

## 一般化されたチャート法によるアブダクション計算

伝 康晴

ATR 音声翻訳通信研究所

〒619-02 京都府相楽郡精華町光台 2-2

統合的自然言語処理は、不完全な部分情報を多様に組み合わせて問題を解決する必要がある音声対話の理解 / 翻訳 / 生成などの目的のためには不可欠である。なかでも、アブダクション計算に基づくモデルは実際的な問題に対して有力である。本研究では、統合的自然言語処理機構におけるアブダクション計算を効率的に行なう手法について述べる。本稿では、先に提案した一般化されたチャート法に基づく演繹計算機構に仮説を扱うための機能を導入することによって、アブダクション計算の手法に拡張する。拡張されたアルゴリズムの詳細とともに統合的自然言語処理への応用を目して簡単な例題を示す。この手法は会話文解釈をアブダクションに基づいて行なうモデルへの応用を目標としている。

## A Generalized Chart Algorithm for Abduction

Yasuharu DEN

ATR Interpreting Telecommunications Research Laboratories

2-2 Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-02, Japan

In the research areas such as interpretation/translation/generation of spoken dialogues, an integrated architecture for natural language processing is strongly required, since no single sort of (partial) information is decisive to derive appropriate results, and thus there should be diverse interactions among several sorts of information. Abduction is one of the powerful approaches to realize practical integrated NLP systems. In this paper, we propose an efficient computation mechanism for abduction based integrated NLP systems. Our algorithm proposed so far, called a *generalized chart based prover*, is modified to treat assumptions, and is extended to an abductive prover. We show details of the new algorithm along with simple examples aiming at its application to integrated NLP systems.

## 1 はじめに

近年、自然言語処理を統合的に行なう機構の研究が注目されている [5]. このような統合的自然言語処理機構は、不完全な部分情報を多様に組み合わせて問題を解決する必要のある音声対話の理解 / 翻訳 / 生成といった目的のためには不可欠である。筆者らは日本語会話文解釈の枠組 [3] を提案しているが、ここでも多様な情報の組合せ的な処理は不可欠である。このモデルは計算機構として Hobbs ら [6] と同様なアブダクション計算を用いている。句構造規則、統語 / 意味 / 談話制約などはいずれも非常に緩く規定されており、制約の組合せによる探索空間の爆発を防ぐためにアブダクション計算を効率的に行なう必要がある。

本研究では、統合的自然言語処理機構におけるアブダクション計算を効率的に行なう手法について述べる。この手法は筆者らの会話文解釈モデルの処理エンジンとして応用することを目標としている。アブダクション計算を効率化するために、本研究では以下の技法を用いる。

1. 目標駆動の上昇型探索
2. チャートによる部分解の再利用

本手法は、構文解析で広く用いられているチャート法の拡張とみることができ、一般化されたチャート法に基づくアブダクション計算 (Generalized Chart based Abductive Prover, GCAP) と呼んでいる。GCAP の中心的な部分はすでに [1] で説明した。本稿では、そこで示した演繹計算の手法 GCP をアブダクション計算の手法 GCAP に拡張する方法について述べる。

本稿の構成は以下の通りである。2 節で GCP のアイデアとアルゴリズムを簡単に復習する。3 節で、それを GCAP に拡張する方法について述べ、GCAP アルゴリズムの形式的な定義を与える。4 節では、簡単な例題を用いて GCAP の統合的自然言語処理への応用をみる。最後に、5 節でまとめを述べる。

## 2 一般化されたチャート法に基づく演繹計算

### 2.1 主辞駆動型導出

ホーン節の演繹計算を単純な下降型 / 上昇型探索で行なうと、以下のような問題が生じる。

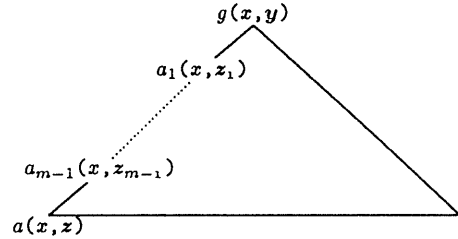


図 1: 主辞駆動型導出

$m(X, [A|Z]) :- h(X, A, Y), m(Y, Z).$   
 $m(\{\square, \square\}, \square).$   
 $h(\{[A|X], Y\}, A, \{X, Y\}).$   
 $h(\{X, [A|Y]\}, A, \{X, Y\}).$

図 2: 主辞駆動型公理の例

下降型探索 規則選択の誤りに起因する探索の行き詰まりによって頻繁な後戻りが生じる。

上昇型探索 初期目標の導出につながらない無駄な命題を数多く導出してしまふ。

しかし、ホーン節がある種の性質を備えているとき、目標駆動の上昇型探索によりこの問題を解決できる。

いま、目標  $g(x, y)$  の導出過程が図 1 に示す形になるとする。すなわち、目標  $g(x, y)$  の第 1 引数  $x$  が左隅にある事実  $a(x, z)$  にいたるまでの経路上で共有されている。ある公理を用いて初期目標を導出する過程において、すべての部分目標について図 1 の形の部分導出木が構成できるとき、この公理は主辞駆動型 (head-driven) であるという。公理が主辞駆動型であるためには、すべての規則  $A :- B_1, \dots, B_n$  について、右辺の先頭の要素式  $B_1$  と左辺の要素式  $A$  の第 1 引数が等しくなければならない。このとき、右辺の先頭の要素式を主辞 (heads) と呼び、その第 1 引数を主辞引数 (head arguments) と呼ぶ。このように主辞駆動型公理では規則の形に制限があるが、GCP では主辞を持たない規則の記述も許している。主辞を持つ規則を連鎖規則 (chain rules)、主辞を持たない規則を非連鎖規則 (non-chain rules) と呼ぶ。図 2 は主辞駆動型公理の例である (公理は Prolog の記法で書く)。

公理が主辞駆動型のとき、目標駆動の上昇型探索によって導出を効率的に行なえる。この導出過程を主辞駆動型導出 (head-driven derivation) と呼ぶ。主辞駆動型導出を図 1 を用いて簡単に説明する。まず、目標  $g(x, y)$  から主辞引数  $x$  を取り出

し、これを索引にして事実  $a(x, z)$  を検索する。次に、この要素式  $a(x, z)$  と単一化可能な主辞を持つ連鎖規則  $a_{m-1}(x, z_{m-1}) :- a(x, z), B_1, \dots, B_n$  を検索し、姉妹  $B_1, \dots, B_n$  と親  $a_{m-1}(x, z_{m-1})$  を導入する。このとき、要素式  $a(x, z)$  をこの導出の核 (pivots) と呼ぶ。姉妹  $B_1, \dots, B_n$  がすべて再帰的に導出されたら、親  $a_{m-1}(x, z_{m-1})$  も導出される。親が目標  $g(x, y)$  と単一化可能なら、親と目標を単一化して目標の導出を完了する。そうでなければ、親を核として再び同様な導出を繰り返す。

## 2.2 GCP の計算手続き

主辞駆動型導出は左隣構文解析 / 意味主辞駆動生成を一般化したものとみなせる。深さ優先の後戻り法に基づくこれらの構文解析 / 生成が部分分解の再計算という問題を抱えているのと同様に、深さ優先の主辞駆動型導出にも同じ問題がある。そこで、構文解析 / 生成の場合 [7][4] にならって、チャート (charts) を用いて部分分解の再利用を行なう。ただし、ここで用いるチャートは以下の点で構文解析の場合のチャートを拡張している [4]。

1. GCP の語彙弧は部分目標の主辞引数を索引として事実を検索することによって導入されるので、語彙弧の導入は漸時的に行なわなければならない。
2. ある部分目標の次に処理される部分目標として、主辞引数の異なる複数のものがありえるので、GCP の弧の接続 (主辞引数の並び) は多対多になる。

GCP の計算手続きは、ボトムアップチャート構文解析の手続きに上記の拡張を施すことによって得られる。図 3 が GCP の計算手続きである。図 2 の主辞駆動型公理を用いて目標  $m(\{[a], [b]\}, Z)$  を GCP アルゴリズムによって導出する過程を図 4、表 1 に示す。多対多になる弧の接続はポイント A-D によって表されている。

## 3 一般化されたチャート法に基づく

### アブダクション計算

#### 3.1 自然言語処理とアブダクション

アブダクションは、不完全な情報に基づく自然言語の理解 / 翻訳 / 生成にとって有効である [6]。例えば、

(1) を考えよう。

(1) IBM 買った。

手続き 1 位置  $[s, t]$  に活性弧  $[\dots[?]B_j\dots]A$  がある。ただし、 $B_j = b_j(x, y)$  は最左空所。主辞引数  $x$  で決まるチャート上の位置を  $[u, v]$  とする。

1.  $[u, v]$  がはじめて導入される位置ならば、以下を行ない、節点  $t$  から  $u$  にポインタをつける。
  - (a) すべての事実  $a(x, z)$  に対し位置  $[u, v]$  に不活性弧  $a(x, z)$  を張る。
  - (b) すべての非連鎖規則  $a(x, z) :- B_1, \dots, B_n$  に対し位置  $[u, v]$  に活性弧  $[[?]B_1\dots[?]B_n]a(x, z)$  を張る。
2.  $[u, v]$  がすでに導入されている位置ならば、単に節点  $t$  から  $u$  にポインタをつける。

手続き 2 位置  $[s, t]$  に不活性弧  $C$  がある。このとき、 $A = C$  であるすべての規則  $A' :- A, B_1, \dots, B_n$  に対し位置  $[s, t]$  に活性弧  $[A][?]B_1\dots[?]B_n]A'$  を張る ( $n = 0$  なら不活性弧  $A'$  を張る)。

手続き 3 位置  $[s, t]$  に活性弧  $[\dots[?]B_j[?]B_{j+1}\dots[?]B_n]A$  があり、位置  $[u, v]$  に不活性弧  $C$  がある。ただし、 $B_j$  は最左空所。さらに、節点  $t$  から  $u$  にポインタがある。このとき、 $B_j = C$  であれば節点  $[s, v]$  に活性弧  $[\dots B_j[?]B_{j+1}\dots[?]B_n]A$  を張る ( $j = n$  なら不活性弧  $A$  を張る)。

図 3: GCP の計算手続き

この文を理解するには、

1. “IBM” と “買った” の間に対象格として解釈できる意味関係がある。
2. “IBM” は “IBM の製品” を提喻している。

ことなどがわかる必要がある。しかし、これらは事実から演繹的には推論されない。むしろ、

1. 対象格は意味関係を形成し得る。  
 $sem(E, X) :- object(E, X).$
2. 生産者と製品の関係は提喻を形成し得る。  
 $metonymy(X, Y) :- produce(X, Y).$

といった公理を用いて、文に含まれている事実  $sem(buy, product)$ ,  $metonymy(ibm, product)$  を説明する仮説  $object(buy, product)$ ,  $produce(ibm, product)$  を導く形で推論しなければならない。この推論はアブダクションである。

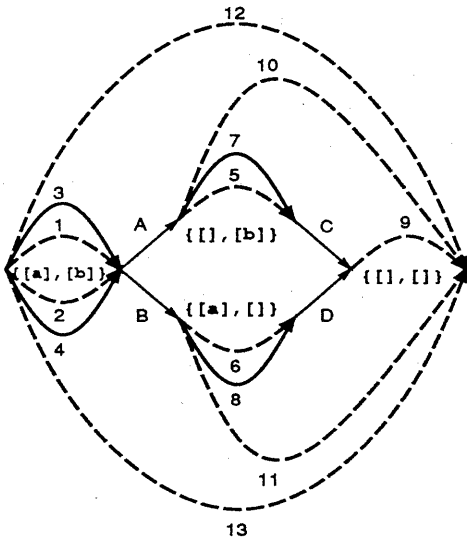


図 4:  $m(\{[a], [b]\}, Z)$  の導出で構成されるチャート

表 1:  $m(\{[a], [b]\}, Z)$  の導出で構成されるチャート

#	Arc	From
1	$h(\{[a], [b]\}, a, \{[ ], [b]\})$	—
2	$h(\{[a], [b]\}, b, \{[a], [ ]\})$	—
3	$[h(\{[a], [b]\}, a, \{[ ], [b]\})$ $[?]m(\{[ ], [b]\}, Z_1)$ $m(\{[a], [b]\}, [a Z_1])$	1
4	$[h(\{[a], [b]\}, b, \{[a], [ ]\})$ $[?]m(\{[a], [ ]\}, Z_2)$ $m(\{[a], [b]\}, [b Z_2])$	2
5	$h(\{[ ], [b]\}, b, \{[ ], [ ]\})$	3
6	$h(\{[a], [ ]\}, a, \{[ ], [ ]\})$	4
7	$[h(\{[ ], [b]\}, b, \{[ ], [ ]\})$ $[?]m(\{[ ], [ ]\}, Z_3)$ $m(\{[ ], [b]\}, [b Z_3])$	5
8	$[h(\{[a], [ ]\}, a, \{[ ], [ ]\})$ $[?]m(\{[ ], [ ]\}, Z_4)$ $m(\{[a], [ ]\}, [a Z_4])$	6
9	$m(\{[ ], [ ], [ ]\})$	7, 8
10	$m(\{[ ], [b]\}, [b])$	$7+C+9$
11	$m(\{[a], [ ]\}, [a])$	$8+D+9$
12	$m(\{[a], [b]\}, [a, b])$	$3+A+10$
13	$m(\{[a], [b]\}, [b, a])$	$4+B+11$

### 3.2 GCAP の計算手続き

アブダクションは演繹の拡張とみることができ、すなわち、部分目標  $G$  を証明する代わりに  $G$  を仮説として仮定するという操作を導入することによって、演繹計算はアブダクション計算に拡張できる。したがって、GCAP の計算手続きは、図 3 にあげた GCP の手続き 1 において、活性弧の最左空所  $b_j(x, y)$  から仮説  $b_j(x, y)$  を導入する操作を追加することによって得られる。

このとき、導入した仮説は不活性弧として扱う。つまり、GCAP において仮説と事実とは全く同等に扱われる。一般に、目標の導出過程において仮説と事実が果たす役割は同じである。仮説が事実と異なるのは、「目標が複数の導出を持つとき、それらの優劣を評価する際の指標としてどのような仮説が作られたかが考慮される」という点である。しかし、このような導出の良さの評価はアブダクションによる計算とは別に（例えば、コスト最小化原理 [6] などに基づいて）行なうので、計算という観点から見れば、仮説と事実を区別する必要はない。もちろん、アブダクションにおいて導出の良さの評価は欠かせないから、チャート中の弧には導出の評価を受けるための指標（すなわち、ある導出によって得られた弧にどのような仮説が内在しているか）が表現されている必要がある。そのために、GCAP では、チャート上の弧に仮説集合 (assumption sets) を付随させる。各弧の仮説集合は次のようにして決定する。

1. 仮説不活性弧  $A$  の仮説集合は  $\{A\}$  である。
2. 事実不活性弧  $A$  の仮説集合は  $\phi$  (空集合) である。
3. 仮説集合  $S$  を持つ不活性弧  $A$  から手続き 2 によって作られる弧  $B$  の仮説集合は  $S$  である。
4. 仮説集合  $S_1$  を持つ活性弧  $A_1$  と仮説集合  $S_2$  を持つ不活性弧  $A_2$  から手続き 3 によって作られる弧  $B$  の仮説集合は  $S_1 \cup S_2$  である。

以上の拡張を施した GCAP の計算手続きは図 5 のようになる (弧  $A:S$  の  $S$  は仮説集合を表す)。

### 3.3 仮定可能性の明示

図 5 にあげた GCAP の計算手続きでは、すべての要素式は仮定可能 (assumable) であるとした (ただし、仮説は手続き 1 によってのみ導入されるので、連鎖規則の主辞は仮定可能でない)。しかし、公理の中には仮定すべきでない要素式が含まれるのが普通

手続き1 位置  $[s,t]$  に活性弧  $[\dots[?]B_j\dots]A$  がある。ただし、 $B_j = b_j(x,y)$  は最左空所。主辞引数  $x$  で決まるチャート上の位置を  $[u,v]$  とする。

1.  $[u,v]$  がはじめて導入される位置ならば、以下を行ない、節点  $t$  から  $u$  にポインタをつける。
  - (a) すべての事実  $a(x,z)$  に対し位置  $[u,v]$  に不活性弧  $a(x,z):\phi$  を張る。
  - (b) すべての非連鎖規則  $a(x,z) :- B_1, \dots, B_n$  に対し位置  $[u,v]$  に活性弧  $[[?]B_1\dots[?]B_n]$   $a(x,z):\phi$  を張る。
  - (c) 位置  $[u,v]$  に不活性弧  $B_j:\{B_j\}$  を張る。
2.  $[u,v]$  がすでに導入されている位置ならば、節点  $t$  から  $u$  にポインタをつける。さらに、 $B_j$  がまだ導入されていない仮説なら、位置  $[u,v]$  に不活性弧  $B_j:\{B_j\}$  を張る。

手続き2 位置  $[s,t]$  に不活性弧  $C:S$  がある。このとき、 $A = C$  であるすべての規則  $A' :- A, B_1, \dots, B_n$  に対し位置  $[s,t]$  に活性弧  $[A[?]B_1\dots[?]B_n]A':S$  を張る ( $n = 0$  なら不活性弧  $A':S$  を張る)。

手続き3 位置  $[s,t]$  に活性弧  $[\dots[?]B_j[?]B_{j+1}\dots[?]B_n]A:S_1$  があり、位置  $[u,v]$  に不活性弧  $C:S_2$  がある。ただし、 $B_j$  は最左空所。さらに、節点  $t$  から  $u$  にポインタがある。このとき、 $B_j = C$  であれば節点  $[s,v]$  に活性弧  $[\dots B_j[?]B_{j+1}\dots[?]B_n]A:S_1 \cup S_2$  を張る ( $j = n$  なら不活性弧  $A:S_1 \cup S_2$  を張る)。

図 5: GCAP の計算手続き

である。例えば、 $s(X,Z) :- pp(X,Y), vp(Y,Z)$  のような句構造規則は演繹的に使われるべきであり、 $pp(X,Y), vp(Y,Z)$  は仮定すべきでない。そこで、要素式の仮定可能性 (assumability) を公理中に明示する。このとき、GCAP の手続き 1 の 1c 項と 2 項は最左空所  $B_j$  が仮定可能な場合のみ適用される。

要素式のうち仮定可能なものには  $\circledast$  を前置する。例えば、意味関係を対象格に解釈する非連鎖規則

- (2)  $sem(s(E,X)) :-$   
 $\quad buy(E),$   
 $\quad product(X),$   
 $\quad \circledast object(s(E,X)).$

において、要素式  $object(s(E,X))$  は仮定可能であり、 $buy(E), product(X)$  は仮定可能でない。こ

で、仮定可能性は述語名に対してではなく、要素式の出現位置に対して定義されることに注意せよ。すなわち、同じ述語名を持つ要素式が、ある規則では仮定可能であり、別の規則では仮定不能な場合がある。例えば、“買った”の語彙は (3) のような規則で記述できるが、そこでは要素式  $buy(E)$  は仮定可能である。一方、この要素式は規則 (2) では仮定不能であった。

- (3)  $vp([買った|Z], Z, E) :-$   
 $\quad \circledast buy(E).$

仮定可能性の違いが意味するのは次のようなことである。仮定可能な要素式は、初期目標を真にすることによって新しく得られる情報を表す (入力から得られる新情報)。例えば、(3) は、“買った”という単語を聞くことによって、 $buy(E)$  なる事態  $E$  の存在が新情報として得られることを表している。一方、仮定不能な要素式は、処理時点で既に持っていなければならない情報を表す (前提となる旧情報)。例えば、(2) は、 $sem(s(E,X))$  から  $object(s(E,X))$  を仮定するには前提  $buy(E)$  および  $product(X)$  が満たされなければならないことを表している。それゆえ、仮定可能 / 不能はコスト付きアブダクション [6] におけるコストの低 / 高に概ね対応する。

ある部分目標  $G_1$  を導出する際に前提として使われる要素式  $A$  は、別の部分目標  $G_2$  の導出過程において仮定されたものであっても構わない。すなわち、複数の部分目標の導出が仮説を共有してよい。これは仮説の因子化 (factoring) に相当する。

#### 4 統合的自然言語処理への応用

GCAP の統合的自然言語処理への応用として、統語 / 意味 / 語用論処理の統合を取り上げる。

図 6 は 3.1 節の (1) を解釈するのに必要な公理である。句構造規則は緩く書かれており、助詞の欠落した格要素を許す。動詞と格要素の結び付きは、動詞の低位範疇化によるのではなく、係り受けの制約によって規定される (したがって、統語的には任意の格要素と動詞の間に係り受けが成立し得る)。係り受け関係は、格要素  $X$  が提論する対象  $Y$  (特別な場合として  $X$  自身を含む) と動詞が指示する対象  $E$  との間に何らかの意味関係が成立する場合に成り立つ。意味関係としては助詞  $ガ$  でマークされるものと助詞  $ヲ$  でマークされるものがあり、助詞が明示されていない (欠落している) 場合はいずれにもなり得る。  $ガ$  でマークされる意味関係

```

% 句構造規則 (統語制約)
s(X, Z, VP) :-
  pp(X, Y, P, NP),
  vp(Y, Z, VP),
  depend(d(P,VP,NP)).
s(X, Z, VP) :-
  np(X, Y, NP),
  vp(Y, Z, VP),
  depend(d(VP,NP)).

pp(X, Z, P, NP) :-
  np(X, Y, NP),
  p(Y, Z, P).

% 係り受け制約
depend(d(R,E,X)) :-
  metonymy(m(X), Y),
  sem(s(R,E,Y)).
depend(d(E,X)) :-
  metonymy(m(X), Y),
  sem(s(E,Y)).

% 語彙
np([IBM|Z], Z, NP) :-
  {inst(NP)},
  @ ibm(NP).
p([を|Z], Z, P) :-
  {inst(P)},
  @ wo(P).
vp([買った|Z], Z, VP) :-
  {inst(VP)},
  @ buy(VP).

% 語用論制約
metonymy(m(A), A).
metonymy(m(A), B) :-
  company(A),
  {inst(B)},
  @ produce(s(A,B)),
  @ product(B).

% 一般知識
company(X) :-
  ibm(X).

% 意味制約
sem(s(R,E,X)) :-
  ga(R),
  ga(s(E,X)).
sem(s(R,E,X)) :-
  wo(R),
  wo(s(E,X)).
sem(s(E,X)) :-
  ga(s(E,X)).
sem(s(E,X)) :-
  wo(s(E,X)).

ga(s(E,X)) :-
  buy(E),
  human(X),
  @ agent(s(E,X)).
wo(s(E,X)) :-
  buy(E),
  product(X),
  @ object(s(E,X)).

```

図 6: 統語 / 意味 / 語用論制約の例

は、 $E$  が「買う (buy)」タイプの事態かつ  $Y$  が「人間 (human)」タイプの個体であるという条件のもとで、行為者格 (agent) に解釈できる。一方、 $\square$  でマークされる意味関係は、 $E$  が「買う (buy)」タイプの事態かつ  $Y$  が「製品 (product)」タイプの個体であるという条件のもとで、対象格 (object) に解釈できる。また、 $X$  が  $Y$  を提喻する関係は、 $X$  が「会社 (company)」タイプの個体であるという条件のもとで、生産者  $X$  と製品  $Y$  の間の関係 (produce) として解釈できる。なお、公理中の  $\{inst(X)\}$  は変数  $X$  にユニークなシンボルを割り当てる手続きであり、新しい個体や事態を導入する操作に相当する。

図 6 の公理を用いて目標  $s([IBM, 買った], \square, S)$  を GCAP アルゴリズムによって導出する過程を図 7、表 2 に示す (ただし、ここでは下降子測フィルタ (top-down prediction) によって、部分目標につながらない弧の導入を避けている)。図から統語 / 意味 / 語用論処理が複雑に入り組んでいる様子がわかる。主辞引数  $[IBM, 買った]$  の位置から出ている弧 1, 3, 4, 8, 29 は統語処理が進んでいく様子を表す。この部分は左隅構文解析と同様に進む。すなわち、まず “IBM” の辞書引きから始まり、ついで動詞句 “買った”、さらにそれらの間の係り受け関係を処理して、最終的に目標  $s([IBM, 買った], \square, b)$  を導出する。一方、

主辞引数  $m(a)$  の位置から出ている弧 11, 16, 18, 20 は語用論処理が進んでいく様子を表す<sup>1</sup>。この過程では、語彙 “IBM” から得られた仮説  $ibm(a)$  を提喻解釈の前提  $company(a)$  を導くために再利用し、新たに仮説  $produce(s(a,c))$ ,  $product(c)$  を仮定することによって、目標  $metonymy(m(a),c)$  を導出する。さらに、主辞引数  $s(b,c)$  の位置から出ている弧 22, 23, 24, 26, 27 は、 $b$  (“買った” が指示する対象) と  $c$  (“IBM” が提喻している対象で “IBM の製品” を表す) の間の意味関係を解釈する処理が進んでいく様子を表す。この過程では、“IBM” と “IBM の製品” の間の提喻関係を解釈する上述の語用論処理において仮定された仮説  $product(c)$  を再利用している。このように、統語 / 意味 / 語用論処理は仮説の共有によって相互に影響を及ぼし合う。

本節では、統語 / 意味 / 語用論処理の統合の簡単な例題を取り上げたが、これらをさらに形態素処理と組み合わせることも可能である (形態素 / 統語 / 意味処理の統合については [1] で示した)。また、形態素より下位のレベル (音素レベル) の処理や文より上位のレベル (談話レベル) の処理もこれに統合できる。実際、これら様々なレベルでの現象がアブダクションによって

<sup>1</sup>主辞引数  $m(a)$  の位置からはポインタ  $F$  を介してもう一つ分枝があるが、これは  $a$  の提喻を  $a$  自身に解釈する場合に対応し、意味制約を満たさないため弧 14 までで行き詰まる。

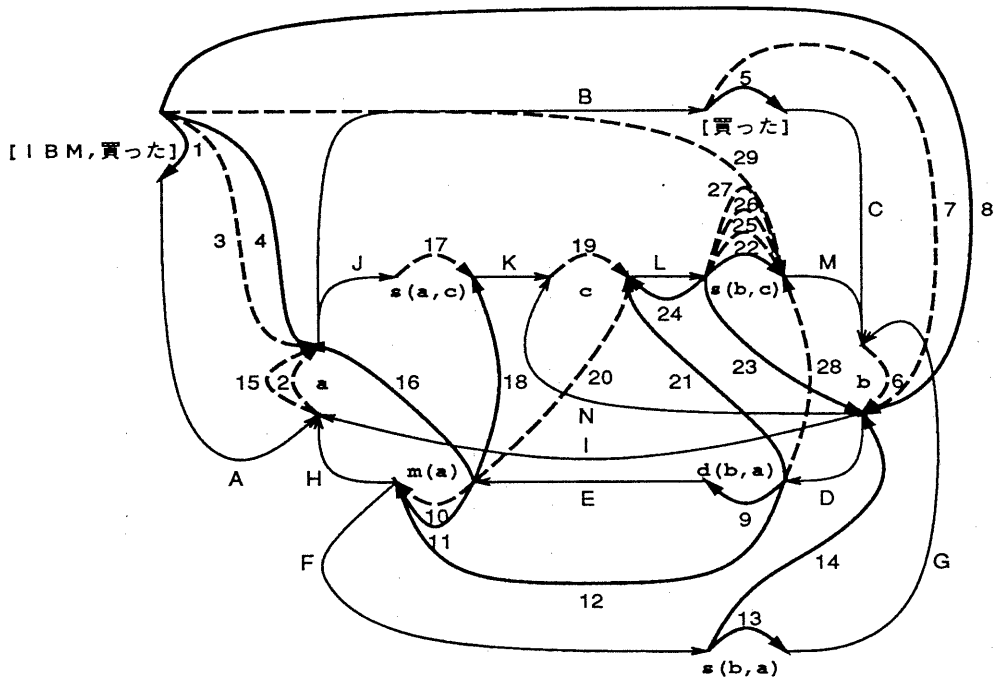


図 7:  $s([IBM, 買った], \square, S)$  の導出で構成されるチャート

統一的にとらえられることが示されている [6][2]. このようなより広範囲にわたる統合的自然言語処理への GCAP の応用については次稿で説明する.

## 5 おわりに

統合的自然言語処理において有効な計算機構と考えられるアブダクション計算を効率的に行なう手法 GCAP を提案した. この手法は日本語会話文解釈モデルの処理エンジンとして応用することを目標としている. 本手法は以下の特徴を持つ.

1. 目標駆動の上昇型探索
2. チャートによる部分解の再利用

本稿では, [1] において説明した GCP アルゴリズムをもとにして, 仮説を扱うための機能を新たに導入し, 演繹計算の手法 GCP をアブダクション計算の手法 GCAP に拡張した. また, 簡単な例題によって GCAP の統合的自然言語処理への応用例を示した. 次稿では, GCAP のより広範囲な問題への応用と計算機への実装について述べる.

## 参考文献

- [1] 伝. 統合的自然言語処理のための一般化されたチャート法. 情報処理学会研究報告, (93-NL-95):17-24, 1993.
- [2] 伝. 音声会話文法の定式化について. 人工知能学会研究会資料, (SIG-SLUD-9302), 1993.
- [3] 伝, 飯田. 情報伝達の観点から見た日常会話文の解析手法. 「自然言語処理の新しい応用」シンポジウム論文集, pp. 40-49. 電子情報通信学会・日本ソフトウェア科学会, 1992.
- [4] 春野, 伝, 松本, 長尾. ボトムアップチャート法に基づく並列文生成. 情報処理学会研究報告, (92-NL-88):95-102, 1992.
- [5] 橋田, 竹沢. 自然言語処理における統合の諸相. コンピュータソフトウェア, 8(6):3-16, 1991.
- [6] J. R. Hobbs, M. Stickel, D. Appelt, and P. Martin. Interpretation as abduction. Technical Report 499, SRI International, 1990.
- [7] M. Kay. Algorithm schemata and data structures in syntactic processing. Technical Report CSL-80-12, XEROX PARC, 1980.

表 2: s([IBM, 買った], □, S) の導出で構成されるチャート

#	Arc	A-Set	From	#	Arc	A-Set	From
1	[[?]*ibm(a) np(Φ, Ψ, a)	φ	-	15	company(a)	{α}	2
2	ibm(a)	{α}	1	16	[company(a) [?]*produce(s(a,c)) [?]*product(c)] metonymy(m(a), c)	{α}	11+H+15
3	np(Φ, Ψ, a)	{α}	1+A+2	17	produce(s(a,c))	{δ}	16
4	[np(Φ, Ψ, a) [?]*vp(Ψ, Z, VP) [?]*depend(d(VP, a))] s(Φ, Z, VP)	{α}	3	18	[company(a) produce(s(a,c)) [?]*product(c)] metonymy(m(a), c)	{α, δ}	16+J+17
5	[[?]*buy(b) vp(Ψ, □, b)	φ	4	19	product(c)	{γ}	18
6	buy(b)	{β}	5	20	metonymy(m(a), c)	{α, γ, δ}	18+K+19
7	vp(Ψ, □, b)	{β}	5+C+6	21	[metonymy(m(a), c) [?]*sem(s(b,c))] depend(d(b, a))	{α, γ, δ}	9+E+20
8	[np(Φ, Ψ, a) vp(Ψ, □, b) [?]*depend(d(b, a))] s(Φ, □, b)	{α, β}	4+B+7	22a	[[?]*buy(b) [?]*human(c) [?]*agent(s(b,c))] ga(s(b,c))	φ	21
9	[[?]*metonymy(m(a), Y) [?]*sem(s(b, Y))] depend(d(b, a))	φ	8	22b	[[?]*buy(b) [?]*product(c) [?]*object(s(b,c))] wo(s(b,c))	φ	21
10	metonymy(m(a), a)	φ	9	23a	[buy(b) [?]*human(c) [?]*agent(s(b,c))] ga(s(b,c))	{β}	22a+M+6
11	[[?]*company(a) [?]*produce(s(a,B)) [?]*product(B)] metonymy(m(a), B)	φ	9	23b	[buy(b) [?]*product(c) [?]*object(s(b,c))] wo(s(b,c))	{β}	22b+M+6
12	[metonymy(m(a), a) [?]*sem(s(b,a))] depend(d(b, a))	φ	9+E+10	24	[buy(b) product(c) [?]*object(s(b,c))] wo(s(b,c))	{β, γ}	23b+N+19
13a	[[?]*buy(b) [?]*human(a) [?]*agent(s(b,a))] ga(s(b,a))	φ	12	25	object(s(b,c))	{ε}	24
13b	[[?]*buy(b) [?]*product(a) [?]*object(s(b,a))] wo(s(b,a))	φ	12	26	wo(s(b,c))	{β, γ, ε}	24+L+25
14a	[buy(b) [?]*human(a) [?]*agent(s(b,a))] ga(s(b,a))	{β}	13a+G+6	27	sem(s(b,c))	{β, γ, ε}	26
14b	[buy(b) [?]*product(a) [?]*object(s(b,a))] wo(s(b,a))	{β}	13b+G+6	28	depend(d(b, a))	{α, β, γ, δ, ε}	21+L+27
				29	s(Φ, □, b)	{α, β, γ, δ, ε}	8+D+28

Φ = [IBM, 買った], Ψ = [買った]  
α = ibm(a), β = buy(b), γ = product(c)  
δ = produce(s(a,c)), ε = object(s(b,c))