自然言語処理　95－3
（1993. 5. 20）

統合的自然言語処理のための一般化されたチャート法

伝 康晴

ATR 音声翻訳通信研究所

統語 / 意味 / 談話の統合的処理は，不完全な情報を多様に組み合わせて処理を行なう必要のある会話文解釈などに不可欠である．このような統合的処理を実現するモデルのうち，アブダクション計算に基づくものが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．本手法では，目標駆動の上昇型探索を用いることによって無駄な導出を回避するとともに，構文解析で用いられるチャートを拡張して用いることにより部分解の再利用を行なう．本稿では，この手法の基本的なアイデアとアルゴリズムの中心部分を簡単な例を用いて説明する．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 as a Uniform Processing Module for Natural Language Processing

Yasuharu DEN

ATR Interpreting Telecommunications Research Laboratories

In the research areas such as interpretation of conversations, an integrated architecture for syntactic, semantic and discourse processing is strongly required, since no single sort of information is decisive to derive an appropriate interpretation, and thus there should be diverse interactions among several sorts of information. Abduction is one of the powerful approaches which realize practical integrated NLP systems. In this paper, we propose an efficient computation mechanism for abduction based integrated NLP systems. This method avoids unnecessary derivations by employing a goal-driven bottom-up searching, and reuses partial results by using a chart-tabulation technique which is extended from chart parsing algorithm. We explain a basic idea of this method, and show the central part of the algorithm with simple examples.

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]．ここでいう統合的自然言語処理は，以下のように区分できる[1]．

1. 統語 / 意味 / 談話処理の統合 [12][6][9]
2. 解析と生成の統合 [11][4][1]
3. 音声処理などと言語処理との統合 [8][9]

このような統合的自然言語処理機構は，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．こういった問題領域においては，単独では不完全な情報を多様に組み合わせて問題を解決する必要があるからである．例えば，音声対話理解においては，音素の脱落 / 置換による音声情報の不完全さ，構成素の省略 / 倒置による文法情報の不完全さ，あるいは，提喩 / 換喩の使用による意味情報の不完全さなどを別の種類の情報で補うことによって，はじめて文の理解が可能になる．

筆者らが提案している日本語会話文解釈の枠組 [2] においても，多様な情報の組合せ的な処理が不可欠である．この枠組の構成は図 1 のようであり，上にあげた統合的自然言語処理の区分の 1 に属すタイプのものである．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．この点は Hobbs らの研究においてはあまり検討されていない．

本稿および後続する予定の稿では，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．ここでは，最初にあげた統合的自然言語処理の区分のうち特に 1 に注目し，図 1 の会話文解釈のモデルに応用することを目標とする．本稿では，まず，その基本的なアイデアと技術上の中心部分に関して説明する．

アブダクション計算の効率化に際して，本手法では，特に以下の改善点を中心にする．

1. 単純な下降型探索における無駄な分岐の削減，および，単純な上昇型探索における無駄な導出の回避．
2. 深さ優先探索における部分解の再計算の回避．

まず，1 を実現するために本手法では**目標駆動の上昇型探索**を用いることによって，必要な情報の入手につながらない無駄な導出を回避できるようにする．次に，2 を実現するために，構文解析で用いられている**チャート**を拡張したデータ構造を導入し，部分解の再利用を行なう．このようにして得られる本手法を，**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**) と呼ぶ．本稿では，この手法の中でアブダクションに関わる側面 (仮説の導入，仮説を用いた導出，仮説の共有など) を取り除いたものに相当する**一般化されたチャート法に基づく演繹計算** (Generalized Chart based Prover, **GCP**) について説明する．**GCAP** は **GCP** を拡張して得られるので，**GCAP** の中心的な部分は本稿において説明されることになる．

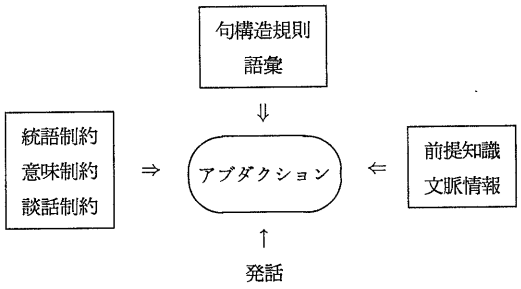


図 1: 会話文解釈モデルの基本構成

本稿の構成は以下の通りである．まず，2 節で **GCP** の基本的なアイデアを説明する．次に，3 節で **GCP** アルゴリズムの形式的な定義を与え，その内容を簡単な例を用いて説明する．4 節では，会話文解釈モデルへの応用を想定して，形態素 / 統語 / 意味処理の統合に関する簡単な例題を与え，**GCP** の有効性を確認する．最後に，5 節でまとめを述べる．

2 主辞駆動型導出

本手法では，計算の際に用いる規則を**ホーン節** (Horn clauses) に限定する．図 2 の公理 (規則 / 事実の集合) を考えよう (以下では，便宜上，規則 / 事実を Prolog の記法で表現する)．いま，この公理を用いて目標 `p(い,Z)` を導出することを考える．簡単にわかるように，この目標は規則 (2) と事実 (5), (7) を用いることにより，単一化 `Z = one` のもとで導出可能である．ここでの問題は，この結果を得るにいたる過程である．

最初に，単純な (深さ優先の) 下降型探索を用いた場合を考える．下降型探索では，まず，規則 (1) を用いて初期目標 `p(い,Z)` を展開するが，この後の部分目標 `q1(い,Y)` は導出

[1] 橋田らの研究 [4][9] は，これらすべてを単一の機構によって実現するという大目標の一環として行なわれている．

```
(1)  p(X, Z) :- q1(X, Y), r(Y, Z).
(2)  p(X, Z) :- q2(X, Y), r(Y, Z).
(3)  q1(a, 1).
(4)  q1(b, 2).
(5)  q2(い, 1).
(6)  q2(ろ, 2).
(7)  r(1, one).
(8)  r(2, two).
```

図 2: 公理の例 1

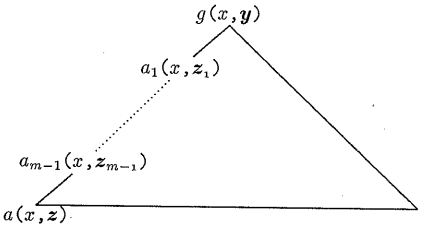


図 3: 主辞駆動型導出

されないので，この探索パスは行き詰まる．そこで後戻りによって，次に，規則 (2) を選択し，最終的に上の結果を得る．ここで，最初に規則 (1) を選択した場合のような行き詰まりが多くある場合や行き詰まりがもっと深いレベルで生じる場合には，規則の選択の誤りによる悪影響はかなり大きなものになる．

次に，単純な上昇型探索を用いた場合を考える．上昇型探索では，まず，規則 (1), (2) を使って事実 (3)-(8) から，4 つの新しい命題 p(a,one), p(b,two), p(い,one), p(ろ,two) が導出される．このうち，初期目標の導出に必要な命題は p(い,one) のみである．ここで，最初の事実から導出される命題が多くある場合やさらにそれらの命題から多くの別の命題が導出される場合には，初期目標につながらない無駄な命題の導出を非常に多く行なってしまう．

この例の規則 / 事実たちは GCP の基本的アイデアを想起させる 2 つの重要な性質を有している．

1 つは，すべての事実がその第 1 引数によって索引づけられていることである．例えば，第 1 引数に項 'い' を持つ事実は一意に q2(い,1) と決まる．この性質は自然言語処理で用いるいくつかの規則の中でみられる．例えば，構文解析で用いる語彙記述は，n([太郎 |Z],Z,taro), v([来る |Z],Z,Sbj,come(Sbj)) のような形をしており，第 1 引数である単語列を索引として，(一意ではないにせよ) 効率的な検索ができる．逆に，構文生成では意味表現を索引として語彙を検索できる．この性質は，要するに，事実に含まれる引数の中に索引の役割を果たすものが 1 つ存在するということである．

もう 1 つは，上で述べた事実を検索する際の索引が，初期目標の第 1 引数としてそのまま現れていることである．すなわち，初期目標 p(い,Z) の第 1 引数 'い' がそのまま事実を検索する際の索引として使われる．左隅構文解析や意味主辞駆動生成 [10] は，まさにこの性質を利用して効率的な探索を行なっている．例えば，左隅構文解析において，非終端記号を s([太郎, が, 来る],[],come(taro)), n([太郎, が, 来る],[が, 来る],taro) のように DCG 流に表現すると，左隅非終端記号の第 1 引数 (単語列の先頭部分) と開始記号の第 1 引数 (単語列の先頭部分) は等しい．それゆえ，開始記号を下降型に展開するのではなく，その第 1 引数を索引としてまず語彙を検索し，そこから上昇型に構文木を作っていくことができる．これは，意味主辞駆動生成においても同様である．

構文解析 (左隅構文解析)/ 構文生成 (意味主辞駆動生成) で典型的に見られる上に述べた性質を，自然言語処理で用いる規則一般に広く見られるものと仮定し，左隅構文解析 / 意味主辞駆動生成で用いるのと同様な効率的な探索を実現しようというのが，GCP の基本的アイデアである．

図 3 はこのアイデアを図式化したものである．目標 $g(x,\boldsymbol{y})$ の第 1 引数 x は左隅にある事実 $a(x,z)$ にいたるまでの経路上で共有されており，かつ，この引数は事実を検索する際の索引になるとする．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 3 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，図 2 の規則 (1), (2) のように，右辺の先頭の要素式の第 1 引数と左辺の要素式の第 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．例えば，規則 (1) の主辞は q1(X,Y) であり，その主辞引数は X である．このように主辞駆動型の公理では，規則の形に制限があるが，場合によっては，この制限に従わない規則を使いたいこともあるかも知れない．そのために，GCP では主辞を持たない規則の記述を許している．以下では，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．

公理が主辞駆動型のとき，左隅構文解析 / 意味主辞駆動生成と類似の探索を利用して目標の導出を効率的に行なえる．この導出過程を**主辞駆動導出** (head-driven derivation) と呼ぶ．主辞駆動導出を図 3 を用いて簡単に説明すると

以下のようになる．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$a_{m-1}(x,z_{m-1})$:- $a(x,z),B_1,\ldots,B_n$を検索し，姉妹$B_1,\ldots,B_n$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$B_1,\ldots,B_n$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．この親が目標$g(x,y)$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]．

本節で導入した主辞駆動型導出は，1節で述べた改善点の1番目のもの，すなわち，探索の枝刈りに関する問題を解決する．次節では，さらに，もう1つの問題を解決するために，構文解析/構文生成で広く用いられている技術を主辞駆動型導出に導入する．

3 一般化されたチャート法に基づく演繹計算

3.1 チャートの導入

GCPが解決すべきもう1つの問題は，部分解の再計算の回避である．

図2の公理に加えて，図4の公理を考える．いま，これらの公理を用いて目標k(i,Y,Z)を導出することを考える．今度は，単一化Y = イ ∧ Z = oneおよびY = ヰ ∧ Z = oneの2通りの導出が可能である(図5)．注目すべきことは，これらの2通りの導出過程において部分目標p(い,Z)が共有されていることである．深さ優先の主辞駆動型導出で2通りの結果を得ようとすると，後戻りによって，この部分目標の導出を2回行なってしまうことになる．これは，左隅構文解析や意味主辞駆動生成で問題となる部分解の再計算と同じ問題である．

そこで，この問題を解決するのに，左隅構文解析/意味主辞駆動生成の場合[7][3]と同じアプローチをとる．すなわち，**チャート**(charts)を用いて，部分解の再利用を行なう．特に，ここで用いるチャートは，意味主辞駆動チャート法[3]の場合と同様に，構文解析で用いるチャートを拡張したものである．GCPの計算手続きは，この拡張を考慮して，ボトムアップチャート構文解析(BCP)の手続きをもとに得られる．

3.2 GCPの計算手続き

BCPでは，語彙弧Aは語彙規則A --> [w]を用いて入力単語wをもとにして導入される．一方，GCPにおける語彙弧の導入は，図3で目標$g(x,y)$が設定されたときに

(9) k(X, Y, Z) :- l(X, Y), m(Y, Z).
(10) m(X, Z) :- n(X, Y), p(Y, Z).
(11) l(i, イ).
(12) l(i, ヰ).
(13) l(ro, ロ).
(14) n(イ, い).
(15) n(ヰ, い).
(16) n(ロ, ろ).

図4: 公理の例2

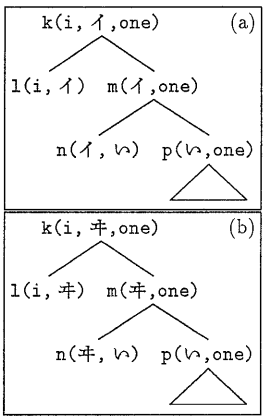


図5: 目標k(i,Y,Z)の2つの導出過程

左隅にある事実$a(x,z)$を導入することである．すなわち，目標$g(x,y)$に対して主辞引数xを索引として事実$a(x,z)$を見つけ，これを語彙弧としてチャートに導入する．ここで，目標とはチャート法でいう活性弧の最左空所のことであるから，GCPでは，新しい活性弧が張られる度にその最左空所の主辞引数を索引として事実を検索し，新しい語彙弧を導入する．

さて，BCPでは弧の位置は単語によって決まるから，弧の接続関係は単語の並びに相当し，1対1の接続関係になる．ところが，弧の位置が主辞引数によって決まるGCPでは弧の接続関係はBCPの場合よりも複雑である．図5の2つの導出過程からわかるように，同じ主辞引数iによって複数の語彙弧l(i,イ), l(i,ヰ)が導入される場合，この隣に来る目標m(イ,Z)およびm(ヰ,Z)の主辞引数は異なる．これは，主辞引数iによって決まる位置に張られた弧が接続する先として，主辞引数'イ'によって決まる位置と主辞引数'ヰ'によって決まる位置の複数が可能だということである．さらに，図5からわかるように，これら2つの異なる主辞引数のもう1つ先の主辞引数は同じ'い'である．それゆえ，異なる位置に張られた弧が同じ位置に接続していく場

[2]これに対して，非連鎖規則は下降型に展開される．

手続き 1　位置 $[s,t]$ に活性弧 $[\cdots[?]B_j\cdots]A$ がある．ただし，$B_j = b_j(x,\boldsymbol{y})$ 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．このとき，

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 $a(x,\boldsymbol{z})$ に対し位置 $[u,v]$ に不活性弧 $a(x,\boldsymbol{z})$ を張る．
   (b) すべての非連鎖規則 $a(x,\boldsymbol{z})$:- $B_1,\ldots,B_n$ に対し位置 $[u,v]$ に活性弧 $[[?]B_1\cdots[?]B_n]a(x,\boldsymbol{z})$ 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' :- $A,B_1,\ldots,B_n$ に対し位置 $[s,t]$ に活性弧 $[A[?]B_1\cdots[?]B_n]A'$ を張る ($n = 0$ の場合は不活性弧 A' を張る)．

手続き 3　位置 $[s,t]$ に活性弧 $[\cdots[?]B_j[?]B_{j+1}\cdots[?]B_n]A$ 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 $[\cdots B_j[?]B_{j+1}\cdots[?]B_n]A$ を張る ($j = n$ の場合は不活性弧 A を張る)．

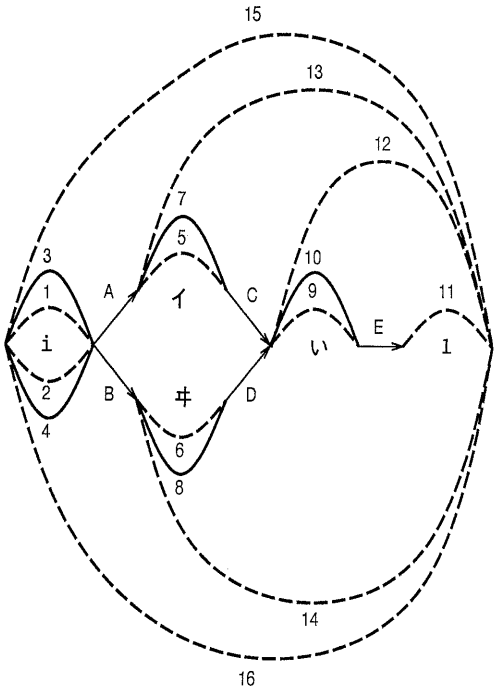
図 6: GCP の計算手続き

合がある．したがって，BCG では弧の接続関係は一般に多対多になるので，弧の接続関係をポインタを用いて表す．

以上の拡張を施した GCP の計算手続きは図 6 のようになる．図 2, 4 の公理を用いて，目標 k(i,Y,Z) を GCP アルゴリズムによって導出する過程を図 7 に示す．構文解析の場合との大きな違いは，弧の接続関係がポインタで表されていることであり，活性弧を隣接する不活性弧で補完する手続き (**手続き 3**) はこの点を考慮して拡張されている．例えば，図 7 では，活性弧 10: [q2(い,1) [?]r(1,Z5)] p(い,Z5) と不活性弧 11: r(1,one) はポインタ E で結ばれることによって隣接しており，これらから補完手続きによって不活性弧 12: p(い,one) を作れる．この違いを除けば，構文解析のチャート法とほぼ同じであるから，図 7 のチャートができる過程を各自追ってみて欲しい．

4　形態素 / 統語 / 意味処理の統合

1 節で述べたように，本研究の目標は GCAP を会話文解釈のモデルに応用することである．そこで，本節では，GCAP による統語 / 意味 / 談話の統合的処理のイメージを



#	Arc	From
1	l(i, イ)	–
2	l(i, キ)	–
3	[l(i, イ) [?]m(イ,Z1)] k(i, イ,Z1)	1
4	[l(i, キ) [?]m(キ,Z2)] k(i, キ,Z2)	2
5	n(イ, い)	3
6	n(キ, い)	4
7	[n(イ, い) [?]p(い,Z3)] m(イ,Z3)	5
8	[n(キ, い) [?]p(い,Z4)] m(キ,Z4)	6
9	q2(い,1)	7, 8
10	[q2(い,1) [?]r(1,Z5)] p(い,Z5)	9
11	r(1,one)	10
12	p(い,one)	10+E+11
13	m(イ,one)	7+C+12
14	m(キ,one)	8+D+12
15	k(i, イ,one)	3+A+13
16	k(i, キ,one)	4+B+14

図 7: 目標 k(i,Y,Z) の導出で構成されるチャート

つかめるよう，簡単な例題に対する GCP の適用を見る．ここであげる例題では，簡単のため談話処理は扱わず，形態素 / 統語 / 意味処理を統合的に行なう．

図 8 は簡単な日本語の文を解釈するための比較的自由度

の高い公理である[3]．句構造規則は通常より緩く書かれている．すなわち，動詞による下位範疇化を仮定せず，任意の助詞句と動詞句が係り受け関係を持ち得るとしている．要素式 sem([P,VP,NP]) は，助詞 P でマークされた名詞句 NP が動詞句 VP に係る際に，それらの間に何らかの意味関係が成り立つことを保証するための制約である (ただし，P, NP, VP はいずれも句の意味情報)．例えば，de でマークされた係り受け関係は de([VP,NP]) という制約によって規定され，さらにこの制約によって，例えば，VP が call のとき NP は制約 call_by(NP) を満たすような対象に制限される．制約 call_by(NP) は「(人を) 呼ぶ手段」になり得るものを規定する選択制限の一種であり，ここでは phone, letter がこの制約を満たすとしている．また，語彙は単語ではなく形態素をもとにして記述されており，これによって形態素解析と構文解析が同時に行なわれる．なお，図 8 の公理のうち意味制約はすべて非連鎖規則として記述している．

図 8 の公理を用いて目標 vp([く，る，ま，で，よ，ぶ], [],VP) を導出する過程を考えよう．これは文『くるまでよぶ』を形態素 / 統語 / 意味解析することに相当する．GCP によるこの目標の導出過程は図 9 のようになる (ただし，下降予測フィルタ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)．図の左半分が形態素 / 統語解析の部分に相当し，右半分が意味解析の部分に相当する．この文の形態素解析は『くるま / で / よぶ (車で呼ぶ)』と『くる / まで / よぶ (来るまで呼ぶ)』の 2 通りに曖昧である．一般に形態素解析が曖昧性を持つ場合は単語の接続が束状になるが，図 9 の左半分での弧間の接続はまさにこの単語束に対応している．したがって，形態素列 [よ，ぶ] に対する部分解が複数の解析過程で共有されている．一方，これらの解析によって得られる意味情報に対する意味解析では，後者の『くる / まで / よぶ (来るまで呼ぶ)』に対応する意味情報 ([made,call,come]) だけが解釈を与えられる．なぜなら，もう一方の解析では『くるま (車)』と『よぶ (呼ぶ)』の間に助詞『で』でマークされるような解釈を与えられないからである．このようにして，目標 vp([く，る，ま，で，よ，ぶ],[],VP) がただ一つの解釈のもとで導出される．

[3]ここでは，すべての規則は演繹的に適用されるとしているが，筆者らの会話文解釈のモデルはアブダクション計算に基づいているため，規則の定式化はここであげたものとは異なる．したがって，図 8 の定式化が会話文解釈のためにふさわしいものであるという主張はここでは行なっていない．例えば，文法的に不完全な会話文を扱うために，句構造規則は下位範疇化を用いずに定式化しているが，意味制約が演繹的に導出されなければならないとすると，下位範疇化と同じくらい強い制約を課してしまうことになり望ましくない．ここでの公理は，あくまでも GCP の動作を知るための例として見て欲しい．

```
% 句構造規則 (統語制約を含む)
vp(X,Z,VP) :-
        pp(X,Y,P,NP),
        vp(Y,Z,VP),
        sem([P,VP,NP]).

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

% 語彙 (形態素制約を含む)
np([く，る，ま|Z], Z, car).
vp([く，る|Z], Z, come).
vp([よ，ぶ|Z], Z, call).
p([ま，で|Z], Z, made).
p([で|Z], Z, de).

% 意味制約
sem([made|Args]) :- made(Args).
sem([de|Args]) :- de(Args).

made([call,X]) :- until(X).
made([write,X]) :- until(X).

de([call,X]) :- call_by(X).
de([write,X]) :- write_with(X).
de([write,X]) :- write_in(X).

% 選択制限
until(X) :- time(X).
until(X) :- event(X).

time(tomorrow).
time(friday).

event(come).
event(write).

call_by(phone).
call_by(letter).

write_with(pencil).
write_with(pen).

write_in(X) :- language(X).

language(japanese).
language(english).
```

図 8: 形態素 / 統語 / 意味制約の例

この例では，形態素 / 統語解析と意味解析が完全に分離しているかのように見える．しかし，例えば『たろうがくるまでよぶ』のように，複数の助詞句が動詞に係る場合や複文を構成する場合を考えると，文全体の形態素 / 統語解析が終わった後で意味解析が行なわれるのではなく，おのおのの係り受け構造ごとに形態素 / 統語解析と意味解析が交互に行なわれることがわかるであろう．これは形としては，形態素 / 統語規則中に意味解析を手続き呼び出しとして埋め込む方法に似ている．しかし，手続き呼び出しを用いる方法では形態

素 / 統語解析モジュールとは別に意味解析モジュールを用意する必要がある．仮に，形態素 / 統語解析をチャート法によって実行するとしても，意味解析モジュールはそれとは別に必要である．ところが，意味解析も，GCP のような主辞駆動型のチャート法などを使わないと効率的には行なえない (例えば，意味制約 sem([made,call,come]) を下降型に導出する場合を考えてみよ)．そうすると，意味解析モジュールも形態素 / 統語解析モジュールと同じ手法で実現する必要があるわけで，それなら，わざわざこれらのモジュールを分ける意味はない．結局，GCP のアイデアのもととなった仮定「自然言語処理で用いる規則は一般に主辞駆動型になる」がこの例においては満たされているので，GCP による計算が勝るのである．そして，この仮定が実際的な問題に対して広く成立するのを確認すること，あるいは，この仮定を満たすようにさまざまな規則を定式化することが今後の重要な課題となる．

5 おわりに

統語 / 意味 / 談話の統合的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．本稿では，まず，GCAP の基本的なアイデアと技術上の中心部分をなす GCP アルゴリズムについて説明した．GCP の特徴は次の 2 点である．

1. 目標駆動の上昇型探索による無駄な導出の回避
2. チャートを用いた部分解の再利用

本稿では，アブダクションに関わる側面 (仮説の導入，仮説を用いた導出，仮説の共有など) については述べなかった．仮説の扱いはより柔軟な制約計算を行なうために重要である．これについては稿を改めて述べる．

謝辞

本研究を進めるにあたって有益なコメントを数多くいただいた松本裕治 (奈良先端科学技術大学院大学)，春野雅彦 (NTT)，新保仁 (京都大学工学部) 各氏に感謝します．

参考文献

[1] 伝．チャート型構文解析・生成の統一的構造．日本ソフトウェア科学会第 9 回大会論文集, pp. 141–144, 1992.

[2] 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.

[3] 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.

[4] K. Hasida. Common heuristics for parsing, generation, and whatever .... In T. Strzalkowski, editor, *Reversible Grammar in Natural Language Processing: Proceedings of a Workshop Sponsored by the Special Interest Groups on Generation and Parsing of ACL*, pp. 81–90. University of California, Berkeley, 1991.

[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 Palo Alto Research Center, 1980.

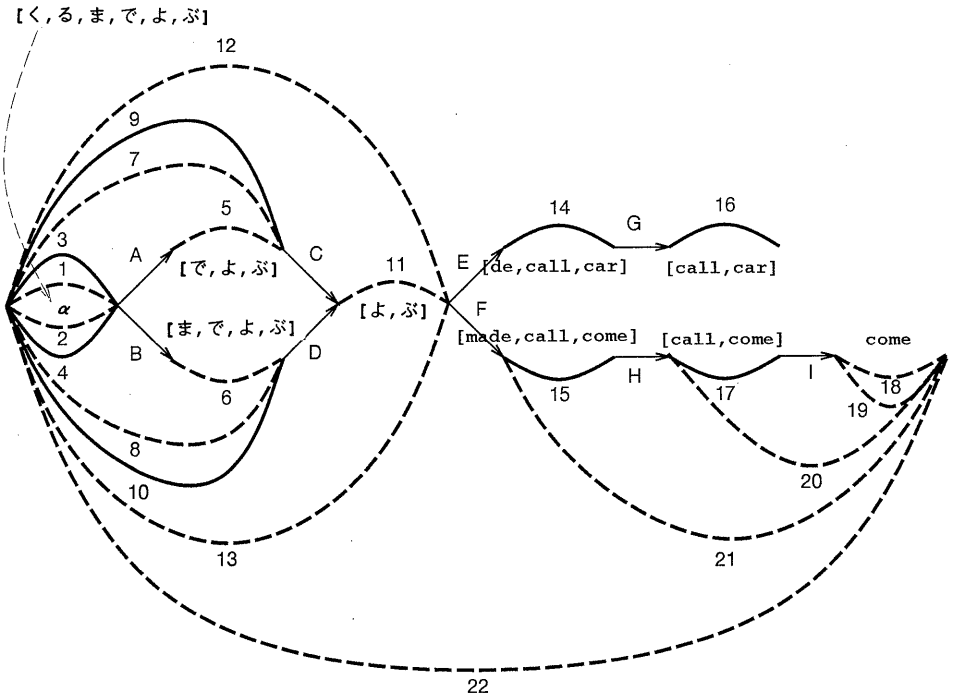
[8] H. Kitano. ΦDM-Dialog: An experimental speech-to-speech dialog translation system. *IEEE Computer*, 24(6):36–50, 1991.

[9] 長尾, 橋田, 宮田．全方向的な情報の流れによる音声言語理解．「自然言語処理における基本的問題」シンポジウム論文集, pp. 97–106. 電子情報通信学会・日本ソフトウェア科学会, 1992.

[10] S. Shieber, G. van Noord, R. Moore, and F. Pereira. Semantic-head-driven generation. *Computational Linguistics*, 16(1):30–42, 1990.

[11] S. M. Shieber. A uniform architecture for parsing and generation. In *Proceedings of the 12th COLING*, pp. 614–619, 1988.

[12] D. L. Waltz and J. B. Pollack. Massively parallel parsing: A strongly interactive model of natural language interpretation. *Cognitive Science*, 9:51–74, 1985.



#	Arc	From
1	np(α,[で,よ,ぶ],car)	–
2	vp(α,[ま,で,よ,ぶ],come)	–
3	[np(α,[で,よ,ぶ],car) [?]p([で,よ,ぶ],Z1,P1)] pp(α,Z1,P1,car)	1
4	[vp(α,[ま,で,よ,ぶ],come) [?]p([ま,で,よ,ぶ],Z2,P2)] pp(α,Z2,P2,come)	2
5	p([で,よ,ぶ],[よ,ぶ],de)	3
6	p([ま,で,よ,ぶ],[よ,ぶ],made)	4
7	pp(α,[よ,ぶ],de,car)	3+A+5
8	pp(α,[よ,ぶ],made,come)	4+B+6
9	[pp(α,[よ,ぶ],de,car) [?]vp([よ,ぶ],Z3,VP3) [?]sem([de,VP3,car])] vp(α,Z3,VP3)	7
10	[pp(α,[よ,ぶ],made,come) [?]vp([よ,ぶ],Z4,VP4) [?]sem([made,VP4,come])] vp(α,Z4,VP4)	8

#	Arc	From
11	vp([よ,ぶ],[],call)	9, 10
12	[pp(α,[よ,ぶ],de,car) vp([よ,ぶ],[],call) [?]sem([de,call,car])] vp(α,[],call)	9+C+11
13	[pp(α,[よ,ぶ],made,come) vp([よ,ぶ],[],call) [?]sem([made,call,come])] vp(α,[],call)	10+D+11
14	[[?]de([call,car])] sem([de,call,car])	12
15	[[?]made([call,come])] sem([made,call,come])	13
16	[[?]call_by(car)] de([call,car])	14
17	[[?]until(come)] made([call,come])	15
18	event(come)	17
19	until(come)	18
20	made([call,come])	17+I+19
21	sem([made,call,come])	15+H+20
22	vp(α,[],call)	13+F+21

α = [く,る,ま,で,よ,ぶ]

図 9: 目標 vp([く,る,ま,で,よ,ぶ],[],VP) の導出で構成されるチャート