8M-06

不完全なGDAタグを持つ文書の構文解析によるタグの詳細化

横山　憲司† 白松　俊† 伊藤　誠悟†
橋田　浩一‡ 奥乃　博†
†東京理科大学 理工学部 情報科学科 ‡電子技術総合研究所

文書に意味構造を明示するタグを付与する作業には人手のコストがかかる。そのコストを抑制するため，まず人がおおまかにタグ付けをしておき，そのタグによる制約と構文解析による制約の双方を用いて，さらに詳細なレベルのタグ付き文書を生成する手法を提案し，実装したシステムを報告する．実装の手法の鍵はタグをはずす時に主辞を抽出しながら文章の簡略化を行い曖昧性の解消を促進していることである．これにより，GDAタグを用いると，自然言語に潜む様々な曖昧性を解消できることを示す．

1. はじめに

1.1 自動GDAタグ詳細化の必要性

GDA(Global Document Annotation)は統語・意味情報が明示できるように設計されたXMLタグセットである．統語的・意味的曖昧性を解消するために，係り受け，代名詞の指示対象，多義語の意味などを明示するタグ付けを想定している．しかし，人間がタグを付ける場合，タグ付けのコストが問題になる．本稿ではタグ付けのコストを抑制するため，構文解析と人間の付けたタグ情報を用いて，おおまかなタグをもつGDA文書からより詳細度の高いGDA文書に変換するシステムを開発したので報告する．

1.2 用語の定義

次節に入る前に，本稿で出てくる用語の定義を述べておく．

エレメント：開始タグから対応する終了タグまでの文字列を**エレメント**（Element）と言う．

構成文字列：エレメントからタグを除いた文字列を**構成文字列**と言う．

主辞：構成文字列の係り受け関係において，2つの構成文字列の一方が他方に係ることにより，受ける側の構成文字列を中心としてひとまとまりの意味を持つ大きな構成文字列を作る．受ける側の構成文字列をその大きな構成文字列の**主辞**(head)と言う．

2. 曖昧性解消と詳細化の方針

曖昧性を解消するために人間が事前にタグ付けをする．このタグが維持された状態でタグの詳細化が行われる．これは人間の付けたタグが最も重視されるべき制約であるという方針による．エレメントの構成文字列から主辞の抽出をすることで得られる簡略化された文についてタグの詳細化をし，詳細化されたタグ付き文書を子供の主辞と詳細化の情報を用いて展開する．これを再帰的に繰り返していく．

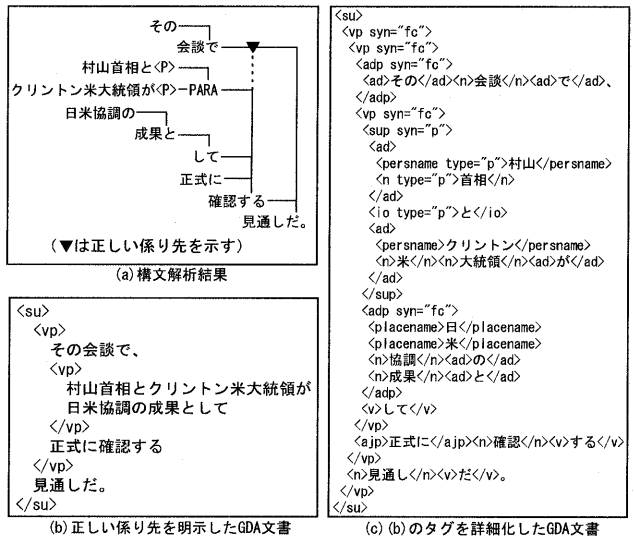

図1 GDAタグ詳細化の例

3. GDAタグ詳細化の手順

現在の構文解析では誤った結果を返す**図1(a)**のような文書があった場合，係り受け関係を明示するため事前に人間がタグを付けておく**図1(b)**．そしてそのタグ付き文書をシステムに通せば，**図1(c)**のような結果を返す．以下に，実装における各工程の概略を述べる．また，その中でも特に重要と思われる工程については詳細を示す．

3.1 各工程の概略

(1) GDA文書をDOM木に変換する．

(2) 既存のツールを用いて形態素解析，構文解析，タギングを順次実行する．

(3) 人間が付けたタグと機械が付けたタグ双方の評価をし，適切なタグ付き文書を生成する．

Refinement of the document which has GDA tags incompletely by the syntactic analysis

†Department of Information Sciences, Science University of Tokyo
‡Information Science Division, Electrotechnical Laboratory

(4) エレメントの構成文字列から主辞を抜き出す（主辞抽出）.

(5) 主辞とDOM木のセットが格納されているリストを用いて，主辞とDOM木を置換する（展開処理）.

(6) 人間による不完全なタグ付き文書のエレメントについて再帰的にタグの詳細化を実行していく.

3.2 (2) の詳細 [ツール]

使用したツール	
ツール	用途
JUMAN	形態素解析
KNP	構文解析
knp2gda	KNP の結果を用いて GDA タグを付ける

3.3 (3) の詳細 [適切なタグの採択]

人間が付けたタグを t_m，t_m で始まるエレメントを e とする．e と同じ構成文字列に対して詳細化システムが自動的に付けたタグを t_s，t_s で始まるエレメントを ε とする．t_m と t_s は同じになるとは限らないので，場合分けが必要となる．

- t_m と t_s が同じ場合—ε を採択.
- t_m と t_s が句か句でないかの違いだけの場合—t_s を t_m に書き換えて，ε を採択.
- t_e と t_ε が異なる場合—詳細化された ε は破棄し，e を採択.

3.4 (4) の詳細 [主辞抽出]

3.4.1 主辞を抽出する必要性

人手によって付けられた不完全なタグからエレメントが得られる．それらエレメントの構成文字列について詳細化を進めていく時，子エレメントから簡略化された文字列を生成することで親エレメントの処理時には子エレメントの構文解析をせずに済む．この方法を用いると，元のタグは維持されたまま，タグの詳細化がDOM木の葉から根に向かって進められる．次の例では，主辞を抽出する意味がわかりやすい．

```
<su>
  <n>風に乗った子供達の楽しそうな歌声</n>
  が聞こえる。
</su>
```

子エレメントは，"<n>風に乗った子供達の楽しそうな歌声</n>" であり，親エレメントに渡される主辞は「歌声」となる．したがって，親エレメントのレベルでは "<su>歌声が聞こえる。</su>" を詳細化することになる．

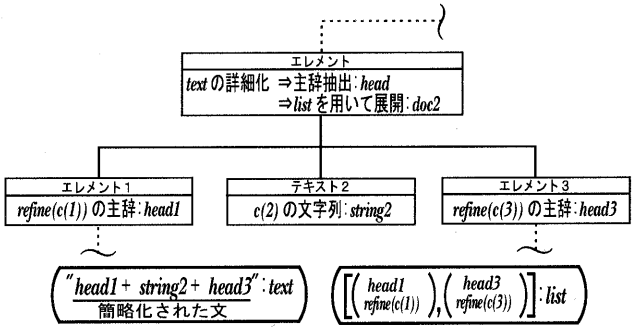

図 2 工程 (6):再帰的関数 refine の図説

3.4.2 主辞を抽出する方法

エレメント e の主辞を抜き出すには，e の子供を後ろから前へ走査し，名詞または名詞句が発見されるまでの文字列をとる.

3.5 (6) の詳細 [タグの詳細化]

この工程 (6) は (1)〜(5) までを再帰的に呼び出して詳細化を行なう本体部分である．具体的なアルゴリズムとその図を示す.

```
Factor refine (node) {
  for i := 0 to node の子供の数
    if(node の i 番目の子供 c(i) のタイプが TEXT_NODE)
      text に c(i) の表す文字列をつなげる;
    else{
      list に refine(c(i)) を加える;
      text に refine(c(i)) の主辞をつなげる;
    }
  text に工程 (1), (2) を適用した結果 -> dom;
  dom と node に工程 (3) を適用した結果 -> doc1;
  doc1 に工程 (4) を適用した結果 -> head;
  list と doc1 に工程 (5) を適用した結果 -> doc2;
  return Factor(head, doc2);
}
```

4. 評　　価

実装はJavaを用いて行なった．毎日新聞データベースからKNPで構築された京大コーパスにはほぼ全文に対して人手が入っており，完全な自動化ができていない．KNPが誤って解析してしまうような文に対しても，事前にごく簡単なタグを入れておけば曖昧性が解消でき，正しいタグが付けられることが確認できた．すなわち，自動解析が誤るような50の例文を用意し，正しい係り受けを明示するための最低限のタグを付けて本システムに通したところ．41の文について正しい結果が得られた.

5. ま と め

本稿では構文解析を用いたGDAタグ詳細化の方法について述べた．今後の予定としては以下のことを考えている．1) EmacsのGDAタギングエディタからタグ付けの詳細化を可能にするため，本システムをプラグインとして改良する，2) 大規模データベースに適用し，GDA文書を自動作成する，3) 事前のタグ付けのレベルと詳細化の精度の関係を実証的に示す.

最後に，knp2gdaを提供いただいたソニーCSL飯田仁博士，慶應大学鈴木潤氏に感謝する.

参 考 文 献

1) 橋田 浩一: 大域文書修飾 Global Document Annotation (GDA): http://www.etl.go.jp/etl/nl/gda/
2) 黒橋禎夫:「結構やるな KNP」: 情報処理 2000/11 vol.41
3) W3C Document Object Model (DOM): http://www.w3.org/DOM/
4) 長尾 真，佐藤理史，黒橋禎夫，角田達彦:『自然言語処理』, 岩波書店, 1996
5) 山本 他: 不完全なタグを持つ文書の照応解析によるタグの洗練化, 8M-08, 情報処理学会第62回全国大会, 2001