A-039

あるクラスの属性グラフ文法に対する属性評価アルゴリズム
Attribute Evaluation Algorithm for A Class of Attribute Graph Grammars

安達 由洋 [†]
Yoshihiro Adachi

1. まえがき

属性グラフ文法はグラフ文法をその各プロダクションに意味規則を付加して意味の計算（属性評価）ができるように拡張した文法である．D. Knuth の提案した属性文法に対しては，導出木上の頂点の辿り方に依存したさまざまな属性評価アルゴリズムが提案されている [1]．一方，属性グラフ文法に対してビジュアル言語の定式化やその処理系に関する研究はいくつか報告されているが，属性評価アルゴリズムに関する研究はほとんどない．記号列を生成する文脈自由文法では記号の並びに左から右への向かう自然な順序があるが，グラフ文法ではプロダクションの右辺グラフや生成されるグラフのノードの間に自然な順序がない．そのため属性グラフ文法の属性評価アルゴリズムを構成するには工夫が必要である．

我々は文献 [2] で，順序グラフを生成する edNCE グラフ文法を基底文法とした属性グラフ文法を用いて，Prolog プログラム図の構文規則とレイアウト規則を定式化した Logichart 属性グラフ文法を定義した．そして，この文法により導出される順序グラフのノードの順序に従って意味規則の評価を行うとすべての属性値が計算できることを示した．ただし，文献 [2] では主に Logichart 属性グラフ文法の定義とレイアウトの評価について記述しており，その基になる属性グラフ文法については記述していない．

本稿では，順序グラフを生成する edNCE グラフ文法を基底文法とする属性グラフ文法を定式化するとともに，その文法の属性評価アルゴリズムについて議論する．

2. 順序グラフを生成する属性グラフ文法

この節では，順序グラフを生成する edNCE グラフ文法 [3] を説明した後，これを基底文法とする属性グラフ文法を定義する．

Σ と Γ 上のグラフとは，3 項組 $H = (V, E, \lambda)$ である．ここで，V はノードの空でない有限集合，$E \subseteq \{(v, \gamma, w) \mid v, w \in V, v \neq w, \gamma \in \Gamma\}$ はエッジの有限集合，そして $\lambda : V \to \Sigma$ はノードラベリング関数である．特に，すべてのノード間に線形順序を定義したグラフを順序グラフという．

2 つのグラフ H と K に対して，全単射 $\theta : V_H \to V_K$ が存在して $E_K = \{(\theta(v), \theta(w)) \mid (v, w) \in E_H\}$，かつ，すべての $v \in V_H$ に対して $\lambda_K(\theta(v)) = \lambda_H(v)$ が成り立つとき，H と K は同形であるといい，θ を H から K への同形写像という．なお，順序グラフに対して同形写像 θ は順序を保存するものとする．$V_H \cap V_K = \phi$ のとき，H と K は互いに素であるという．Σ と Γ 上のすべてのグラフの集合を $GR_{\Sigma,\Gamma}$ で表す．

定義 1 edNCE グラフ文法は，6 項組 $G = (\Sigma, \Delta, \Gamma, \Omega, P, S)$ で表される．ただし，

(1) Σ はノードラベルのアルファベットである．

(2) $\Delta \subseteq \Sigma$ は終端ノードラベルのアルファベットである．また，$\Sigma - \Delta$ は非終端ノードラベルのアルファベットである．

(3) Γ はエッジラベルのアルファベットである．

(4) $\Omega \subseteq \Gamma$ は終端エッジラベルのアルファベットである．また，$\Gamma - \Omega$ は非終端エッジラベルのアルファベットである．

(5) P はプロダクションの有限集合である．プロダクションは $(X ::= D, C)$ の形式で表現される．ここで，$X \in \Sigma - \Delta$ であり，$D \in GR_{\Sigma,\Gamma}$ は順序グラフである．また，$C \subseteq \Sigma \times \Gamma \times \Gamma \times V_D \times \{\text{in}, \text{out}\}$ は結合関係である．C の要素 $(\sigma, \beta, \gamma, x, d)$ $(\sigma \in \Sigma, \beta, \gamma \in \Gamma, x \in V_D, d \in \{\text{in}, \text{out}\})$ を結合命令という．

(6) $S \in \Sigma - \Delta$ はスタートラベルである．スタートグラフ H_s は，S でラベル付けされた唯一つのノードからなる順序グラフである． ■

2 つのプロダクション $(X_1 ::= D_1, C_1)$ と $(X_2 ::= D_2, C_2)$ は，$X_1 = X_2$ かつ D_1 から D_2 への同形写像 f が存在して $C_2 = \{(\sigma, \beta, \gamma, f(x), d) \mid (\sigma, \beta, \gamma, x, d) \in C_1\}$ が成り立つとき，同形であるという．$p \in P$ に同形なプロダクションの集合を $copy(p)$ で表す．$copy(p)$ の要素を G の p に対するプロダクションコピー，あるいは単にプロダクションコピーという．$copy(P) = \cup_{p \in P} copy(p)$ とする．

この edNCE グラフ文法 $G = (\Sigma, \Delta, \Gamma, \Omega, P, S)$ に基づくグラフの書き換えは次のように行われる．$H \in GR_{\Sigma,\Gamma}$ を書き換えられる前の順序グラフ (ホストグラフ) とする．また，H の順序のなかで最初の非終端ノードを v とし，ノード v のラベルが X であるとする．そして，$p' = (X ::= D', C')$ を G のあるプロダクション p に対するプロダクションコピーとする．このとき，以下の手順で新しいグラフ $H' \in GR_{\Sigma,\Gamma}$ を構成する．

(1) X とラベル付けされたノード v と，v につながっているエッジをすべて取り除く．そして得られたグラフ (レストグラフ) を H^- とする．

(2) グラフ H^- にドーターグラフ D' を加える．

(3) C' の結合命令にしたがって D' のノードと H^- のノード間のエッジを新しく作り，この結果得られるグラフを H' とする．

(4) H と D' の順序に基づいて，自然な方法で H' に順序を入れる．すなわち，H のノードの順序を $(v_1, \cdots, v_{i-1},\ v_i,\ v_{i+1}, \cdots, v_j)$，$v = v_i$，そして

† 東洋大学工学部情報工学科

D' のノードの順序を $(w_1, \cdots, w_k)$ とする．このとき，H' のノードの順序は $(v_1, \cdots, v_{i-1}, w_1, \cdots, w_k, v_{i+1}, \cdots, v_j)$ である．

グラフ H にプロダクションコピー p' を適用して書き換えグラフ H' を得るとき $H \Rightarrow_{p'} H'$ と書き，H から H' への最左導出ステップと呼ぶ．また，導出ステップの系列を最左導出と呼ぶ．edNCE グラフ文法 $G = (\Sigma, \Delta, \Gamma, \Omega, P, S)$ に対して，最左導出 $H_0 \Rightarrow_{p'_1} H_1 \Rightarrow_{p'_2} \cdots \Rightarrow_{p'_n} H_n, n \geq 0$, は H_0 と $rhs(p'_i), 1 \leq i \leq n$，が互いに素であるとき生成的であるという．今後，生成的な最左導出のみを扱うことにする．上記のような最左導出が $H_0 = H$ と $H_n = H'$ について存在するとき $H \Rightarrow^* H'$ と書く．スタートグラフ H_S に対して $H_S \Rightarrow^* H$ であるグラフ $H \in GR_{\Sigma,\Gamma}$ を G の文形式という．スタートグラフおよびプロダクションの右辺がすべて順序グラフなので，G の文形式も順序グラフとなる．$\mathcal{L}(G) = \{[H] \mid H \in GR_{\Delta,\Omega}$ かつ $H_S \Rightarrow^* H\}$ を G によって生成されるグラフ言語という．ここで，$[H]$ はグラフ H をノード間の順序を無視して順序のないグラフとみなし，この H に同形なすべてのグラフの集合（アブストラクトグラフ）を表すものとする．

定義 1 の edNCE グラフ文法では，プロダクションの右辺グラフのノードが順序付けられているので文脈自由文法と同様な方法で導出木の概念を定義することができる [3].

定義 2 $G = (\Sigma, \Delta, \Gamma, \Omega, P, S)$ を edNCE グラフ文法とする．G の（c-ラベル付き）導出木とは，その頂点がプロダクションコピーでラベル付けられた根付き順序木 t で次の条件 (1) と (2) を満たすものである．

(1) t の頂点をラベル付けするすべてのプロダクションコピーの右辺グラフは互いに素である，

(2) t の頂点 v がラベル $p = (X ::= D, C)$ を持てば，v の子頂点 $w_1, w_2, \cdots, w_n$ は D の非終端ノードであり，t での順序は D での順序と同じである； さらに，各々の子頂点 w_i に対して t での w のラベルの左辺は D の中の w のラベルに等しい． ■

本論文で扱う属性グラフ文法は，edNCE グラフ文法にそのグラフ構文規則に関連する情報を形式的に計算するメカニズムを付加した拡張文法であり，文献 [4] を参考にして定義している．

定義 3 edNCE 属性グラフ文法は 3 項組 $AGG = (G_u, A, F)$ である．

(1) $G_u = (\Sigma, \Delta, \Gamma, \Omega, P, S)$ は edNCE グラフ文法であり，AGG の基底文法という．また，グラフ $D \in GR_{\Sigma,\Gamma}$ について，$Lab_N(D) = \{\lambda_D(v) \in \Sigma - \Delta \mid v \in V_D\}$，$Lab_T(D) = \{\lambda_D(v) \in \Delta \mid v \in V_D\}$ とする．

(2) G_u の各ノードラベル $X \in \Sigma$ に対して，互いに素な 2 つの有限集合，継承属性の集合 $\mathcal{I}(X)$ と合成属性の集合 $\mathcal{S}(X)$ が付随している．そして X の属性全体の集合を $\mathcal{A}(X) = \mathcal{I}(X) \cup \mathcal{S}(X)$ で表す．$\mathcal{A} = \cup_{X \in \Sigma} \mathcal{A}(X)$ を AGG の属性集合という．$\mathcal{I}(S) = \phi$ と仮定する．ノードラベル X の属性 a を $a(X)$ で表す．

(3) P の各プロダクション $p = (X ::= D, C)$ に対し，$\mathcal{S}(X) \cup \bigcup_{Y \in Lab_N(D)} \mathcal{I}(Y) \cup \bigcup_{Y \in Lab_T(D)} \mathcal{A}(Y)$ の属性のみをすべて定義する意味規則の集合 F_p が付随している．集合 $F = \cup_{p \in P} F_p$ を AGG の意味規則集合という． ■

3. 属性評価アルゴリズム

定義 2 の導出木上の頂点の辿り方に依存して，文献 [1] で紹介された属性文法に対する純粋多重戦略あるいは単純多重戦略の種々の属性評価アルゴリズムと同様のアルゴリズムを属性グラフ文法に対しても定義できる．図 1 に属性グラフ文法に対する単純 1 パス属性評価アルゴリズムを示す．属性グラフ文法は単純 1 パス属性評価アルゴリズムですべての属性値を求めることができるとき，単純 1 パスであるという．文献 [2] で報告した Logichart 属性グラフ文法は単純 1 パス属性グラフ文法である．

Procedure DEVAL(v) ; 導出木の頂点 v ;
{v のラベルを $p = (X ::= D, C)$ とし，D の中のノードをその順序に従って $w_1, w_2, \cdots, w_n$ とする．}
begin
  if $X \neq S$ **then** $\mathcal{I}(X)$ のすべての属性値を求める ;
  for $i := 1$ **to** n **do**
    begin
      if w_i が非終端ノード **then** DEVAL(w_i) ;
      else $\mathcal{A}(\lambda_D(w_i))$ のすべての属性値を求める ;
    end
  $\mathcal{S}(X)$ のすべての属性値を求める ;
end

図 1: 単純 1 パス属性評価アルゴリズム

4. まとめ

順序グラフを生成する edNCE グラフ文法を基底文法とする属性グラフ文法を定義し，その導出木上での属性評価アルゴリズムについて議論した．本稿の属性グラフ文法を用いると，属性評価アルゴリズムの議論を見通しよく進めることができ，また手続き型プログラムによる属性評価器の実現にも役立つものと思われる．

参考文献

[1] 情報処理学会（監修），西野哲朗，片山卓也，佐々政孝（編）：属性文法入門，共立出版 (1996)

[2] 安達由洋，“Prolog プログラム図の属性グラフ文法に基づく定式化”, 情報処理学会第 33 回数理モデルと問題解決研究会, (2001).

[3] J. Engelfriet and G. Rozenberg, “Node Replacement Graph Grammar”, in *Handbook of Graph Grammars and Computing by Graph Transformation* (Rozenberg, G., eds), World Scientific, 1–94 (1997).

[4] Nishino T.:, Attribute graph grammars with applications to Hichart editors, *Adv. Software Sci. & Tech.*, 1, pp.426–433 (1989).