Towards a Semantic Analysis of Bahasa Indonesia for Question Answering

Septina Dian Larasati and Ruli Manurung
Faculty of Computer Science
University of Indonesia
Depok 16424, Indonesia
septina.dian@ui.edu, maruli@cs.ui.ac.id

Abstract

This paper presents a model of deep syntactic and semantic processing to support question-answering for Bahasa Indonesia. Starting from an existing unification-based grammar, we specify lexical semantics for each lexeme and semantic attachment rules for each grammar rule using lambda-calculus notation. This approach enables us to obtain semantic representations of Indonesian sentences in the form of first order logic literals. These representations are used by a question-answering module which stores declarative sentences as facts in a knowledge-base, and answers queries by unifying the question representation with known facts. We discuss an implemented prototype of this model using Prolog, and conclude with the remaining work that has to be done to realise our goal of a fully-fledged question-answering system for Bahasa Indonesia.

1 Background

In this section we present some information on the relevant issues to our research: question answering, specifically for Bahasa Indonesia, and existing semantic approaches. In Section 2 we give an overview of our system, before presenting all the details of the semantic analysis and representation of Indonesian sentences in Section 3. A Prolog implementation of this approach is discussed in Section 4, before we conclude by suggesting further avenues of work.

1.1 Semantic approaches to question answering

Question Answering, or QA, is a subtask of the more general Information Retrieval (IR) task. A QA system seeks to provide answers to questions expressed in natural language, where the answers are to be found in a given collection of documents. QA systems typically require more sophisticated linguistic analysis than conventional IR, as they need to reason about the types of questions, predicate argument structure, result aggregation, etc.

Most QA systems employ a pipeline composed of the following stages: first, the question is syntactically analysed to determine its type, and the type of the expected answer. Second, "shallow" statistical methods derived from the IR field, e.g. keyword-based retrieval, are used to retrieve a subset of documents that may potentially contain the answer. Finally, a finer-grained module processes this subset by focusing on specific passages and extracting candidate answers. QA systems often exploit data redundancy, where correct answers are more likely to appear multiple times in the corpus than false positives.

More recently, work has been done in developing QA systems that obtain semantic representations of both the query and (a subset of) the documents, e.g. Narayanan and Harabagiu (2004). The resulting logical forms abstract away purely syntactic variations from facts and queries, enabling more precise identification of answers to questions. They also capture semantic functions, which simple statistical systems that treat texts as bags-of-words fail to account for, e.g. the difference between *"USA invades Iraq"* and *"Iraq invades the USA"*. Additionally, semantic representations enable logical inference, allowing the QA systems to answer more complex queries by exploiting knowledge encoded in ontologies such as WordNet. Such an approach is exemplified by recent work in the textual entailment task, e.g. Bos and Markert (2006).

1.2 Question answering in bahasa Indonesia

Bahasa Indonesia (hereinafter simply 'Indonesian') is the official language of Indonesia, spoken

Proceedings of the 10th Conference of the Pacific Association for Computational Linguistics, pages 273–280.

by over 100 million people. Given this fact, we believe it is underrepresented in terms of research into Indonesian QA, and Indonesian NLP in general.

There has been some work on developing QA systems for Indonesian, e.g. Wijono et al. (2006); Purwarianti et al. (2007). To our knowledge, all previous systems adopt a purely statistical method.

Wijono et al. (2006) sought to achieve multilingual QA by answering queries in Indonesian based on English documents. Questions are classified based on a manually constructed taxonomy of Indonesian questions. The query is then automatically translated into English using a commercial translator available online[1], and from then on is handled as a purely English QA task.

Purwarianti et al. (2007) uses a machine learning method to develop the question and answer classifier modules based on a corpus of raw text.

In this paper, we propose a model of an Indonesian QA system that adopts a deeper linguistic approach, leveraging a previously built syntactic parser for the Indonesian language (Joice, 2002).

2 Framework

The overall framework of our system consists of the following modules: a syntactic parser, a semantic analyser, and a question-answering module. We describe these three modules in the following subsections.

2.1 Syntactic parser

Our starting point was a computational account of Indonesian declarative sentences described in Joice (2002), which in turn is based on the official prescriptive grammar of formal Indonesian presented in Alwi et al. (1998). This grammar was developed using PC-PATR (McConnel, 1995), an implementation of the PATR-II formalism (Shieber, 1986).

Unfortunately, although it superficially used a constraint-based formalism, the grammar failed to elegantly capture regularities in Indonesian. For instance, various selectional restrictions were stipulated by explicitly enumerating all possible combinations of subcategories between, say, transitive verbs and their objects. Additionally, certain constructions best handled by recursive rules, e.g. adjectival phrases, were simulated up to a finite depth by explicit enumeration. This resulted in an explosion of rules once the grammar was transformed into a definite clause grammar, or DCG[2], to be used with our system (see Sect. 4).

Another drawback was that the system had a trivial account of morphology: the parser required a lexicon containing fully inflected words. Given the rich derivational morphology in Indonesian, this resulted in a very unwieldy dictionary.

Nevertheless, the grammar exhibited fairly wide coverage – see Joice (2002) for results of an experiment conducted with a corpus of sentences obtained from a national Indonesian newspaper.

To support our prototype question-answering application, we augmented the existing grammar with rules to account for interrogative sentences. More specifically, we handle both *in-situ* and *ex-situ* variations of wh-questions using *apa* (what), *siapa* (who), *mana* (where), *mengapa* (why), *kapan* (when), and *berapa* (how many/much).

2.2 Semantic analyser

The core of our research is concerned with the development of this module, i.e. a module that transduces semantic representations from parse trees. The semantic representations are designed to abstract away syntactic variations, allowing sophisticated automated processing of Indonesian texts.

We adopt a 'flat' semantic representation (Hobbs, 1985), details of which are presented in Sect. 3.1. Adopting the well-known rule-to-rule hypothesis (Bach, 1976), we decided to augment the lexicon with semantic information (Sect. 3.2), and develop semantic attachment rules for each grammar rule (Sect. 3.3).

Such a deeply handcrafted approach is at odds with the predominant approach in Information Retrieval of corpus-based statistical methods, which for much of the last decade has proved to be more successful at scaling up to the large resources available. However, we are not aware of similar previous work being done for Indonesian, and thus believe that our work establishes a foundation that will be of value for future research into, among others, Indonesian machine translation, dialogue systems, and natural language generation.

2.3 Question-answering module

Currently, we have implemented a prototype query processor in Prolog. The semantic representations

[1] `http://www.toggletext.com`

[2] a notation for CFGs with variable unification, commonly implemented in most Prolog systems

```
<expression>   :=  <literal>
<expression>   :=  <literal> ∧ <expression>
   <literal>   :=  <predicate>(<term>,<term>)
      <term>   :=  <variable>|<concept>
  <variable>   :=  [A-Z][a-z|A-Z|0-9]+
   <concept>   :=  [a-z][a-z|A-Z|0-9]+
 <predicate>   :=  [a-z][a-z|A-Z|0-9]+
```

Figure 1: BNF syntax of logical expressions

of Indonesian declarative sentences, i.e. as found within a collection of documents, are stored in a clausal knowledge base. Subsequently, the semantic representation of queries are transformed into Prolog rules which, when unified with the clause database, yields the appropriate answer. See Sect. 4 for details on this implementation.

3 Semantic representation

In this section we present all the details concerning the semantic representations of Indonesian sentences, i.e. the syntax of logical expressions, the content of lexical semantics, and how the semantic attachment rules are defined and applied. Our approach is reminiscent of Alshawi and van Eijck (1989).

3.1 Logical expressions

As mentioned in Sect. 2.2, we adopt a simple 'flat' semantic representation (Hobbs, 1985), where a logical expression is a conjunction of first order logic literals. The arguments of these literals represent domain concepts such as objects and events, while the functors state relations between these concepts. All variables are existentially quantified with the widest possible scope. The BNF syntax corresponding to our representation is given in Figure 1.

Additionally, following the approach in Durme et al. (2003), literals are divided into two categories, extrinsic and intrinsic literals. An **extrinsic literal** defines a relationship between two variables, whereas an **intrinsic literal** defines a relationship between a variable and its referent as being some semantic concept in some underlying ontology.

Table 1 shows a partial list of the intrinsic literals, whereas Table 2 shows a partial list of the extrinsic literals. Both types of literals are stored within the lexical semantics entries of the words that convey their meaning, which specify the Y variable (see Section 3.2).

Our semantic representation falls into the category of so called neo-Davidsonian approaches, where intrinsic literals are predicates over objects and events, and arguments and modifiers are specified via the thematic relations specified by the extrinsic literals.

$\lambda X time(X,Y)$	X is temporal object Y
$\lambda X location(X,Y)$	X is location object Y
$\lambda X object(X,Y)$	X is inanimate object Y
$\lambda X person(X,Y)$	X is animate object Y
$\lambda X event(X,Y)$	X is event "object" Y
$\lambda X property(X,Y)$	X is property "object" Y
$\lambda X quantity(X,Y)$	X is cardinal number Y
$\lambda X grade(X,Y)$	X is ordinal number Y
$\lambda X unit(X,Y)$	X is quantity unit Y

Table 1: List of some intrinsic literals

$\lambda X \lambda Y\ agent(X,Y)$	X is the agent of Y
$\lambda X \lambda Y\ patient(X,Y)$	X is the patient of Y
$\lambda X \lambda Y\ theme(X,Y)$	X is the theme of Y
$\lambda X \lambda Y\ attrib(X,Y)$	X is an attribute of Y
$\lambda X \lambda Y\ possess(X,Y)$	X possesses Y
$\lambda X \lambda Y\ purpose(X,Y)$	X is the purpose Y
$\lambda X \lambda Y\ cause(X,Y)$	X causes Y
$\lambda X \lambda Y\ manner(X,Y)$	X is a manner of Y
$\lambda X \lambda Y\ instrument(X,Y)$	X is an instrument of Y
$\lambda X \lambda Y\ condition(X,Y)$	X is a condition Y
$\lambda X \lambda Y\ result(X,Y)$	X is a result of Y
$\lambda X \lambda Y\ complement(X,Y)$	X is a complement of Y

Table 2: List of some extrinsic literals

3.2 Lexical semantics

Lexical entries of open class words are associated with exactly one intrinsic literal which asserts a reference to the domain concept the word is "about". The domain concepts can belong to any underlying ontology, e.g. WordNet (Fellbaum, 1998).

For example, given the WordNet verb synset labelled with the ID `s201132466`, whose gloss is "take in solid food", we can specify the following lexical semantics entries for *"memakan"* and *"menyantap"*, both Indonesian synonyms for the verb *"eat"*:

- *"memakan"*: $\lambda E \lambda A \lambda P\ \ event(E,\texttt{s201132466}) \wedge agent(E,A) \wedge patient(E,P)$
- *"menyantap"*: $\lambda E \lambda A \lambda P\ \ event(E,\texttt{s201132466}) \wedge agent(E,A) \wedge patient(E,P)$

Unfortunately, a mapping from WordNet synsets to an Indonesian lexicon does not currently exist – an ongoing project of ours aims at addressing this issue. Until then, we arbitrarily choose the root form of a synonym to act as the conceptual

symbol. For instance, the ID `s201132466` above is replaced with the token *makan*.

Note that the verbs *"memakan"* and *"menyantap"* above also specify the extrinsic literals $agent(E,A)$ and $patient(E,P)$. These are thematic relations that must be specified by complements within its syntactic projection. The variables will be subsequently bound with the variables of the subject and object through the lambda calculus operation of β-reduction (see Section 3.3 below).

In Alwi et al. (1998), there are several subcategories of nominals, e.g. temporal, location, object, person, etc. The lexical semantics of nominals is simply the appropriate intrinsic literal, e.g. the semantics of *"dapur"* (kitchen) is $\lambda X \ \ location(X,dapur)$ and the semantics of *"ayah"* (father) is $\lambda X \ \ person(X,ayah)$.

Adjunct modifiers such as adjectives and adverbials are associated with a logical expression containing the appropriate intrinsic literal coupled with an extrinsic literal that specifies the thematic relation between the modifier and its head. For example, the semantics of *"indah"* (beautiful) is $\lambda A \lambda T \ \ property(A,indah) \wedge attrib(T,A)$.

The lexical semantics of prepositions and words which coordinate and/or subordinate other clauses is simply the appropriate extrinsic literal which specifies the relation between the prepositional phrase and its head or the clauses being coordinated. For example, the semantics of *"karena"* (because) is $\lambda X \lambda Y \ \ cause(X,Y)$.

Question words, i.e. *wh* words in Indonesian, e.g. *"apa"* (what), *"siapa"* (who), *"mana"* (mana), are associated with a logical expression that contains two literals. The first is the appropriate literal which would typically be associated with the answer, but instead of specifying the domain concept as the second argument, it is given a variable *Ans*. The second literal is a special $query(Ans)$ literal that indicates a question that is to be processed by the question-answering module (see Section 4 for details). For example the lexical semantics of *"siapa"* is $\lambda X \ \ person(X,Ans) \wedge query(Ans)$.

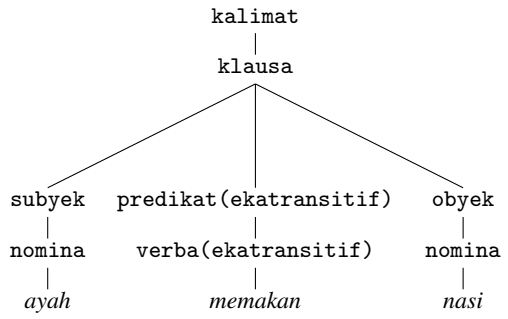
Finally, there are several special cases of lexical semantics where morphological processes introduce literals. For example, the suffix *"-nya"* amounts to a possessive pronoun, requiring the addition of the literals $person(O,owner)$ and $possess(O,X)$ to the lexical semantics. For example, we assume that the lexical semantics for the word *"bukunya"* (his/her book) is $\lambda X \ \ object(X,buku) \wedge person(O,owner) \wedge possess(O,X)$.

3.3 Semantic attachment rules

Montague's principle of compositionality of semantics states that the meaning of a complex expression is determined by the meanings of its parts, and the way in which those parts are combined. However, it says nothing about how exactly these meanings can be constructed. For this, we turn to the well-known rule-to-rule hypothesis (Bach, 1976), which states that semantic interpretation rules are defined on the syntactic rules and structures. As a result, we develop *semantic attachment rules* for each syntactic rule in our grammar.

These semantic attachment rules define how the lexical semantics of the constituent words are combined, and in particular how the correct predicate-argument structure is specified. The most common approach is to use *lambda calculus* notation, where predicate-argument structure is controlled through the operation of β-reduction. See Jurafsky and Martin (2000) for a clear discussion of this approach (note that they call the process lambda-reduction).

To see an example of the semantic attachment rules and how they are combined, observe the following example, which constructs the semantic representation of the simple declarative sentence *"Ayah memakan nasi"* (father eats rice).



The sentence is parsed by our grammar (Sect. 2.1) as follows: a `kalimat` (sentence) can consist of a single `klausa` (clause), which in turn expands to a transitive verbal predication. The `subyek` and `obyek` categories simply consist of a single `nomina` lexeme, whereas a transitive `predikat` consist of a single transitive `verba` lexeme (note the `ekatransitif` subcategorization value). Rules (1)-(4) below show the required syn-

tax rules and corresponding semantic attachment rules, whereas rules (5)-(7) show the lexical semantics entries (see Section 3.2 for discussion of these values):

1. `klausa -> subyek, predikat(ekatransitif), obyek`
   $\lambda M\ \ predikat.sem(M)(K)(N) \wedge subyek.sem(K) \wedge obyek.sem(N)$

2. `subyek -> nomina`
   $\lambda X\ \ nomina.sem(X)$

3. `predikat(ekatransitif) -> verba(ekatransitif)`
   $\lambda Y\ \ verba.sem(Y)$

4. `obyek -> nomina`
   $\lambda Z\ \ nomina.sem(Z)$

5. `nomina -> [ayah]`
   $\lambda D\ \ person(D,ayah)$

6. `verba(ekatransitif) -> [memakan]`
   $\lambda E \lambda A \lambda P\ \ event(E,makan) \wedge agent(E,A) \wedge patient(E,P)$

7. `nomina -> [nasi]`
   $\lambda S\ \ object(S,nasi)$

The *.sem* operator indicates the logical expression of the indicated syntactic category. The β-reduction proceeds as follows:

1. Lexical semantics are copied over to the `subyek`, `predikat`, and `obyek` categories:

   (2) & (5): `subyek.sem=`
   $\lambda X\ (\lambda D\ person(D,ayah))(X)$
   reduces to
   $\lambda X\ person(X,ayah)$

   (3) & (6): `predikat.sem=`
   $\lambda Y\ (\lambda E \lambda A \lambda P\ \ event(E,makan) \wedge agent(E,A) \wedge patient(E,P))(Y)$
   reduces to
   $\lambda Y \lambda A \lambda P\ \ event(Y,makan) \wedge agent(Y,A) \wedge patient(Y,P)$

   (4) & (7): `obyek.sem=`
   $\lambda Z\ (\lambda S\ object(S,nasi))(Z)$
   reduces to
   $\lambda Z\ object(Z,nasi)$

2. At the `clause` rule, the semantics of `subyek`, `predikat`, and `obyek` are substituted and reduced:

   (1),(2),(3) & (4): `klausa.sem=`
   $\lambda M\ \ (\lambda Y \lambda A \lambda P\ \ event(Y,makan) \wedge agent(Y,A) \wedge patient(Y,P))(M)(K)(N) \wedge (\lambda X\ person(X,ayah))(K) \wedge (\lambda Z\ object(Z,nasi))(N)$
   reduces to
   $\lambda M\ \ \ event(M,makan) \wedge agent(M,K) \wedge patient(M,N) \wedge person(K,ayah) \wedge object(N,nasi)$

The semantic representation of `kalimat`, the sentence, is simply the semantics of the `clause` as shown above.

4 Initial implementation

Our initial implementation is written in Prolog, to exploit its facilities of unification and search. However, a preliminary step was required to translate the rules from Joice (2002), which were written in the PC-PATR format, into a form that could be used by our implementation. To achieve this, we wrote a Perl script that processed a PC-PATR rule and wrote out the equivalent DCG rules. In particular, since PC-PATR rules allow for disjunction and optional constituents, a single PC-PATR rule was translated into a set of DCG rules that enumerated all possible configurations. Due to the very inelegant definition of rules (see Section 2.1), our initial attempt at translating the 61 PC-PATR rules resulted in over seven thousand DCG rules, which contained massive redundancy. At the expense of reduced coverage, we then decided to prune the grammar down to an elementary form by omitting all optional constructions. Further elimination of redundant rules provided us with a "skeletal" grammar of 421 rules.

Each of the syntactic rules from this grammar was augmented with semantic attachment rules (see Section 3.3), and we also constructed a handcrafted lexicon where words were associated with lexical semantics as discussed in Section 3.2.

A prototype Prolog parser with semantic representation building, using the associated attachment rules, was developed to handle our resources. Testing on a set of 496 sentences obtained from a national newspaper revealed that 66 sentences were successfully parsed. This low percentage of coverage was due to the aforementioned grammar pruning and limited lexicon. Manual observation of the parsed sentences showed that the correct semantic representations were being constructed.

The next step was to develop a question-answering module. In general, we assert new facts to the knowledge base by "telling" it Indonesian declarative sentences. This can be repeated for as many sentences as necessary. Finally, we issue a query to the knowledge base by "asking" it an Indonesian interrogative sentence.

A typical evaluation of a question answering system requires a standardized set of documents and a series of question whose answers are to be found within the documents, such as those used by the TREC and CLEF challenges. Unfortunately no such resource currently exists for Indonesian, so we performed a very limited testing of the sys-

tem's ability to extract answers from some artificially created questions. This testing confirmed that our system can currently account for some lexical and syntactic variation, and can correctly provide multiple answers to a question, if warranted by the input texts.

For example, say we assert the following sentence:

Presiden	*mengumumkan*	*susunan*	*kabinet*
The president	announced	roster	cabinet
baru	*kepada*	*wartawan*	*di*
new	to	reporters	at
istana.			
palace			

("The president announced the new cabinet roster to reporters at the palace")

using the following Prolog statement:

```
tell_KB([presiden,mengumumkan,susunan,
         kabinet,baru,kepada,wartawan,
         di,istana]).
```

The knowledge base will assert all the individual literals contained in the resulting semantic representation as follows:

```
event(x1, umum).
agent(x1, x2).
patient(x1, x3).
object(x2, presiden).
object(x3, susunan).
object(x4, kabinet).
nn(x3, x4).
adjective(x5, baru).
attrib(x4, x5).
object(x6, wartawan).
purpose(x1, x6).
object(x7, istana).
di(x1, x7).
```

Note that the predicate argument structure is now represented through atomic Prolog terms `x1`, `x2`, etc. This is because the scope of Prolog variables is only local to the clause it appears in.

Finally, we can query the database the following question:

Siapa	*mengumumkan*	*susunan*	*kabinet*	*baru*	?
Who	announced	roster	cabinet	new	?

("Who announced the new cabinet roster?")

using the following Prolog query:

```
ask_KB([siapa,mengumumkan,susunan,
        kabinet,baru], Ans).
```

This will construct a Prolog `query/1` rule by constructing the semantic representation of the question, and extracting the *query* literal contained by the *wh*-lexeme (see Section 3.2) from the expression and making it the head of the rule.

The resulting rule is as follows:

```
query(Ans) :- event(A, umum),
              agent(A, B),
              patient(A, D),
              object(B, Ans),
              object(D, susunan),
              object(E, kabinet),
              adjective(F, baru),
              attrib(E, F),
              nn(D, E).
```

Executing this query will unify the query variable with the correct result: `presiden`.

The following sentence is a slightly more complex one:

Wartawan	*melaporkan*	*bahwa*	*harga*	*susu*
Journalists	report	that	price	milk
meningkat	*karena*	*pasokan*	*susu*	*impor*
increase	because	supply	milk	import
berkurang	*di*	*Indonesia.*		
decrease	in	Indonesia		

("Journalists have reported an increase in the price of milk due to a decrease in imported milk supplies in Indonesia.")

Given this sentence, our semantic analyser constructs the following semantic representation:

```
event(x1, lapor).
agent(x1, x2).
patient(x1, x3).
profession(x2, wartawan).
complement(x3, x4).
event(x4, naik).
agent(x4, x5).
object(x5, harga).
object(x6, susu).
nn(x5, x6).
event(x7, kurang).
agent(x7, x8).
object(x8, pasokan).
object(x9, susu).
object(x10, impor).
nn(x9, x10).
nn(x8, x9).
location(x11, indonesia).
di(x7, x11).
cause(x4, x7).
```

This enables us to query the database with, for example, the following question:

Pasokan	*apa*	*yang*	*berkurang*	?
Supplies	what	which	decreased	?

("What supplies have decreased?")

```
ask_KB([pasokan,apa_yang,
        berkurang,?], Ans).
```

The resulting rule is as follows:

```
query(Ans) :- event(B, kurang),
              agent(B, C),
              location(C, pasokan),
              object(D, Ans),
              nn(C, D).
```

Executing this query will unify the query variable *Ans* with the correct result: `susu`.

5 Further work & summary

We believe that our implementation forms an important basis for the construction of a question-answering system for Bahasa Indonesia based on deep linguistic analysis. However, several issues must be addressed before this can be realised:

1. The source grammar we started with, Joice (2002), was linguistically unelegant. For our prototype, we extracted a "skeletal" grammar that was a suitable size to work with. Unfortunately, coverage of the grammar was significantly compromised. The next step is to gradually reintroduce various syntactic constructions that were omitted as a result.

2. Our implementation is essentially a proof-of-concept system. It would certainly be naive to expect we could handle a corpus in the order of millions of words by asserting them as logical facts in a Prolog knowledge base. We plan to embed our system within a more conventional statistical QA system, where it would work with a small passage of texts that had previously been filtered through standard IR techniques.

3. Our account of lexical semantics is very naive in that we arbitrarily choose some synonym as a token representing a domain concept. As work on an Indonesian version of WordNet progresses, we hope to gradually update our lexicon.

The system described in this paper can be downloaded from our Information Retrieval Lab webpage at `http://ir.cs.ui.ac.id`.

References

Hiyan Alshawi and Jan van Eijck. 1989. Logical forms in the core language engine. In *Proceedings of the 27th Annual Meeting of the Association for Computational Linguistics*, pages 25–32. Association for Computational Linguistics, Vancouver, Canada.

Hasan Alwi, Soenjono Dardjowidjojo, Hans Lapoliwa, and Anton Moeliono. 1998. *Tata Bahasa Baku Bahasa Indonesia*. Balai Pustaka, Jakarta, Indonesia, third edition.

Emonn Bach. 1976. An extension of classical transformational grammar. In *Problems of Linguistic Metatheory: Proceedings of the 1976 Conference*, pages 183–224. Michigan, USA.

Johan Bos and Katja Markert. 2006. When logical inference helps determining textual entailment (and when it doesn't). In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*. Venice, Italy.

Benjamin Van Durme, Yifen Huang, Anna Kupść, and Eric Nyberg. 2003. Towards light semantic processing for question answering. In *Proceedings of the HLT-NAACL 2003 Workshop on Text Meaning*, pages 54–61. Association for Computational Linguistics, Edmonton, Canada.

Christiane Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database*. MIT Press.

Jerry Hobbs. 1985. Ontological promiscuity. In *Proceedings of the 23rd Annual Meeting of the Association for Computational Linguistics*, pages 61–69. The Association for Computational Linguistics, Chicago, USA.

Joice. 2002. Pengembangan lanjut pengurai struktur kalimat bahasa indonesia yang menggunakan constraint-based formalism. Undergraduate thesis, Faculty of Computer Science, University of Indonesia.

Daniel S. Jurafsky and James H. Martin. 2000. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice-Hall.

Stephen McConnel. 1995. *PC-PATR Reference Manual*. Summer Institute for Linguistics. `http://www.sil.org/pcpatr/manual/pcpatr.html`.

Srini Narayanan and Sanda Harabagiu. 2004. Question answering based on semantic struc-

tures. In *Proceedings of the 20th international conference on Computational Linguistics*.

Ayu Purwarianti, Masatoshi Tsuchiya, and Seiichi Nakagawa. 2007. A machine learning approach for indonesian question answering system. In Vladan Devedzic, editor, *Proceedings of the International Conference on Artificial Intelligence and Applications (AIA 2007)*. IASTED, Innsbruck, Austria.

Stuart Shieber. 1986. An introduction to unification-based approaches to grammar. CSLI Lecture Notes 4, Center for the Study of Language and Information, Stanford, USA.

Sri Hartati Wijono, Indra Budi, Lily Fitria, and Mirna Adriani. 2006. Finding answers to indonesian questions from english documents. In *Working Notes of the Workshop in Cross-Language Evaluation Forum (CLEF)*. Alicante.