自然言語処理 71－1
（1989. 3. 18）

自然言語解析における意味的曖昧性を増進的に解消する計算モデル

奥村 学，田中 穂積
東京工業大学 工学部

[概要]
コンピュータによる自然言語解析で最も困難な問題は，文に含まれる様々な意味的曖昧性をどのように解消するかということである．文を解析するとき，曖昧性を文末まで解消しない方法をとると，一般に文が長くなるとその曖昧性の数は組合せ的に爆発してしまう．そのため，文末まで読み進み，膨大な数の曖昧性を得て，その各々について意味的妥当性を検討し曖昧性を解消しようとするシステムは非現実的であると言える．また，文に含まれる曖昧性は先行文や後続文を考慮しなければ解消できない場合もある．従って，曖昧性を解消するための望ましい方法は，文を解析する過程で得られる情報（制約）を増進的に蓄積し，できるかぎり早期の段階で曖昧性を段階的に解消していくことである．我々はこのような計算モデルを増進的曖昧性解消モデルと呼ぶ．

曖昧性解消を行うための情報は，解析結果を絞り込むための制約とみなすことができる．制約プログラミングは，バックトラックしない，実行が進むにつれ探索空間が徐々に小さくなるという特徴を持っている．一方，増進的曖昧性解消モデルは，文頭から文末への後戻りのない解析，得られた制約による増進的曖昧性解消という特徴を持っている．以上のことからわかるように，増進的曖昧性解消モデルは制約プログラミングの枠組みとよく整合する．本研究で提案する計算モデルは，論理型プログラム言語 Prologの拡張として，制約プログラミングを実現したものと考えることができる．

Incremental Disambiguation Model of Natural Langauge Analysis

OKUMURA Manabu and TANAKA Hozumi
Department of Computer Science, Tokyo Institute of Technology
(2-12-1 Oookayama Meguro-Ku Tokyo 152 Japan)

Abstract
Semantic disambiguation is a difficult problem in natural language analysis. The meaning of words is ambiguous because it cannot be uniquely determined unless information of other words in the sentence is obtained. A possible strategy for semantic disambiguation is to determine the meaning of words at the end of the sentence. This strategy might cause combinatorial explosion of the number of possible meanings, if the sentence is long. So we think it is impracticable for a system to determine the meaning of words from a number of candidates after it has finished reading the whole sentence. A desirable strategy for semantic disambiguation is to accumurate information obtained during the analysis process of a sentence and disambiguate the meaning incrementally by using the information. In this paper, we propose such a computational model of natural language analysis, called incremental disambiguation model.

1 はじめに

コンピュータによる自然言語解析で最も困難な問題は，文に含まれる様々な意味的曖昧性をどのように解消するかということである．自然言語に意味的曖昧性が遍在するのは，文を構成する各部分の情報がどれも部分的で，文の他の部分（先行文や後続文も含む）の情報を用いないと曖昧性を解消できないことに起因している．例えば，「はしをかけた川がはんらんした.」という文を考えてみよう．4章で述べるように，「はし」まで読み進んだ段階では「はし」の意味的曖昧性を解消できず，後続する「かけた」まで読み進み「かける」の意味を考慮しないと「はし」の意味的曖昧性を解消できない．一方，「かける」の意味は，「はし」や「川」の意味を考慮しないと決定できない．

文を解析するとき，曖昧性を文末まで解消しない方法をとると，一般に文が長くなるとその曖昧性の数は組合せ的に爆発してしまう．そのため，文末まで読み進み，膨大な数の曖昧性を得て，その各々について意味的妥当性を検討し曖昧性を解消しようとするシステムは非現実的であると言える．また，文に含まれる曖昧性は先行文や後続文を考慮しなければ解消できない場合もある．一般に自然言語解析において，完全に曖昧性が解消できる時点はないと言える．従って，曖昧性を解消するための望ましい方法は，文を解析する過程で得られる情報（制約）を増進的に蓄積し，できるかぎり早期の段階で曖昧性を段階的に解消していくことである．我々はこのような計算モデルを増進的曖昧性解消モデルと呼ぶ．これは人間の行っている曖昧性解消の方式に近いと考えられる．

意味的曖昧性を増進的に解消するモデルを実現する際には，以下に示す2つの問題を解決しなければならない：

- 曖昧性を含んだ意味解析結果をどのように表現するか，
- その表現を用いてどのように増進的に意味的曖昧性を解消していくか．

文の解析中途では，解析を終えた部分の情報は利用できるが，これから解析しようとする部分の情報は利用できない．そのため，未決定の部分が残った意味解析結果を得ることになる．このような意味解析結果は，解析が進み，後から得られる情報により，未決定の部分が新しく決定される表現でなければならない．本研究では，3章で述べる不定項という概念を提案し，このような表現を実現している．不定項は，意味的曖昧性を解消していく過程を表現すると同時に，解析の過程で得られ曖昧性解消に用いられる制約をも蓄積していく．

解析中途で得られる不定項は，文頭から文末へ文を読み進む過程で徐々に未決定の部分が決定されていく．これはBUP-UTI[1]と呼ばれる機構によって実現する．BUP-UTIは，解析の進む過程に沿って情報を伝播する機構であり，文を読み進む過程で伝播された不定項に対して，増進的に曖昧性を解消することができる．BUP-UTIの詳細に関しては，[1]を参照して頂きたい．曖昧性解消の基本計算機構は，3章で述べる不定項上での拡張ユニフィケーションを導入して実現する．

曖昧性解消を行うための情報は，解析結果を絞り込むための制約とみなすことができる．新しいプログラミング・パラダイムとして最近登場した制約プログラミング[5]は，「制約を能動的に用いて」，探索空間を段階的に小さくしていく手法であり，次の特徴を持っている：

- バックトラックしない，
- 実行が進むにつれ，探索空間が徐々に小さくなる．

一方，増進的曖昧性解消モデルは次の特徴を持っている：

- 文頭から文末への後戻りのない解析，
- 得られた制約による増進的曖昧性解消．

以上のことからわかるように，増進的曖昧性解消モデルは制約プログラミングの枠組みとよく整合する．従って，本研究で提案する計算モデルは，論理型プログラム言語 Prologの拡張として，制約プログラミングを実現したものと考えることができる．

本研究で提案する増進的曖昧性解消モデルと関連する研究として，Hirst[6]，Sowa[9]の研究があるが，2章で述べるように，これらは我々の増進的曖昧性解消モデルの計算機構の一部が実現されているにすぎない．拡張ユニフィケーションに関する研究としては，CIL[3]，赤間[2]があるが，3章で述べるように，これらは本研究で提案する拡張ユニフィケーションのサブセットである．

2章では意味的曖昧性のタイプについて述べる．3章では，2章で述べた2つのタイプの意味的曖昧性を統一的に処理する増進的曖昧性解消モデルを提案する．4章では，3章で提案した増進的曖昧性解消モデルを用いた自然言語解析の例を示す．5章ではまとめと今後の研究課題を述べる．

2 意味的曖昧性のタイプ

自然言語に含まれる意味的曖昧性を段階的に解消するためには，大別して次の2つのタイプの意味的曖昧性を考慮する必要がある：

(a) 意味するものとして複数の可能性が存在し，そのうちの1つに決定できない；
多義性(polysemy)や同音異義性(homonymy) などは

```
[ operate:
  [ cause-to-function
    agent          SUBJ
    patient        SUBJ,OBJ
    instrument     SUBJ,with
    method         by
    ... ]
  [ perform-surgery
    agent          SUBJ
    patient        upon,on
    instrument     with
    method         by
    ... ]]
```

図 1: 動詞'operate'のポラロイド語

このタイプである．単語に多義性があるとは，例えば「子供」のように，「(親に対する) 子」と，「(おとなに対する) 少年・少女」の少なくとも2通りの意味がある場合である．単語に同音異義性があるとは，「はし」のように，「橋」と「箸」の少なくとも2通りの意味がある場合である．このタイプの意味的曖昧性が解消されるということは，その可能性の数が減少することに対応する．

(b) 意味するものが十分に限定されていない；
例えば，現時点で「人間」であることがわかっている対象が，他の情報が得られた結果，「人間」より限定された「男」や，「年齢が27才の人間」に限定する場合である．この場合，この対象は，より限定された表現に変化しうるという意味で意味的曖昧性を持つ．そして，より限定された表現に変化していくことが意味的曖昧性の解消に対応する．未知語の意味を特定する処理や，「彼」,「それ」,「その男」などの指示語の指示対象を決定する処理（照応参照処理）にはこのタイプの意味的曖昧性解消が含まれる．

(a)のタイプの意味的曖昧性を解消する手法として，Hirst[6] は，ポラロイド語(Polaroid words)という考え方を提案している．ポラロイド語は，複数の可能性のある単語の意味をそのまま表現することが可能なデータであり，ポラロイド語同士が情報を交換し合うことで，曖昧性が解消されていく自律的なメカニズムである．図 1([6]より引用)に，ポラロイド語の例を示す．動詞'operate'には，'cause-to-function'と'perform-surgery'の2つの意味があり，それぞれ異なる格フレームを持つことを図 1は表現する．

(b)のタイプの意味的曖昧性に関して，Sowa[9]は，概念を表現したグラフの形成規則(formation rule)としてRestrictとJoinを定義している．Restrictは，同一化しようとしているグラフ中の対応するノード同士を比べ，片方がもう片方の下位概念の場合には，下位概念の方に置き換える規則である．Restrictにより，「人間」を表すグラフと「男」を表すグラフを同一化すると，「人間」はより限定された「男」に変化する．Joinは，同一化可能な2つのグラフをマージする演算である．「年齢が27才の男」を表すグラフと「名前が太郎である男」を表すグラフは，矛盾する部分がないので同一化可能であり，両者の間でJoinを行うと「年齢が27才の，太郎という名前の男」を表すグラフを得る．

本章で述べた2つのタイプの意味的曖昧性を解消するモデルについては，上に述べたように，それぞれHirstとSowaが個々に研究を進めているが，両者を統一的に処理する曖昧性解消のモデルにはなっていなかった．本研究で提案する増進的曖昧性解消モデルは，この2つのタイプの意味的曖昧性を統一的に処理する計算モデルである．次章では，このモデルについて述べる．

3 意味的曖昧性の増進的解消モデル

本章では，前章で述べた2つのタイプの意味的曖昧性を統一的に処理する増進的曖昧性解消モデルを提案する．3.1節では，増進的曖昧性解消モデルの中核となる計算機構として提案する拡張ユニフィケーションの基本的考え方について述べる．3.2節では，それをどのように実現するかを本研究で提案する不定項の説明とともに示す．

3.1 拡張ユニフィケーションの基本的考え方

本節では，前章で述べた2つのタイプの意味的曖昧性を統一的に解消するのに必要なユニフィケーションの機能について例を用いて述べる．

ユニフィケーションとは2つの表現を同一化しようとすることである．例えば，「庭に紫のバラが咲いている．私はその甘いにおいの花が好きだ．」という文では，「その甘いにおいの花」は「紫のバラ」を指す．これは照応関係と呼ばれるが，この照応関係は「甘いにおいの花」の表現と「紫のバラ」の表現をユニファイすることである．

図 2はユニフィケーションの例である．概念名の「花」と「バラ」をユニファイすると，「花」の下位概念である「バラ」がユニフィケーションの結果として得られる．これは，意味するものがより限定されたということで，(b)のタイプの意味的曖昧性解消を行なっていることになる．色は「紫」と，「紫」を要素として持つ集合（この色に関する集合は，「花」の色はこのうちのどれかであるという常識

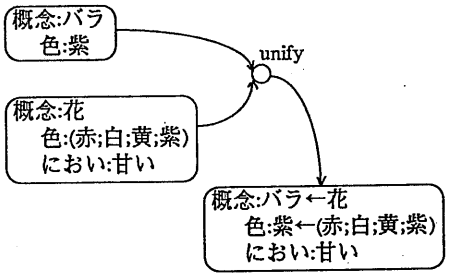

図 2:「紫のバラ」と「甘いにおいの花」のユニフィケーション

から得られるものとする）をユニファイすることにより，結果として「紫」を得る．これは，「色」という属性の値が複数の可能性の中から1つに決定されたということで，(a)のタイプの意味的曖昧性解消を行なっていることになる．また，「色」以外に新しく得られた「におい」に関する制約が表現に付加される．新しく付加された制約により表現している対象がより限定されるので，これは(b)のタイプの意味的曖昧性解消を行なっていることになる．

ユニフィケーションの結果得られた表現で，矢印は値の変化過程を表現する．値の変化は，曖昧性解消過程で新しく得られた制約により引き起こされる．従って，値の変化過程は，それらの制約の蓄積過程に対応する．また，「におい」のように，表現をより限定する制約（属性）は，表現中に増進的に蓄積される．このように表現中に制約を増進的に蓄積することの利点については5章で述べる．

3.2 増進的曖昧性解消モデルの実現

本節では，我々の増進的曖昧性解消モデルを Prolog 上でどのように実現するかについて述べる．

まず，2章で述べた2つのタイプの意味的曖昧性が Prolog 上でどのように表現できるかを見ることにする．複数の可能性がある場合，Prolog では，それらをそれぞれ1つの節に対応させ，複数の節で表現する．例えば，述語'p'を満足する値として 1, 3, 5 があり，'p'を満足するものを解として求める場合を考えてみよう．この場合，値として可能性のある 1, 3, 5 について，

```
p(1).    p(3).    p(5).
```

のように記述する．求める解が曖昧である（複数存在する）ことは，ゴール
$?-p(X).$
を実行し，バックトラックにより3つの節を順々に探索し解として 1, 3, 5 を得ることで判断される．しかし，この表現方法では，ある可能性（例えば，3）を吟味している時にはそれ以外の可能性(1, 5)が存在するかどうかがわからず，求める解が曖昧であるかどうかをゴールの実行中に判断することはできない．従って，この表現方法は，解析中に出現した意味的曖昧性を表現しておき，それを増進的に解消していく計算モデルの実現に不適当である．

増進的曖昧性解消を行なうためには，解析中に出現した意味的曖昧性を，上で述べたような潜在的な方法ではなく，明示的に表現しなければならない．例えば，述語'p'の例では，
$p((1;3;5)).$
のようにである（';'は選言(OR)を表現する）．項(1;3;5)は，'p'を満足する値として 1, 3, 5 の3つがあることを明示している．このように表現された曖昧性の解消過程は以下のようである．求める解の条件（制約）が'p'かつ'q'の場合を考える．
$q((3;5;7)).$
であるとすると，この場合求める解は(3;5)である．ゴール
$?-p(X), q(X).$
を実行すると，まず'p'を満足する集合(1;3;5)を得，次にこの集合が制約'q'を満足するかどうかを検査することになる．この検査は，集合(1;3;5)と(3;5;7)の積集合(intersection)を取ることに対応する．この検査はユニフィケーションにより実行されるが，Prolog のユニフィケーションはこのような機能を持たない．従って，このような表現方法を用いて曖昧性解消を行なうためには，ユニフィケーションの機能を拡張する必要がある．

値が増進的に限定されていくタイプの意味的曖昧性を Prolog の論理変数で表現するのは困難である．Prolog の論理変数は1度値が束縛されると，2度とその値を書き変えられないからである（値が束縛された時点までバックトラックし，値を他のものに変更することは可能であるが，これは今述べている値を限定する場合にはあたらない）．従って，増進的に値が変更可能な項を導入する必要がある．

以上述べたように，2つのタイプの意味的曖昧性を表現し，それを用いて増進的に曖昧性を解消していくためには，増進的に値が変更可能な項を導入するとともにユニフィケーションの機能を拡張する必要がある．以下では，我々の増進的曖昧性解消モデルの表現形式である不定項および拡張ユニフィケーションについて説明する．

まず，曖昧性を含んだ意味解析結果の表現形式として，不定項を導入する．不定項は，次の2種類に分類される．

1. 単純不定項

2. 複合不定項

複合不定項は，表現するものが対象（名詞句）であるのか，イベント（動詞句）であるのかにより，次の表現を取る．

1. 名詞句　$object(Con, Slots)$
   ここで,

   Con 表す概念

   Slots 持っている属性

   である.

2. 動詞句　$event(Con, Slots, Pol)$
   ここで,

   Con 表す概念

   Slots 取っている格スロット

   Pol 極性（そのイベントが成り立っている(1), 成り立っていない(0) を記述する）

   である.

$Slots$ は, $(SlotName_1 : SlotValue_1, ..., SlotName_n : SlotValue_n, \_)$ の形式であり, 属性（スロット）集合を表現する. 後述する CIL における部分項と同様に情報（制約）が付加され単純に増加していくデータ構造である. このデータ構造は, 3.1 節で述べた新しく得られる制約（属性）を蓄積していくものである. 図 2に示したユニフィケーションは, 図 3の不定項間のユニフィケーションとして実現される. ユニフィケーション後の不定項の属性を表す第2引数に「におい」に関する制約が付加されていることに注意して欲しい. 複合不定項の概念を表す部分(第1引数), スロット（属性）値を表す部分($SlotValue_i$)には, 単純不定項を記述する. 単純不定項は, $(SI_1, SI_2, ..., SI_n, \_)$ の形式であり, 項の最初の値が SI_1であり, その後新しい情報（制約）が得られた結果, 項の値が $SI_2, ..., SI_{n-1}$と変化し, 現在は SI_nであることを表現する. 単純不定項により, 図 2の矢印で示した値の変化過程を表現する. 図 3のユニフィケーション後の不定項の概念および色のスロット値を表す部分に注意して頂きたい. 2 章で述べた(b)のタイプの意味的曖昧性解消は単純不定項により実現される. また, SI_iの値として, アトムだけでなく,

- 選言(disjunction)

  $$(v_1; v_2; ...; v_n)$$

  ここで, v_iはアトムである.

- 否定(negation)

  $$-v$$

  ここで, vはアトムまたは選言である.

を許すことで, (a)のタイプの意味的曖昧性も表現し増進的に解消することが可能である. 図 3のユニフィケーション後の不定項は,「花」であった概念が「バラ」に限定され, また, 色も4つの可能性の中から「紫」に決定されている.

```
unify(object((バラ,_),(色:(紫,_),_)),
      object((花,_),(色:((赤;白;黄;紫),_),
                      におい:(甘い,_),_)))
                      ↓
      object((花,バラ,_),
             (色:((赤;白;黄;紫),紫,_),
              におい:(甘い,_),_))
```

図 3: 不定項間のユニフィケーション

次に, 複合不定項を扱えるようにユニフィケーションを拡張する. 複合不定項上でのユニフィケーションは以下の処理を行なう.

概念部 概念階層を検索し, 概念同士が同一化可能かどうかを調べる. 同一化可能な場合, より下位の概念に同一化する.

スロット（属性）部 同一スロット名のスロットはスロット値のユニフィケーションを行なう. また, 新しく得られたスロットの付加を行なう.

例えば,「犬」と「猫」の同一化は, それらが排他的な概念であることから失敗する.「人間」と「動物」の同一化は,「人間」が「動物」の下位概念であることから成功し, 結果として「人間」を得る. このような概念の同一化機構は, Sowa[9]や Dahlgren[4]などでも用いられている.

属性（スロット）集合間のユニフィケーションは, 集合間の和集合(union)を取る演算とみなされる. 例えば, 属性集合 {名前 : 太郎,性別 : 男,年齢 : 27} と {名前 : 太郎,趣味 : 音楽観賞 } をユニファイすると, 結果として, 属性集合 {名前 : 太郎,性別 : 男,年齢 : 27,趣味 : 音楽観賞 } を得る. しかし, 属性集合 {名前 : 太郎,性別 : 男,年齢 : 27} と {名前 : 花子,趣味 : 編みもの } のユニフィケーションは,「名前」スロットの値がユニファイ不可能なため失敗する.

また, 単純不定項の値として選言, 否定を許したことにより, ユニフィケーションも選言, 否定を扱えるように拡張する. 選言, 否定を含む項上でのユニフィケーションは図 4(⊗はユニフィケーションを表す演算記号とする）に示すようになる[8,7]. 図 4で, 矢印の右辺はユニフィケーションの結果を, ifの右側はユニフィケーションが成功するための条件を表す. また, '\+'は Prolog の否定を表し, $(a_i \mid P(a_i))$は, Pを満足する a_iを選言でつないだ集合を表す. $union(a, b)$は, a,b の和集合を表す関数とする.

$A \otimes B$:

1. $a \otimes b$ (アトム同士)
   $\longrightarrow a\ if\ a == b$
   例:$a \otimes a \longrightarrow a$
   $a \otimes b \longrightarrow$ 失敗

2. $a \otimes -b$ (アトムと否定)
   $\longrightarrow a\ if \setminus + unify(a,b)$
   例:$a \otimes -b \longrightarrow a$
   $a \otimes -a \longrightarrow$ 失敗

3. $a \otimes (b_1;...;b_n)$(アトムと選言)
   $\longrightarrow a\ if\ \exists j\ unify(a,b_j)$
   例:$a \otimes (c;a;t) \longrightarrow a$
   $a \otimes (d;o;g) \longrightarrow$ 失敗

4. $(a_1;...;a_n) \otimes -b$ (選言と否定)
   $\longrightarrow (a_i \mid \setminus + unify(a_i,b))$
   例:$(a;b;c) \otimes -b \longrightarrow (a;c)$

5. $(a_1;...;a_n) \otimes (b_1;...;b_n)$(選言同士)
   $\longrightarrow (a_i \mid \exists j\ unify(a_i,b_j))$
   例:$(a;b;c) \otimes (b;c;d) \longrightarrow (b;c)$

6. $-a \otimes -b$ (否定同士)
   $\longrightarrow -union(a,b)$
   例:$-a \otimes -b \longrightarrow -(a;b)$

図 4: 選言, 否定のために拡張されたユニフィケーション

CIL[3]には, 部分項という概念が導入されている. 部分項はラベルと値の対の集合として定義される. 具体的には, $\{a_1/b_1,.....a_n/b_n\}$ の形式で記述する. ここで, a_iがラベルであり, b_iが値である. 部分項はその名前の通り, 対象の持っている情報の一部を表現したものであり, 実行が進むにつれラベル-値の対が付加・蓄積されていく. しかし, 部分項は単調に増加するデータ構造にすぎず,

- 選言, 否定的制約を扱えない.
- 新しい情報 (制約) により値が限定されていくような場合を扱えない.

という問題点がある.

PAL の集合束縛変数[2]は, 変数に制約としてユニファイ可能な集合を指定し, $X^{\wedge}$ < 制約 >のような記述を許すものである. 例えば, $X^{\wedge}$動物は, Xが「動物」の下位概念の個体としか束縛されないことを表現する. また, $X^{\wedge}${ 犬 ,猫,豚,鶏 } のように, いくつかの要素からなる集合を記述するデータ型もある. これは Xが「犬」,「猫」,「豚」,「鶏」のどれかであることを表現する. 問題点としては,

- 3 つの型の集合束縛変数がカバーする範囲が離散的である.
- 異なる型の変数同士のユニフィケーションができない.

ことが挙げられる.

4 選択制限を用いた増進的曖昧性解消

本章では, 我々の計算モデルによる選択制限を用いた増進的曖昧性解消の例を示す.

例えば, 図 5のような格フレームを考えてみよう. 各スロットの$と→の間の部分が選択制限の記述である.

選択制限は, 動詞の格フレームの各スロットを満たすもの (フィラーと呼ばれる) に関する条件 (制約) であり, これを満足しない意味は不適格であるとされる. 従って, 選択制限に違反する意味を排除することにより, フィラーの意味的曖昧性は (部分的に) 解消される. 未知語の意味は, その未知語が満たす格スロットの選択制限により (部分的に) 特定できる. 照応参照処理でも同様に選択制限の情報を用いることができる.

また, 動詞の格フレームが動詞の意味と 1 対 1 対応をしているという前提に立つと, 名詞句が動詞の格スロットを満たす際, 選択制限を用いて複数ある格フレームのうち妥当なもののみを選択することにより, 動詞の意味的曖昧性を (部分的に) 解消することができる.

フィラーが選択制限を満足するかどうかの検査は, 前章で述べた拡張ユニフィケーションを用いて実行する. 選択

```
かける
  [ 欠ける::
    [ が $ Con -> 対象 ]]
  [ 賭ける::
    [ が $ Hum -> 動作主 ]
    [ に $ Abs -> 目標 ]
    [ を $ Qua -> 対象 ]]
  [ 掛ける::
    [ が $ Hum;Org;Ani -> 動作主 ]
    [ に $ Loc -> 場所 ]
    [ を $ Pro -> 対象 ]]
  [ 駆ける::
    [ が $ Hum;Ani -> 動作主 ]
    ( を $ Loc -> 経由 )]
```

図 5: 動詞「かける」の複数の格フレーム

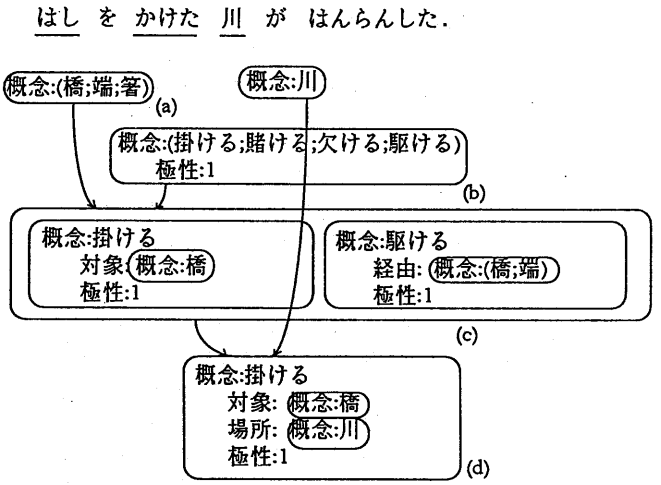

図 6:「はしをかけた川がはんらんした.」の意味解析過程

制限の検査を拡張ユニフィケーションで行なうことにより，選択制限がフィラーに対し制約として課され，フィラーの不定項はより限定された表現となり（概念名やスロット値がより限定された値になるか，または新しいスロット名の属性が付加され），フィラーの意味的曖昧性が解消される．

拡張ユニフィケーションを実行する述語をプリミティブとして用いた自然言語解析の例を図 6(本図では，値の変化過程は省略し，最新の値のみを記述する）に示す．図 6 の例では，「はし」に 3 通りの意味的曖昧性があり（表現(a)参照），「かける」に 4 通りの意味的曖昧性がある（表現(b)参照）．「はしを」まで読み進んだ段階では「はし」の意味的曖昧性は解消されないが，「かけた」を読んだ時点で「はし」の意味は「橋」と「端」のどちらかであると限定することができる（表現(c)参照）．同時に「かける」の意味も「掛ける」と「駆ける」のどちらかであると限定される．これは，図 5に示した「かける」の格フレームから，

1. 「欠ける」は，「を」格スロットを持たないので不適格である．
2. 「賭ける」の「を」格スロットの選択制限を「はし」の 3 つの意味がいづれも満足しないので，「賭ける」は不適格である．
3. 「掛ける」の「を」格スロットの選択制限を「橋」は満足するが，「箸」と「端」は満足しない．
4. 「駆ける」の「を」格スロットの選択制限を「橋」と「端」は満足するが，「箸」は満足しない．

ことが，拡張ユニフィケーションにより判断された結果である．その後，「川」まで読み進むと，依然曖昧であった「かける」の 2 通りの意味のうち，「駆ける」の意味が排除され，「かける」の意味は「掛ける」に決定される（表現(d)参照）．「川」が「駆ける」の残された格スロットである「が」格スロットの選択制限を満足しないからである．その結果，「はし」の意味的曖昧性も解消され，残っていた 2 通りの意味のうち，「橋」に「はし」の意味が決定される．

しかし，選択制限の利用には，以下に示す比喩文，否定文，疑問文，異常事態を記述する文を解析する際に問題がある．

比喩文「アイデアが開花する.」

否定文「ねずみは空を飛ばない.」

疑問文「あひるは本を読みますか？」

異常事態を記述する文「赤ちゃんがビー玉を飲む.」

上の文中の下線部の句が動詞の選択制限に違反するにもかかわらず，意味的には妥当であるからである．この問題は次のようにして解決できると考えられる：

> 選択制限の記述に優先性(preference)を導入し，曖昧性解消時に，選択制限に違反する意味を排除しないで，その優先度を下げるにとどめる．

5 おわりに

自然言語解析の計算モデルとして，文を解析する過程で得られる情報（制約）を増進的に蓄積し，できるかぎり早期の段階で意味的曖昧性を段階的に解消していくモデルを

提案した．提案した計算モデルは，2章で述べた2つのタイプの意味的曖昧性を統一的に処理することができる．提案した計算モデルは，論理型プログラム言語 Prolog の拡張として，制約プログラミングを実現したものと考えることができる．

我々の計算モデルは，曖昧性解消過程で得られた制約を増進的に蓄積することで，意味的曖昧性を増進的に解消している．制約を増進的に蓄積することの利点は次のようなものである．

1. 意味解析で用いる選択制限は絶対的なものではなく，選択制限は優先性として考える必要があることを前章で述べた．そのように選択制限を記述した場合も，優先度があるしきい値を超えた意味しか妥当なものでないとして，選択制限を満足しない意味を排除することで，意味的曖昧性の組合せ的爆発を回避し，増進的曖昧性解消を可能にする．また，そのようにして選択制限を満足しない意味を排除した結果，解析に失敗した場合にも，値の変化過程として蓄積していた意味のうち，排除したものを再び考慮することにより，選択制限に違反する意味的に妥当な文の処理を容易に実現することができる．

2. 3章で述べたように，得られた制約の蓄積過程として値の変化を記録しておくことにより，最初は全くわからない未知語の意味や，「それ」などの指示語の指示する対象の意味の特定が容易に実現できる．

本研究で提案した増進的曖昧性解消モデルに関する今後の研究課題としては，以下のものが挙げられる．

1. 3.2 節で述べたように，対象に対する制約は，その対象を表現する複合不定項の属性（スロット）の部分に単に蓄積しているだけである．それらの制約は，他の制約と矛盾する可能性がある．従って，スロット間のユニフィケーションには，無矛盾性検査の機能を付加する必要がある．

2. 本研究では，曖昧性解消に用いることができる情報の1つとして選択制限を用いた，主に1文を対象とした増進的曖昧性解消について述べた．選択制限以外の情報，例えば，常識的知識を用いた曖昧性解消については今後研究する予定である．これらも本研究で提案した計算モデルで自然に実現できると考える．

謝辞

ICOT 自然言語理解(NLU)ワーキンググループの皆様には，大変有意義なコメントを頂きました．ここに感謝の意を表します．

参 考 文 献

[1] 奥村学, 田中穂積．BUP系解析システム上でのトップダウンな情報の制御について．情報処理学会論文誌, 29(11):1043–1050，1988．

[2] 赤間清．集合束縛変数に基づく意味表現とユニフィケーション．情報処理学会自然言語処理研究会報告, 62:53–60，1987．

[3] *CIL* 言語マニュアル．新世代コンピュータ技術開発機構，第3版，1988．

[4] K. Dahlgren and J. McDowell. Kind types in knowledge representation. In *Proc. of the 11th International Conference on Computational Linguistics*, pages 216–221, 1986.

[5] M. Dincbas. Constraints, logic programming and deductive databases. In *Proc. of the France-Japan Artificial Intelligence and Computer Science Symposium'86*, pages 1–27, 1986.

[6] G. Hirst. *Semantic Interpretation and the Resolution of Ambiguity. Studies in Natural Language Processing*, Cambridge University Press, 1987.

[7] L. Karttunen. Features and values. In *Proc. of the 10th International Conference on Computational Linguistics*, pages 28–33, 1984.

[8] R.T. Kasper and W.C. Rounds. A logical semantics for feature structures. In *Proc. of the 24th Annual Meeting of the Association for Computational Linguistics*, pages 257–266, 1986.

[9] J.F. Sowa. *Conceptual Structures : Information Processing in Mind and Machine.* Addison-Wesley, 1984.