Taking Generation as Abduction Seriously

Masato Ishizaki and *Akira Ishikawa*
Univesity of Edinburgh Sophia Univesity

This paper introduces 'assumptions', one-sided mutual beliefs, to construct a bidirectional architecture incorporating Hobbs et al.'s abduction. The integration of deductive parsing and abductive semantic interpretation may provide a basis for a bidirectional architecture. However, the idea of cost, the principal mechanism in Hobbs et al.'s abduction, is not workable for generation because it simply amounts to the assumability of parts of logical forms for the purpose of interpretation. We introduce explicit 'assumptions' attached to particular expressions so as not only to control generation, but also to lighten the burden of abduction. This paper also discusses implementation issues on the data structure and the generation algorithm.

1 Introduction

Hobbs et al. (1990) explained semantic interpretation as a process of abduction, in which the logical form corresponding to the utterance is proved against a knowledge base using rules abductively when appropriate until some parts are left unproven, and assumed to be the new information in the utterance. They suggest a combination of their abductive semantic interpretation with deductive parsing to use rules bidirectionally. They use the notions of cost and weight for controlling the process of abductive semantic interpretation. The two notions are intended to indicate the degree of assumability of parts of the logical form for interpretation. However, they cannot be used in controlling generation processes.

Figure 1 shows Hobbs et al.'s architecture. Parsing transforms the input string into an interpreted logical form in which the proved parts are distinguished from the assumed parts by referring to the costs associated with them by the grammar rules. However, the nature of assumptions changes in their account of generation. What is assumed in generation are claimed to be terminal categories, and since they have all to be grounded to produce grammatical sentences, the nature of assumptions in generation is totally different from that for interpretation.

In interpretation, assumptions correspond to the new information in the utterance, which is calculated from the logical form associated with the utterance using rules abductively defining possible explanations with different degrees of plausibility. Assumptions are what cannot be explained away abductively or directly by referring to the knowledge base. Abductive use of rules is restricted by the costs assigned to each part of the logical form and the weights assigned to the rules, from which the costs of the parts of the new logical form to be introduced by a particular rule are calculated abductively. If it is more costly to use any rule abductively to prove a particular part of the logical form because it introduces a new logical form with higher costs, you just assume the part with its costs. Thus, the costs work as the mechanism for differentiating new information from old, which is the proper task of interpretation. By contrast, the costs cannot control generation processes to arrive at an adequate surface sentence reflecting the distinction between the old and new information in the input logical form. As we have noted, every terminal category whether indicating new or old information has to be assumed according to Hobbs et al.. So the question is whether the costs assigned to the input logical form for generation carries enough information for properly selecting the

Language, Information and Computation
Chungmin Lee and Beom-mo Kang, editors, 1993, Seoul: Thaehaksa

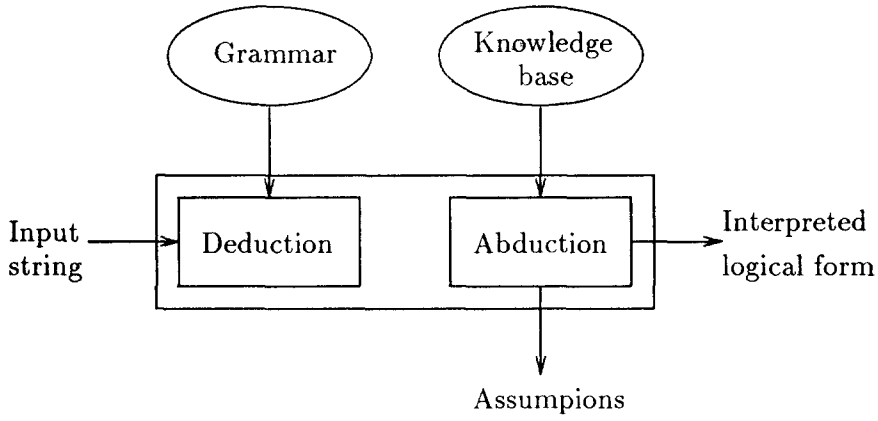

Figure 1: Hobbs et al.'s architecture.

sentence patterns and lexical items(terminal categories) of the surface sentence. As we will argue later, this is not the case. Although they claim a bidirectional use of rules, their cost and weight are meant only for interpretation, and, therefore, assumptions produced using them do not have meaning to generation. Thus, in order to attain a genuine bidirectional architecture, we have to examine their cost and weight by taking generation processes into account.

Generation is characterized as decision making or constraint satisfaction processes. In either case, selection of lexical items, sentence patterns, and discourse structures is the vital part of the generation process. The input data for generation must be fully prepared for the proper selection. As we saw, the costs and weights have no bearing on such selection, connecting not the parts of sentences but those of logical forms with the knowledge base. To choose proper expressions, further data must be attached to the input logical forms. We propose to use one-sided mutual beliefs as such data. One-sided mutual beliefs have suitable properties for characterizing the appropriateness conditions of expressions. It distinguishes the speaker's beliefs from the hearer's , which is important for handling miscommunication. When a one-sided mutual belief is no longer supported by context or contradicts context, it can be canceled. Moreover, it can be introduced into context without much effort. We call such one-sided mutual beliefs 'assumptions'. Ishizaki (1991a,1991b) proposed approaches to a reversible architecture with text generation through planning, in which he underscored, without giving implementation details, important roles played by assumptions. This paper extends this idea somewhat.

The present paper is organized as follows: firstly, different kinds of assumptions in natural language processing are examined; secondly, some problems with Hobbs et al.'s abductive generation are discussed and our new data structure is proposed; and lastly, implementation with a Prolog meta-interpreter is discussed.

2 'Assumptions' in Natural Language Processing

2.1 Word-related 'assumptions'

McCawley (1968) examined the differences among "kill", "murder" and "assassinate", where the word "kill" has the most general meaning. Of the other two, "murder" has the assumption of illegality of the killing event, whereas "assassinate" the assumptions of illegality and the fame of the person who is killed. In either case, the assumptions are not part of the truth conditions of the sentence in which these words appear; rather, they are taken to provide the appropriateness conditions for the use of these words.

2.2 Phrase-related 'assumptions'

Expressions involving indefiniteness, such as indefinite noun phrases, usually carry new information. In semantic interpretation processes, such information cannot be proved because of its novelty. Accordingly, it must be assumed to continue the interpretation process. By contrast, in generation, whether the information of an expression is new or not is for the speaker to decide. This means the novelty of information is to be checked with the speaker's mental state. On the other hand, if the novelty is judged as a mutual belief, the information can be obtained only under very special conditions shown in Halpern and Moses (1984). Assumption approaches can get around this problem.

2.3 Sentence-related 'assumptions'

Cleft sentences require novelty and uniqueness to be associated with the focused part. In interpretation, this information is communicated to the hearer. In generation, it must be assumed so long as it does not contradict the speaker's mental state. This information can be thought of as appropriateness conditions for the use of this sentence pattern. We call such assumptions *lc-assumptions* (linguistic assumptions). Thus, *lc-assumptions* can be defined as appropriateness conditions of certain linguistic expressions, including sentence patterns. One caveat is that, in generation, *lc-assumptions* must be prepared separately from the semantic content of the sentence. In other words, they are provided during the so-called 'what to say' process. But this topic is beyond the scope of the present paper.

2.4 Discourse-related 'assumptions'

The hearer's mental state can affect his understanding of the relations between the sentences constituting a discourse. Oberlander and Lascarides (1992) classified this situation by the difference in knowledge between the speaker and the hearer: the speaker S knows more about the proposition p than the hearer H; S knows p but isn't sure if H does; H as advisor, S as pupil; and S thinks H is mistaken. Their example of the first case is "Max fell. John pushed him". If H does not know the causal rule 'pushing someone causes his falling', H takes this sentence order to reflect the event order. However, if H knows the causal rule, he can understand that Max's falling was caused by John's pushing. Thus, if the speaker believes the hearer does not know the causal rule, he must construct sentences such as "Max fell because John pushed him," or "John pushed Max. Max fell". This causal rule is an example of what we will call *wk-assumptions* (world assumptions).

Notice the relation in strategy between interpretation and generation: if explicit cues do not exist, you should interpret the sentences by using default rules, whereas, in generation, if the speaker is aware of the hearer's lack of certain knowledge, he must provide some cues to the right interpretation.

3 Generation as Abduction

3.1 Generation in Hobbs et al.'s abduction

Hobbs et al. (1990) proposed the idea that semantic interpretation is characterizable as abduction, which is equivalent to proof with assumptions. Abductive interpretation proceeds under the guiding principle of minimizing the assumptions. For this purpose, they proposed a mechanism based on costs and weights, which are associated with the literals constituting logical forms. The cost on a literal inversely represents the assumability of the literal. In other words, it is taken to represent the cost of assuming the literal without proving it against the knowledge base. Weights, on the other hand, are used to determine the cost of using a rule abductively whose consequent unifies with a literal with a certain cost. A schematic example of a logical form is given in (1), where LF_i is a literal, and $\$C_i$ is the cost associated with it.

$$LF_1^{\$C_1} \wedge \ldots \wedge LF_m^{\$C_m} \tag{1}$$

An example of a Hobbs et al.'s rule is shown in (2), where ω_j is the weight associated with the literal P_j.

$$P_1^{\omega_1} \wedge \ldots \wedge P_n^{\omega_n} \supset Q \tag{2}$$

When Q unifies with a literal $LF_k^{\$C_k}$, the cost of using the rule is calculated by adding up the products $\$C_k \times \omega_j$. If $\omega_1 + \cdots + \omega_n > 1$, the total cost of the antecedent becomes greater than $\$C_k$. On the other hand, if $\omega_1 + \cdots + \omega_n < 1$, it become smaller than $\$C_k$. The sum of ω_1 through ω_n is set slightly more or less than one so that, even when a rule has the sum which is more than one, using the rule reduces the total cost if some of the resultant literals from the antecedent can be proved but the original literal $LF_k^{\$C_k}$ cannot. Thus, assumption minimization can be achieved with this mechanism with the sum of the antecedent costs controlling the direction of proof. Their integration of deduction and abduction yields grammar rules with a mixture of costed and costless literals in the antecedent.

$$Q_1 \wedge \ldots \wedge Q_m \wedge P_1^{\$C_1} \wedge \ldots \wedge P_n^{\$C_n} \supset Q \tag{3}$$

where Q_1 through Q_m and Q correspond to syntactic categories and $P_i^{\$C_i}$ to parts of the logical form. (4) is one example of such rules.[1]

$$np(i,j,x) \wedge v(j,k,p) \wedge np(k,l,y) \wedge p'(e,x,y)^{\$3} \wedge Req(p,x)^{\$10} \wedge Req(p,y)^{\$10} \supset s(i,k,e) \tag{4}$$

According to Hobbs et al., generation proceeds as shown in Fig. 2.

One problem with this algorithm is that it cannot select appropriate surface expressions. The procedure TRANSFORMIF in the algorithm abduces semantic expressions which cannot be bound or covered by the rule. As a result, the costs associated with the input logical form cannot serve as the direct clue for selecting appropriate surface expressions.

$$det(i,j,a) \wedge n(j,k,w) \wedge w(x)^{\$5} \supset np(i,k,x) \tag{5}$$
$$det(i,j,the) \wedge n(j,k,w) \wedge w(x)^{\$10} \supset np(i,k,x) \tag{6}$$

(5) and (6) are the rules for indefinite and definite noun phrases, respectively. When the cost of the final logical form can be transformed through the rules, the original cost recomputed cannot be a clue to selecting a proper word.

The other problem is that Hobbs et al.'s rules cannot represent the distinction between the speaker's and the hearer's mental states. This means they cannot distinguish good explanations from bad explanations.

$$Seg(i,j,e1) \wedge Seg(j,k,e2) \wedge CohRel(e1,e2,e)^{\$20} \supset Seg(i,k,e) \tag{7}$$
$$Exp(e1,e2,e1)^{0.8} \supset CohRel(e1,e2,e) \tag{8}$$
$$cause(e2,e1)^{0.8} \supset Exp(e1,e2,e1) \tag{9}$$

Rule (7) states segment from i to j and segment from j to k which are connected by the relation 'CohRel' make segment from i to k. Rule (8) indicates one kind of coherence relation is explanation. Rule (9) states that if events $e1$ and $e2$ are related with cause, event $e1$ is an explanation of event $e2$. If the speaker knows that the hearer does not know rule (9), it is not appropriate to use rule (7) to produce sentences connected by the intended coherence relation.

3.2 Generation with assumptions

To avoid the problems mentioned above, we attach to rules 'assumptions', which equal one-sided mutual beliefs.[2]

The revised rules are composed of expressions defined as follows:

$$\{MS_1\}expression\{MS_2\} \tag{10}$$

Description (10) says that the original mental state MS_1 can be changed into the revised state MS_2 by the expression. When there exist differences between MS_1 and MS_2, the expression is taken to introduce *lc-assumptions* or *wk-assumptions*.

```
PROCEDURE GENERATE(LF,SYN,STRING)
INPUT  LF     % a target logical form (a conjunction of LFi)
       SYN    % a target syntactic category
OUTPUT STRING % a string to be generated
BEGIN
  select a rule whose consequent is unifiable with
  the logical form LF and syntactic category SYN
  (IF FAILURE is notified from the level below in the course of recursion
   THEN select another rule);
  IF there does not exist such a rule THEN FAILURE;
  ELSE
    IF there are unbound variables in the logical forms (= LF_A)
       in the antecedent of the rule
    THEN
       TRANSFORMIF(LF_A,LF);
    IF the antecedent in the rule is a terminal string
    THEN
      RETURN the string;
    ELSE
      WHILE (LFi,SYNi) IN the antecedent of the rule
      BEGIN
        GENERATE(LFi,SYNi,STRINGi);
      END
END

TRANSFORMIF(LF_A,LF)
INPUT  LF_A  % the logical form in the antecedent which
             % has unbound variables (a conjunction of LF_Ai)
       LF    % the input logical form (a conjunction of LFj)
BEGIN
  IF there does not exist LF_Ai to be examined THEN SUCCESS;
  ELSE IF
   there does not exist LFi THEN FAILURE;
  ELSE
    select a rule which can unify the consequent with LF_Ai
    and the antecedent a set of LFj(*)
    (IF FAILURE is notified from the level below in the course of recursion
     THEN select another rule);
    IF there does not exist such a rule THEN FAILURE;
    ELSE
      newLF_A := LF_A - LF_Ai; newLF := LF - a set of LFj;
      TRANSFORMIF(newLF_A,newLF);
END (*) Here supposed to include examination of transitive closure.
```

Figure 2: Hobbs et al.'s generation algorithm.

Incorporation of mental states into rules such as (4) results in a rule such as (11). It represents how the mental states are composed.

$$\{MS_1\}Q_1^{\omega_1}\{MS_{1'}\} \wedge \ldots \wedge \{MS_n\}Q_n^{\omega_n}\{MS_{n'}\} \wedge P_1 \wedge \ldots \wedge P_m \supset \{MS_1, \ldots, MS_n\}Q\{MS_{1'}, \ldots, MS_{n'}\} \quad (11)$$

(12) shows an example of a *lc-assumption* which has to do with the appropriateness condition $\{new(\omega(x))\}$ associated with indefinte expressions.[3]

$$\{\}det(i,j,a)\{new(\omega(x))\} \wedge n(j,k,w) \supset \{\}np(i,k,x)\{new(\omega(x))\} \quad (12)$$

The *lc-assumption* $\{new(\omega(x))\}$ means that the speaker has a one-sided mutual belief that $\omega(x)$ is new to the hearer.[4]

$$np(i,i+1,it) \wedge v(i+1,i+2,is) \wedge np(i+2,j,x) \wedge wh(j,j+1,that) \wedge verb(j,k,p) \wedge np(k,l,y) \wedge p'(e,x,y)^{\$3} \wedge Req(p,x)^{\$10} \wedge Req(p,y)^{\$10} \supset \{\}s(i,k,e)\{unique(x), novel(x)\} \quad (13)$$

(13) is a rule for it-cleft sentences, showing the *lc-assumptions* $unique(x)$ and $novel(x)$ are associated with the construction as a whole. Thus, the mental states can register not only appropriateness conditions for phrases, but also those for sentences. Rule (14) states that if explanation is used as a coherence relation, the speaker believes the hearer believes that event $e1$ causes event $e2$.[5]

$$\{\}Exp(e1,e2,e1)\{cause(e2,e1)\} \supset \{\}CohRel(e1,e2,e)\{cause(e2,e1)\} \quad (14)$$

We revise the algorithm for generation by introducing mental states as shown in Fig. 3.

4 Implementation

Let us illustrate how our idea of explicit assumptions is implementable using a meta-interpreter proposed in Akama and Ishikawa (1988). In generation, the input is a logical form representing the semantic contents and a set of assumptions providing enough information for selecting an appropriate surface string. According to our revised algorithm, there are two distinct phases in each generation step. That is, one phase is concerned with proving logical literals in the antecedent of a rule, and the other with processing syntactic categories. Akama and Ishikawa's meta-interpreter provides a means of separating the two phases in a natural way. Our meta-interpreter approach is also motivated by the fact that the grammar rules contain not only individual variables but also predicate variables as seen in (12), where w is a predicate variable.

$$\{\}det(i,j,a)\{new(\omega(x))\} \wedge n(j,k,w) \supset \{\}np(i,k,x)\{new(\omega(x))\} \quad (12)$$

It is also easy to handle mental states by the meta-interpreter because they involve the mixing of levels when certain semantic expressions are treated as arguments to such higher-level predicates as 'new'.

Akama and Ishikawa's meta-interpreter, called SCP, is motivated by the idea of amalgamation of the object and meta-levels proposed by Bowen and Kowalski (1982). Their meta-interpreter is defined as follows.

```
meta(true).
meta(A & B) :- meta(A), meta(B).
meta(A) :- clause(A,B), meta(B).
meta(A) :- A.
```

```
PROCEDURE GENERATE'(LF,SYN,ASS,STRING)
INPUT  LF       % a logical form (a conjunction of LFi)
       SYN      % a target syntactic category
OUTPUT STRING   % a string to be generated
STACK  ASTACK   % a stack for assumptions
BEGIN
  PUSH(ASS,ASTACK);
  select a rule whose consequent is unifiable with the logical form LF and
  syntactic category SYN, and whose assumptions are included in assumptions ASS
  (IF FAILURE is notified from the level below in the course of recursion
   THEN POP(ASS,ASTACK); execute GENERATE' to select another rule);
  IF there does not exist such a rule THEN FAILURE;
  ELSE
    ASS := ASS - the assumptions in the consequent;
    IF there are unbound variables in the logical forms (= LF_A) in
       the antecedent of the rule
    THEN
      TRANSFORMIF'(LF_A,LF,ASS);
    IF the antecedent in the rule is a terminal string
    THEN
      RETURN the string;
    ELSE
      WHILE (LFi,SYNi) IN the antecedent of the rule
      BEGIN
        GENERATE'(LFi,SYNi,ASS,STRINGi);
      END
END

TRANSFORMIF'(LF_A,LF,ASS)
INPUT LF_A    % the logical form in the antecedent
              % which has unbound variables (a conjunction of LF_Ai)
      LF      % the input logical form (a conjunction of LFj)
STACK ASTACK' % a stack for assumptions
BEGIN
  PUSH(ASS,ASTACK');
  IF there does not exist LF_Ai to be examined THEN SUCCESS;
  ELSE IF
   there does not exist LFi THEN FAILURE;
  ELSE
    select a rule which can unify the consequent with LF_Ai
    and the antecedent a set of LFj(*);
   (IF FAILURE is notified from the level below in the course of recursion
    THEN POP(ASS,ASTACK'); execute TRANSFORMIF' to select another rule);
    IF there does not exist such a rule THEN FAILURE;
    ELSE
      newLF_A := LF_A - LF_Ai; newLF := LF - a set of LFj;
      ASS := ASS - the assumptions in the consequent;
      TRANSFORMIF'(newLF_A,newLF,ASS);
END (*) Here supposed to include examination of transitive closure.
```

Figure 3: The proposed generation algorithm.

In SCP, another layer of meta-level control is added with the constraint predicate, which makes it possible to deal with semantic constraints as a separate module.

```
meta(true,Th,P,P).
meta(A & B,Th,P1,P3) :-
  constraint(prime(A,B)), meta(A,Th,P1,P2), meta(B,Th,P2,P3).
meta(A<-B,Th,P1,P3) :-
  constraint(prime(A,P1,P2)), meta(B,Th,P2,P3).
meta(A,Th,P1,P2) :-
  constraint(prime(A),P1,P2), A.

constraint(A,P,P).
constraint(A & B,P1,P3) :-
  constraint(A,P1,P2), fconstraint(B,P2,P3).
constraint(A,P1,P3) :-
  clause(A,B,P1,P2), constraint(B,P2,P3).
```

Constraints are associated with the head of each rule, which is mapped by the prime function into its name prime(A). The last two arguments of each non-terminal represent the differential lists in the same manner as in Definite Clause Grammar proposed by Pereira and Warren (1980). They represent how the set of assumptions changes during proof. This extension of the syntactic component by means of the semantic component is called a conservative extension because the set of theorems does not change. It is also well established through the reflection principle that proofs in the object level and its simulation in the meta level are equivalent.

We use this extra meta-level in order to handle the phase in which logical literals and mental states are unified with the input logical form while the meta predicate takes care of the syntactic phase.

The rule format (11) is encoded as a term of the following form.

$$\{MS_1\}Q_1^{\omega_1}\{MS_{1'}\} \wedge \ldots \wedge \{MS_n\}Q_n^{\omega_n}\{MS_{n'}\} \wedge P_1 \wedge \ldots \wedge P_m \supset \{MS_1, \ldots, MS_n\}Q\{MS_{1'}, \ldots, MS_{n'}\} \tag{11}$$

```
clause(Q([MS1,..,MSn],[MS1',..,MSn']),
       [P1,..,Pn],
       [Q1(MS1,MS1'),..,Qm(MSm,MSm')])
```

In their formulation of abductive generation as well as interpretation, Hobbs et al. do not distinguish between syntactic categories and logical literals in conducting proofs. Although this may be one consequence of combining deductive parsing and abductive reasoning, more control is necessary at least for generation.

More specifically, instead of using costs and weights to control how far a proof should continue, we have decided to use assumptions for the purpose. Thus, an individual variable occurring as an argument to a predicate is not to be instantiated relative to the knowledge base if the predicate is in the scope of 'new'. Otherwise, it must be instantiated. This way, we can abolish the cost distinction between definite and indefinite noun phrases. Similarly, those predicates associated with high costs are all unspecified ones such as 'rel', 'Req', and 'nn'. In general, such predicates serve to explain the disparity between parts of a logical form and their syntactic realization. For example, 'rel(x,y)' rather than 'rel(x,x)' indicates the existence of a metonymy, which should be expanded in the direction of interpretation, as in their example "The Boston office called," which corresponds to '$call'(e, J1) \wedge person(J1) \wedge work\text{-}for(J1, O1) \wedge office(O1) \wedge Boston(B1) \wedge in(O1, B1)$'. The rule responsible for introducing 'rel' is as follows(p.21):

$$(\forall i, j, k, y, p, e, x)np(i, j, y) \wedge verb(j, k, p) \wedge p'(e, x) \wedge rel(x, y) \wedge Req(p, x) \supset s(i, k, e)$$

If we could regard 'rel' as a kind of variable, abduction starts without the triggering by high costs. Owing to our use of a meta-interpreter, we can adopt this option by defining specific rules for instantiating predicate variables: e.g.,

```
instantiate_pred(rel(x,y)) :- abduce(rel(x,y),X).
```

Then, abduction rules can introduce a marker for metonymy as follows:

```
abduce(rel(x,y),metonymy(work-for(x,y),y)).
```

In the direction of generation, as with 'new', 'metonymy' serves to select a particular syntactic option by consuming '$person(J1) \wedge work\text{-}for(J1, O1)$' at the same time to select 'office'.

Thus, by stretching the interpretation of variables somewhat, we can get rid of costs altogether. So, we can avoid the problem of handling costs sensibly in generation. It is as yet not certain how to say when to stop an abductive reasoning process without using weights. But their behavior in this regard is not specified in Hobbs et al.'s paper either.

We revise the original SCP in the following way. The meta-predicate is essentially concerned with processing syntactic categories. In generation, given a target syntactic category and a logical form, it looks for a relevant rule and try to unify the logical literals in the rule with the input logical form by using the constraint-predicate. The Pi's stand for sets of logical literals (that is, logical forms). The first P's correspond to unprocessed logical literals — the input logical form at the outset — and the second to processed ones. In generation, when a logical literal is subsumed by a matching literal in the rule, it is moved from the first set to the second set. At the end of the generation process, 'global_ cond' checks to make sure that the first set is empty. In interpretation, since there is no information as to what logical literals are contained in the output logical form, the first set starts and remains as empty, but all logical literals from rules are registered in the second set during the interpretation process because they constitute the output logical form. As in the case of generation, 'global_ cond' checks the input set (vacuously, though), and the output set to make sure that every 'variable' has been instantiated, which is vacuously satisfied in the case of generation. Abduction rules are called by 'global_ cond' as well as 'unify', which is called by 'constraint/3' shown below.

```
meta(true,Th,P1,P2) :-
  global_cond(Th,P1,P2).
meta(A & B,Th,P1,P3) :-
  meta(A,Th,P1,P2), meta(B,Th,P2,P3).
meta(A,Th,P1,P3) :-
  member(clause(A,B,P),Th), constraint(A,P,P1,P2), meta(B,Th,P2,P3).
meta(A,Th,P1,P2) :- A.
```

The constraint-predicate takes care of logical literals including those from mental states. 'Prime' is used to extract the set of logical literals associated with the head syntactic category of a rule. This use of 'prime' is quite different from that of the original SCP, where it is used to make the name of a category. 'Constraint/4' just combines the sets A and P to make the argument P' for 'constraint/3'.

```
constraint(A,P,P1,P2) :-
  prime(A,A'), union(A',P,P'), constraint(P',P1,P2).

constraint([],P,P).
constraint(A & B,P1,P3) :-
  constraint(A,P1,P2), constraint(B,P2,P3).
constraint(A,P1,P2) :-
  unify(A,P1,P2).
constraint(A,P1,P3) :-
  clause(A,B,P1,P2), constraint(B,P2,P3).
```

5 Conclusion

We have incorporated 'assumptions' into Hobbs et al.'s framework to achieve a genuine bidirectional architecture with enough mechanism for selection in generation and avoid unnecessary abduction in interpretation. We have also discussed an implementation by means of a meta-interpreter, and a generation algorithm. Our

assumptions replace Hobbs et al.'s costs and weights as principal mechanism for controlling an abductive use of rules. The problem of how far abductive reasoning should continue for a single process of interpretation or generation is left untouched in this paper. As we have argued for the vital role of one-sided mutual beliefs in generation, this problem also seems to require an approach taking the speaker's mental state into consideration.

Acknowledgements

The authors wish to extend their sincere gratitude to the members of the Human Communication Research Centre and the Centre for Cognitive Science of the University of Edinburgh for their encouragements.

Notes

[1]Every variable v in the formula is assumed to be bound by universal v.

[2]Perrault (1990) used one-sided mutual belief to handle the infinite nature of mutual beliefs in finite manner.

[3]Although every expression has its associated mental states, we omit them if both are {} for readability.

[4]The predicate 'new' should be defined relative to time, which indicates our 'assumptions' need to be structured for revision.

[5]The weight in this rule needs to be recomputed with regard to other rules.

References

Akama, S and Ishikawa, A (1988) "Semantically Constrained Parsing" in Abramson, H. and Rogers, M.H., eds., *Meta-Programming in Logic Programming*, The MIT Press, Cambridge, Massachusetts, 157-168.

Akama, S. and A. Ishikawa (1990) "A Semantic Interface for Logic Grammars," in *Proceedings of the First Seoul International Conference on Natural Language Processing*, 65-76.

Bowen, K. A. and R. Kowalski (1982) "Amalgamating Language and Metalanguage in Logic Programming," in Clark, K L and Tarnlund, S A, eds., *Logic Programming*, Academic Press, New York, 153-172.

Halpern, J.Y. and Y. Moses (1984) "Knowledge and Common Knowledge in a Distributed Environment," in *Proceedings of the Third ACM Symposium on the Principles of Distributed Computing*, 50-61.

Hobbs, J. R., M. Stickel, D. Appelt, and P. Martin (1990) "Interpretation as Abduction," *SRI International Technical Note* 499.

Ishikawa, A. and S. Akama (1991) "A Semantic Interface for Logic Grammars and its Application to DRT," in *Proceedings of the Third International Workshop on Natural Language Understanding and Logic Programming*, 290-304.

Ishizaki, M. (1991a) "Handling Pragmatic Information with a Reversible Architecture," in *Proceedings of ACL'91 Workshop on Reversible Grammar in Natural Language Processing*, 119-128.

Ishizaki, M. (1991b) "Linguistic Realization Through Planning," in *Proceedings of IJCAI'91 Workshop on Decision Making throughout Generation Process*, 26-33.

Kowalski, R. (1979) *Logic for Problem Solving*, Elsevier Science Publishing Company, New York.

Kamp, H. (1984) "A Theory of Truth and Semantic Representation," in Groenendijk, J. and T. M. V. Janssen, eds., *Truth, Interpretation and Information*, Foris, Dordrecht, 1-41.

Lascarides, A. and N. Asher (1991) "Discourse Relations and Defeasible Knowledge," in *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics*, 55-62.

McCawley, J.D. (1968) "The Role of Semantics in Grammar," in Bach, E. and Harms, eds., ***Universals in Linguistic Theory***, Holt, Rinehart and Winston, New York, 124-169.

Oberlander, J. and A. Lascarides (1992) "Preventing Temporal Implicatures: Iterative Defaults for Text generation," in *Proceedings of the 14th International Conference on Computational Linguistics*, 721-727.

Perrault, C.R. (1990) "An Application of Default Logic to Speech Act Theory," in Cohen, P. R., J. Morgan and M. E. Pollack, eds., *Intentions in Communication*, The MIT Press, Cambridge, Massachusetts, 161-185.

Pereira, F. C. N. and D. H. D. Warren (1980) "Definite Clause Grammars for Language Analysis — A Survey of the Formalism and Comparison with Augmented Transition Networks," ***Artificial Intelligence*** **13**, 231-278.