自 然 言 語 処 理 83−10
（1991. 5. 17）

非文の解析ーチャートに基づく新たな手法

加藤　恒昭

NTT情報通信処理研究所

脱落、挿入、置換等の誤りを含んだ非文を解析し、誤り箇所と誤り種別を同定するための非文解析の手法を提案する。本手法は、上昇型と下降型のチャート解析を拡張して融合した解析手法であり、Mellishが提案した手法の改良と位置づけられる。特定の文法に依存しないこと、整った文の解析に悪影響を及ぼさないことに加え、本手法では、探索空間の刈り込みに相当する処理を探索に先だって行ない、探索状態の大域的な尤もらしさを付与されたデータ構造を用いるために、複雑なヒューリスティクスを必要とすることなく、従来手法よりも効率的に最も尤もらしい解析結果をすべて求めることができる。

Yet Another Chart-Based Technique for Parsing Ill-Formed Input

Tsuneaki Kato

NTT Communications and Information Processing Laboratories

1-2356 Take, Yokosuka-shi, kanagawa 238-03, Japan

A new chart-based technique for parsing ill-formed sentences is proposed. This can process sentences with unknown/misspelled words, omitted words or extra noise words. Similar to Mellish's notorious algorithm, this method has the advantage that it doesn't affect on parsing process of well-formed sentences and it is independent on any particular grammar. Moreover, early reduction of search space and exploiting global plausibility of the searching state improves the efficiency for getting all plausible analysises of ill-formed sentences.

1 はじめに

非文(ill_formed sentence)が入力された際、単にリジェクトするのではなく、それを解析し、誤り箇所や誤り内容を同定することは、自然言語インタフェースシステムにおいて、ユーザの入力誤りや、システムが持つ文法や辞書の不備を救済するために重要である。更に、このような非文解析技術は、話し言葉によく現われる、文法から逸脱した文や文の断片(Fragment)の解析にも応用できるものであると考えられる。

未知語を含んだ文の解析[6][7]という観点や、音声認識結果に対する解析[4][5]という観点などからのものを含めて、従来より、非文解析のための手法が幾つか提案されている[1]~[7]。この中で、Mellishの提案する下降型のチャート解析[8][9]を拡張した解析手法[1]は、チャートを利用することにより誤りの右側の情報も利用できる点、特定の文法に依存しない枠組みである点、未知語のみでなく、挿入、脱落誤りまでを対象としている点で興味深い。しかし、この手法では、誤りの探索をヒューリスティクスに基づくAgenda制御により行なっており、加えてそのヒューリスティクスは、6つのパラメータにもとづく複雑なものである。このため、最も尤もらしい解析結果をすべて求めるために、整った文(well-formed sentence)の解析の7倍程度の時間を要する。

本稿では、チャートに基づく非文解析の新しい手法を提案する。本手法は、上昇型と下降型のチャート解析をそれぞれ拡張して融合した解析手法であり、探索空間の刈り込みに相当する処理を探索に先立って行なうこと、探索状態の大域的な尤もらしさを付与したデータ構造を用いることにより誤り箇所の探索を効率化し、複雑なヒューリスティクスを必要とすることなく、整った文の解析の4倍程度の時間で、最も尤もらしい解析結果をすべて求めることができる。もちろん、特定の文法に依存しない、整った文の解析に悪影響を及ぼさないなど、Mellishの手法の利点をすべて引き継いでいる。また、本稿でも、Mellishにならって単純なCFGを例に議論を進めるが、本手法はDCG[10]等の論理文法や、LFG等の単一化に基づく文法[11]などに拡張可能である。

以下、2章でMellishの手法を概観する。3章でここで提案する手法について説明する。4章では簡単な実験による評価を示す。5章では本手法について考察し、更なる改善について示唆する。

2 Mellishの手法の概要とその問題点

Mellishの手法では、まず上昇型左隅解析を行ない、それが失敗した場合に、その原因となった誤りを同定するための拡張した下降型解析が行なわれる。入力が整ったものであれば、前半の上昇型解析によって解析結果が得られるため、一般の解析と同じ効率で結果が得られる。また、入力が非文である場合も、この前半の解析によって、入力中に存在する可能な部分構造が非活性弧(Inactive-Edge)として、誤りによって中断された解析の途中結果が活性弧(Active-Edge)として残される。ATNを用いる手法[2]とは異なり、チャートに基づく解析であるため、誤り箇所の右側についても弧(Edge)が残されていることがこの手法の着眼点である。

後半の拡張された下降型解析では、これら残された弧を利用して、特に非活性弧とのcompletionによって、誤り箇所を特定してゆき、そこに適当な仮設を立てることで非活性弧を得る。この仮定された非活性弧を利用して解析を続行し、解析が成功すればこの誤りに関する仮設が採用される、つまり、誤りが同定されたことになるわけである。

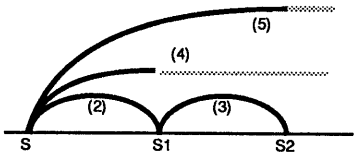
この手法の実現のために、従来の活性弧、非活性弧の拡張である一般化弧(Generalized-Edge)と名付けられたデータ構造が提案されている。一般化弧は以下の様に表現される。

<C from S to E needs
Cs_1 from S_1 to E_1 , ... , Cs_n from S_n to E_n>

ここで、Cはカテゴリシンボル、Cs_iはカテゴリシンボルの列を示す。S,S_i,E,E_iは入力単語列中の位置（特殊記号として、不定を示す記号*がある）を示し、始点、終点と呼ばれる。本稿では、needsの左側のCをラベル、needsの右側を不定部分、そこに現われるカテゴリ列のうち最左のもの、つまり、Cs_1を最左カテゴリ列、最左カテゴリ列に含まれる最左のカテゴリを最左カテゴリと呼ぶこととする。この構造は、入力単語列中の位置S_1からE_1にカテゴリ列Cs_1に一致する構造があり、…、S_nからE_nにカテゴリ列Cs_nに一致する構造があれば、SからEにカテゴリCの構造が見つけられたことになることを示している。また、特に不定部分が空であるような一般化弧、つまり、非活性弧に相当するものは以下の様に表現される。

<C from S to E needs nothing>

例えば、文法規則として図1に示す(1)があり、入力中の位置Sから位置S1の間に統語カテゴリC1が、位置S1から位置S2の間に統語カテゴリC2が存在した場合、上昇型解析によって残される弧は、一般化弧による表現を用いると、図1(2)~(5)の様になる。ここで、(2)(3)が非活性弧に、



C0 -> C1 C2 C3 (1)
<C1 from S to S1 needs nothing> (2)
<C2 from S1 to S2 needs nothing> (3)
<C0 from S to * needs [C2 C3] from S1 to *> (4)
<C0 from S to * needs [C3] from S2 to *> (5)

図1　一般化弧の例

(4)(5)が活性弧に相当する（以下では弧は全て一般化弧の形式で表現する。活性弧、非活性弧と呼ぶのは、正確にはそれらに相当する一般化弧である）。

誤り同定のための拡張された下降型解析は、以下の解析規則によって表現されるチャート解析である。

Top-down rule　一般のTop-down ruleに同じ

Fundamental rule　活性弧と非活性弧とに相当する一般化弧のcompletionを行なう。ただし、一般のチャート解析では、活性弧の最左カテゴリだけがcompletionの対象となるのに対し、この規則では、一般化弧の最左カテゴリ列中の任意のカテゴリが対象となる。

Simplification rule　拡張に伴うHouse-keepingのための規則であり、始点と終点が一致し、かつ空であるような最左カテゴリ列を持つ一般化弧から、そのカテゴリ列（空列）を除去した弧を生成する。

Garbage rule　挿入誤りを処理するための規則で、始点と終点が一致せず、かつ空であるような最左カテゴリ列を持つ一般化弧から、そこに挿入誤りがあると見做して、そのカテゴリ列を除去した一般化弧を生成する。

Empty Category rule　脱落誤りを処理するための規則で、始点と終点が一致し、かつ空でないような最左カテゴリ列を持つ一般化弧から、そこに脱落誤りがあると見做して、そのカテゴリ列を除去した一般化弧を生成する。

Unknown word rule　置換誤り、未知語を処理するための規則で、始点と終点が一致せず、かつ、空でないような最左カテゴリ列を持つ一般化弧について、その最左カテゴリが、Lexical Categoryであり、かつ、そのカテゴリに属す単語がその位置に存在しない場合、そこに置換誤りもしくは未知語の存在があると見做して、そのカテゴリを除去した一般化弧を生成する。

前半の3つの規則により、一般の下降型解析と同様な手順で、ただし、completionの対象を左隅に限定しないで、解析が進み、より限定された（狭い）不定部分を持つ弧が生成されてゆく。後半の3つの規則が適用できる弧が得られると、それらの規則の適用により誤りが仮定されて、誤りを修正した結果に相当する弧が生成され、解析が続けられる。この解析の制御は、付与された得点が大きい弧を優先的に処理するというAgenda制御によって行なわれる。この得点付与はヒューリスティクスによって行なわれ、Fundamental ruleによって生成された弧をTop-down ruleによって生成されたものより優先するなど、6つの観点に基づいている。

Mellishによって提案されたこの手法は、誤りの右側の情報も利用できる点、特定の文法に依存しない手法である点で優れている。しかし、一方で、誤り箇所の発見にTop-down ruleを重点的に用いる拡張された下降型解析は、膨大な数の弧を生成する可能性を含んでいる。つまり、誤り箇所であるLexical CategoryまでTop-down ruleによる展開を行なった場合、数多くのオルタナティブが発生する。加えて、生成される弧は独立しており、局所的な情報のみを表現し、それが生成された経緯等を含んでいないから、弧に含まれる情報だけからでは、その弧を展開すべきなのかの判定が行なえない。非文解析においては、語彙レベルで解析がアンカリングされる必要がなく、誤りを仮定することで解析が続行できるうえ、Top-down ruleを適用する対象が上昇型解析の結果得られた弧を含むために、開始記号によってアンカリングされている保証もない。このため、これらの問題は一般の下降型解析に比べて、極めて深刻なものとなる。Mellishの手法では、このような問題から生じる多くの潜在的に無駄な探索を、Agenda制御だけを利用して抑制している。しかし、ヒューリスティクスに基づく複雑な得点付けによるこのような制御は、見通しが悪いうえ、そのような制御によって、最適な探索が行いうるかが疑問である。

3　提案する手法

概要

前章で述べた問題に着目し、ここで提案する手法では、以下を考える。まず、制御の一部をAgenda制御から独立させ、探索空間の刈り込みに相当する処理を先行して行う。次に、活性弧とのcompletionという規則を導入することで、Top-

down ruleへの依存度を減らす。更に、大域的な尤もらしさを保持するデータ構造を採用することで制御を容易にすると共に、探索が開始記号によってアンカリングされるようにする。

本手法は、上昇型左隅解析が失敗した後に起動される点でMellishのそれと同様であるが、後半の解析が2つのステップに分割されている点で大きく異なる。第一のステップでは、上昇型の解析により、弧の完備化が行なわれる。ここで言う完備化とは、左から右へという左隅解析の方向性が原因で生成されないすべての活性弧を生成することである。例えば、図2で、文法規則(1)が存在し、(2)(3)の2つの非活性弧が存在したとき、上昇型の左隅解析によって得られる弧は、(4)だけである。弧の完備化では、これに加えて(5)(6)の弧を生成する。この完備化により、弧における不定部分は、解析が行なわれて（試されて）いない部分を含むことなく、純粋にそこに対応する構造がないことを表わすことになる。

第二のステップは、探索である。完備化を先行させることにより、ある弧の不定部分は（未解析の部分を含まず）誤りの存在のみを示しているので、この不定部分を追跡してゆくことにより、誤り箇所を同定することができる。探索は、入力単語列全体から開始記号をカテゴリとする構造を探すことから始められ、不定部分にカテゴリCを含み、Cを含むカテゴリ列の始点、終点がS,Eであるような弧と、Cをラベルとし、そのラベルの始点、終点がS',E'（ただしS≤S'かつE'≤E）であるような弧とをcompletionしてゆくことで、不定部分を絞り込んでゆく。直感的には、図3に示すような関係を持つ2つの一般化弧(1)(2)を組み合せ、(3)の弧を生成することによって不定部分を絞り込んでゆく。このようなcompletionによりTop-down ruleへの依存度を減少させることができる。不定部分がLexical Categoryにまで絞り込まれた時点で、挿入、脱落、置換に対する規則が適用され、ひとつの解が得られる。

前述のように、弧の不定部分は誤りの存在する場所を示している。従って、不定部分として存在するカテゴリの数は、誤りの数を越えることはなく、予想される誤り数の最小値に相当する。このカテゴリ数をガイドとして、つまり、このカテゴリ数の少ないものから優先的に探索を行なうことで、探索を制御でき、誤り数の少ないものから順に解を得てゆくことが可能となる。

以下に各ステップの詳細を述べる。

弧の完備化

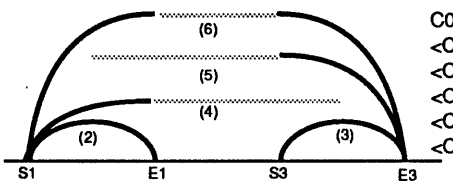
このステップでは、データ構造として前章で示した一般化弧を用いる。その処理は図4に示す2種類の規則によって表現されるチャート解析である。第一の規則Bottom-up ruleは、上昇型左隅解析によって得られた非活性弧に対して文法右辺左隅以外とのマッチングを行ない、新たな弧を生成する。第二の規則Fundamental ruleは、活性弧の任意の位置の不定部分と不活性弧のcompletionを行なう。これは新たな活性弧が生成されなくなるまで繰り返される。ここでは単純化して記述してあるため、このままでは同じ弧を重複して生成するが、簡単な場合分けにより、重複する弧を生成しないようにすることができる。また、Mellishの Simplification ruleに相当するものが必要となるが、ここでは省略した。

探索処理

このステップで用いるのは以下に示すデータ構造で、これを探索子と呼ぶこととする。

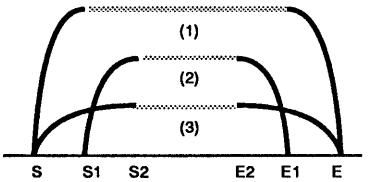
<hole:N err:M

Cs_1 from S_1 to E_1 , ... , Cs_n from S_n to E_n>



C0->C1 C2 C3 (1)
<C1 from S1 to E1 needs nothing> (2)
<C3 from S3 to E3 needs nothing> (ただし、E1≤S3) (3)
<C0 from S1 to * needs [C2 C3] from E1 to *> (4)
<C0 from * to E3 needs [C1 C2] from * to S3> (5)
<C0 from S1 to E3 needs [C2] from E1 to S3> (6)

図2　弧の完備化処理の例



<C from S to E needs [C1] from S1 to E1> (1)
<C1 from S1 to E1 needs [C2] from S2 to E2> (2)
<C from S to E needs [C2] from S2 to E2> (3)

図3　探索処理のイメージ

Bottom-up rule　（左隅以外からの生成を行う）
<C from S to E needs nothing>なる弧があり
文法中に、C1 -> ...Cs1 C Cs2... なる規則が存在すれば、（ただし、Cs1は空でない）
<C1 from * to E2 needs Cs1 from * to S , Cs2 from E to E2>なる弧を生成する
ただし、Cs2が空の時 , E2=E それ以外では E2=*

Fundamental rule　（左隅以外にも適用するように拡張）
<C from S to E needs ... , [...Cs11,C1,Cs12...] from s1 to e1 , ...>なる弧があり
<C1 from S1 to E1 needs nothing>なる弧があれば
<C from S to E needs ... , Cs11 from s1 to S1 , Cs12 from E1 to e1,...>なる弧を生成する
ただし、s1 ≤ S1 もしくは s1=*、かつ、E1 ≤e1 もしくはe1=*

図4　弧の完備化処理の規則

ここで、holeの部分の数値Nは、この探索子の不定部分のカテゴリ数であり、errの部分の数値Mは、この探索子を得るまでに修正された誤り数である。それ以外の部分は一般化弧からneedsの左側を除いたものである。処理は、この探索子とこれまでの処理によって得られている弧とから、新しい探索子を生成することによって進められる。図5に示すように、不定部分の追跡のために2種類の規則、Top-down ruleとActive Fundamental ruleが、誤り部分発見のために3種類の規則が利用される。Top-down ruleは、文法規則を用いて誤り部分を下降的に探索する。Active Fundamental ruleは、以前の解析で得られた活性弧とのcompletionによって誤り部分を下降的に探索する。中心的な規則は、このActive Fundamental ruleであり、これにより、Top-down ruleへの依存を減らし、効率的に誤り部分に達することができる。探索は以下の探索子から始められる。

<hole:1 err:0 [S] from 0 to L>

ここで、Sは開始記号、Lは入力の単語長である。この探索子から規則に従って新しい探索子を生成してゆき、次の探索子が得られた時点で、ひとつの解が得られる。

<hole:0 err:_ nothing>

制御はどの探索子に優先して規則を適用するかによって行なわれ、holeの値とerrの値の和がの小さい探索子が、その値が同じものの中では、holeの値の小さいものが優先される。この制御により、誤り数の少ないものから順に解が得られることになる。

実行例

本手法の実行例を図6に示す。図中では、上から、文法、入力例、上昇型解析後に残される弧の一部、弧の完備化によって生成される弧の一部、探索の過程を示している。完備化によって左隅解析で得られる右側に不定部分を持つ活性弧だけでなく、左側や両側に不定部分を持つ弧が生成される。探索は、この例ではTop-down ruleを適用する必要はなく、すべてActive Fundamental ruleによって行なわれ、最後にUnknown word ruleが適用される。図中に(1)の探索子にTop-down ruleを適用して得られる探索子(2')が示されているが、この探索子はholeの値が大きいので、その後の処理は後回しとなる。

実際、Top-down ruleの適用は、ある文法規則の右辺に並ぶすべての構造に誤りが含まれているときにのみ必要となる。そうでない場合には、弧の完備化によって相当する活性弧が既に生成されているからである。従って、誤り数がひとつのときには、右辺に現われるカテゴリ数が1であるような規則を除けば、Top-down ruleを適用する必要はない。

4　評価

本方式を評価するために、簡単な実験を行なった。規則数158、カテゴリ数48の極めて小規模なCFG（ε規則は含まない）について、本方式を適用し、ひとつの弧に関する処理を1サイクルとして、サイクル数で評価した。この評価法はMellish[1]のものと同様である。入力はまず、その文法の受理できる与えられた単語数の文を考え、そこに対象となる誤りをひとつ含ませたものを利用した。その結果を表1に示す。BUCycleは、最初の上昇型解析が終了するまでのサイクル数である。この時点では、サイクル数と弧の数は一致するので、ここは非活性弧数／活性弧数を記述した。ECCycleは弧の完備化に費やされたサイクル数である。Bottom-up ruleは、非活性弧の数だけのサイクルを必要とし、それ以外はFundamental ruleに費やされたサイクルである。探索は最初に得られた解と同じ誤り数を持つ解がすべて得られた時点で終了するようにした。First,Last,TDCycleは、それぞれ、最初の解が得られるまでに要したサイクル数、解の数／最後の解が得られるまでに要したサイクル数、探索が終了するまでに要したサイクル数である。

Top-down rule
<hole:N err:M [C1...Cs1] fromS1 to E1 , ...>なる探索子があり
文法中に、C1 -> RHSなる規則が存在すれば、
<hole: N+(length of RHS)-1 err:M [...RHS ...Cs1] fromS1 to E1 , ...>

Active Fundamental rule （活性弧どうしのcompletionに相当）
<hole:N err:M [...Cs11,C1,Cs12...] from s1 to e1 , ...>なる探索子があり
<C1 from S to E needs Cs1 from S1 to E1 , ... , Csn from Sn to En >なる弧があれば
<hole:N+∑(length of Csi)-1 err:M
Cs11 from s1 to S , Cs1 from S1 to E1 , ... , Csn from Sn to En , Cs12 from E to e1 , ...>
なる探索子を生成する

Garbage rule 挿入誤りの規則
<hole:N err:M [C1...Cs1] from s1 to e1 , ...> なる探索子があり（ただし、C1は語彙記号）
<C1 from S1 to E1 needs nothing>なる弧があれば （ただし、s1 ≤ S1）
<hole:N-1 err:M+(S1-s1) Cs1 from E1 to e1 , ...>なる探索子を生成する

Unknown word rule 未知語、置換誤りの規則
<hole:N err:M [C1...Cs1] from s1 to e1 , ...> なる探索子があり（ただし、C1は語彙記号）
<C1 from s1 to s1+1 needs nothing>なる弧がなければ
<hole:N-1 err:M+1 Cs1 from s1+1 to e1 , ...>なる探索子を生成する

Empty category rule 脱落誤りの規則
<hole:N err:M Cs1 from s to s , Cs2 from s2 to e2 , ...>なる探索子があれば
<hole:N-(length of Cs1) err:M+(length of Cs1) Cs2 from s2 to e2 , ...>なる探索子を生成する

図5 探索処理の規則

文法規則 S -> NP VP ...(G0) NP -> N NP -> Det N
NP -> NP C NP VP -> TV NP VP -> IV
VP -> VP PP PP -> P NP ...(G1)

入力 The lady bought cakes an the shop
0 1 2 3 4 5 6 7

初期状態（部分） <NP from 0 to 2 needs nothing>
<TV from 2 to 3 needs nothing>
<NP from 3 to 4 needs nothing>
<NP from 5 to 7 needs nothing> ...(i0)
<S from 0 to * needs [VP] from 2 to *> ...(a)
<VP from 2 to * needs [PP] from 4 to *> ...(b)
<VP from 2 to * needs [NP] from 3 to *> ...(c)
<NP from 3 to * needs [C NP] from 4 to *> ...(d)

弧の完備化で得られる弧の例
<PP from * to 7 needs [P] from * to 5> ...(e) (G1)と(i0)より、Bottom-up ruleを適用
<NP from 3 to 7 needs [C] from 4 to 5> ...(f) (d)と(i0)より、Fundamental ruleを適用

探索処理の過程
(1) <hole:1 err:0 [S] from 0 to 7> 探索開始時の探索子
(2) <hole:1 err:0 [VP] from 2 to 7> (1)と(a)より、Active Fundamental ruleを適用
(2') <hole:2 err:0 [NP VP] from 0 to 7> (1) と(G0)より、Top-down ruleを適用

(3) <hole:1 err:0 [PP] from 4 to 7> (2)と (b)より、Active Fundamental ruleを適用
(4) <hole:1 err:0 [P] from 4 to 5> (3)と (e)より、Active Fundamental ruleを適用
(5) <hole:0 err:1 nothing> (4)より、Unknown word ruleを適用
'an'が前置詞(P)である単語と置換されている

(6) <hole:1 err:0 [NP] from 3 to 7> (2)と (c)より、Active Fundamental ruleを適用
(7) <hole:1 err:0 [C] from 4 to 5> (6)と (f)より、Active Fundamental ruleを適用
(8) <hole:0 err:1 nothing> (7)より、Unknown word ruleを適用
'an'が接続詞(C)である単語と置換されている

図6 実行例

表1　簡単な文法規則による評価

誤り種類	単語数	BUCycle	ECCycle	First	Last	TDCycle
無し	6	20/51				
	9	25/58				
	12	49/106				
一語の削除	6	14/39	160	5	7/16	17
	9	15/33	118	8	4/14	20
	12	40/94	426	5	16/32	65
未知語一語の挿入	6	15/44	170	9	3/14	28
	9	20/51	187	8	1/8	27
	12	39/92	399	16	2/39	67
既知語一語の挿入	6	20/56	215	15	4/34	35
	9	25/63	232	14	2/32	34
	12	42/97	426	17	2/40	68
未知語との一語置換	6	10/26	101	8	3/13	19
	9	15/33	118	13	1/13	21
	12	40/94	426	5	12/32	65
既知語との一語置換	6	12/31	114	8	3/13	19
	9	17/38	131	13	1/13	21
	12	46/114	509	5	24/44	89

結果を概観してみると、弧の完備化に普通の解析の約3倍のサイクル数を要し、探索自体は、普通の解析と同等以下のサイクル数しか要していない。解析全体として、整った文の解析に必要とするサイクル数の約4倍で、誤りを一箇所含んだ文の解析が行なえる。Mellishの提案した手法では、同じ処理に整った文の解析に必要とするサイクル数の約7倍を要している。ただし、本手法では、尤もらしい最初の誤りをひとつ求めるまでにも、尤もらしいすべての解を求めるのと同程度のサイクル数を必要とするが、これについては、Mellishの提案した手法の方がより少ないサイクル数で解に到達できる。

5　考察

解析結果の取得について

チャート解析においては、下降型の解析においても、Fundamental rule (completion)により上昇的に解析結果を取得することができる。本手法の探索においては、これに相当する過程が存在しないため、解析結果、例えば、統語構造の取得も下降的に行なう必要がある。これを実現するために、変数とそれへの代入という形式で、解析結果を保存する。まず、上昇型左隅解析と弧の完備化の過程で得られる一般化弧については、それに関連する文法規則を用いて、必要な情報を取得し、不定部分に相当する情報はそれに対応付けられた変数で埋める。例えば、規則C1 <- C2 C3 C4に関連し、C3を不定部分とする一般化弧は、以下のように表現される。

<C1/T1(T2 V3 T4) from S1 to E1
needs [C3/V3] from S3 to E3>

ここで、カテゴリの/以下に記述されているのが統語構造など解析において取得すべき情報である。ラベルについては変数V3を含んだ形式で情報が記述されている。この変数V3が統語カテゴリC3から得られるものであることが、C3/V3の形式で不定部分に記述される。Tnは変数を含まない情報である。同様に、規則C3 <- C5 C6に関連し、C5を不定部分とする一般化弧は、以下のように表現される。

<C3/T3(V5 T6) from S3 to E3
needs [C5/V5] from S3 to E5>

探索子には、subst:という情報を付加し、ここに探索中に得られる情報を代入の列の形式で記述する。探索におけるこれらの扱いをActive Fundamental ruleを例に考える。探索は、空の代入の列を持ち、解析結果全体が代入される変数V1と開始記号C1とを対応付けた探索子

<hole:1 err:0 subst:[] [C1/V1] from 0 to L>

から始められる。新しい探索子が生成される度に、対応する変数への代入が代入リストに加えられてゆく。上に示した2つの一般化弧とのActive Fundamental rule適用を以下に示す。

<hole:1 err:0 subst:[V1=T1(T2 V3 T4)]
[C3/V3] from S3 to E3>
<hole:1 err:0
subst:[V1=T1(T2 V3 T4),V3=T3(V5,T6)]
[C5/V5] from S5 to E5>

解が得られた時点で、この代入を評価することで、解析結果全体を示す変数に解析結果情報が得られる。上の例に続いて、V5=T5なる代入が得られた場合、代入リストを評価して得られる情報は、T1(T2 T3(T5 T6) T4) となる。

効率の改善について

本手法を更に効率化するひとつの方法として、誤り数の上限を設定することが挙げられる。本方式において、時間を大きく消費しているのは、前半の弧の完備化の過程である。この過程で得られるある種の弧は、探索においてそれを利用すると、必然的に一定数以上の誤りを仮定することになるもの、つまり、誤り数の上限が定まっていれば不要であるような弧である。前述した例

The lady bought cakes an the shop
0 1 2 3 4 5 6 7

では、<NP from 0 to 7 needs [C] form 2 to 5>なる弧は、等値接続詞Cを構成する最大単語数が1であることを考えれば、2つの単語の挿入を仮定することになるし、<S from 5 to 7 needs [VP] from 7 to *>なる弧は、先の文法では少なくとも5つの単語の挿入と1つの単語の欠落を仮定することになる。誤り数の上限を設定し、これらの弧の生成を抑制することで効率を改善することができる。探索の段階でも、同様に、探索子のhole数やerr数を利用してTop-down ruleの適用を抑制することができる。

また、探索の過程で、同じhole数を持った探索子内の優先順位を規定することによって最初の解をより早い時点で発見することも可能と考えられるが、これについてはMellishの提案している得点計算の一部が利用できると思われる。

重複した解析について

本手法における後半の探索過程は、単純な下降型の解析と似たメカニズムで行なわれている。このため、チャート解析における下降型解析とは異なり重複した解析を行なうことがある。先の例で、文法に新たな規則N -> N PPを加えたとする。このとき、<N from 3 to * needs [PP] from 4 to *>の一般化弧が得られるが、これを利用して行なわれる解析は、<VP from 2 to * needs [PP] from 4 to *>を利用して行なわれる解析と重複する。

このような解析の重複を回避するためには、探索子を大域的な探索の状態を保存するものとせず、一般のチャート解析のように局所化し、これらからFundamental rule(completion)による弧の生成に相当するものによって解析の途中結果を保存しなければならない。しかし、このような情報の局所化（大域的な探索状態の放棄）とFundamental rule(completion)の利用は大きな時間を消費する。一方重複している部分の解析は、straightforwardに行なわれる。この点を考慮すると、現実的な場面で解析の途中結果の保存が効率的であるかは疑問が残る。

今後の課題

本手法を自然言語インタフェースなどで利用するためには、より大きな規模の現実的な文法で有効性を検証する必要がある。同時に、単一化文法やDCG等の拡張CFGで、実際の動作を確認する必要もある。日本語への応用では形態素解析との結合が問題となろう。また、このような応用においては、得られた誤り候補の中から誤り数以外の基準付けを用いて更にどれが尤もらしいかを判定するメカニズムも必要である。ひとつの方法として、文脈情報を利用して発話予測と関係付けた尤もらしさを利用することが考えられる。更にこのような発話予測が入力の構造を非常に高い確度で制約するのであれば、Bottom-up ruleの適用、特に上部構造に関するものを制約することで、効率のよい非文復旧が可能であると考えられる。

参考文献

[1] Mellish , Chris S. : Some Chart-Based Techniques for Parsing Ill-Formed Input. 1989 Proceeding of 27th ACL pp102~109

[2] Weishedel , Ralph M. et.al. : Meta-Rules as a Basis for Processing Ill-Formed Input. 1983 AJCL 9 (3-4) pp161~177

[3] Kudo , Ikuo et.al : Schema Method : A Framework for Correcting Grammatically Ill-formed Input. 1988 Proceeding of COLING88 pp341~347

[4] Saito , Hiroaki et.al : Parsing Noisy Sentences. 1988 Proceeding of COLING88 pp561~566

[5] 劉学敏他：統合パーサによるノイズを含んだ文の理解. 1990 情報処理学会自然言語研究会79-2

[6] Lang , Bernard : Parsing Incomplete Sentences. 1988 Proceeding of COLING88 pp365~371

[7] 神岡太郎他：仮説生成機構を用いた未知語を含む文の解析 1988 人工知能学会論文誌 3(5) pp627~638

[8] Kay , Martin : Algorithm Schemata and Data Structures in Syntactic Processing. 1980 Research Report CSL-80-12 Xerox PARC

[9] Gazdar , Gerald et.al. : Natural Language Processing in LISP. 1989 Addison-Wesley

[10] Pereira , Fernando C. N. et.al. : Definite Clause Grammars for Language Analysis - A Survey of the Formalism and a Comparison with Augmented Transition Networks. 1980 Artificial Intelligence 13(3) pp231~278

[11] Shieber , Stuart M. : An Introduction to Unification-Based Approaches to Grammar. 1986 CSLI Lecture Notes 4