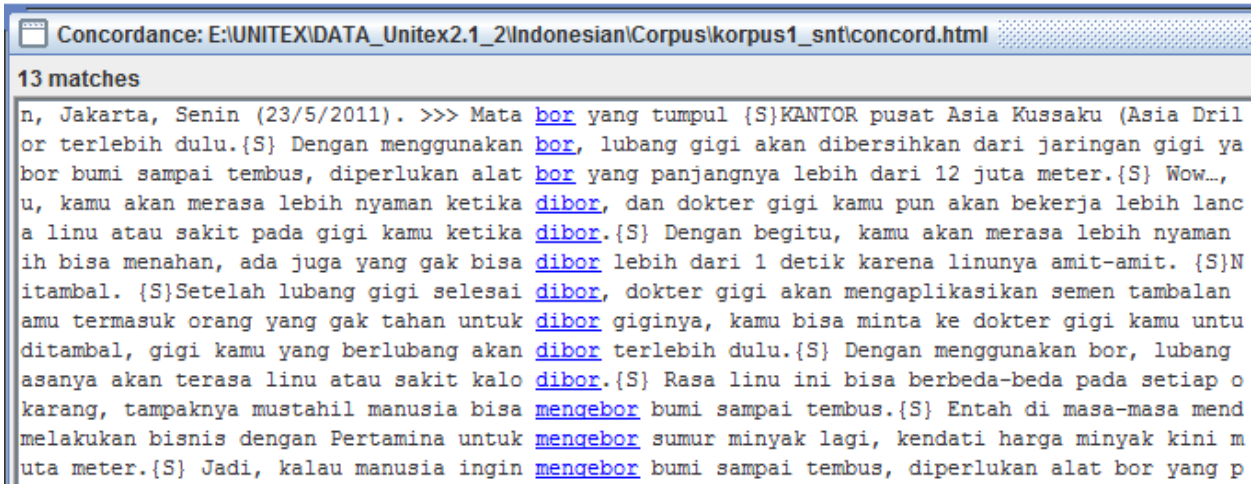
4.3.1 Word Form Based Extraction (Regular Expressions)

The search function in UNITEX is carried out by pattern matching operation. Word form search intends on retrieving and extracting both basic forms and inflected forms. This operation allows users to perform one search only for all the target strings. As an illustration, for the word *ambil*, if users want to

retrieve *ambil* and its variations like *ambil, diambil, mengambil*, most likely they have to perform three queries (for each word forms, excluding the canonic form).

With word forms search, the users need only one time query for the whole searching. This operation retrieves the preprocessed corpus, matching the target query and its inflected forms. In UNITEX, the regular expression for this operation is angel bracketing on the query: <query>. When performing search with graph, the syntax must be expressed in the same way as in the regular expression. By employing this operation, *bor, dibor,* and *mengebor* can be retrieved and extracted by only one query: <bor>. Consider the concordance for query <*bor*> in figure 17:

Figure 17.Concordance for Query <bor>



4.3.2 Code-based Extraction (Regular Expression)

The search can also be performed by using codes written in the dictionary (reference). The more complicated code has consequence on tightening the constraints. For example, query <TOKEN> retrieves and extracts all word forms. With this query, all words, regardless of the POS, simple or inflected, are retrieved and extracted. If the constraint is tightened to just verbs, then the query should be <V>. This query is meant for retrieving and extracting all verbs regardless of the word forms (It cane inflected or uninflected forms) as illustrated by figure 18:

Figure 18. Concordance for Query <V>

Figure 18 is the concordance for query <V> which calls all verbs regardless of the word forms. It shows *bor-mengebor-dibor* from the basic form *bor*, *tanam-menanam* from the basic form *tanam*, *mengaji-mengkaji* from basic form *kaji*. This constraint can still be tightened. Query <V:a> retrieves and extracts only verbs that are prefixed with rule *a*, where the underlying form of *N* (of *meN*-) is realized as zero. Consider figure 19:

Figure 19. Concordance for Query <V:a>

udhoyono menyatakan negara tidak bisa melarang warganya untuk memilih berobat ke luar negeri
Saya tidak boleh mengeluarkan Keppres melarang rakyat kita untuk berobat ke luar negeri.{S} T
ar hal besar." >>> Politik Pencitraan Memakan Korban {S}Rabu, 09 Februari 2011 | 07:{S}08 WIB
ik pula.{S} Begitu pun sebaliknya. {S}Memakan korban {S}Setidaknya ada tiga cara dalam memban
nesia harus mengambil inisiatif untuk memulai pembicaraan soal perjanjian ekstradisi dengan S
ntoran dan mahasiswa yang betul-betul memulai dari nol dan kami betul-betul membentuknya," ka
ANTARA News/Reuters) - Barcelona akan memulai perburuan mereka merebut gelar juara La Liga un

Each prefixing type in the machine readable dictionary in this research takes different codes. Hence, users might focus their search only one prefixing types, or a collection of few. So far, pattern matching operation is performed via regular expression. However, regular expression is effective when the query is not beyond word level. When more complex query is requested (multi word), a graph will be more effective for a retrieve and extraction task.

4.3.3 Complex Extraction (Graph)

A query might not just be limited to word forms, but a query also allows for more complex extraction (longer strings) like compounds or phrases. For a complex extraction, the pattern matching operation is much more comfortable to be carried out with an LGG instead of a regular expression. Consider a transducer graph in figure 20, and the result in figure 21.

Figure 20. A Transducer Graph for Pattern Matching and Extraction of Passive Agent

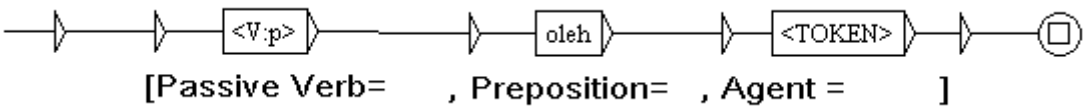


Figure 21. Extraction Result for Passive Agent

Concordance: E:\UNITEX\DATA_Unitex2.1_2\Indonesian\Corpus\korpus1_snt\concord.html

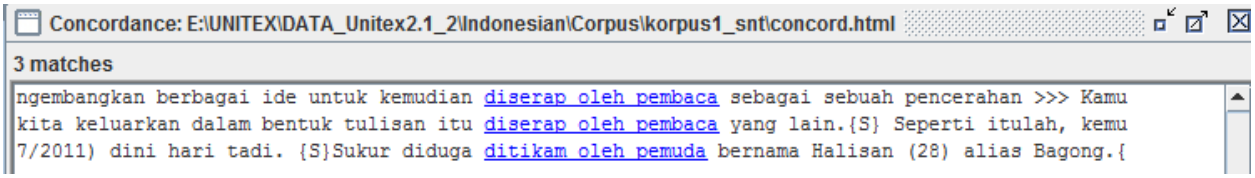
3 matches

ngembangkan berbagai ide untuk kemudian [Passive Verb= diserap, Preposition= oleh, Agent = pembaca]
kita keluarkan dalam bentuk tulisan itu [Passive Verb= diserap, Preposition= oleh, Agent = pembaca]
7/2011) dini hari tadi. {S}Sukur diduga [Passive Verb= ditikam, Preposition= oleh, Agent = pemuda]

The above search result, where output 'Passive Verb', 'Preposition', and 'Agent' are available, made possible with transducer mechanism that can be carried out with LGGs. Search with regular expressions to obtain the above result is not possible. Tags above might not be obtained unless the LGG is a

transducer type. With the following regular expression query (<V:p>oleh<TOKEN>), UNITEX will extract only target strings without giving any output as in figure 21. Please consider figure 22:

Figure 22. Extraction with Regular Expression



More advanced user might want to use left context operation. This operation order the left part of the strings to be the target retrieval context as it is illustrated in figure 24, with LGGs in figure 23:

Figure 23. Using (Left) Context

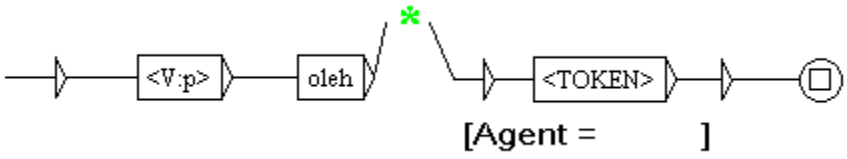
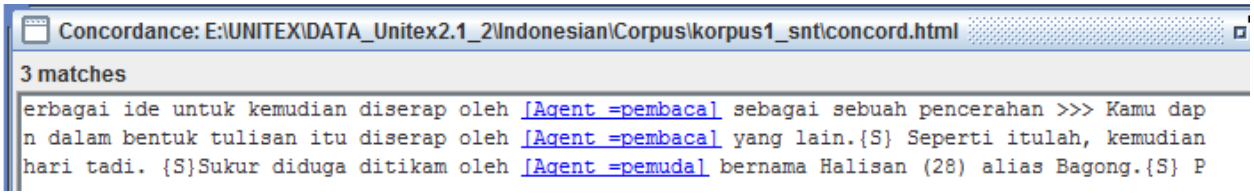


Figure 24. Result Extraction with (Left) Context



The LGG represented in figure 23 indicates that the target query is only the agent. To make the pattern matching accurate, context is included. The target query must be preceded by two tokens: the first one is verb in passive form, and another one is preposition *oleh*. From the LGG, the result in figure 24 is obtained. See the difference with figure 21 and 22, where context is not established. In the previous result, the passive verbs and the preposition are included in the result, where in the search with context, they are not included in the research.

5. Conclusion

By focusing on some particular type of stems, I have described my proposal for the automatic retrieval and extraction of *meN-* prefixed word forms. Morphophonemic constraints of prefix *meN-* have been formalized with LGGs so that it can further be used to carrying various NLP tasks. I have demonstrated how the linguistic resources proposed in this research can be used to perform pattern matching and extraction tasks.

There are three important linguistic resources proposed in this research: the uninflected word dictionary, the inflected word dictionary, and the inflection LGGs, which are all dedicated to

Indonesian language. These resources are maintainable and open for further development. But the most important future work is building a complete uninflected word dictionary for Indonesian. Only by having this resource the subsequent resources can effectively be created and employed, and in turn can perform a sufficient corpus annotation.

Another important work to do is to standardize (and widely publish) the morphophonemic constraints such as for reduplications, allowed affixes (*meN-guna-kan* and *meN-guna-*i*), double prefixes (*menggunakan-mempergunakan-mem*ergunakan*) [xiv] , consonant cluster (*meN-transfer* realized as *mentransfer* instead of *men*ransfer*), and other irregular morphophonemic constraints. If they are already standardized, then LGGs can be used to formalize the constraints, and in turn, an inflected form dictionary can be created and used to perform various NLP tasks in Indonesian corpus.

References

Amas, S. (2009). Bentuk Memperhatikan dan Memperkosa. *Jurnal sosioteknologi : membaca arah pembangunan Indonesia melalui sejarah bangsa* : 8 (18): 745 - 747

Alwi, H- D, et al. (1998). *Tata Bahasa Baku Bahasa Indonesia*. Balai Pustaka: Jakarta

Chomsky, N (1956). Three models for the description of language. *IRE Transactions on Information Theory (2)*: 113–124. http://www.chomsky.info/articles/195609--.pdf.

Departemen Pendidikan Nasional, (2008). *Kamus Besar Bahasa Indonesia*. Edisi Keempat. Jakarta : PT Gramedia Pustaka Utama.

Chomsky, N, and M. Halle. 1968. *The Sound Pattern of English*. New York: Harper and Row. xiv, 470 pages. Reprinted 1991, Boston: MIT Press.

Ermanto.(2008). Fungsi dan Makna Afiks Infleksi pada Verba Afiksasi Bahasa Indonesia: Tinjauan dari Perspektif Morfologi Derivasi dan Infleksi.*Bahasa dan Seni, Jilid 36, No 1:Fakultas Sastra Universitas Negeri Malang.*

Gross, M. (1993). *Local Grammars and their Representation by Finite Automata*. In Data, Description, Discourse. Papers on the English Language in honour of John McH Sinclair, M. Hoey (Ed.) (1993) 26-38.

Lewis, M.P. (ed.). 2009. *Ethnologue : Languages of the World*. 16th edition. Dallas : SIL International. Online version at http://www.ethnologue.com/.

_____. (1968). *Grammaire transformationnelle du francais*., vol. 1, Syntaxe du verbe. Larousse.

Martin-Vide.C. (2004). Formal Grammar and Language, in *The Oxford Handbook of Computational Linguistics*. Mitkov, R (Ed) (2004): 157-177.

Max, S. (1993). *Dictionnaires électroniques et analyse automatique de textes : le système INTEX*. Masson Ed.: Paris Press: Oxford.

Paumier, S. (1998). *Unitex Manual 2.1*. Université Paris-Est Marne-la-Vallée: Paris

Pischeldo, F., et al. *A Two-Level Morphological Analyser for the Indonesian Language*. In Proceedings of the 2008 Australasian Language Technology Association Workshop (ALTA 2008), pages 142– 150. Hobart, Australia

Throst, Harold. (2004). Morphology, in *The Oxford Handbook of Computational Linguistics*. Mitkov, R (Ed) (2004): 25-48.

Endnotes

[i] Abbreviations used in this paper: LG (Local Grammar), LGG (Local Grammar Graph), NLP (Natural Language Processing), POS (Part of Speech), NP (Noun Phrase), PP (Prepositional Phrase)
[ii] *N* refers to nasal sounds, which might be represented as /m/, /n/, /ɲ/ or /ŋ/
[iii] http://igm.univ-mlv.fr/~unitex/ (no module is available yet for Indonesian Language)
[iv] Nonsense string
[v] Glosses taken from IMT (interlinear Morpheme Translation): PASS (Passive), ACT (Active)
[vi] For computer processing, abstract underlying form of nasal (-*N*) is removed. Thuse, basic prefix is considered to be *me-*. This will not affect the prefixing results. The generated prefixed words will still be in correct forms.
[vii] Alwi et al (1988) noted that as Indonesian language keeps improving complying to the language use in social life, changes concerning the rules may apply.
[viii] ng = /ŋ/, ny= /ɲ/
[ix] ? signifies any letters
[x] n signifies the order of the letter to be removed
[xi] To solve this problem, one must apply disambiguation grammar.
[xii] News/Newspaper websites: Kompas (www.kompas.com) , Seputar Indonesia (www.seputar-indonesia.com) , Tempo (www.tampo.co) , Antara (www.antaranews.com)
[xiii] Blogs: www.blogspot.com, www.wordpress.com, www.kompasiana.com
[xiv] In KBBI (Indonesian Standard Dictionary), there are two circumfixed (*me-kan*) form of *perkara*, which is *memperkarakan* (as a sub entry of *perkara*), *memerkarakan* (as a synonym of *mengacarakan*, sub entry of *acara*). If it is assumed that *p* is deleted only when it is the initial phoneme of a canonic word, then *memerkarakan* should be the correct form instead of *memperkarakan*. This conforms to *mempergunakan*, since the canonic form is *guna*, and prefixed by *per* (deletion does not apply). See Amas (2009)