ISSN: 2087-1643

MALINDO 2010

PROCEEDINGS

The 4th International MALINDO Workshop

indonesia, August 2nd 2010

Organizer :

UNIVERSITAS INDONESIA

NUS National University of Singapore

MULTIMEDIA UNIVERSITY

 ISSN: 2087-1643

Programme Committee

Chair:

- Mirna Adriani
  Universitas Indonesia, Indonesia

Co-Charis:

- Stéphane Bressan
  National University of Singapore, Singapore
- Bali Ranaivo-Malançon
  Multimedia University, Malaysia

Members:

- Alvin Yeo
  University Malaysia Sarawak, Malaysia
- Cheah Yu-N
  Universiti Sains Malaysia, Malaysia
- Chooi-Ling Goh
  NICT, Japan
- Didier Schwab
  University Mendhs-France, Grenoble, France
- David Gil
  Max Planck Institute, Germany
- Dwi Hendratmo Widyantoro
  Institut Teknologi Bandung, Indonesia
- Gilles Serasset
  University Joseph-Fourier, France
- Hendra Setiawan
  University of Maryland, USA
- Indra Budi
  Universitas Indonesia, Indonesia
- Myrna Laksman
  Universitas Indonesia, Indonesia
- Ng Hwee Tou
  National University of Singapore, Singapore
- Preslav Nakov
  National University of Singapore, Singapore
- Rila Mandala
  Institut Teknologi Bandung, Indonesia
- Ruli Manurung

ISSN: 2087-1643

Universitas Indonesia, Indonesia

- Sakriani Sakti
  NICT, Japan
- Tan Tien Ping
  Universiti Sains Malaysia, Malaysia
- Yves Lepage
  Waseda University, Japan

Local Organizers:

- Ruli Manurung
  Universitas Indonesia, Indonesia
- Indra Budi
  Universitas Indonesia, Indonesia
- Lelya Rimadhiana
  Universitas Indonesia, Indonesia
- Bayu Distiawan Trisedya
  Universitas Indonesia, Indonesia
- Metti Zakaria Wanagiri
  Universitas Indonesia, Indonesia
- Clara Vania
  Universitas Indonesia, Indonesia
- Penny Hutabarat
  Universitas Indonesia, Indonesia
- Zainal Hidayat
  Universitas Indonesia, Indonesia

ISSN: 2087-1643

Programme

08.00 - 08.30	Registration
08.30 - 08.45	Opening Ceremony
08.45 - 09.45	Keynote Speech : Prof Rachel Edita Roxas De La Salle University, Philippines Practical Applications on Human Language Technologies: the Philippine
09.45 - 10.00	Coffee break
10.00 - 12.00	Session I: I Wayan Arka. Dynamic and Stative Passives in Indonesian and their Computational Implementation Angga Kho Meidy and Ruli Manurung. An Initial Indonesian Semantic Analyser that Leverages SUMO Inferential Power Hapnes Toba, Annisa Ihsani, and Ruli Manurung. Discourse Representation Structures for an Indonesian Question Answering System Herry Sujaini, Kuspriyanto, and Arry Ahmad. Indonesian Part of Speech Analysis for Automatic Translation Process Alfan Farizki Wicaksono and Ayu Purwarianti. HMM Based Part-of-Speech Tagger for Bahasa Indonesia Hassan Mohamed, Nazlia Omar, and Mohd Juzaidin Ab Aziz. An Evaluation of Trigram Malay Part-of-Speech Tagger
12.00 - 13.00	Lunch
13.00 - 15.00	Session II: Gatot Wahyudi. Indonesian Named Entity Recognition Using Support Vector Machine Antoine Veillard, Elvina Melissa, Cassandra Theodora, and Stephane Bressan. Learning to Rank Indonesian-English Machine Translations Septina Dian Larasati and Vladislav Kubon. A Study of Indonesian-to-Malaysian MT System Metti Zakaria Wanagiri and Mirna Adriani. Indonesian-English Machine Translation

	Masayu L. Khodra, Dwi H. Widyantoro, E. Aminuddin Aziz, and Bambang R. Trilaksono. Extracting Topic Sentences of Indonesian Expository Paragraphs Indra Budi and Randy Halim. Word Sequences in Automated Essay Scoring for Indonesian Language with LSA method
15.00 - 15.15	**Coffee Break**
15.15 - 17.15	**Session III:** Amalia Zahra and Julie Carson-Berndsen. GMM-Based Identification of Indonesian Speech Sakriani Sakti, Shinsuke Sakai, Ryosuke Isotani, Hisashi Kawai, and Satoshi Nakamura. Quality and Intelligibility Assessment of Indonesian HMM-Based Speech Synthesis System Ronny and Mirna Adriani. Developing Bilingual Indonesian-English Speech Recognition Haris Hasanudin. BAIK Language for Visual Programming with Indonesian Natural Language Yugo Kartono Isal, Belawati H. Widjaja, and Erik Dominikus. Compressing Text in Bahasa Indonesia using Word-Based Block Sorting Natasha and Ruli Manurung. Building an Indonesian News Aggregator using Naive Bayes Classification and Non-Negative Matrix Factorization Clustering Algorithms
17.15 - 17.45	**Discussion & Closing Ceremony**
19.00	**Dinner**

ISSN: 2087-1643

Table of Contents

Discourse Representation Structures for an Indonesian Question Answering System

Hapnes Toba
Faculty of Computer Science
Universitas Indonesia
Depok, Indonesia
hapnes.toba@ui.ac.id

Annisa Ihsani
Faculty of Computer Science
Universitas Indonesia
Depok, Indonesia
nisaihsani@yahoo.com

Ruli Manurung
Faculty of Computer Science
Universitas Indonesia
Depok, Indonesia
maruli@cs.ui.ac.id

Abstract—**This paper presents an initial attempt to develop a semantic analyzer component of a question answering system that handles linguistic information at the discourse level. Using the well-known Discourse Representation Theory as implemented in Blackburn & Bos' systems, linguistic rules for Indonesian are adapted from an existing system and used to construct a Discourse Representation Structure (DRS) of an Indonesian utterance. An ability to do unification-based question answering over the resulting DRS structures is demonstrated.**

Keywords: question answering, discourse representation theory, computational semantics, Indonesian

I. Introduction

A question answering (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 information retrieval, as they need to reason about various other factors, among others the types of questions, predicate argument structure, and result aggregation. One such example of a sophisticated linguistic approach is semantic analysis, which defines precise models for transducing natural language into semantic representations, typically some variant of first order logic [1]. These representations afford the ability to accurately validate answers.

Initial work on a deep linguistic approach for question answering has been reported in [2,3], which uses a unification-based grammar augmented with lambda calculus rules that constructs semantic representations of Indonesian declarative and interrogative sentences. The semantic representations are conjunctions of first order logic literals where all predicate arguments are elaborated as specific roles based on reified event variables [4]. These previous works used simple unification as its only inferential operator, and the knowledge representation scheme did not yet support complex reasoning. In particular, while it supports simple factoid question answering, it has no concept of discourse structure or consistency, which is essential for answering more complex question types.

In this paper we present an initial attempt to extend the work reported in [2,3] to a discourse level using Discourse Representation Theory [5,6]. In Sections 2 we first discuss previous work. Section 3 presents the inference tasks introduced in [1], and the problems faced when applying them to our previous knowledge representation scheme. The basis for the solution, Discourse Representation Structures, are provided in Section 4, and finally in Section 5 we describe our partial solution for question answering over these structures.

II. Previous Work

In previous work [2,3], a system that builds semantic representations of Indonesian sentences was developed using a syntax driven semantic analysis method [7], which uses a unification-based grammar augmented with lambda calculus rules that constructs semantic representations of Indonesian declarative and interrogative sentences.

The semantic expression represents the meaning of a sentence in a logical form, which 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 [3]. Based on the argument types, literals are divided into two categories: intrinsic and extrinsic literals. Intrinsic literals define a relationship between a variable and a concept. An example of an intrinsic literal is **`λx.event(x,concept)`**, which states that **`x`** is an event object of **`concept`**. Extrinsic literals define a relationship, attribute, or modification between two variables. An example of an extrinsic literal is **`λe.λy.agent(e,y)`**, which states that **`y`** is an agent of **`e`** [2].

The system requires various resources, i.e. (i) a grammar and lexicon, written in Prolog DCG rules, (ii) lexical semantics for each item, e.g. mapping the noun *nasi* (rice) to the lambda expression **`λx.object(x,nasi),`** and (iii) semantic attachment rules for each grammar rule that define how the lexical semantics of the constituent words are combined, e.g.:

Grammar rule:
`sentence -> subj, pred, obj`

Semantic attachment rule:
`λt.(pred.sem(t)(n)(l) ∧ subj.sem(n) ∧ obj.sem(l))`

For example, given the simple sentence *Ayah makan nasi*, the system is able to produce the following semantic expression:

```
event(t,makan) ∧agent(t,n) ∧  patient(t,l) ∧
person(n,ayah) ∧object(l,nasi)
```

This expression is then asserted to the Prolog clause database, as follows:

```
tell_KB([ayah,makan,nasi],X).
X = [event(x1,makan), agent(x1,x2),
     patient(x1,x3), person(x2,ayah),
     object(x3,nasi)].
```

The system supports question answering by matching the semantic expression of an interrogative question against the clause database, e.g. given the question *Siapa makan nasi?* (who eats rice?), the system is able to respond as follows:

```
ask_KB([siapa,makan,nasi,?],X).
X = ayah;
```

III. The Problems with Inference

Blackburn and Bos introduce three inference tasks for natural language processing [1]:

1. The **querying** task: given a model *M* and a first-order formula φ, is φ satisfied in *M* or not?
2. The **consistency checking** task: given a first-order formula φ, is φ consistent (satisfiable) or inconsistent (unsatisfiable)?
3. The **informativity checking** task: given a first-order formula φ, is φ informative (invalid) or uninformative (valid)?

To draw inferences on first order logic, semantic tableau and resolution proof methods are implemented. However, they are only useful for very simple problems. Therefore, off-the-shelf theorem provers and model builders are used. Theorem provers check the validity of a formula. Therefore, using theorem provers we can check whether an arbitrary first-order formula is inconsistent (unsatisfiable) or uninformative (valid). However, theorem provers do not show non-validity. This means they do not provide a positive check for consistency and informativity. However, we can perform a partial check using model builders. Given a formula, a model builder will try to build a model that satisfies it.

In [1], a model implementation of the above approach is provided in the form of CURT (Clever Use of Reasoning Tools), also implemented in Prolog. We mapped the linguistic resources from [2,3] to the specific format used in CURT, to enable richer inference on semantic representations of bahasa Indonesia.

However, the problem of quantifier scoping needs to be solved first. Consider the quasi-logical form representation without explicit existential quantifiers in the sentences *Ayah makan nasi* (Father eats rice) and *Ibu tidak makan roti* (Mother does not eat bread) as follows:

Ayah makan nasi:

```
event(A,makan)∧agent(A,B) ∧patient(A,C) ∧
object(C,nasi)∧person(B,ayah)
```

Ibu tidak makan roti:

```
not(event(D,makan)∧agent(D,E) ∧ patient(D,F))
∧ object(F,roti)∧person(E,ibu).
```

Due to the absence of quantifiers, the variables in the arguments are not bound, and treated as atoms. The resolution method tries to unify the Prolog variables in:

```
event(A,makan) ∧agent(A,B) ∧patient(A,C)
```

with Prolog variables in:

```
not(event(D,makan) ∧agent(D,E) ∧ patient(D,F))
```

resulting in a contradiction, regardless of the scope of the variables. In another scenario that uses a theorem prover and model builder to check the satisfiability of the representation in a model, the scoping problem also appears.

When existential quantifiers are added to the representations, for example, *Ayah makan nasi* is represented as:

```
∃E ∃A ∃P(event(E,makan)∧agent(E,A) ∧
patient(E,P) ∧ object(P,nasi)∧
person(A,ayah)).
```

and *Ayah tidak makan nasi* is represented as:

```
∃E ∃A ∃P(not(event(E,makan)∧agent(E,A) ∧
patient(E,P)) ∧ object(P,nasi)∧
person(A,ayah)).
```

The representation of the sentences will be:

```
and(some(A,some(B,and(some(C,and(and(event
(A,makan),and(agent(A,B),patient(A,C))),
object(C,nasi))),person(B,ayah))))),some(D,
some(E,and(some(F,and(not(and(event(D,makan)
,and(agent(D,E),patient(D,F)))),object(F,
nasi))),person(E,ayah)))))
```

The variables in the `event` predicates were bound to different quantifiers and thus produce no contradiction.

IV. Discourse Representation Structures

The first order logic representation as explained in the previous section is very effective to handle scope ambiguities by introducing quantifiers in sentences. However such representation can face very serious problem when sentences are considered as pieces of information that are taken for granted in a context, or consist of a series of sentences that form a particular context as a whole.

In the example from the previous section, i.e. *Ayah makan nasi* continued by the next sentence *Ayah tidak makan nasi*, those are not really clear in their context. Either there is only one agent 'father' or two agents (different 'fathers') that eat(s) some rice in one or different events. Suppose the second sentence is changed by replacing father into *dia* (he) and introducing a finite determiner *sepiring* (a plate of) next to the 'rice', as follows:

Dia tidak makan sepiring nasi.
(He does not eat a plate of rice.)

A first order representation would have trouble in recognizing the context of the sentence, either it is the same object 'rice' that agent 'father' is supposed to eat or a different object 'rice' from some other agent.

A specific problem, namely pronoun referencing, occurs in these sentences, and at the same time, the consistency and informativity should remain in the scope of the existential variables, which could lead to the presupposition problem. Those problems cannot be handled directly using first order logic representations. The context of the sentences should first be explicitly affirmed by applying Discourse Representation Structures (DRS) [8].

A DRS is a box-like structure that implements Discourse Representation Theory (DRT). DRT was originally developed in [5,6]. The concept was originally called File Change Semantics, which tries to recognize the change in context in which subsequent sentences will be interpreted. In [9], a DRS is viewed as a mental model constructed during the process of discourse comprehension, i.e. as a picture that at the same time also has a dynamic perspective which insists that DRS are programs.

A DRS is a pair consisting of a finite set of discourse referents and a finite set of conditions. In the Prolog-based system developed in [9], a DRS is encoded as a term of the form **drs(Ref,Cond)**, where **Ref** is a list of terms representing the discourse referents, and **Cond** is a list of other terms representing the DRS conditions. From the example *Ayah makan sepiring nasi*, the following semantic representation is created (see also Figure 1):

```
drs([A, B, C], [pred(ayah,A), pred(nasi,
B), pred(makan, C), rel(agent, C, A),
rel(patient, C, B), pred(nonreflexive, C),
pred(event, C)])
```

This leads to the following DRS:

```
 ___________________
| x3 x2 x1          |
|-------------------|
| ayah(x3)          |
| nasi(x2)          |
| makan(x1)         |
| agent(x1,x3)      |
| patient(x1,x2)    |
| nonreflexive(x1)  |
| event(x1)         |
|___________________|
```

The **x1**, **x2** and **x3** variables are the discourse referents that bind the vocabularies (*ayah*, *makan*, *sepiring*, *nasi*) to some conditions in the semantic rules. The same lexicon and grammar constructions that have been built in the previous mapping could be directly used in the DRS semantic rules.

A. Mapping of Lexical Semantics

To extend the representation of lexical semantics into DRS, the **semlex/2** predicate need to be reconstructed into the DRS. For example the pronoun **pn** is represented as:

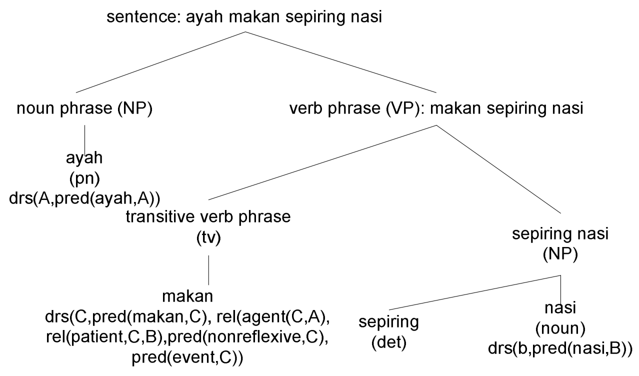

Figure 1. Parse tree of the sentence *Ayah makan sepiring nasi*

```
semLex(pn,M):-
   M = [symbol:Sym,
sem:lam(P,alfa(nam,drs([X],[pred(Sym,X)]),
app(P,X)))].
```

In this example, the bound occurrences of variable **X** indicate that pronouns are associated with a DRS. The DRS need to be supplied with a term predicate that is plugged into it in the indicated position. In this way a partial DRS becomes a full DRS.

B. Mapping of the Semantic Rules

The grammar in the representation is a DCG which uses predicates for functional application to manipulate the representations handed up by lexical entries. It gives thus flexibility whether these representations are first order formulas mixed with lambdas, or even DRS mixed with lambdas. For this purpose the macros for the lexical entries and the merging process for the predicates needs to be defined. For the example above, the merging among the components in the verb phrase (VP) *makan sepiring nasi*, will occurs in two steps. First, the process needs to know which lexical entries need to be merged, and secondly the merge process itself.

The definition for the macro that was needed, is given by the DCG notation, in the example is the VP part:

```
vp([coord:no,inf:I,num:Num,gap:G,sem:VP]-->
tv([inf:I,num:Num,ref:Ref,sem:TV]),
np([coord:_,num:_,gap:G,ref:Ref,sem:NP]),
   {combine(vp:VP,[tv:TV,np:NP])}.
```

This rule combines the semantics of the transitive verb (TV) and object noun phrase (NP), where each is a different DRS to be merged in our grammar. The merging process comes in the semantic rule **combine/2**. For the example above this rule will be:

```
combine(t:Drs,[s:S,t:T]):-
betaConvert(mergeDrs(app(S,lam(E,drs([],
[pred(event,E)]))),T),Drs).
combine(vp:app(A,B),[tv:A,np:B]).
```

And the definition of merge rule in **mergeDRS/2** is as follows:

```
mergeDrs(merge(G1,G2),drs(Disc3,Cond3)):-
    mergeDrs(G1,drs(Disc1,Cond1)),
    mergeDrs(G2,drs(Disc2,Cond2)),
    appendLists(Disc1,Disc2,Disc3),
    appendLists(Cond1,Cond2,Cond3).
```

This merge process is used to build a complex DRS, i.e. the process itself is a result of another merge process.

C. *Reference and Presupposition Resolution*

The DRS that need to be built can be complex enough to form subordinates DRS, i.e. a box inside another box. This property will be useful to detect anaphoric references that occur in a discourse.

Equality conditions are the mechanism used in DRS to resolve anaphors and presuppositions. DRS allows the condition **x=y** to be added, that holds the discourse referents in the universe of a DRS, if **y** is accessible from a DRS whose universe contains **x**. In other words, a discourse referent could have the same reference in contrast to another referent if they occur in the same box or subordinate box.

For example, if some sentences were typed into the system, as follows:

```
> ayah makan sepiring nasi.
Curt: OK.

> dia tidak makan sepiring nasi.
Curt: OK.
```

These sentences will have DRS interpretations as follows:

```
Interpretation 1: consistent, informative,
0 local violations.
 _________________________
| x3 x5 x4                |
|-------------------------|
| neuter(x3)              |
| ayah(x3)                |
| nasi(x5)                |
| makan(x4)               |
| agent(x4,x3)            |
| patient(x4,x5)          |
| nonreflexive(x4)        |
| event(x4)               |
|      __________________ |
|     | x2 x1            | |
| __  |------------------| |
|   | | nasi(x2)         | |
|     | makan(x1)        | |
|     | agent(x1,x3)     | |
|     | patient(x1,x2)   | |
|     | nonreflexive(x1) | |
|     | event(x1)        | |
|     |__________________| |
|_________________________|
```

The above consistent interpretation says something about the coreference problem that has been resolved. The system knows that the vocabularies **ayah (x3)** and **dia = neuter (x3)** refer to the same thing.

More interesting is that the negation that was asserted in the second sentence would give an interpretation that the same event of *makan nasi* (eating rice) cannot be occurring at the same time (**event(x1)** and **event(x4)**), or they will be interpreted as inconsistent, as stated in the following second interpretation:

1. Generate a DRS for the input sentence with all the elementary anaphor and presuppositions given as beta and alpha substitutions.
2. Merge this DRS with the DRS that has been constructed so far.
3. Traverse the DRS locally (subordinate DRS = local constraints), and when an alpha substitution is applied, try to:
   a. Link the anaphor information to an accessible subordinate DRS (in this way, the algorithm will try to resolve the DRS using free variable check mechanism).
   b. If it fails, accommodate the information to a superordinated level of discourse.
4. Remove those DRS from set of potential interpretations that violate the acceptability constraints, i.e. the free variable check.

Figure 2. Reference and presupposition resolution algorithm

```
Interpretation 2: inconsistent.
 _________________________
| x3 x5 x4                |
|-------------------------|
| neuter(x3)              |
| ayah(x3)                |
| nasi(x5)                |
| makan(x4)               |
| agent(x4,x3)            |
| patient(x4,x5)          |
| nonreflexive(x4)        |
| event(x4)               |
|      __________________ |
|     | x2 x1            | |
| __  |------------------| |
|   | | nasi(x2)         | |
|     | makan(x1)        | |
|     | agent(x1,x3)     | |
|     | patient(x1,x2)   | |
|     | nonreflexive(x1) | |
|     | event(x1)        | |
|     |__________________| |
|_________________________|
```

The algorithm to resolve the reference and presupposition problem can be summarized as the algorithm in Figure 2 [9]. Following this algorithm, there could be more than one interpretation present when the program traverses the DRS locally in its subordinates and superordinates.

The tricky step after the anaphoric parts of an input have been resolved is the mechanism to ensure that the anaphoric resolution satisfies the consistency and informativity checks that are performed respectively using a first order logic theorem prover (Otter) and model builder (Mace). The translation from DRS into first order logic is performed by mapping the discourse referents in the universe of a DRS to existentially quantified variables, and then recursively translating the conditions. In Prolog, this could be done as:

```
drs2fol(drs([X|Refs],Conds),
some(X,Form)):-drs2fol(drs(Refs,Conds),Form.
```

The final judgment is now left to the human user to manually select the best interpretation that actually happens in the discourse. This selection will be the final decision which plays a further role in the whole universe of discourse.

V. Question Answering Strategy

A partial strategy of a question answering mechanism has been developed, that checks the unifiability of a question against the first order representation of the selected discourse. During a question answering session, a question from the user will be parsed using the grammar rules for a question in the following form:

```
q([sem:Sem])-->
  whnp([num:Num,sem:NP]),
  vp([coord:_,inf:fin,num:Num,gap:[],se
  m:VP]),
  {combine(q:Sem,[whnp:NP,vp:VP])}.
```

This rule says that a question should be parsed as a combination of a question word (as an NP constituent) and the rest of the question as a VP constituent. There could be some other rules developed, but in this paper only questions that can be answered with a proper noun will be described. Further in the grammar rules, the rule for an NP constituent is as follows:

```
whnp([num:sg,sem:NP])-->
   qnp([mood:int,sem:QNP]),
   {combine(whnp:NP,[qnp:QNP])}.
```

Using this rule, the system will recognize an interrogative sentence that has a **qnp** rule, as follows:

```
qnp([mood:M,sem:Sem])-->
  {lexEntry(qnp,[symbol:Symbol,syntax:Word,
  mood:M,type:Type])}, Word,
  {semLex(qnp,[type:Type,symbol:Symbol,sem:
  Sem])}.
```

In the lexicon entry, the question type needs to be described, in the form of a proper name. This mechanism is done by using a prolog variable called `Person` as the `symbol` in the lexicon.

```
lexEntry(qnp,[symbol:Person,syntax:[who],
mood:int,type:wh]).
```

The `Person` variable will be later unified with the first order logic formula of the selected discourse representation.

In order to bring the question type into the same representation of the discourse, a lexical semantic rule of the `Person` variable needs to be developed that merges it into the DRS of the rest of the question by using the `combine/2` predicate. The lexical semantic rule of the variable takes the following form:

```
semLex(qnp,M):-
   M = [type:wh, symbol:Sym,
   sem:lam(P,alfa(nam,drs([X],[pred(Sym,X)]),
   app(P,X)))].
```

And the combination rule takes the following form:

```
combine(q:app(A,B),[whnp:A,vp:B]).
combine(whnp:A,[qnp:A]).
```

To check the unifiability of the question formula against the formula of the discourse, the built-in Prolog predicate `unifiable/3` is used. This predicate returns a list of variable-value pairs that will unify the two formulas. If the formulas are unifiable then it is ensured that the answer can be found in one of the variable-value pairs. To find the answer, the pairs that have no proper name in it need to be deleted, and return the proper name as the final answer.

For example, once again using the simple sentence *Ayah makan sepiring nasi* as the contents of the knowledge base (see Section IV), the results of the question answering mechanism when applied to the query *Siapa makan sepiring nasi?* (who eats a plate of rice?) are as follows:

```
siapa makan sepiring nasi?

FOL of the selected discourse:
some(_G6487,some(_G6490,some(_G6493,and(pred(ayah
,_G6487),and(pred(nasi,_G6490),and(pred(makan,_G6
493),and(rel(agent,_G6493,_G6487),and(rel(patient
,_G6493,_G6490),and(pred(nonreflexive,_G6493),pre
d(event, _G6493)))))))))

FOL of the query:
some(_G6241,some(_G6244,some(_G6247,and(pred(_G62
53,_G6241),and(pred(nasi,_G6244),and(pred(makan,
_G6247),and(rel(agent,_G6247,_G6241),and(rel(pati
ent,_G6247,_G6244),and(pred(nonreflexive,_G6247),
pred(event, _G6247)))))))))
```

The list of unifiable variable-value pairs of the two formulas, i.e. the discourse model and the query, is: `[_G6562=ayah, _G6568=_G6569, _G6574=_G6575, _G6580=_G6581]`

After the non-proper noun variable-value pairs have been deleted, the proper noun `ayah` will be returned, and the engine considers it as the final answer.

VI. Discussion and Summary

In this paper the process of performing Indonesian semantic analyzer in first-order logic and discourse modeling architecture has been discussed. Both architectures can be very useful to do some inference tasks. First-order logic representation has its limitations in variable scoping and modeling the universe of a discourse. To model contextual scope of semantic representation, the first-order logic representation needs to be extended into a more acceptable context representation, namely the discourse representation structures.

In order to use the DRS in a question answering session, a partial strategy that uses unifiability mechanism has been developed. This strategy depends strongly on the first-order formula of the selected discourse. If the first-order formula of the question is not in the same form of the selected discourse, than it will fail to return an answer.

The question answering mechanism needs to be further developed so that the system can infer an answer from the model of the selected discourse. In other words, it will be necessary to make sure that an answer is satisfiable in a discourse model. If the question answering mechanism can infer from the model, then it can be used to answer a series of questions, and can also handle multiple named entities.

REFERENCES

[1] P. Blackburn and J. Bos, "Representation and Inference for Natural Language: A First Course in Computational Semantics," Center for the Study of Language and Information (CLSI) Publications, 2005.

[2] S. D. Larasati and R. Manurung, "Towards a semantic analysis of bahasa Indonesia for question answering," Proceedings of the 10th Conference of the Pacific Association for Computational Linguistics (PACLING 2007), Melbourne, Australia, 19-21 September 2007.

[3] R. Mahendra, S. D. Larasati, and R. Manurung, "Extending an Indonesian Semantic Analysis-based Question Answering System with Linguistic and World Knowledge Axioms," Proceedings of the 22nd Pacific Asia Conference on Language, Information, and Computation (PACLIC 2008), pp.262-271, Cebu, Philippines, 20-22 November 2008.

[4] J. Hobbs, "Ontological promiscuity," Proceedings of the 23rd Annual Meeting of the Association for Computational Linguistics, pp.61–69. The Association for Computational Linguistics, Chicago, USA, 1985.

[5] I. Heim, "The Semantics of Definite and Indefinite Noun Phrases," Ph.D. thesis, University of Massachusetts. 1982.

[6] H. Kamp, "A Theory of Truth and Semantic Representation," In J. Groenendijk, T.M. Janssen, and M. Stokhof (Eds.), Truth, Interpretation and Information: Selected Papers from the Third Amsterdam Colloquium, pp. 1-41. Dordrecht, 1984.

[7] D. S. Jurafsky and J. H. Martin, "Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition," Prentice-Hall, 2000.

[8] P. Blackburn and J. Bos, "Computational Semantics," Theoria 18(1), pp. 27-45, 2003.

[9] P. Blackburn and J. Bos, "Working with Discourse Representation Theory: An Advanced Course in Computational Semantics," book draft available at: http://www.blackburnbos.org. M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.