数理解析研究所講究録 1426 巻 2005 年 26-31

構文解析木の類似度の判定アルゴリズム

椎名 広光 (Hiromitsu Shiina), 秋友 克俊 (Katsutoshi Akitomo)
岡山理科大学 総合情報学部 (Okayama University of Science)

1 はじめに

Web などにおける検索システムは，テキストデータ中の文字列を基本としてサーチをしている．これに対して，本来 Web データの文章は日本語として構造がつけられており，意味的な類似性を考えるのであるならば，構文構造を主眼とした類似性を図る必要性がある．本研究では，類似度の計算方法のひとつとして 2 つの構文解析木の類似度を，それぞれの構文解析木の部分木が完全一致する個数を基にする方法と部分木の編集距離を基にしてする方法を提案し，その計算方法について述べる．編集距離については，対象を構文解析木としているために，編集距離の計算に文法規則を用いようとするものである．

2 諸定義

本研究で利用する記号について定義をする．文法 $G=(N,T,P,S)$, N:非終端記号, T:終端記号, P:文法規則, S:開始記号に対して，文法から生成される終端記号列の対する構文解析木を T で表す．ただし，終端記号列に対して構文解析木は複数あってもかまわないが，アルゴリズム等の表現上の簡便性のために文法規則は Chomsky 標準形とする．また，構文解析木 T_A の部分木集合を $SF(T_A)$=$\{\alpha_{A,1},\alpha_{A,2},\ldots,\alpha_{A,N_A}\}$ とする (N_A は $SF(T_A)$ の部分木の個数)．また，文法規則の 1 つから生成される 1 つの構文解析木の部分木集合 (以下，文法規則部分木と省略) を $IF(T_A)$=$\{\alpha^1_{A,1},\alpha^1_{A,2},\ldots,\alpha^1_{A,N^1_A}\}$ であらわすことにする (N^1_A は $IF(T_A)$ の部分木の個数).

3 文法と構文解析木の例

本稿では，文法例 $G_1=\{N_1,T_1,P_1,S\}$, $N_1=\{S,A,B\}$, $T_1=\{a\}$, $P_1=\{S\rightarrow AA, S\rightarrow AB, A\rightarrow AA, B\rightarrow AA, B\rightarrow BA, A\rightarrow a\}$ に対して，次の 3 つの構文解析木 T_A,T_B,T_C 用いる (図 1)．また，T_A,T_B,T_C の部分木集合の個数は，$N_A=14$, $N_B=6$, $N_C=9$ である (図 3, 図 4 参照)．なお，構文解析木 T_A,T_B,T_C には終端記号を除いている．

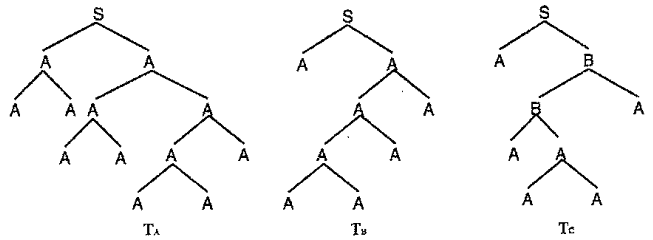

図 1: 構文解析木の例 T_A,T_B,T_C

4 部分木の一致度による類似度

部分木の一致度による類似度では，例えば構文解析木 T_A と T_B を部分的に見てみると同じ部分木が含まれている (図 2)．このように，同じ部分木の頻度を構文解析木の類似度として計算する．

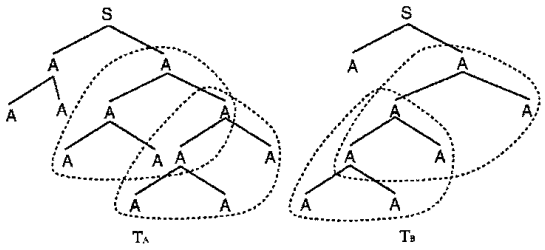

図 2: 構文解析木 T_A と T_B の部分木の削除による類似

部分木の一致度による類似度の計算は，構文解析木の部分木集合 $SF(T_A)$, $SF(T_B)$ 内の部分木 $\alpha_k \in SF(T_A)$, $\beta_l \in SF(T_B)$ が完全に一致する個数を計算する．計算式は次のとおりである．

$$sim_1(T_A, T_B) = \frac{2}{N_A + N_B} \sum_{k=1}^{N_A} \sum_{l=1}^{N_B} com_1(\alpha_k, \beta_l)$$

$$com_1(T_A, T_B) = \begin{cases} 1, & \alpha_k = \beta_l, \\ 0, & \alpha_k \neq \beta_l. \end{cases}$$

例えば，構文解析木 T_A, T_B の部分木の一致による類似度の計算を行うとする．ここで計算を行うには，比較する構文解析 2 つの部分木集合 $SF(T_A)$，$SF(T_B)$ を計算する必要があり，図 3,4 にそれらの示す．

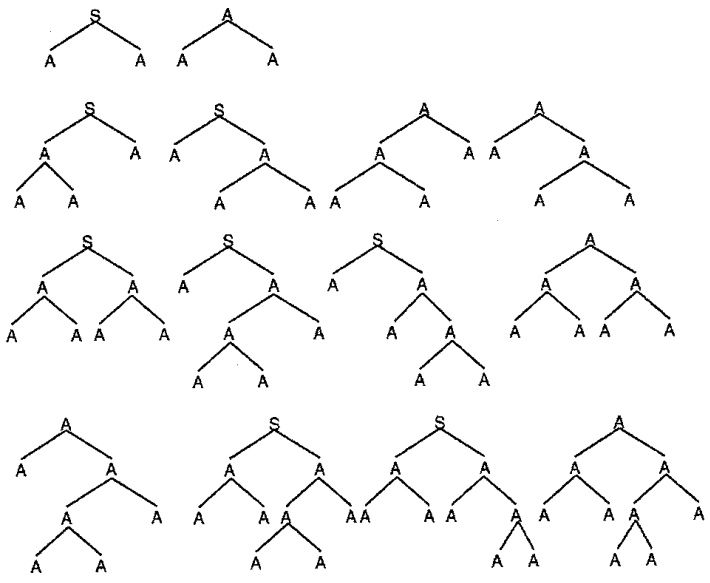

図 3: 構文解析木 T_A の部分木

また，部分木集合 $SF(T_B)$ に対して，$SF(T_A)$ の部分木に一致しているものに○をつけたものを図 4 に示す．

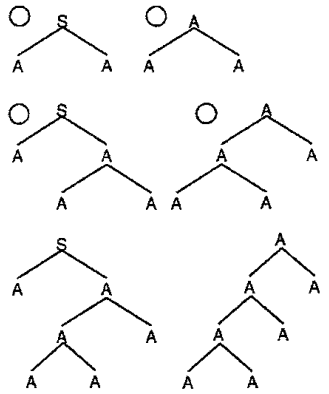

図 4: 部分木集合 T_A と一致する部分木集合 T_B

構文解析木 T_A と T_B の類似度は，$sim_1(T_A, T_B) = \frac{2}{14+6} \cdot 4 = 0.4000$ となる．また，構文解析木 T_A と T_C の類似度は $sim_1(T_A, T_C) = \frac{2}{14+9} \cdot 1 = 0.0870$ である．

5 構文解析木の編集距離による類似度の計算

部分木の一致度の他にも構文解析木に対して部分木を追加したり削除の操作をおこない，その操作回数を基にして２つの構文解析木の類似度を計算する．部分木の操作による類似度の計算は，比較対象の構文解析木を分解して，部分木を合成しながら一方の構文解析木を得ようとするものである．部分木の合成に関していえば，ボトムアップに合成してゆく．

以下では，構文解析木の分解操作と合成操作について述べる．ただし，それぞれの操作を行うのに次の記号を定義する．構文解析木 T_C をもとにして構文解析木 T_A へ変形してゆく操作において，i 回目の操作中でできる部分木集合 (以下，操作部分集合と省略する) を $CF(T_A, T_C, i)$=$\{\beta_{C,1}, \beta_{C,2}, \ldots, \beta_{C,n_i}\}$ とする また，$CF(T_A, T_C, 0)$=$\{T_C\}$ である．

5.1 部分木の分解操作

操作部分木集合 $CF(T_A, T_C, i)$ の１つの部分木 $\beta_C \in CF(T_A, T_C, i)$ に対して，その一番上の親を削除し，新しい部分木 $\beta_{C,n+1}, \beta_{C,n+2}$ を作成し追加する．

$$CF(T_A, T_C, i+1) = CF(T_A, T_C, i) \cup \{\beta_{A,n+1}, \beta_{A,n+2}\} - \{\beta_{A,1}\}$$

図５は，１回操作分に当たる分割例で，１つの部分木が一番上の親の削除によって分割操作が行われ，２つの部分木が作成されている．

5.2 部分木の合成操作

部分木の合成は，部分木集合の１つの部分木でも類似度を計算する対象の構文解析木 T_A へ近づけるようにする．合成の種類には２種類の場合があり，文法規則１つから作られる部分木を親に追加する場合と文法規則１つから作られる部分木と T_A の部分木集合 $SF(T_A)$ の要素を合成して追加する場合がある．

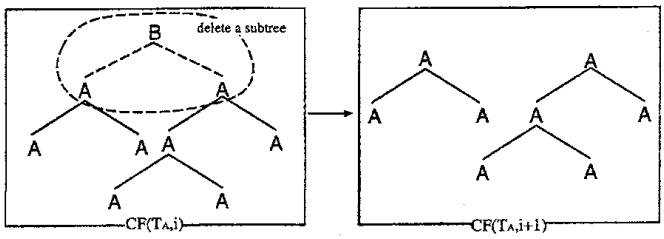

図 5: 分割操作

5.2.1 文法規則の部分木との合成操作

部分木を分解して出来た操作部分木集合 $CF(T_A, T_C, i)$ 内の部分木 1 つを $\beta_C \in CF(T_A, T_C, i)$ とし，類似度を計算する対象となる T_A の文法規則 1 つから作られる文法規則部分木集合 $IF(T_A)$ の部分木を $\alpha_A^1 \in IF(T_A)$ とする．

この 2 つの部分木 β_C と α_A^1 に対し，α' が親 α_A^1 を左側の子供になるように合成し，部分木 α'_A を合成し，α' が親 α_A^1 を右側の子供になるように合成し，部分木 α''_A を合成し，操作部分木集合 $CF(T_A, T_C, i+1)$ に追加する．ただし，合成のときに親と子供の文法記号が一致しない場合は，新たに部分木はできない．加えて，α', α'' が $SF(T_A)$ に属していない場合も，追加しない．

$$
\begin{aligned}
&CF(T_A, T_C, i+1) = CF(T_A, T_C, i) \cup \{\alpha'_A, \alpha''_A\}, \\
&\alpha'_A = \alpha_A^1 \odot (\beta_C, \phi), \alpha''_A = \alpha_A^1 \odot (\phi, \beta_C), \\
&\alpha_A \in IF(T_A), \beta_C \in CF(T_A, T_C, i)
\end{aligned}
$$

なお，$\alpha \odot (\beta, \gamma)$ は，α を親部分，β を左の部分木，γ を右の部分木となるように合成したした木を表し，$\odot$ は合成操作を表す．ただし，上に述べ立てているように，合成できない場合は木は合成できない．

図 6 は，文法規則の部分木の合成を図示したものである．

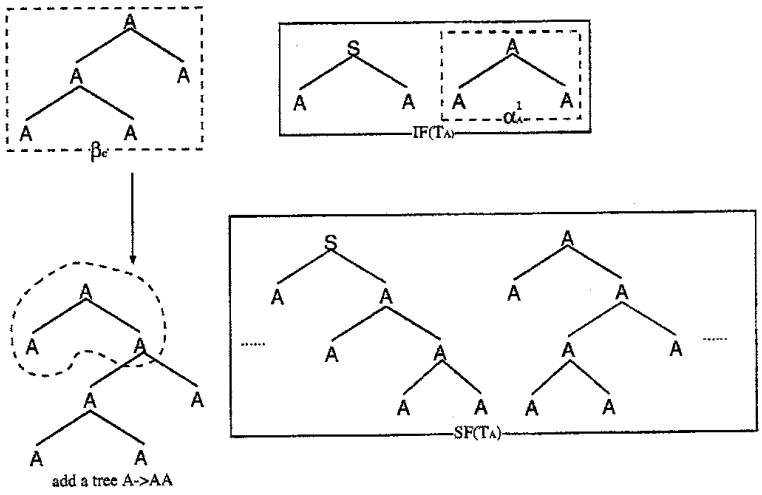

図 6: 部分木 1 つを追加して合成

5.2.2 部分木集合からの合成操作

現在合成している操作部分木集合 $CF(T_A, T_C, i)$ を $\beta_C \in CF(T_A, T_C, i)$，類似度を計算する対象となる T_A の部分木集合 $SF(T_A)$ を部分木 $\alpha_A \in SF(T_A)$，T_A の文法規則1つから作られる文法規則部分木集合 $IF(T_A)$ を部分木 $\alpha_A^1 \in IF(T_A)$ とする．これら3つの部分木 α_A, α_A^1,β_C に対して，α_A^1 を親とし，α_A と β_C を左右の子供とする部分木 α'_A と α''_A を合成し，操作部分木集合に追加する．ただし，**5.2.1** と同様に，α'_A と α''_A の合成は親と子供の部分木の文法規則が一致しないと作成できない．また α' や α'' が $SF(T_A)$ に属していないか T_A と一致しない場合も，追加しない．

$$
\begin{aligned}
&CF(T_A, T_C, i+1) = CF(T_A, T_C, i) \cup \{\alpha'_A, \alpha''_A\} \\
&\alpha'_A = \alpha_A^1 \odot (\beta_C, \alpha_A), \alpha''_A = \alpha_A^1 \odot (\alpha_A, \beta_C), \\
&\alpha_A^1 \in IF(T_A), \alpha_A, \beta_C \in CF(T_A, T_C, i)
\end{aligned}
$$

図7は，部分木集合との部分木の合成を図示したものである．

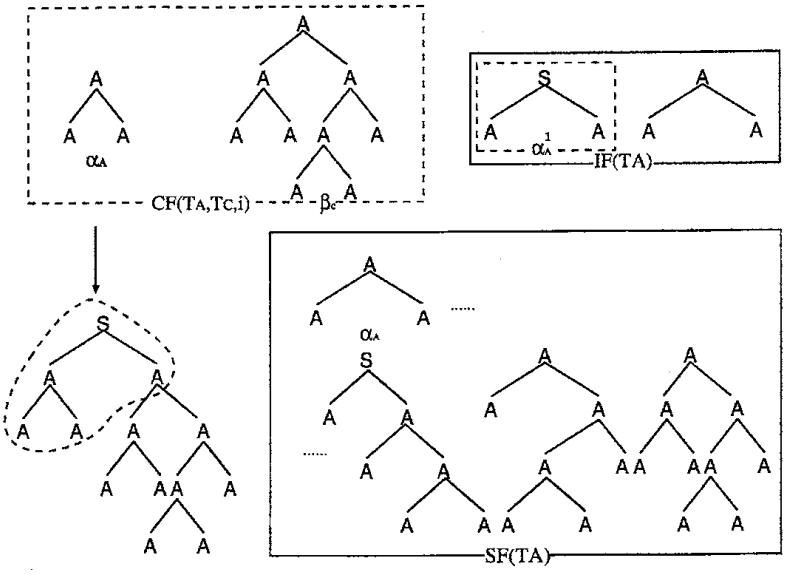

図 7: 部分木集合との合成合成

5.2.3 分割操作と合成操作の制御

5.2.1 と **5.2.2** の部分木の分割と合成操作の制御は，$CF(T_A, T_C, i)$=$\{T_C\}$, $i = 0$ から開始される．分割操作と合成操作は操作部分木集合に対して行われるが，どちらかの操作が行われるたびに i に1加えられる．合成操作において，類似度を求めようとしている目的の構文解析木 T_A が操作部分木集合に含まれているなら操作を終了させ，$com_2(T_A, T_C) = i$ とする．

5.2.4 編集距離類似度

操作回数 $com_2(T_A, T_C)$ から編集距離類似度 $sim_2(T_A, T_C)$ を次のように定義する．

$$
sim_2(T_A, T_C) = \frac{(N_A - com_2(T_A, T_C)) + (N_C - com_2(T_A, T_C))}{N_A + N_C}
$$

図 8 は例の構文解析木 T_C から T_A へ分割と合成を行った様子である．分割操作が 3 回，合成操作が 3 回で合計 6 回で，合成操作のうち文法規則からなる部分木のみの合成は 2 回で，T_A の部分木との合成は 1 回である．

よって，$com_2(T_A, T_C) = 6$ であり，編集距離類似度 $sim_2(T_A, T_C) = \frac{(14-6)+(9-6)}{14+9} = 0.4782$ となる．また，構文解析木 T_A,T_B の $com_2(T_A, T_B) = 4$ であり，編集距離類似度は $sim_2(T_A, T_B) = \frac{(14-4)+(6-4)}{14+6} = 0.600$ である．

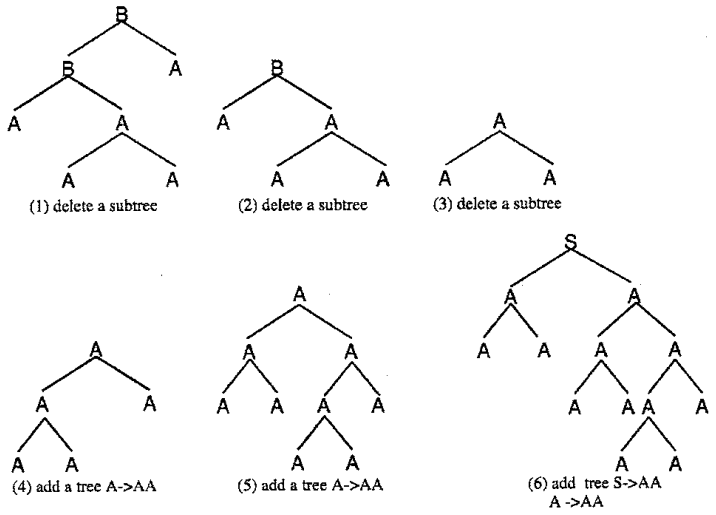

図 8: 構文解析木 $T_C \rightarrow T_A$ への編集操作

6 まとめ

本研究では，部分木の一致する個数を基にする一致類似度と，編集操作を元にした編集距離類似度の 2 種類の構文的類似度について述べた．編集距離類似度については，計算する構文解析木の条件によって編集距離の計算の高速化が可能であることが見込まれる．例えば，操作部分木集合 CF へ追加する部分木の削減や合成操作 CF から選ぶ部分木の選択方法についても改善される．

また，日本語文に対する構文解析済みデータに対する一致類似度については，[6] にて報告予定である．今後は構文的類似度の自然言語における有効性について調査する必要がある．

参考文献

[1] E. Charniak : "Statistical Language Learning", The MIT Press (1993).

[2] 深谷, 山村, 工藤, 松本, 竹内, 大西, 単語と頻度統計を用いた類似性の定量化, 信学論文誌 Vol.J87-DII No.2 pp.611-672(2004)

[3] 高橋, 乾, 松本, テキストの構文的類似度の評価方法について, 情報処理学会 自然言語処理研究会 NL-150-24 , 163-170(2002)

[4] S. Dulucq and H. Touzet, "Analysisy of Tree Edit Distance Algorithms", LNCS 2676, 83-95(2003).

[5] K. Zhang and D. Shasha, "Simple fast alhorithms for the editing distance betwee trees and related problems", SIAM Journal of Computing, Vol18-6, p1245-1262 (1989).

[6] 秋友, 椎名, 構文解析木間の類似度に関する研究, 2005 年電子情報通信学会総合大会, D-5-7 pp45(2005).