構造化文書体系(ODA)の処理:
文書プログラミングのための文書構造変換の形式的記述

4C−8

千葉　和也、京嶋　仁樹
富士ゼロックス(株)システム技術研究所

1. はじめに

オフィスにおいて、形式の決まった文書から形式の決まった文書を作成することは多い。文書プログラミング[1]は、そのような文書処理の自動化のために提案された。

本稿では、構造化文書、特にODA文書[2]におけるそのような文書処理の自動化を扱い、入力の文書から出力の文書への論理構造の変換に形式的記述を与えることを試みる。以下、単に文書の構造というときは論理構造をさすものとする。

構造化文書間の構造の変換を形式化した例としては、NST(Nested Sequence of Tuples)上の代数に基づいたもの[3]がある。ここでは、入力文書(とみなせるもの)にはスキーマがあり、そのスキーマに各種の演算を施すことによって、目的の出力文書を得るようになっている。

本稿では、構造の組み換えでできるような文書処理を対象として、CFG(Context-free Grammar)に基づいた文字列間の変換であるSDT(Syntax-Directed Translation)[4]に基づいた、文書構造間の変換の宣言的な形式的記述が提案される。

この記述では、入出力文書のジェネリック構造がそれぞれ明示的に表現されるため、入出力文書の形式が決まっている場合の処理をうまく記述できる。また、ODAにおいては、その出力文書のジェネリック構造に基づいたジェネリック割付け構造によって、出力文書の割付けが行える。その他に、入出力のジェネリック構造の記述は対等であり、逆向きにも実行可能な変換が記述できるという特徴がある。

2. CFGを用いた文書クラスの記述

まず、木構造の文字列による表現を与える。

【定義】(T')　アルファベットの集合A、Bおよび特別な記号"[","]"(アルファベットの一種として扱う)から作られる文字列の集合T'(A,B)は、次のように帰納的に定義される。(ただしn≧0とする)

- $b \in B \Rightarrow b \in T'(A,B)$
- $a \in A, x_1,\cdots,x_n \in T'(A,B) \Rightarrow a[x_1\cdots x_n] \in T'(A,B)$ □

ここで、$a[x_1\cdots x_n]$を、図1の木構造を示していると解釈する。つまり、T'(A,B)は、Aをリーフ以外のノード、Bをリーフノードとしてもつ木に対応する文字列の集合である。

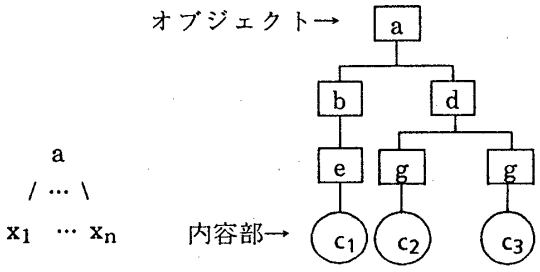

図1　木構造

図2　構造化文書の例

【定義】$T'^*(A,B) \equiv \{x_1\cdots x_n | x_1,\cdots,x_n \in T'(A,B)\} \cup \{\varepsilon\}$ $(n \geqq 1)$ □

【定義】CFG　$G:<N,\Sigma\cup\{"[","]"\},P,S>$　のうち、生成規則の集合P中の全ての生成規則$A\rightarrow\gamma$について$\gamma \in T'^*(\Sigma, N\cup\Sigma)$を満たすようなものを、T-CFG(Tree-preserving Context-free Grammar)という。□

さて、本稿で対象とする構造化文書(図2)およびジェネリック構造を定める。文書の各オブジェクトを1つのアルファベットで表し、1つの文書は、そのオブジェクトの作る木構造を示す文字列σで表す。例えば図2の文書はa[b[e]d[gg]]で表される。ジェネリック構造に相当するものが1つのT-CFGである。σがT-CFG　Gで生成できれば、文書σはそのジェネリック構造Gを満たすという。

3. SDTTを用いた文書構造変換の記述

【定義】木構造(に対応する文字列)間の変換であるSDTT(Syntax-Directed Tree Translation)　Tは、次のような5項組$<N,\Sigma,\Delta,R,S>$である。

"Processing of Structured Documents (ODA): A Formal Description of Translations between Document Structures for Document Programming"
Kazuya CHIBA and Masaki KYOJIMA
System Technology Research Lab., Fuji Xerox Co., Ltd.

- Nは非終端記号の有限集合であり、$N \cap (\Sigma \cup \Delta) = \emptyset$である。
- Σは入力アルファベットの有限集合である。
- Δは出力アルファベットの有限集合である。
- Rは$A \rightarrow \beta, \gamma$の形のルールの有限集合であり、次の条件をみたす。
  - $A \in N, \beta \in T'^*(\Sigma, N \cup \Sigma), \gamma \in T'^*(\Delta, N \cup \Delta)$
  - β中に現れる非終端記号は、γ中に現れる非終端記号と、種類および数が一致する。β中のそれぞれの非終端記号について、γ中にある同じ非終端記号が1対1に対応する。同じ非終端記号が2つ以上ある場合は添字を用いて対応を示す。

また、T-CFG $<N, \Sigma \cup \{"[","]"\}, P_i, S>$、$<N, \Delta \cup \{"[","]"\}, P_o, S>$をそれぞれTの入力文法、出力文法と呼ぶ。ただし、$P_i = \{A \rightarrow \beta | A \rightarrow \beta, \gamma \in R\}$、$P_o = \{A \rightarrow \gamma | A \rightarrow \beta, \gamma \in R\}$とする。□

【定義】(変換対) SDTT $T = <N, \Sigma, \Delta, R, S>$ の変換対π(T)は、次のように定義される。

- $(S,S) \in \pi(T)$であり、2つのSは対応している。
- $A \rightarrow \beta, \gamma \in R$かつ$(\beta', \gamma') \in \pi(T)$であり、β',γ'はそれぞれAを含みその2つが対応しているとき、β',γ'中の上記Aのところで、それぞれ生成規則$A \rightarrow \beta, A \rightarrow \gamma$を適用して得られたものをそれぞれβ",γ"とすると、$(\beta", \gamma") \in \pi(T)$である。□

【定義】(変換集合) SDTT $T = <N, \Sigma, \Delta, R, S>$ の変換集合τ(T)は、G_iを入力文法、G_oを出力文法とすると、G_iの生成する言語$L(G_i)$および$L(G_o)$の間の関係(relation)であり、次の集合である。

$\tau(T) = \{(x,y) | x \in L(G_i), y \in L(G_o), (x,y) \in \pi(T)\}$。□

$(\sigma_i, \sigma_o) \in \tau(T)$を満たすとき、Tにより、文書$\sigma_i$が文書$\sigma_o$に変換されるという。1つの文書に対する変換の出力文書が複数ある場合もあることに注意する。

文書構造変換の記述例を示す。図3に入出力のジェネリック構造をODAでの記述を用いて示した。ここ

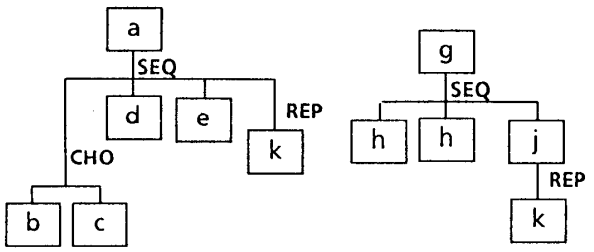

図3 入力ジェネリック構造(左)と出力ジェネリック構造

で、(1)出力側の左側のhには、入力側のbまたはcに付いていた内容部を付け、(2)出力側の右側のhには、入力側のeに付いていた内容部を付け、(3)出力側のkの並びは、入力側のkの並びをそのまま持ってくるような変換は、SDTTを用いて次のように記述できる。(ルールのみを示した。)

S→a[LdEK],	g[LEj[K]]
E→e,	h
L→b,	h
L→c,	h
K→kK,	kK
K→k,	k

例えば、文書a[bdekkk]は文書g[hhkkk]に変換される。出力側の、終端記号で示される最下位オブジェクト(上のh,k)には、対応する入力側の最下位オブジェクトから内容部が付け替えられる。

4. おわりに

CFG間の変換に基づいて、文書構造変換を記述する形式的な枠組みを与えた。T-CFGは、CFGに基づいているので、クラス記述力は強く、例えばNSTのスキーマによっては記述できない、ODAの複雑な従属生成子を記述できる。またそのため、SDTTにより様々な構造変換を記述できる。

さらに複雑な処理、例えば入力側の内容として表現される情報によって、出力側の構造の生成が影響を受けるような変換処理の記述のために、属性文法(Attribute Grammar)を用いてSDTTを拡張することが考えられる。その際に、内容部の情報、例えば、「入力のある項目がいくつあるか」「「可」か「不可」か」などを、属性文法における1つの属性の値として表現することができるだろう。

SDTTの実現は、一般的なCFGの構文解析アルゴリズムを用いれば容易である。実際に文書を生成する際は、出力の論理構造を生成したのち、ODAに代表されるような自動割付け処理を行う必要がある。

今後の課題としては、SDTTに基づいた文書構造変換のシステムとしての実現、属性文法を用いたSDTTの拡張、およびユーザーインターフェースについての考察があげられる。

参考文献

[1] 京嶋仁樹, 上林憲行, N.V.Gopal : 文書プログラミングについての一考察, 情報処理学会 文書処理とヒューマンインタフェース研究会資料, 89-DPHI-23 (1989).

[2] ISO(edt.) : Open Document Architecture (ODA) and Interchange Format (1987).

[3] Guting, R. H., Zicari, R. and Choy, D. M. : An Algebra for Structured Office Documents, *ACM Trans. Office Information Systems*, Vol. 7, No. 4, pp. 123-157(1989).

[4] Lewis II, P. M. and Stearns, R. E. : Syntax-Directed Transduction, *Journal of the ACM*, Vol. 15, No. 3, pp. 465-488(1968).