A Decidable Linear Logic for Speech Translation

Tsutomu Fujinami
School of Knowledge Science
Japan Advanced Institute of Science and Technology
1-1 Asahidai, Tatsunokuchi
Ishikawa, Japan 923-1292
fuji@jaist.ac.jp

Abstract

The structure of objects employed in the study of Natural Language Semantics has been increasingly being complicated to represent the items of information conveyed by utterances. The complexity becomes a source of troubles when we employ those theories in building linguistic applications such as speech translation system. To understand better how programs operating on semantic representations work, we adopt a logical approach and present a monadic and multiplicative linear logic. In designing the fragment, we refine on the multiplicative conjunction to employ both the commutative and non-commutative connectives. The commutative connective is used to glue up a set of formulae representing a semantic object conjointly. The non-commutative connective is used to glue up a list of formulae representing the argument structure of an object, where the order matters. We also introduce to our fragment a Lambek slash, the directional implication, to concatenate the formula representing the predicative part of the object and the list of formulae representing the argument part. The monadic formulae encode each element of the argument part by representing its sort with the predicate and the element as the place-holder. The fragment enjoys the nice property of being decidable. To encode contextual information involved in utterances, however, we extend the fragment with the exponential operator. The context is regarded as a resource available as many as required, but not infinitely many. We encode the items of context with the exponential operator, but ensure that the operator should appear only in the antecedent. The extention keeps the fragment decidable because the proof search will not fall into an endless search caused by the coupling of unlimited supply and consumption. We show that the fragment is rich enough to encode and transform semantic objects employed in the contemporary linguistic theories. The result guarantees that the theories on natural language semantics can be implemented reasonably and safely on computers.

Keyward: multiplicative linear logic, natural language semantics, monadic logic, Lambek calculus

1 Introduction

The recent progress in the theories of natural language semantics enables us to capture a great deal of subtleties in the meanings conveyed by utterances. Even several decades ago, when Montague initiated the program of research, we could already handle beliefs, quantification, tenses and aspects and so on. The field of study has since then been greatly expanded to include attitudes, mental states, communication states among many others, all pointing to the real life rather than the unreal such as unicorn or the nonexistent French King. Through inference we can find something more interesting items of information, not limited to a simple answer, "yes" or "no."

The progress was however not achieved without paying a price. The objects to represent the meaning of utterances have got complicated and look sometimes very exotic to outsiders who have not seen anything more curious than $\Box$ or $\Diamond$. Even insiders suffer from the complexity when they try to build, based on a theory, something useful such as a speech translation system as the definition of each semantic object becomes too long to view on the computer display at a glance. The definition may sometimes not be formatted in one page.

One of the important lessons we have learnt through implementing linguistic theories, is that theories should be as simple as possible so that we can check whether an algorithm implementing an idea terminates successfully or unsuccessfully. Theories should also require the least computational power to run so that we will not be frustrated in front of the computer display, waiting for a return.

How far can we reasonably get into details of the meaning, without requiring too much computational power? This is the issue we address in this paper. We take a logical approach to study the computational aspects of the theories on meanings and show how we can specify the semantic objects employed in the contemporary linguistic theories in linear logic. Linear logic particularly suits us because its rich set of connectives allows us to specify varieties of objects and its close connection to computation brings us a clear insight into what class of computational complexity is expected when we encode and operate on those objects on computers.

The minimum requirement for our project is to ensure that there is a decidable method whether or not a translation or inference is valid. Because we specify semantic objects as formulae and translation rules as axioms, translating a semantic object into that of the target language is seen naturally as an inference in linear logic. The decidability is particularly important because it ensures that we can test whether our translation rules work properly as we expect provided enough time to run. Once we are assured that there is a decidable method, then we are concerned with the computational complexity. We discuss the issue briefly in this paper, too.

The body of this paper is divided into two parts. In the first part we show how we can specify semantic objects in a monadic, multiplicative linear logic, which is decidable. In the second part we extend the fragment with the exponential so that we can handle contexts that relate the meaning of utterances to the extra-linguistic factors and propose a way to keep the fragment decidable by imposing some restriction on the use of the exponential. The body parts are followed by discussions on logical and computational issues and we conclude the paper by claiming our contributions to computational linguistics.

2 Encoding and transforming structured objects

2.1 Structured objects

We start by presenting an example of a semantic object to represent the meaning of the sentence, "I have a car." The basic idea is to find out the elements comprising the meaning and encode each element as a term. We can know by hearing the utterance that there is a person who addresses him/herself as the speaker and claimes that he or she has something, which is categorized as car, but not determined yet in that context. To express these items of information we need the following four terms:

(1) a. speaker(i)
b. have(i, x)
c. car(x)
d. undef(x)

Intuitively these terms are equivalent to a logical formula such as

$$\exists i\ \exists x(\ \text{speaker}(i) \wedge \text{have}(i, x) \wedge \text{car}(x) \wedge \text{undef}(x)\ )$$

except of the last one, undef(x), which does not denote a thing in the world, but specifies the way the word, 'car', is interpreted.

We do not care about in this paper how the expression should be interpreted against the world, but assume that there exists a systematic way to relate it to the real world to judge whether the given sentence is true or false. We are only concerned with the issues concerning the representation itself. Once we confine ourselves to investigating the issues of representation, the points to note on the approach to represent the meaning are as follows:

1. The meaning is represented conjointly by a set of terms,
2. Each term is consisted of the relation and argument, and
3. The term itself cannot be an element of arguments.

The last point is the most important because we do not need to consider complex terms such as `have( speaker(i), x )` and the degree of complexity is greatly decreased compared with the first order logic.[1] Here we recognize a chance to construct a logic lighter than the first order logic, but still suitable for our purpose.

2.2 Encoding structured objects

We encode the representation of meanings in linear logic. One of the reasons why we turn to linear logic is that the logic provides us with a rich set of connectives, with which we can construct various kinds of compound objects. The other reason is that we can estimate what class of computation is required when we implement our ideas on computers. We start by showing how we utilize the connectives provided in linear logic.

Let $[\![.]\!]$ be a function to translate the terms into linear logic. The first connective we employ is the multipicative conjunction, $\otimes$. We follow other researchers such as [Dalrymple et al. (1993)] in employing the connective to glue up terms, that is,[2]

[1] The technique to simplify terms is often called Neo-Davidsonean approach.
[2] We assume that the variables, `i` and `x`, are existentially bound by $\exists$.

$[\![$(speaker(i), have(i, x), car(x), undef(x))$]\!]$
$= [\![$speaker(i)$]\!] \otimes [\![$have(i, x)$]\!] \otimes [\![$car(x)$]\!] \otimes [\![$undef(x)$]\!]$

We now have to regress from the first order logic to encode the formula such as `have(i,x)` because the first order logic is known to be undecidable. Our source of inspiration comes from the treatment of relations by Montague, who considered relations such as 'have(a, b)' as a function such as 'have(b)(a)', a function to return 'true' if 'a' and 'b' are given in that order. All we need are then some connective to distinguish the predicate, 'have', from the argument, 'a' and 'b', and the other connective to keep the order among elements of argument. For the former we employ the Lambek slash, /, and for the latter the non-commutative multiplicative conjunction, i.e., $\oslash$.

The Lambek slash is a directional version of the linear implication, $\multimap$, in that $A/B\ B \Rightarrow A$ is valid while $B\ A/B \Rightarrow A$ is not valid inference. Because the formula A/B behaves like function, we use the connective to distinguish the predicate part from the argument part. The term, $[\![$have(i,x)$]\!]$, is for example translated to $have/[\![$(i,x)$]\!]$.

The non-commutative multiplicative conjunction, $\oslash$, is useful for keeping the order among the elements of argument. While the commutative multiplicative conjunction, $\otimes$, allows to swap composites, e.g., $A \otimes B \Rightarrow B \otimes A$, it is not the case for the non-commutative one. Thus, the expression, $have/[\![$(i,x)$]\!]$, can further be reduced to $have/([\![$i$]\!] \oslash [\![$x$]\!])$ without the loss of the order between `i` and `x`.

To go further we have to partially return to the first order logic because of the variables `i` and `x`. We allow for one-place predicates in our logic because the monadic logic is known to be decidable, thus will not ruin our enterprise. The variables may hold a place of a predicate, that is, p(i) or q(x), but what is the most suitable ontologically for those predicates p and q? I believe that it is most natural if we encode the sort or type information of individuals or variables as predicate. The sort information will be particularly valuable when we apply the logic to specifying the change as inference because we can narrow down the number of axioms applicable to particular formulae owing to the sort information.

Assuming for the sake of explanation that `i` is of the sort `person` and `x` of the sort `vehicle`, the term, $have/([\![$i$]\!] \oslash [\![$x$]\!])$, is finally reduced to $have/(person(i) \oslash vehicle(x))$. The other terms are similarly translated to $speaker/person(i)$, $car/vehicle(x)$, and $undef/vehicle(x)$, respectively.

Table 1 shows our fragment of linear logic. We customary let (capital) alphabets range over single formulae and greek letters over the strings of formulae. The fragment is basically the (associative) Lambek Calculus as presented in [Abrusci (1996)], but extended by admitting both the commutative and non-commutative multiplicative conjunctions, each indicated as $\otimes$ and $\oslash$, respectively. Note that the exchange rule is only applicable to the commutative one. (The rule of the other non-commutative multiplicative conjunction in the opposite direction is suppressed for simplicity.)

2.3 Transforming structured objects

Linear logic enables us not only to specify structured objects to represent meanings but also to specify the transformation of or the operation on those objects. The ability of specifying changes is valuable for us because we can study the logical and computational aspects of machine translation. In the following we assume that the translation occurs at the level of semantics, that is, a sentence in the source language is analyzed to extract the semantic information, which is defined as we have seen above. The semantic

Exchange	$\dfrac{\Gamma, A \otimes B, \Delta \vdash C}{\Gamma, B \otimes A, \Delta \vdash C}$		
Identity	$\dfrac{}{A \vdash A}$	$\dfrac{\Gamma \vdash A \quad \Pi, A, \Delta \vdash B}{\Pi, \Gamma, \Delta \vdash B}$	Cut
1 Left	$\dfrac{\Gamma, \Delta \vdash A}{\Gamma, \mathbf{1}, \Delta \vdash A}$	$\dfrac{}{\vdash \mathbf{1}}$	**1** Right
$\otimes$ Left	$\dfrac{\Gamma, A, B, \Delta \vdash C}{\Gamma, A \otimes B, \Delta \vdash C}$	$\dfrac{\Gamma \vdash A \quad \Delta \vdash B}{\Gamma, \Delta \vdash A \otimes B}$	$\otimes$ Right
$\oslash$ Left	$\dfrac{\Gamma, A, B, \Delta \vdash C}{\Gamma, A \oslash B, \Delta \vdash C}$	$\dfrac{\Gamma \vdash A \quad \Delta \vdash B}{\Gamma, \Delta \vdash A \oslash B}$	$\oslash$ Right
$\backslash$ Left	$\dfrac{\Gamma \vdash A \quad \Pi, B, \Delta \vdash C}{\Pi, \Gamma, A \backslash B, \Delta \vdash C}$	$\dfrac{A, \Gamma \vdash B}{\Gamma \vdash A \backslash B}$	$\backslash$ Right
/ Left	$\dfrac{\Gamma \vdash A \quad \Pi, B, \Delta \vdash C}{\Pi, B/A, \Gamma, \Delta \vdash C}$	$\dfrac{\Gamma, A \vdash B}{\Gamma \vdash B/A}$	/ Right

Table 1: Sequent calculus formalisation of the fragment

representation is then transferred to that of the target language and some sentence in the target language is generated by referring to the transferred semantic representation. We confine ourselves in the paper to analyzing the semantic transfer.[3]

Let us consider how the sentence, "I have a car", is translated to some other language, say, Japanese. The Japanese sentence, shown as TR1, is an example of dative subject construction, in which an animate NP, usually an Experiencer, appears overtly in the dative case, while another NP appears in the form usually associated with subjects[Trask (1993), page 71].

TR1 watashi ni kuruma ga aru
pron. dative car subject exist
(A car exists to me = I have a car)

The meaning of the sentence may be encoded in linear logic as below:

(2) $aru/(vehicle(x) \oslash person(i)) \otimes speaker/person(i) \otimes kuruma/vehicle(x) \otimes undef/vehicle(x)$

which differs from that of the srouce sentence in that 'have' is replaced by 'aru', the order between $person(i)$ and $vehicle(x)$ is changed, and 'car' is replaced by 'kuruma'. We reproduce the semantic representation of the original sentence below for reference:

(3) $have/(person(i) \oslash vehicle(x)) \otimes speaker/person(i) \otimes car/vehicle(x) \otimes undef/vehicle(x)$

The following set of axioms suffices to infer (2) from (3) in the fragment of our logic:

(4) a. $have/(person(x) \oslash vehicle(y)) \vdash aru/(vehicle(y) \oslash person(x))$
b. $car/vehicle(x) \vdash kuruma/vehicle(x)$

[3] Our approach is however not limited to studying the semantic transfer module only. The same approach can be applied to studying parsing and generating sentences and we can study the whole process of translation in our framework once we combine them all into one system.

As illustrated above we can specify the translation in linear logic by defining the set of translation rules as axioms. We can claim that that kind of simple translation can be formalized in a decidable fragment of linear logic, i.e., a monadic, multiplicative linear logic. But for better translations we need richer systems.

3 Encoding and transforming structured objects in contexts

3.1 Structured objects in contexts

We have so far considered sentences independent of the context in which it is uttered. For better translations, however, we have sometimes to refer to the context. Let us examine the sentence, "I have a car", for example. The sentence may be uttered by either a male or female speaker, but it does not matter in English and the first person singular pronoun, 'I', is always employed. The pronoun is better translated to 'boku' in Japanese, however, instead of 'watashi' if the speaker is male.[4]

The items of information that are not explicit in given sentences are often expressed separately as an item of background knowledge. The extra-linguistic knowledge that the speaker is male may be indicated by describing the fact with some conjunction like *where*:

(5) a. speaker(i)
b. have(i, x)
c. car(x)
d. undef(x)
e. **where** male(i)

3.2 Encoding structured objects in contexts

When the extra item is added as in (5), the effect is usually taken into account by restricting the domain where the semantic representation should be interpreted. That is, the representation should only be interpreted against a domain in which the speaker is male. While truth-conditionally the treatment is enough to accommodate the additional items of information, we cannot resort to something outside representations when we work in a proof-theoretic framework. How should we then deal with those extra linguistic items?

Our idea is to regard those items to represent contexts as resources available as many as required. The information contained in sentences will be transformed through the process of translation and the original items of information will no longer be available after transformation. The items comprising contexts are on the other hand stand there no matter how the information contained in the sentence undergoes transformation.

Once we are contended with the treatment it is straight forward to encode the replicable items with the exponential, !, in linear logic as below:

(6) $have/(person(i) \oslash vehicle(x)) \otimes speaker/person(i) \otimes car/vehicle(x) \otimes undef/vehicle(x) \otimes\ !\ (male/person(i))$

[4]Needless to say, we simplify the matter drastically for the sake of explanation. The pronoun, 'watashi', is still applicable in formal settings.

3.3 Transforming objects under context

We have to be careful not to push our fragment of linear logic beyond the boundary of decidable fragments. It is known that the decidability of multiplicative-exponential linear logic (MELL) is unknown [Lincoln (1995)]. We therefore lose the decidability unless we impose some restriction on the use of the exponential.

We observe that the source of difficulty lies in allowing for both unlimited supply and consumption of resources. For our purpose, however, we can happily give up with the latter, the unlimited consumption, because transformed representations do not need to be referred to repeatedly. Once we restrict the use of exponential to the antecedent, the decidability of the fragment is trivial. Reusable resources are referred to as many times as a demand arises and will be erased out when they are not required anymore.

Table 2 shows the additional part of our fragment concerning the exponential. The weakening is as usual, but the contraction rule only allows to generate A without the exponential. With the side condition that A should be neither in Γ nor in Δ, the rule prohibits the logic from generating a number of unused $!A$s. We exclude the rules for dereliction (Table 3). The same effect by Dereliction Left rule can be derived in our logic by combining the Contraction, $\otimes$ Left, and Weakening rules. Dereliction Right rule does not conform to our idea of eliminating any occurrences of the exponential in the consequent, thus it is eliminated.[5]

Weakening $\dfrac{\Gamma, \Delta \vdash B}{\Gamma, !A, \Delta \vdash B}$	$\dfrac{\Gamma, !A \otimes A, \Delta \vdash B}{\Gamma, !A, \Delta \vdash B}$	Contraction $(A \notin \Gamma, \Delta)$

Table 2: The rules of the exponential

Dereliction Left $\dfrac{\Gamma, A \vdash B}{\Gamma, !A \vdash B}$	$\dfrac{!\Gamma \vdash A}{!\Gamma \vdash !A}$	Dereliction Right

Table 3: The dereliction rules (Not in our fragment)

With the restriction imposed on, we can encode the rule to transfer `speaker(i)` to `speaker(i)`$\wedge$`male(i)` by referring to the context as below:

(7) $speaker/person(i) \otimes male/person(i) \vdash speaker/person(i) \otimes male/person(i)$

When combined with the set of axioms (4), the axiom turns the give semantic representation into the ohter below:

(8) $aru/(vehicle(x) \oslash person(i)) \otimes speaker/person(i) \otimes male/person(i) \otimes kuruma/vehicle(x) \otimes undef/vehicle(x)$

3.4 Points to note

3.4.1 The distribution of the exponential

We ensure that the exponential, !, should not distribute under any other connectives but the multiplicative one. That is, we allow for the consturctions such as $!p \otimes q$ but not for other forms of constructions such as $r \backslash !s$ where the exponential is put under

[5] It causes in fact nothing harmful if we retain Dereliction Left rule, but we eliminate it because the symmetric rule, Dereliction Right, is dropped off.

the other connective, \. Although the restriction appears to be rather tight, the terms definable in the syntax are rich enough to encode the semantic objects we are concerned with.

3.4.2 Less strict fragment

From purely theoretical point of view, we can loosen the above restriction upon the syntax that the exponential should not distribute under any other connectives but the multiplicative one, as long as only the formulae in the antecedent are allowed to reproduce themselves. The formula, $r\backslash !s$, will be for example within the language because $!s$ is kept to the antecedent in the deduction. (Check how \ Left rule in Table 1 works to keep the formula to the lefthand side.) But another formula, $!r\backslash s$, should not be in the language because it leads to the appearance of $!r$ in the consequence.

The syntactic constraint we would like to impose on the logic can better be expressed if we define the logic as one-sided. We can then simply say that only the formulae marked with the negation, $\perp$, are replicable. We dare not to take the approach because it obscures our principles in employing the connectives, especially the Lambek slashes. We would like to preserve a tradition in semantic theories initiated by Montague or Lambek unless we intend to look into the *semantics* of our fragment.[6]

3.4.3 Distinction between semantic and contextual objects

A potential setback in encoding the context in our fragment is that we cannot distinguish the semantic objects from their contexts when translated because we cannot mark them with the exponential, !. (If we allow the operator to appear in the consequence, the fragment is no more decidable.) The feature will not hinder our attempt, however, because we do not think that we need to change through translation any items of information given as context. The items in the context are extracted to supplement the linguistic information contained in the sentence for better results. Once we have collected enough items of inforamtion to translate the sentence properly, we can simply discard the items given as context. It does not matter for the sentence generation module if a particular part of a semantic representation is obtained from the sentence itself or its context.

The other point to note is that the translation rules specified as axioms too can not distinguish the context from the semantic objects. We could of course allow the exponential to appear in the antecedent of axioms if we have extended our fragment of logic so that $!A \Rightarrow !A \otimes !A$ is a valid inference. But we have not done so because it becomes difficult to control the number of copies to be produced and the more computational power is required. The encoder who compiles the set of translation rules as axioms may want to distinguish the meaning and its context, but we had better not distinguish them when we formalize the translation in logic.[7]

[6] The last use of 'semantic' may sound strange to some audience. The word, semantics, means in this case another attempt to relate the logic present here to a mathematical, abstract structure or a set of objects agains which they can be interpreted.

[7] Gianluigi Bellin suggested to me that the use of the exponential, !, at both sides may not lead to undecidability if the fragment is monadic. I have not tried it out, but it is a possibility.

4 Discussions

4.1 Decidability

Our fragment of linear logic is decidable, that is, there exists for any inference a method for deciding whether it is valid. The core part of our logic is the multiplicative propositional linear logic, which is known to be decidable. The core is extended to allow for one-place predicates. The resulting logic is monadic logic and known to be decidable, too. We have extended the fragment by introducing to it the exponential, but with some care. Because the decidability of multiplicative-exponential linear logic (MELL) is unknown, we have to somehow restrict the use of the exponential. Our strategy is to allow the exponential only to appear in the antecedent. We then make copies as many as needed, but not infinitely many, because the number of consumers is limited as the result of imposing the restriction. The proof search thus always terminates no matter how it is successful or unsuccessful.

4.2 Computational complexity

The complexity class of the monadic, multiplicative linear logic is known to be PSPACE-complete. The introduction of the exponential does not lead to the combinatorial explosion in proof search, thus, the complexity remains to be PSPACE.

4.3 Linguistic coverage

The technique to encode semantic representations as a set of terms is devised in the speech translation project, Verbmobil [Kay et al. (1994)], in which the author was involved. Our work is based on Discourse Representation Theory (DRT) [Kamp and Reyle (1993)] and extends the machinery to cover broader ranges of (ordinary) linguistic phenomena such as greetings or making an appointment. DRT has been used to study various semantic phenomena for several decades and currently covers the broadest ranges of phenomena among semantic theories. The power of DRT resides in its strategy such that it allows to employ rich structures to represent subtle meanings as long as they can be related to some model systematically. Putting the issues of models and systematic interpretations aside, we think that our logic proposed in this paper is sufficient to specify most representations as employed in DRT. We believe thus that our logic helps to study broad ranges of linguistic phenomena that we are concerned with for the moment.

4.4 Simultaneous abstractions

How powerful computational machinery we have devised, compared with the one devised by, say, Montague? Looking back at his work, we can realize that his theory is based on an amalgamation of the first order logic, the lambda calculus, and the modal logic. Putting the modal logic aside, what we have achieved with the help of linear logic is to integrate the first two elements, the first order logic and the lambda calculus, into one frame. While Montague had to resort to the lambda abstraction to capture the dynamic aspects of the semantic analysis, the same effect can be achieved logically and better we enjoy some flexibility.

Recall that abstractions and substitutions must observe some order. A function such as $have(y)(x)$ can only be abstracted over x and y in that order, i.e., $\lambda x.\lambda y.have(y)(x)$. The same can be encoded in our logic as

$(r_1(x) \oslash r_2(y))\backslash(have/(person(x) \oslash vehicle(y))))$

where r_1 and r_2 serves as a reference or role to index the parameters. The substitution function is encoded as $r_1(a) \oslash r_2(b)$ and the formula is reduced to $have/(person(a) \oslash vehicle(b))$ when the substitution formula is placed to the left, concatenated by $\otimes$. (More precisely the substitution is inferred.)

The order of abstractions and substitutions can be more flexible in our logic when we use the comutative connective as follows:

$(r_1(x) \otimes r_2(y))\backslash(have/(person(x) \oslash vehicle(y))))$

The formula to encode the substitution, too, can be lazy as follows:

$r_1(a) \otimes r_2(b)$

The flexibility gives us a room to refer to something outside the sentence, e.g., the context, while parsing sentences. While the process of semantic construction had to be parallel to parsing, notably in Montague semantics, the construction can be more independent of parsing. The flexibility is the essential difference from the machinery employed by Montague.

5 Conclusion

We have proposed a fragment of linear logic to encode and transform semantic objects employed in linguistic theories. The fragment is monadic, multiplicative linear logic extended by the exponential. The use of the exponential is however restricted to appear only in the antecedent to make the logic decidable. We have shown that the fragment can specify semantic objects employed in DRT.

Linear logic helps us on one hand to estimate what class of computational power is required when we implement a theory on the meaning and guides us to design lighter schemes of representations computationally. The logic on the other hand enables us to integrate some background theories of natural language semantics, namely, the lambda calculus and the first order logic, providing us with a clearer view of computational aspects. Linear logic also provides us with a flexible mechanism for abstractions and substitutions, allowing us to study the roles of context in understanding sentences.

Acknowledgements

The ideas presented here have grown out of some of my other papers, e.g., [Fujinami (1997)]. When I wrote the paper I could not think of the fragment I presented here. My standing point was somehow ambiguous as I proposed in that paper to encode Discourse Representation Structures in a fragment of the π-calculus and employ a decidable liner logic to impose some typing scheme on them. I have made the ideas sharper in that the current work is purely logical and there is no reference to the π-calculus. I am grateful to Prof. Okada for reminding me of the fact that monadic logic is decidable at a lecture given at our institute in 1999. Everything has fitted into a picture slowly since then. I am also greatful to Josef van Genabith and Dick Crouch for inviting me to a seminar titled Linear Logic and Applications held at Schloss Dagstuhl in Germany from 22nd to 27th August in 1999. I present there the core idea of this paper and received from audience many valuable comments and feedbacks.

References

Abrusci, V. Michele. 1996. Lambek calculus, cyclic multiplicative-additive linear logic, non-commutative multiplicative-additive linear logic: Language and sequent calculus. In V. Michele Abrusci and Claudia Casadio, editors, *Proceedings of 1996 Roma Workshop on Proofs and Linguistic Categoris*, pages 21–48.

Dalrymple, Mary, John Lamping, and Vijay Saraswat. 1993. LFG semantics via constraints. In *Proceedings of the 6th Meeting of the European Association for Computational Linguistics.*

Fujinami, Tsutomu. 1997. A decidable linear logic for transforming DRSs in context. In Paul Dekker and Martin Stokhof, editors, *Proceedings of the 11th Amsterdam Colloquium*, pages 127–132, The Netherlands, December. ILLC/Department of Philosophy, University of Amsterdam.

Kamp, Hans and Uwe Reyle. 1993. *From Discourse to Logic: Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory.* Dordrecht: Kluwer.

Kay, Martin, Jean Mark Gawron, and Peter Norvig. 1994. *Verbmobil - a translation system for face-to-face dialog.* Number 33 in CSLI Lecture Notes. Center for the Study of Language and Information.

Lincoln, Patrick. 1995. Deciding provability of linear logic formulas. In Jean-Yves Girard, Yves Lafont, and Laurent Regnier, editors, *Advances in Linear Logic.* Cambridge University Press, pages 109–122.

Trask, Robert Lawrence. 1993. *A Dictionary of Grammatical Terms in Linguistics.* Routledge.