Atomic characterizations of uniform multi-pass attribute grammars

By E. GOMBÁS and M. BARTHA

1. Introduction

Several reasonable classes of attribute grammars can be defined based on the concept of computation sequence [1]. A computation sequence for a derivation tree is intended to describe a systematic evaluation of all the attributes of the tree without violating their dependencies. The attributes are evaluated during a walk through the tree. This walk starts at the root, and once it arrives at (enters) a node, it must return to that node later for exiting it. Between any successive entering and exiting a node — this period is called a visit to the node — the walk can make several visits to the sons of the node. If in any derivation tree, the number of visits to a node required to evaluate all the attributes of it, and the set of attributes of the node evaluated in each of these visits can both be determined in a top-down manner, i.e. independently of the subtree below that node, then the grammar is called uniform [5]. It was proved in [5] that an attribute grammar (*AG*) is uniform iff it is absolutely noncircular (anc). The latter property is investigated e.g. in [3], where an efficient evaluator is given for these grammars. As the anc property can be decided in polynomial time, the class of uniform *AG* is practically more interesting than the class of simple multi-visit *AG* introduced in [1]. (Recall from [1] that the problem deciding whether an *AG* is simple multi-visit is *NP*-complete.) However, if we ask whether an *AG* is uniform *m*-visit for a fixed $m \in \mathrm{N}$, then the answer cannot be given in polynomial time, generally. Thus, to answer this question in polynomial time we have to restrict ourselves to a smaller class of *AG*. In this paper we investigate the class of uniform multi-pass *AG*. A pass to a node is a visit such that during it each son of the node is visited exactly once in a left-to-right order. We present a natural characterization of this class and give an algorithm that provides the minimal number *m* for which an *AG* is uniform *m*-pass in polynomial time.

2. Definitions and basic concepts

An attribute grammar [4] $\mathcal{G}$ consists of the following objects.

(i) A reduced context-free grammar $G=(T, N, P, Z)$.

(ii) A finite nonempty set A such that $A=A_s\cup A_i$ and $A_s\cap A_i=\emptyset$. The elements of A_s and A_i are called synthesized (s-) and inherited (i-) attributes, respectively.

(iii) A function v which assigns each nonterminal $F\in N$ a nonvoid subset of A. We assume that the start symbol Z has only s-attributes and it does not occur on the right-hand side of any production. $S(F)$ and $I(F)$ will denote $v(F)\cap A_s$ and $v(F)\cap A_i$, respectively, and an occurrence of an attribute $a\in v(F)$ will often be referenced as $a(F)$.

(iv) A set $V(a)$ of possible values for each attribute a.

(v) A set r_p of semantic rules associated with each production $p\in P$. If p: $F_0\to w_0F_1\dots F_kw_k$ $(F_i\in N,\ w_i\in T^*)$, then a rule of r_p is a formal equation:

$$a_0(F_{i_0}) = f(a_1(F_{i_1}), \dots, a_m(F_{i_m})),$$

where $0\leqq i_j\leqq k$ $(0\leqq j\leqq m)$, $a_j\in v(F_{i_j})$ and f: $V(a_1)\times\dots\times V(a_m)\to V(a_0)$ is a (computable) function. This equation is interpreted by saying that $a_0(F_{i_0})$ depends on $a_1(F_{i_1}), \dots, a_m(F_{i_m})$ in p by f. We assume that $\mathcal{G}$ is in Bochmann normal form, i.e. r_p defines all and only the occurrences of attributes $S(F_0)\cup(\cup(I(F_j)|\,j\in[k]))$ using as arguments only of the occurrences of $I(F_0)\cup(\cup(S(F_j)|\,j\in[k]))$. ($[k]$ denotes the set $\{1, 2, \dots, k\}$.)

D_F will denote the set of derivation trees with root labelled by F. Trees of D_Z are called complete derivation trees. If t is a tree representing a derivation $F\overset{*}{\underset{G}{\Rightarrow}}\alpha$, where α is not necessarily a terminal string, then t is called a cut; in notation, $t\in C_F$. Clearly we have $D_F\subseteqq C_F$ for all $F\in N$. By a node of t we always mean a nonterminal node, and if there is no danger of confusion, we identify the node with its label. U_t, $U_t(F)$ and $rt(t)$ will denote the set of all nodes of t, the set of all F-labelled nodes of t and the root of t, respectively.

The semantic rules are used to assign meanings to derivation trees of G in the following way. Let t be a complete derivation tree, $u\in U_t$, and assume that p: $F_0\to w_0F_1\dots F_kw_k$ is the production applied at u. For each $a_0\in S(F_0)$, the function f occurring in the rule $a_0(F_0)=f(a_1(F_{i_1}), \dots, a_m(F_{i_m}))$ can be used to determine the value of a_0 at u when the values of all the neighbouring attributes $a_1(F_{i_1}), \dots, a_m(F_{i_m})$ have been determined. Similarly, the rule with left-hand side $b(F_j)$ $(j\in[k], b\in I(F_j))$ can be used to determine the value of attribute b at the j-th son of u. If it is possible to determine the values of all the attributes at any node of t in the above way, then the meaning of t is the set $\{(u, \{v_a(u)|a\in v(u)\})|u\in U_t$ and $v_a(u)$ is the value of attribute a at $u\}$.

If all the complete derivation trees have a meaning, then $\mathcal{G}$ is called well-defined or noncircular.

The dependency graph for the production p: $F_0\to w_0F_1\dots F_kw_k$ (denoted by $dp(p)$) has as nodes the disjoint union of $v(F_i)$ $0\leqq i\leqq k$, and there is an arc from $a_1(F_{i_1})$ to $a_2(F_{i_2})$ iff $a_2(F_{i_2})$ depends on $a_1(F_{i_1})$ in p. A graph with nodes $v(F)$ $(F\in N)$ and some arcs is called a dependency graph (d-graph) for F. For p: $F_0\to w_0F_1\dots F_kw_k$ and d-graphs $\gamma_1, \dots, \gamma_k$ for $F_1, \dots, F_k$ define the substitution $dp(p)(\gamma_1, \ldots, \gamma_k)$ of $\gamma_1, \ldots, \gamma_k$ into $dp(p)$ by adding all the arcs of γ_i $i \in [k]$ to $dp(p)$, i.e. fitting γ_i on $dp(p)|v(F_i)$. This substitution induces a d-graph for F_0 by restricting the transitive closure of $dp(p)(\gamma_1, \ldots, \gamma_k)$ to $v(F_0)$. Now, the induced dependency graph for symbol $F \in N$ $(ids(F))$ is defined as the least d-graph for F such that for any production p: $F_0 \to w_0 F_1 \ldots F_k w_k$, the d-graph for F_0 induced by the substitution $dp(p)(ids(F_1), \ldots, ids(F_k))$ is a subgraph of $ids(F_0)$. The induced dependency graph for production p is $idp(p) = dp(p)(ids(F_1), \ldots, ids(F_k))$. We write $a_1(F_{i_1}) <_p a_2(F_{i_2})$ if there is a nonempty path in $idp(p)$ from $a_2(F_{i_2})$ to $a_1(F_{i_1})$. Similarly, $a <_F b$ denotes that there is an arc in $ids(F)$ from b to a.

The induced dependency graph for a cut t $(idt(t))$ is obtained by pasting together the idp's of all the productions t consists of. Let u_1 and u_2 be two nodes of t, $a_1 \in v(u_1)$, $a_2 \in v(u_2)$. As above, $a_1(u_1) <_t a_2(u_2)$ denotes that there is a path in $idt(t)$ from $a_2(u_2)$ to $a_1(u_1)$. We write $a_1(u_1) <_t^R a_2(u_2)$ if this path contains an R-arc. Recall from [2] that an R-arc leads to an i-attribute of a node from an s-attribute of itself or one of its right neighbours. G is called absolutely noncircular (anc) if $<_t$ is a strict partial order for every cut t.

The following AG will be used as an example throughout the paper. G has five nonterminals with attributes:

$$v(Z) = \{a_0\}, \quad v(A) = \{a_0, a_1, a_2, b_1, b_2\}$$
$$v(B) = \{a_1, a_2, b_1, b_2\}, \quad v(C) = \{a_1, b_1\}, \quad v(D) = \{a_1, a_2, b_1, b_2\}.$$

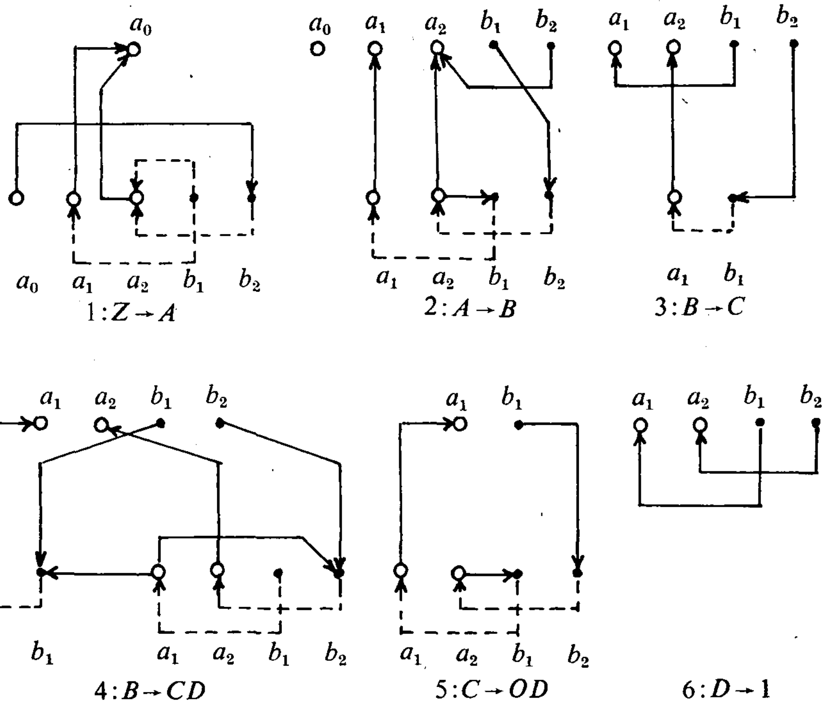

Fig. 1.

$A_s=\{a_0, a_1, a_2\}$, $A_i=\{b_1, b_2\}$. The productions with the corresponding *dp*-graphs are listed in Fig. 1. Dotted lines denote *idp*-arcs, 0 and 1 are terminal symbols.

A partial computation sequence for a derivation tree $t\in D_F$ is a sequence h of so called basic actions ([1]), where each basic action is either the evaluation of some *i*-attributes of a node, called entering the node, or the evaluation of some *s*-attributes of a node, called exiting the node. Thus, a basic action can be represented by a basic action symbol (ba-symbol) $i(u, B)$ or $s(u, A)$, where $u\in U_t$, $A\subseteqq S(u)$ and $B\subseteqq I(u)$. The order of evaluation is systematic and it cannot violate the dependencies of the attributes. By this we mean that h must obey the following restrictions.

1. The first and the last ba-symbol of h is $i(rt(t), B)$ and $s(rt(t), A)$, respectively, where $A\subseteqq S(F)$ and $B\subseteqq I(F)$.

2. For any two contiguous ba-symbols $\dots x_1(u_1, A_1)x_2(u_2, A_2)\dots$ in h, one of the following conditions holds.

(i) u_2 is a son of u_1 and $x_1=x_2=i$,
(ii) u_2 is the father of u_1 and $x_1=x_2=s$,
(iii) u_2 is a brother of u_1 and $x_1=s$, $x_2=i$,
(iv) $u_2=u_1$ and $x_1\neq x_2$.

3. For every $u\in U_t$, if

$$i(u, B_1)s(u, A_1)\dots i(u, B_m)s(u, A_m)$$

is the sequence of all the ba-symbols for u occurring in h (from left to right), then $(B_1\cup A_1, \dots, B_m\cup A_m)$ is an ordered subpartition of $v(u)$. This subpartition will be denoted by $E_h(u)=(E_1(u), \dots, E_m(u))_h$ or $E(u)$ if h is understood. By an ordered subpartition of a set C we mean a sequence of sets $(C_1, \dots, C_m)$ such that

$$\bigcup_{i=1}^{m} C_i \subseteqq C \quad\text{and}\quad C_i\cap C_j = \emptyset \quad\text{if}\quad 1 \leqq i \neq j \leqq m.$$

4. For any production p, consider an arbitrary occurrence of p in t, and let u_1, u_2 and a_1, a_2 such nodes and attributes of this occurrence that $a_2(u_2)$ depends on $a_1(u_1)$ in p. If $a_2(u_2)$ occurs in h (i.e. there exists a ba-symbol $x(u_2, A)$ in h with $a_2\in A$), then so does $a_1(u_1)$, and the occurrence of $a_1(u_1)$ precedes that of $a_2(u_2)$.

If $E_h(u)$ is a complete partition (i.e. $\cup E_h(u)=v(u)$) for all $u\in U_t$, then h is a (total) computation sequence. If h satisfies 1, 2 and 3 only, then it is called a walk. A walk h is a pass if:

— each node is entered and exited (i.e. visited) exactly once;

— during the visit to a node, all its sons are visited in a left to right order.

h is an m-pass walk ($m\in\mathbb{N}$) if $h=h_1\dots h_m$ and h_i is a pass for all $i\in[m]$.

A ba-symbol $x(u, A)$ is empty if $A=\emptyset$. A pass is called empty if all the ba-symbols occurring in it are empty.

Example 2.1. Let t be the complete derivation tree of our example *AG* illustrated on the left-hand side of Fig. 2. The graph on the right-hand side indicates the dependencies between the attributes of t. A 4-pass computation sequence for t is the following.

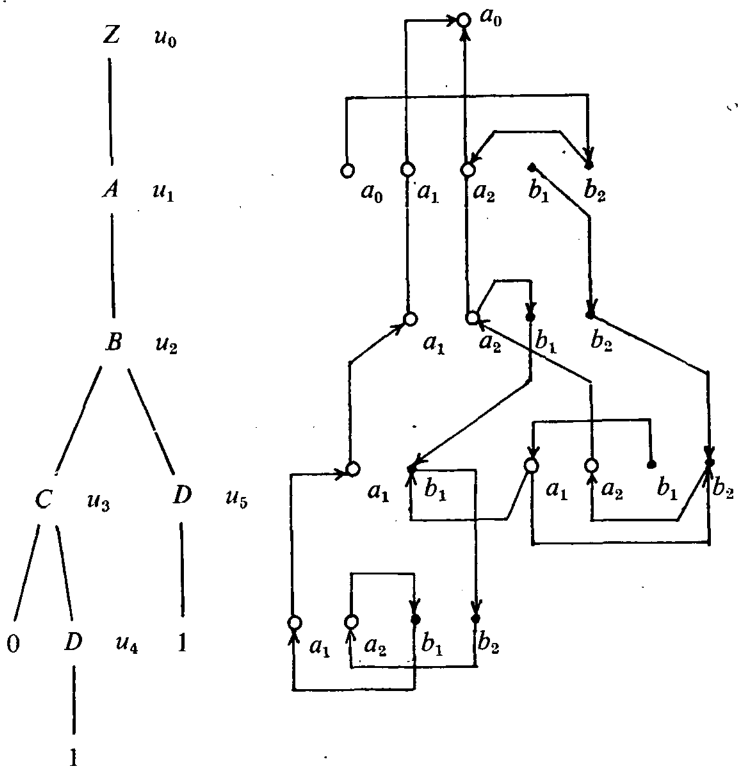

Fig. 2.

$h = h_1 h_2 h_3 h_4$, where

$$h_1 = i(u_0, \emptyset)\, i(u_1, \{b_1\})\, i(u_2, \{b_2\})\, i(u_3, \emptyset)\, i(u_4, \emptyset)\, s(u_4, \emptyset)\, s(u_3, \emptyset)$$
$$i(u_5, \{b_1\})\, s(u_5, \{a_1\})\, s(u_2, \emptyset)\, s(u_1, \{a_0\})\, s(u_0, \emptyset),$$
$$h_2 = i(u_0, \emptyset)\, i(u_1, \emptyset)\, i(u_2, \emptyset)\, i(u_3, \emptyset)\, i(u_4, \emptyset)\, s(u_4, \emptyset)\, s(u_3, \emptyset)$$
$$i(u_5, \{b_2\})\, s(u_5, \{a_2\})\, s(u_2, \{a_2\})\, s(u_1, \emptyset)\, s(u_0, \emptyset),$$
$$h_3 = i(u_0, \emptyset)\, i(u_1, \emptyset)\, i(u_2, \{b_1\})\, i(u_3, \{b_1\})\, i(u_4, \{b_2\})\, s(u_4, \{a_2\})$$
$$s(u_3, \emptyset)\, i(u_5, \emptyset)\, s(u_5, \emptyset)\, s(u_2, \emptyset)\, s(u_1, \emptyset)\, s(u_0, \emptyset),$$
$$h_4 = i(u_0, \emptyset)\, i(u_1, \{b_2\})\, i(u_2, \emptyset)\, i(u_3, \emptyset)\, i(u_4, \{b_1\})\, s(u_4, \{a_1\})$$
$$s(u_3, \{a_1\})\, i(u_5, \emptyset)\, s(u_5, \emptyset)\, s(u_2, \{a_1\})\, s(u_1, \{a_1, a_2\})\, s(u_0, \{a_0\}).$$

Let $p\colon F_0 \to w_0 F_1 \dots F_k w_k \in P$ and $t_j \in D_{F_j}$ for each $j \in [k]$. Let

$$\pi = (A_1 \cup B_1, \dots, A_m \cup B_m)$$

be an ordered subpartition of $v(F_0)$ with $A_i \subseteq S(F_0)$, $B_i \subseteq I(F_0)$ $(i \in [m])$, and let $h_j = h_1^{(j)} \dots h_{m_j}^{(j)}$ be an m_j-pass walk $(m_j \leqq m)$ for each t_j. $\pi \circ (h_1, \dots, h_k)$ will denote the m-pass walk $i(u_0, B_1)h_1^{(1)}\dots h_1^{(k)}s(u_0, A_1)\dots i(u_0, B_m)h_m^{(1)}\dots h_m^{(k)}s(u_0, A_m)$ for $F_0(w_0t_1\dots t_kw_k)\in D_{F_0}$, where u_0 is the root and $h_i^{(j)}$ is empty if $i>m_j$.

The proof of the following two easy lemmas are left to the reader.

Lemma 2.1. Let t be a complete derivation tree and $u\in U_t(F)$. Denote $t'\in D_F$ the subtree of t below u, and let $h=h_1\dots h_m$ and $l'=l_1'\dots l_m'$ be m-pass computation sequences for t and t', respectively. Each h_i ($i\in[m]$) can be written in the form $h_i=\alpha_i l_i\beta_i$, where $l=l_1\dots l_m$ is an m-pass computation sequence for t'. If $E_h(u)=$ $=E_{l'}(rt(t'))$, then $h'=h_1'\dots h_m'$ — where $h_i'=\alpha_i l_i'\beta_i$ — is also an m-pass computation sequence for t.

Let $\pi_1=(A_1, \dots, A_n)$ and $\pi_2=(B_1, \dots, B_m)$ be two ordered subpartitions of a set D. Construct the ordered subpartition

$$\text{merge}\,(\pi_1, \pi_2)=(C_1, \dots, C_{\max(n,m)})$$

as follows:

(i) $C_1 = A_1\cup B_1$,

(ii) $C_{i+1} = A_{i+1}\cup B_{i+1}\setminus \bigcup_{j=1}^{i} C_j$ for each $1 \leqq i < \max(n, m)$.

Lemma 2.2. Let h_i ($i=1, 2,$) be m_i-pass partial computation sequences for $t\in D_F$, $m=\max(m_1, m_2)$. Construct an m-pass walk merge (h_1, h_2) as follows. For each $u\in U_t$ let

$$E_{\text{merge}(h_1, h_2)}(u)=\text{merge}\,(E_{h_1}(u),\ E_{h_2}(u)).$$

Then merge (h_1, h_2) is a partial computation sequence.

Note that an m-pass walk for t is completely determined by the set $\{E(u)|u\in U_t\}$. The operation merge can be extended to any number of walks by:

$$\text{merge}\,(h_1, \dots, h_{n+1}) = \text{merge}\,(\text{merge}\,(h_1, \dots, h_n), h_{n+1}).$$

3. The atomic characterization and decidability results

An atomic pass-description for $\mathcal{G}$ is a five-tuple $\mathcal{D}=(\mathcal{A}, \mu, \chi, \varrho, \pi_0)$, where

(i) $\mathcal{A}$ is a finite nonempty set, called the set of atoms.

(ii) μ: $N\to P(\mathcal{A})$ assigns each nonterminal a subset of $\mathcal{A}$.

(iii) $\chi=\{\chi_F|F\in N\}$, where χ_F: $\mu(F)\to\mathcal{P}(v(F))$ is a function such that

$$\bigcup_{c\in\mu(F)} \chi_F(c) = v(F) \quad\text{and}\quad \chi_F(c_1)\cap\chi_F(c_2) = \emptyset \quad\text{if}\quad c_1 \neq c_2.$$

(iv) $\varrho=\{\varrho_p|p\in P\}$ is a family of mappings such that if p: $F_0\to w_0F_1\dots F_kw_k$, then

$$\varrho_p\colon \mu(F_0) \to \mathcal{P}(\bigcup_{j=1}^{k} (\{j\}\times\mu(F_j))).$$

As in the case of attributes, $c(F)$ indicates an occurrence of an atom $c\in\mu(F)$, and we prefer the notation $c(F_j)\in\varrho_p(c_0)$ to $(j, c)\in\varrho_p(c_0)$ if no confusion arises.

(v) π_0 is an ordered partition of $\mu(Z)$.

Let π be an ordered subpartition of $\mu(F)$. We say that an m-pass partial computation sequence h for $t\in D_F$ respects $\mathcal{D}/\pi$ if it satisfies the following conditions.

1. For every node $u\in U_t(Y)$ $(Y\in N)$ there exists an ordered subpartition $\pi_h(u)= =(\mathcal{A}_1, \dots, \mathcal{A}_m)$ of $\mu(Y)$ such that

$$(E_i(u))_h = \bigcup(\chi_Y(c)|c\in\mathcal{A}_i) \quad \text{for all} \quad i\in[m].$$

2. For any production p: $F_0\to w_0 F_1 \dots F_k w_k$ consider an arbitrary occurrence of p in t. Let $u_0, u_1, \dots, u_k$ be the corresponding nodes of t, respectively. If $\pi_h(u_l)=(\mathcal{A}_1^{(l)}, \dots, \mathcal{A}_m^{(l)})$ $(0\leqq l\leqq k)$, then
a) for every $i\in[m]$ and $j\in[k]$

$$\{c(F_j)|c\in\mathcal{A}_i^{(j)}\}\subseteqq \varrho_p(\mathcal{A}_i^{(0)})\cap\mu(F_j);$$

b) if $c(F_j)\in\varrho_p(c_1)\cap\varrho_p(c_2)$ for some $c_1\neq c_2$ such that $c\in\mathcal{A}_i^{(j)}$ and $c_l\in\mathcal{A}_{i_l}^{(0)}$ $(l=1, 2)$, then $i\leqq\min(i_1, i_2)$.
3. $\pi_h(rt(t))=\pi$.

Definition 3.1. $\mathcal{G}$ is atomic m-pass with respect to $\mathcal{D}$ if every complete derivation tree has an m-pass computation sequence respecting $\mathcal{D}/\pi_0$. $\mathcal{G}$ is atomic multi-pass (amp) if there exist m and $\mathcal{D}$ such that $\mathcal{G}$ is atomic m-pass with respect to $\mathcal{D}$.

Example 3.1. In our example AG let $\mathcal{D}=(\mathcal{A}, \mu, \chi, \varrho, \pi_0)$, where

$$\mathcal{A} = \{d_1, d_2\}\cup\{(c_1, i)|i\in[3]\}\cup\{(c_2, i)|i\in[2]\}\cup$$
$$\cup\{(c_0, i), (c, i)|i\in[4]\};$$

$$\mu(Z) = \{(c_0, i)|i\in[4]\},\ \mu(A) = \{(c_0, 1), d_1\}\cup\{(c, i)|i\in[4]\},$$
$$\mu(B) = \{(c_1, i)|i\in[3]\}\cup\{(c_2, i)|i\in[2]\}\cup\{d_1, d_2\},$$
$$\mu(C) = \{(c_1, i)|i\in[2]\}\cup\{d_1\},\ \mu(D) = \{(c_1, 1), (c_2, 1)\};$$

$$\chi_Z(c_0, i) = \begin{cases}\emptyset & \text{if } i\leqq 3\\ \{a_0\} & \text{if } i=4\end{cases} \quad \chi_A(c, i) = \begin{cases}\emptyset & \text{if } i\leqq 3\\ \{a_1, a_2, b_2\} & \text{if } i=4\end{cases}$$

$$\chi_A(c_0, 1) = \{a_0\} \quad \chi_A(d_1) = \chi_B(d_1) = \{b_1\} \quad \chi_B(d_2) = \{b_2\}$$

$$\chi_B(c_1, i) = \begin{cases}\emptyset & \text{if } i\leqq 2\\ \{a_1\} & \text{if } i=3\end{cases} \quad \chi_B(c_2, i) = \begin{cases}\emptyset & \text{if } i=1\\ \{a_2\} & \text{if } i=2\end{cases}$$

$$\chi_C(c_1, i) = \begin{cases}\emptyset & \text{if } i=1\\ \{a_1\} & \text{if } i=2\end{cases} \quad \chi_C(d_1) = \{b_1\}$$

$$\chi_D(c_1, 1) = \{a_1, b_1\} \quad \chi_D(c_2, 1) = \{a_2, b_2\};$$

$$\varrho_1(c_0, 1) = \{(c_0, 1)(A), d_1(A), (c, 1)(A)\}, \quad \varrho_1(c_0, 2) = \{(c, 2)(A)\},$$
$$\varrho_1(c_0, 3) = \{(c, 3)(A)\}, \quad \varrho_1(c_0, 4) = \{(c, 4)(A)\},$$
$$\varrho_2(c_0, 1) = \varrho_2(d_1) = \emptyset, \quad \varrho_2(c, 1) = \{d_2(B), (c_2, 1)(B)\},$$
$$\varrho_2(c, 2) = \{(c_2, 2)(B), (c_1, 1)(B)\},$$
$$\varrho_2(c, 3) = \{(c_1, 2)(B), d_1(B)\}, \quad \varrho_2(c, 4) = \{(c_1, 3)(B)\},$$

$$\begin{aligned}
&\varrho_3(c_1, 1) = \varrho_3(c_1, 2) = \varrho_3(c_1, 3) = \emptyset,\\
&\varrho_3(c_2, 1) = \{(c_1, 1)(C)\}, \quad \varrho_3(c_2, 2) = \{(c_1, 2)(C)\},\\
&\varrho_4(d_1) \quad = \varrho_4(d_2) = \emptyset, \quad \varrho_4(c_1, 1) = \{(c_1, 1)(D)\},\\
&\varrho_4(c_1, 2) = \{d_1(C),\ (c_1, 1)(C)\}, \quad \varrho_4(c_1, 3) = \{(c_1, 2)(C)\},\\
&\varrho_4(c_2, 1) = \{(c_1, 1)(D)\}, \ \varrho_4(c_2, 2) = \{(c_2, 1)(D)\},\\
&\varrho_5(d_1) \quad = \emptyset, \quad \varrho_5(c_1, 1) = \{(c_2, 1)(D)\}, \quad \varrho_5(c_1, 2) = \{(c_1, 1)(D)\},\\
&\varrho_6(c_1, 1) = \varrho_6(c_2, 1) = \emptyset;\\
&\qquad\qquad \pi_0 = (\{(c_0, 1)\},\ \{(c_0, 2)\},\ \{(c_0, 3)\},\ \{(c_0, 4)\}).
\end{aligned}$$

It is not difficult to check that the example AG is atomic 4-pass with respect to $\mathcal{D}$. For example, the computation sequence h of Example 2.1 respects $\mathcal{D}/\pi_0$.

Readers who are less familiar with attribute evaluation procedures are advised to read section 4 before going further.

Lemma 3.1. Let h_i $(i=1, 2, ..., n)$ be m_i-pass partial computation sequences for $t\in D_F$ respecting $\mathcal{D}/\pi_i$. Then merge $(h_i|i\in[n])$ respects $\mathcal{D}$/merge $(\pi_i|i\in[n])$.

Proof. Obvious.

Definition 3.2. $\mathcal{G}$ is absolutely non-R-recursive (anr) if it is anc and there is no cut t for which the following holds. There is a leaf $u\in U_t$ having the same label F as $u_0=rt(t)$ and an attribute $a\in v(F)$ such that: $a(u_0)<_t^R a(u)$ or $a(u)<_t^R a(u_0)$.

Let $\mathcal{G}$ be anr. We shall prove that it is amp, too, moreover, we give an algorithm that provides the minimal number m for which $\mathcal{G}$ is atomic m-pass. In the first step the algorithm computes a relative difference $r(F, a, b)$ for each triple $F\in N$, $a<_F b$. $\{r(F, a, b)|F\in N, a<_F b\}$ is the system of minimal numbers such that for any p: $F_0\to$ $\to w_0 F_1 \dots F_k w_k$ and idp (p)-path

$$a_0(F_0) <_p a_1(F_{i_1}) <_{F_{i_1}} b_1(F_{i_1}) <_p \dots <_p a_n(F_{i_n}) <_{F_{i_n}} b_n(F_{i_n}) <_p b_0(F_0)$$

containing l R-arcs,

$$r(F_0, a_0, b_0) \geqq l + \sum_{j=1}^{n} r(F_{i_j}, a_j, b_j).$$

$r(F, a, b)$ expresses that, during an amp evaluation of derivation tree, at any F-labelled node, if $b(F)$ can be evaluated in the i-th pass, then $a(F)$ cannot be evaluated sooner than in the $i+r(F, a, b)$-th pass. This statement will be proved in Lemma 3.6. In the second step a pass-number $q(F, a)$ is computed for each $F\in N$, $a\in S(F)$ such that if $t\in D_F$ is an arbitrary derivation tree, and the values of all the i-attributes at the root are available, then — supposing an amp evaluation — $a(rt(t))$ can be evaluated in the $q(F, a)$-th pass, but not sooner in general. If all these numbers are finite, then the required atomic description can be constructed easily.

Algorithm 3.1. *Input:* An anr AG $\mathcal{G}$.
Output: pass-numbers $q(F, a)$ for each $F\in N$, $a\in S(F)$;
inherited pass-numbers $q_p(F_j, a, a_0)$ for each
p: $F_0 \to w_0 F_1 \dots F_k w_k$, $j\in[k]$, $a\in v(F_j)$, $a_0\in S(F_0)$ such that $a_0(F_0)<_p a(F_j)$.

```
    (1) Compute for each $F\in N$, $a<_F b$ the number $r(F,a,b)$ as follows.
        (a) Let $r_0(F,a,b)=0$ and set $i=0$.
        (b) For each $p\colon F_0\to w_0F_1\dots F_kw_k\in P$
            begin for each $b_0\in I(F_0)$
                begin let $H_0=\{b_0(F_0)\}$, $x(b_0(F_0))=0$ and set $n=0$.
            (i) Let
        $H_{n+1}=H_n\cup\{a(F_j)\mid a\in S(F_j),\ j\in[k],\ a(F_j)<_p b_0(F_0)$ and for any $a'(F_m)\in\bigcup$
$\bigcup(S(F_l)\mid l\in[k])\cup H_0$, $a(F_j)<_p a'(F_m)<_p b_0(F_0)$ implies that $a'(F_m)\in H_n\}$.
        If $H_{n+1}=H_n$, then goto step (ii), else for each $a(F_j)\in H_{n+1}\setminus H_n$ let

    $x(a(F_j))=\max(\max(r_i(F_j,a,b)+x(a'(F_m))\mid a(F_j)<_{F_j}b(F_j)<_p a'(F_m)\in H_n)$,
            $\max(r_i(F_j,a,b)+x(a'(F_m))+1\mid a(F_j)<_{F_j}b(F_j)<^R_p a'(F_m)\in H_n))$

(note that $\max(\emptyset)=0$ by definition);
set $n=n+1$ and repeat step (i).
            (ii) For each $a_0<_{F_0}b_0$ let

            $r^{(p)}_{i+1}(F_0,a_0,b_0)=\max(x(a(F_j))\mid a(F_j)\in H_n,\ a_0(F_0)<_p a(F_j))$;

                end
            end;

For each $F\in N$, $a<_F b$ let

                $r_{i+1}(F,a,b)=\max(r^{(p)}_{i+1}(F,a,b)\mid p\in P)$.

If $r_{i+1}(F,a,b)=r_i(F,a,b)$ for all $F\in N$, $a<_F b$, then let $r(F,a,b)=r_i(F,a,b)$,
else set $i=i+1$ and repeat step (b).
    (2) Compute for each $F\in N$, $a\in S(F)$ the number $q(F,a)$ as follows.
        (c) Let $q_0(F,a)=1$ and set $i=0$.
        (d) For each $p\colon F_0\to w_0F_1\dots F_kw_k$
            begin let $M_0=\emptyset$ and set $n=0$.
            (iii) Let

    $M_{n+1}=M_n\cup\{a(F_j)\mid j\in[k],\ a\in S(F_j)$ and for any $a'(F_m)$ $(m\in[k],\ a'\in S(F_m))$,
                $a(F_j)<_p a'(F_m)$ implies that $a'(F_m)\in M_n\}$.

        If $M_{n+1}=M_n$, then goto step (iv), else for each $a(F_j)\in M_{n+1}\setminus M_n$ let
            $y(a(F_j))=\max(q_i(F_j,a),\ \max(\max(r(F_j,a,b)+y(a'(F_m))\mid$
                    $a(F_j)<_{F_j}b(F_j)<_p a'(F_m)\in M_n)$,
        $\max(r(F_j,a,b)+y(a'(F_m))+1\mid a(F_j)<_{F_j}b(F_j)<^R_p a'(F_m)\in M_n)))$;

set $n=n+1$ and repeat step (iii).
            (iv) For each $a_0=S(F_0)$ let

            $q^{(p)}_{i+1}(F_0,a_0)=\max(y(a(F_j))\mid a(F_j)\in M_n,\ a_0(F_0)<_p a(F_j))$

            end;
```

For each $F \in N$, $a \in S(F)$ let

$$q_{i+1}(F, a) = \max \left(q_{i+1}^{(p)}(F, a) \mid p \in P \right).$$

If $q_{i+1}(F, a) = q_i(F, a)$ for all $F \in N$, $a \in S(F)$, then let $q(F, a) = q_i(F, a)$, else set $i = i+1$ and repeat step (d).

(3) Compute for each $p\colon F_0 \to w_0 F_1 \dots F_k w_k$, $j \in [k]$, $a \in v(F_j)$, $a_0 \in S(F_0)$ such that $a_0(F_0) <_p a(F_j)$ the number $q_p(F_j, a, a_0)$ as follows.

Let $N_0 = \{a_0(F_0)\}$, $q_p(F_0, a_0, a_0) = q(F_0, a_0)$ and set $n = 0$.

(v) Let

$$N_{n+1} = N_n \cup \{ a(F_j) \mid j \in [k],\ a \in v(F_j),\ a_0(F_0) <_p a(F_j) \text{ and for any } a'(F_m)$$
$$(0 \leqq m \leqq k,\ a' \in v(F_m)),\ a(F_0) \leqq_p a'(F_m) <_p a(F_j) \text{ implies that } a'(F_m) \in N_n \}.$$

If $N_{n+1} = N_n$ then halt, else for each $a(F_j) \in N_{n+1} \setminus N_n$;

if $a \in I(F_j)$, then let

$$q_p(F_j, a, a_0) = \min \left(q_p(F_j, a', a_0) - r(F_j, a', a) \mid a' \in S(F_j),\ a'(F_j) \in N_n \right);$$

else (i.e. if $a \in S(F_j)$) let

$$q_p(F_j, a, a_0) = \min \big(\min \left(q_p(F_m, a', a_0) \mid a'(F_m) \in N_n,\ a'(F_m) <_p a(F_j) \right),$$
$$\min \left(q_p(F_m, a', a_0) - 1 \mid a'(F_m) \in N_n,\ a'(F_m) <_p^R a(F_j) \right) \big);$$

set $n = n+1$ and repeat step (v).

Example 3.2. Executing the algorithm on our example AG we get that:

$$\begin{aligned}
&r(D, a_1, b_1) = r(D, a_2, b_2) = 0, \quad r(C, a_1, b_1) = 1,\\
&r(B, a_1, b_1) = r(B, a_2, b_2) = 1, \quad r(A, a_2, b_2) = 0,\\
&r(A, a_2, b_1) = 1, \quad r(A, a_1, b_1) = 3;\\
&q(D, a_1) = q(D, a_2) = 1, \quad q(C, a_1) = q(B, a_2) = 2,\\
&q(B, a_1) = 3,\ q(A, a_2) = 2 (\text{but } q_1(A, a_2, a_0) = 4),\\
&q(A, a_1) = 4, \quad q(A, a_0) = 1, \quad q(Z, a_0) = 4.
\end{aligned}$$

Lemma 3.2. For every $i \geqq 1$, $F \in N$ and $a <_F b$, if $r_i(F, a, b) > 0$, then there exists $t \in D_F$ such that $a(rt(t)) <_t^R b(rt(t))$.

Proof. Trivial, by construction.

Lemma 3.3. If $\mathcal{G}$ is anr, then the numbers $r(F, a, b)$ and $q(F, a)$ computed in steps (1) and (2) are all finite.

Proof. (a): $r(F, a, b)$ is finite.

Let $i_0 = \|\{(F, a, b) \mid F \in N,\ a <_F b\}\| + 1$ ($\|B\|$ denotes the cardinality of set B), and suppose that $r_{i_0+1}(F_0, a_0, b_0) > r_{i_0}(F_0, a_0, b_0) = r_0$ for some $a_0 <_{F_0} b_0$. Then there exists $p\colon F_0 \to w_0 F_1 \dots F_k w_k \in P$ such that $r_{i_0+1}^{(p)}(F_0, a_0, b_0) > r_0$, i.e. for some $a(F_j) \in H_n$ we have

$$a_0(F_0) <_p a(F_j) \leqq_p a'(F_m) <_{F_m} b'(F_m) <_p b_0(F_0),$$

where $x(a(F_j)) > r_0$, and consequently $r_{i_0}(F_m, a', b') > r_{i_0-1}(F_m, a', b')$ for some $m \in [k]$, $a' <_{F_m} b'$. (Note that $i_0 \geqq 2$, else we have nothing to prove.) Let $\alpha_0 =$

$=(F_0, a_0, b_0)\equiv(F^{(0)}, a^{(0)}, b^{(0)})$, $\alpha_1=(F_m, a', b')\equiv(F^{(1)}, a^{(1)}, b^{(1)})$, and construct $\alpha_2=$ $=(F^{(2)}, a^{(2)}, b^{(2)}), \ldots, \alpha_{i_0-1}$ in the same way. It follows that for any $0\leqq i<j\leqq i_0-1$ there exists a cut $t\in C_{F^{(i)}}$ and a leaf u of t labelled by $F^{(j)}$ for which we have:

$$a^{(i)}(rt(t)) <_t a^{(j)}(u) <_{F^{(j)}} b^{(j)}(u) <_t b^{(i)}(rt(t)). \tag{1}$$

Moreover, by Lemma 3.2, if all the cuts t with property (1) are such that neither

$$a^{(i)}(rt(t)) <_t^R a^{(j)}(u) \text{ nor } b^{(j)}(u) <_t^R b^{(i)}(rt(t)), \tag{2}$$

then $r_{i_0-i+1}(\alpha_i)=r_{i_0-j+1}(\alpha_j)$. By the choice of i_0, $\alpha_i=\alpha_j$ for some $0\leqq i<j\leqq i_0-1$. In this case, however, $r_{i_0-i+1}(\alpha_i)=r_{i_0-j+1}(\alpha_j)$ is impossible, thus (2) contradicts the anr property. We conclude that $r_{i_0+1}(F, a, b)=r_{i_0}(F, a, b)$ for all $F\in N$, $a<_F b$, which was to be proved.

(b): $q(F, a)$ is finite.

Let $n_0=\|\{(F, a)|F\in N, a\in S(F)\}\|+1$, and suppose that $q_{n_0+1}(F_0, a_0)>$ $>q_{n_0}(F_0, a_0)=q_0$ for some $a_0\in S(F_0)$. Then there exists p: $F_0\to w_0F_1\ldots F_kw_k\in P$ such that $q_{n_0+1}^{(p)}(F_0, a_0)>q_0$, i.e. for some $a(F_j)\in M_n$ we have $a_0(F_0)<_p a(F_j)$ and $y(a(F_j))>q_0$. Choose this $a(F_j)$ so that

(i) $y(a(F_j))=q_{n_0}(F_j, a)$;

(ii) if $a_0(F_0)<_p a'(F_m)<_p a(F_j)$, then $y(a'(F_m))>q_{n_0}(F_m, a')$.

Clearly, $q_{n_0}(F_j, a)>q_{n_0-1}(F_j, a)$. Let $\beta_0=(F_0, a_0)\equiv(F^{(0)}, a^{(0)})$, $\beta_1=(F_j, a)\equiv$ $\equiv(F^{(1)}, a^{(1)})$, and construct $\beta_2=(F^{(2)}, a^{(2)}), \ldots, \beta_{n_0-1}$ in the same way. It follows that for any $0\leqq k<m\leqq n_0-1$ there exists a cut $t\in C_{F^{(k)}}$ and a leaf u of t labelled by $F^{(m)}$ for which:

(i) $a^{(k)}(rt(t))<_t a^{(m)}(u)$;

(ii) if $a^{(k)}(rt(t))\equiv_t^R a^{(m)}(u)$, then $q_{n_0-k+1}(\beta_k)=q_{n_0-m+1}(\beta_m)$.

By the choise of n_0, $\beta_k=\beta_m$ for some $0\leqq k<m\leqq n_0-1$. As in the part (a) of the proof, $q_{n_0-k+1}(\beta_k)=q_{n_0-m+1}(\beta_m)$ is again impossible, thus (3) contradicts the anr property. Consequently, $q_{n_0+1}(F, a)=q_{n_0}(F, a)$ for all $F\in N$, $a\in S(F)$.

Now we construct $\mathscr{A}$, μ, χ, ϱ and π_0 such that $\mathscr{G}$ is amp with respect to $\mathscr{D}=(\mathscr{A}, \mu, \chi, \varrho, \pi_0)$. Let

$\mathscr{A} = A_i\cup\{(a, i)|a\in A_s,\ 1 \leqq i \leqq \max(q(F, a)|F\in N \text{ such that } a\in S(F))\}$.

Define the mappings α: $\mathscr{A}\to A$ and β: $\mathscr{A}\to\mathbf{N}$ by

$$\alpha(c)=\begin{cases} b & \text{if } c=b\in A_i, \\ a & \text{if } c=(a, i),\ a\in A_s \end{cases} \qquad \beta(c)=\begin{cases} 1 & \text{if } c\in A_i, \\ i & \text{if } c=(a, i),\ a\in A_s. \end{cases}$$

To simplify the formalism let $q(F, b)=1$ for each $b\in I(F)$. For $F\in N$ let

$$\mu(F)=\{c|\alpha(c)\in v(F) \text{ and } \beta(c)\leqq q(F, \alpha(c))\},$$

and if $c\in\mu(F)$, then

$$\chi_F(c)=\begin{cases} \{\alpha(c)\} & \text{if } \beta(c)=q(F, \alpha(c)), \\ \emptyset & \text{otherwise.} \end{cases}$$

For p: $F_0\to w_0F_1\ldots F_kw_k$ and $a_0\in S(F_0)$ consider the inherited pass-numbers $q_p(F_j, a, a_0)$ computed in step (3) of Algorithm 3.1. Extend q_p to triples (F_j, c, a_0) $c\in\mu(F_j)$ by

$$q_p(F_j, c, a_0)=q_p(F_j, \alpha(c), a_0)-(q(F_j, \alpha(c))-\beta(c)).$$

For each $c_0 \in \mu(F_0)$, if $c_0 \in \alpha^{-1}(A_s)$, then let

$$\varrho_p(c_0) = \{c(F_j) | \alpha(c_0)(F_0) <_p \alpha(c)(F_j) \text{ and } \beta(c_0) = q_p(F_j, c, \alpha(c_0))\},$$

else let $\varrho_p(c_0)=\emptyset$. Finally, let $\pi_0=(\mathscr{A}_1, \ldots, \mathscr{A}_m)$, where $m=\max(q, (Z, a)|a \in v(Z))$ and for every $i \in [m]$ $\mathscr{A}_i=\{c \in \mu(Z)|\beta(c)=i\}$.

The reader is advised to construct the atomic pass description of the example AG. Since this is similar to that of Example 3.1, we do not detail it here.

Lemma 3.4. Let $F \in N$, $a \in S(F)$ and $t \in D_F$ be arbitrary, $\pi=(\mathscr{A}_1, \ldots, \mathscr{A}_n)$ an ordered subpartition of $\mu(F)$ such that

(i) $\bigcup_{j=1}^{n} \mathscr{A}_j = \{c \in \mu(F)| a \leqq_F \alpha(c)\}$,

(ii) $(a, q(F, a)-i) \in \mathscr{A}_{n-i}$ for every $0 \leqq i < q(F, a)$,

(iii) if $a <_F b$ and $b \in \mathscr{A}_j$, then $j \leqq n-r(F, a, b)$.

There exists an n-pass partial computation sequence for t respecting $\mathscr{D}/\pi$.

Proof. Induction on the depth of t. For $t=F(w)$ ($F \to w \in P$ for some $w \in T^*$) the statement is obvious. Let $t=F_0(w_0 t_1 \ldots t_k w_k)$ such that the top production of t is p: $F_0 \to w_0 F_1 \ldots F_k w_k$, and let $a_0 \in S(F_0)$. It is enough to prove the statement for the subpartition $\bar{\pi}_0=(\mathscr{A}_1^{(0)}, \ldots, \mathscr{A}_{n_0}^{(0)})$ of $\mu(F_0)$ which satisfies (i)—(iii), moreover, $n_0=q(F_0, a_0)$ and $b_0 \in \mathscr{A}^{(0)}_{n_0-r(F_0, a_0, b_0)}$ if $a_0 <_{F_0} b_0$. For $j \in [k]$ and $a \in S(F_j)$ such that $a_0(F_0) <_p a(F_j)$ let $\pi_j(a)=(\mathscr{A}_1^{(j)}, \ldots, \mathscr{A}_{n_j}^{(j)})$, where $n_j=q_p(F_j, a, a_0)$ and

$$\mathscr{A}_i^{(j)} = \{c(F_j)| a \leqq_{F_j} \alpha(c) \text{ and } q_p(F_j, c, a) = i\}.$$

As $t_j \in D_{F_j}$ and $\pi_j(a)$ satisfy the requirements of the lemma, by the induction hypothesis there exist n_j-pass partial computation sequences $h_j(a)$ for t_j respecting $\mathscr{D}/\pi_j(a)$. Let

$$h_j = \operatorname{merge}(h_j(a)|a \in S(F_j),\ a_0(F_0) <_p a(F_j)).$$

By Lemma 3.1, h_j respect $\mathscr{D}/\pi_j$, where $\pi_j=(\mathscr{B}_1^{(j)}, \ldots, \mathscr{B}_{n_j}^{(j)})$ is an appropriate ordered subpartition of $\mu(F_j)$. Observe that $n_j \leqq n_0$, and by the construction of ϱ we have $\mathscr{B}_i^{(j)}=\varrho_p(a_0, i) \cap \mu(F_j)$. This implies that $(\chi_{F_0}(\mathscr{A}_1^{(0)}), \ldots, \chi_{F_0}(\mathscr{A}_{n_0}^{(0)})) \circ (h_1, \ldots, h_k)$ is an n_0-pass partial computation sequence for t respecting $\mathscr{D}/\pi_0$, which was to be proved.

Proposition 3.1. If $\mathscr{G}$ is anr, then it is amp.

Proof. First suppose that $\mathscr{G}$ is connected, i.e.:

a) none of the i-attributes of any $F \in N$ is isolated in ids (F);

b) for every p: $F_0 \to w_0 F_1 \ldots F_k w_k$ and $a \in S(F_j)$ ($j \in [k]$) there exists $a_0 \in S(F_0)$ such that $a_0(F_0) <_p a(F_j)$.

By Lemmas 3.1 and 3.4 every $t \in D_Z$ has a (total) computation sequence respecting $\mathscr{D}/\pi_0$, thus $\mathscr{G}$ is atomic m-pass with respect to $\mathscr{D}$. To treat the general case extend $\mathscr{G}$ as follows. For each $F \in N$ let $\bar{v}(F)=v(F) \cup \{\bar{a}\}$, where $\bar{a}$ is a new s-attribute. Correspondingly, for each p: $F_0 \to w_0 F_1 \ldots F_k w_k$ add the rule $\bar{a}(F_0)=\bar{f}(b_1(F_0), \ldots$ $\ldots, b_i(F_0), a_1(F_{j_1}), \ldots, a_n(F_{j_n}), \bar{a}(F_1), \ldots, \bar{a}(F_k))$ to r_p, where $b_1, \ldots, b_i$ are all the isolated i-attributes of ids (F_0), $a_1(F_{j_1}), \ldots, a_n(F_{j_n})$ are all those s-attributes that cannot be connected with any $a_0 \in S(F_0)$ in $\mathrm{idp}(p)$ and $\bar{f}$ is a hypothetical function. Let $\bar{\mathcal{G}}$ denote this extended grammar. As $\bar{\mathcal{G}}$ is connected, we can construct $\bar{\mathcal{A}}$, $\bar{\mu}\,\bar{\chi}$, $\bar{\varrho}$ and $\bar{\pi}_0$ as above. Returning to the original grammar $\mathcal{G}$, let $\mathcal{A} = \bar{\mathcal{A}}$, $\mu(F) = \bar{\mu}(F)$ for each $F \in N$,

$$\chi_F(c) = \begin{cases} \bar{\chi}_F(c) & \text{if} \quad \alpha(c) \neq \bar{a}, \\ \emptyset & \text{if} \quad \alpha(c) = \bar{a}, \end{cases}$$

$\varrho = \bar{\varrho}$ and $\pi_0 = \bar{\pi}_0$. It is clear that $\mathcal{G}$ is amp with respect to $\mathcal{D} = (\mathcal{A}, \mu, \chi, \varrho, \pi_0)$ defined in this way.

A *top-down assignment* of partitions for $\mathcal{G}$ is a quadruple $\mathcal{T} = (\mathcal{Q}, \tau, \varphi, q_0)$, where

(i) $\mathcal{Q}$ is a finite nonempty set, the set of states;

(ii) $\tau = \{\tau_F | F \in N\}$ such that τ_F is a mapping of $\mathcal{Q}$ into the set Π_F of all ordered partitions of $v(F)$;

(iii) $\varphi = \{\varphi_p | p \in P\}$ such that if p: $F_0 \to w_0 F_1 \ldots F_k w_k$, then φ_p is a partial mapping of $\mathcal{Q}$ into $\mathcal{Q}^k$.

(iv) $q_0 \in \mathcal{Q}$ is the initial state.

Concerning states, $\mathcal{T}$ is considered as an ordinary deterministic top-down tree automaton working on the derivation trees of G. $s_{qt}(u)$ will denote the state in which $\mathcal{T}$ passes through a node u of a derivation tree t, starting from state q at the root. q will be omitted if $t \in D_Z$ and $q = q_0$.

Let t be a derivation tree and $q \in \mathcal{Q}$. An m-pass computation sequence h for t is said to respect $\mathcal{T}/q$ if $E_h(u) = \tau_F(s_{qt}(u))$ for all nodes $u(\in U_t(F))$.

Definition 3.3. $\mathcal{G}$ is generalized uniform m-pass with respect to $\mathcal{T}$ if every complete derivation tree has an m-pass computation sequence respecting $\mathcal{T}/q_0$. $\mathcal{G}$ is generalized uniform multi-pass (gump) if there exist $m \in \mathrm{N}$ and $\mathcal{T}$ such that $\mathcal{G}$ is generalized uniform m-pass with respect to $\mathcal{T}$. $\mathcal{G}$ is uniform multi-pass (ump) if $\mathcal{T}$ can be chosen so that $\mathcal{Q} = \bigcup(\Pi_F | F \in N)$, $\tau_F(\pi) = \pi$ for all $F \in N$, $\pi \in \Pi_F$, and φ is a top-down assignment of partitions in the sense of [5].

Lemma 3.5. If $\mathcal{G}$ is gump, then it is ump, too.

Proof. Let $\mathcal{G}$ be gump with respect to $\mathcal{T}$. For each $F \in N$ let

$$\mathcal{Q}_F = \{s_t(u) | t \in D_Z, \ u \in U_t(F)\}$$

be the set of possible states at an F-labelled node. Consider an arbitrary $t \in D_Z$ and a node $u \in U_t(F)$. Intervene in the work of $\mathcal{T}$ on t at node u as follows.

(i) Force $\mathcal{T}$ to change the state $s_t(u)$ to any other state $q \in \mathcal{Q}_F$ for which $\tau_F(q) = \tau_F(s_t(u))$.

(ii) Let it continue working as if q were the state in which it had reached u.

Let $s'(v)$ denote the new state assigned to any $v \in U_t$ in the above way. By Lemma 2.1 there exists a computation sequence h' for t such that $E_{h'}(v) = \tau_Y(s'(v))$ for all nodes $v \in U_t(Y)$. By successive interventions $\mathcal{T}$ can be forced to perform a uniform top-down assignment on t, thus $\mathcal{G}$ can be made uniform. Moreover, the pass-number m also remains the same.

Proposition 3.2. If $\mathcal{G}$ is amp, then it is ump.

Proof. Let $\mathcal{G}$ be atomic m-pass with respect to $\mathcal{D}=(\mathcal{A}, \mu, \chi, \varrho, \pi_0)$. Define $\mathcal{T}=(\mathcal{Q}, \tau, \varphi, q_0)$ as follows. $\mathcal{Q}=\bigcup(\{F\}\times\Pi_{\mu(F)}|F\in N)$, where $\Pi_{\mu(F)}$ denotes the set of all ordered m-partitions of $\mu(F)$; $q_0=(Z, \pi_0)$. For each $F\in N$ let

$$\tau_F(F, (\mathcal{A}_1, \ldots, \mathcal{A}_m)) = (\chi_F(\mathcal{A}_1), \ldots, \chi_F(\mathcal{A}_m)).$$

If $p\colon F_0\to w_0F_1\ldots F_kw_k\in P$, then

$$\varphi_p(F_0, \pi^{(0)}) = ((F_1, \pi^{(1)}), \ldots, (F_k, \pi^{(k)})),$$

where $\pi^{(n)}=(\mathcal{A}_1^{(n)}, \ldots, \mathcal{A}_m^{(n)})$ $0\leqq n\leqq k$, and for any $j\in[k]$ and $c\in\mu(F_j)$, $c\in\mathcal{A}_i^{(j)}$ iff i is the minimal number such that $c(F_j)\in\varrho_p(\mathcal{A}_i^{(0)})$. Let h be an m-pass computation sequence for $t\in D_F$ and $\pi\in\Pi_{\mu(F)}$. An easy induction on the depth of t shows that h respects $\mathcal{D}/\pi$ iff it respects $\mathcal{T}/(F, \pi)$. Thus, $\mathcal{G}$ is gump with respect to $\mathcal{T}$, and by Lemma 3.5 it is ump, too.

Proposition 3.3. If $\mathcal{G}$ is ump, then it is anr.

Proof. Let $t\in D_Z$, $u\in U_t(F)$ and $a<_Fb$ be arbitrary. It was proved in [5] (Lemma 4) that if h is a uniform computation sequence for t, then $b(u)$ must be evaluated sooner than $a(u)$ in h. Suppose $\mathcal{G}$ were not anr. It is clearly anc, so there must be a cut $t\in C_F$, a leaf $u\in U_t(F)$ and an attribute $a\in v(F)$ such that $a(rt(t))<_t^R a(u)$, or vice versa. Let t^k denote the cut which can be obtained by k-fold composition of t at node u. Since $\mathcal{G}$ is reduced, there exists a complete derivation tree t_0 that contains t^k. But in idt (t_0) there is a path containing at least k R-arcs, thus $\mathcal{G}$ cannot be uniform m-pass for any $m\leqq k$. This is a contradiction, since k is arbitrary.

By Propositions 3.1, 3.2 and 3.3 we have proved

Theorem 3.1. $\mathcal{G}$ is anr iff it is amp iff it is ump.

Let $\mathcal{G}$ be uniform m-pass with respect to τ. If $\pi=(A_1, \ldots, A_m)\in\mathcal{Q}$ and $a\in\bigcup_{i=1}^{m} A_i$, then let $i_\pi(a)$ denote the number i for which $a\in A_i$.

Lemma 3.6. (a) For every $F\in N$, $a<_Fb$ and $\pi\in\mathcal{Q}_F$:

$$i_\pi(a)-i_\pi(b) \geqq r(F, a, b).$$

(b) For every $F\in N$, $a\in S(F)$ and $\pi\in\mathcal{Q}_F$, $i_\pi(a)\geqq q(F, a)$.

Proof. (a) Let $\bar{r}(F, a, b)=\min(i_\pi(a)-i_\pi(b)|\pi\in\mathcal{Q}_F)$. We have to prove that $\bar{r}(F, a, b)\geqq r(F, a, b)$. Fix $F_0\in N$, $a_0<_{F_0}b_0$ and $p\colon F_0\to w_0F_1\ldots F_kw_k$ arbitrarily, and let $\pi_0\in\mathcal{Q}_{F_0}$. If $\varrho_p(\pi_0)=(\pi_1, \ldots, \pi_k)$, then there exists a complete derivation tree t and nodes $u_0, u_1, \ldots, u_k$ of t (u_n labelled by F_n) representing an occurrence of p such that $s_t(u_n)=\pi_n$ for every $0\leqq n\leqq k$. Let

$$a_0(F_0) <_p a_1(F_{i_1}) <_{F_{i_1}} b_1(F_{i_1}) <_p \ldots <_p a_n(F_{i_n}) <_{F_{i_n}} b_n(F_{i_n}) <_p b_0(F_0)$$

be any path in idp (p) containing l R-edges. Since there exists an m-pass computation sequence for t respecting τ/q_0,

$$i_{\pi_0}(a_0)-i_{\pi_0}(b_0) \geqq l+\sum_{j=1}^{n}(i_{\pi_{i_j}}(a_j)-i_{\pi_{i_j}}(b_j)) \geqq l+\sum_{j=1}^{n}\bar{r}(F_{i_j}, a_j, b_j).$$

Thus, $\bar{r}(F_0, a_0, b_0) \geqq l + \sum_{j=1}^{n} \bar{r}(F_{i_j}, a_j, b_j)$. However, by construction $r(F_0, a_0, b_0)$ is the minimal number satisfying this property, i.e. $r(F_0, a_0, b_0) \leqq \bar{r}(F_0, a_0, b_0)$ for all $F_0 \in N$, $a_0 <_{F_0} b_0$. Statement (b) can be proved similarly.

Theorem 3.2. If $\mathcal{G}$ is anr, then $m_0 = \max(q(Z, a) | a \in v(Z))$ is the minimal number m for which $\mathcal{G}$ is atomic m-pass.

Proof. Suppose $\mathcal{G}$ were atomic m-pass for some $m < m_0$. Then, by Proposition 3.2 $\mathcal{G}$ is uniform m-pass, too, which contradicts (b) of Lemma 3.6.

Now we give an algorithm that decides the anr property. It is assumed that the well-known test for the anc property has been executed before. In the first step we construct the set

$R = \{(F, a, b) | F \in N$, there exists $t \in D_F$ such that $a(rt(t)) <_t^R b(rt(t))\}$.

Step (2) constructs for each $F \in N$ and $a \in S(F)$ the set

$R(a(F)) = \{a'(Y) | a' \in S(Y)$ and there is a cut $t \in C_F$ and a leaf $u \in U_t(Y)$ such that $a(rt(t)) <_t^R a'(u)\}$,

while step (3) constructs for each $F \in N$ and $b \in I(F)$ the set

$R(b(F)) = \{b'(Y) | b' \in I(Y)$ and there is a cut $t \in C_Y$ and a leaf $u \in U_t(F)$ such that $b(u) <_t^R b'(rt(t))\}$.

Algorithm 3.2. *Input:* An anc AG $\mathcal{G}$.

Output: "yes" if $\mathcal{G}$ is anr, "no" otherwise.

(1) Construct the set $R \subseteq N \times A_s \times A_i$ as follows.

(a) Let $R_0 = \emptyset$ and set $i = 0$.

(b) For each p: $F_0 \to w_0 F_1 \dots F_k w_k \in P$ let

$$R_{i+1}^{(p)} = \{(F_0, a_0, b_0) | a_0(F_0) <_p^R b_0(F_0) \quad \text{or} \quad a_0(F_0) <_p a(F_j) <_{F_j} b(F_j) <_p b_0(F_0)$$
$$\text{for some} \quad (F_j, a, b) \in R_i\}.$$

Let $R_{i+1} = R_i \cup (\cup (R_{i+1}^{(p)} | p \in P))$. If $R_{i+1} = R_i$, then let $R = R_i$, else set $i = i+1$ and repeat step (b).

(2) Construct for each $F \in N$, $a \in S(F)$ the set $R(a(F))$ as follows.

(c) Let $L_0(a(F)) = \{a(F)\}$, $R_0(a(F)) = \emptyset$ and set $i = 0$.

(d) For each p: $F_0 \to w_0 F_1 \dots F_k w_k \in P$ and $a_0 \in S(F_0)$ let

$$L_{i+1}^{(p)}(a_0(F_0)) = \cup (L_i(a(F_j)) | j \in [k], \ a_0(F_0) <_p a(F_j));$$

$$R_{i+1}^{(p)}(a_0(F_0)) = \cup (R_i(a(F_j)) | j \in [k], \ a_0(F_0) <_p a(F_j)) \cup$$

$$\cup (L_i(a(F_j)) | j \in [k], \ a_0(F_0) <_p^R a(F_j), \quad \text{or} \quad a_0(F_0) <_p a'(F_m) <_{F_m}$$

$$<_{F_m} b'(F_m) <_p a(F_j) \quad \text{and} \quad (F_m, a', b') \in R).$$

Let $L_{i+1}(a(F)) = L_i(a(F)) \cup (\cup (L_{i+1}^{(p)}(a(F)) | p \in P))$;

$$R_{i+1}(a(F)) = R_i(a(F)) \cup (\cup (R_{i+1}^{(p)}(a(F)) | p \in P)).$$

If both $L_{i+1}(a(F)) = L_i(a(F))$ and $R_{i+1}(a(F)) = R_i(a(F))$ for all $F \in N$, $a \in S(F)$, then let $R(a(F)) = R_i(a(F))$, else set $i = i+1$ and repeat step (d).

(3) Construct for each $F \in N$, $b \in I(F)$ the set $R(b(F))$ as follows.

(e) Let $L_0(b(F)) = \{b(F)\}$, $R_0(b(F)) = \emptyset$ and set $i=0$.

(f) For each $p\colon F_0 \to w_0 F_1 \dots F_k w_k \in P$, $F \in N$ and $b \in I(F)$ let

$$L_{i+1}^{(p)}(b(F)) = \bigcup (L_i(b_0(F_0)) \mid F = F_j \text{ for some } j \in [k] \text{ and } b(F_j) <_p b_0(R_0))$$

$$R_{i+1}^{(p)}(b(F)) = \bigcup (R_i(b_0(F_0)) \mid F = F_j \text{ and } b(F_j) <_p b_0(F_0)) \cup$$

$$\cup (L_i(b_0(F_0)) \mid F = F_j \text{ and } b(F_j) <_p^R b_0(F_0)) \cup$$

$$\cup (L_i(b_0(F_0)) \mid F = F_j \text{ and } b(F_j) <_p a'(F_m) <_{F_m} b'(F_m) <_p b_0(F_0) \text{ for}$$

$$\text{some } m \in [k], \quad (F_m, a', b') \in R).$$

Let $L_{i+1}(b(F)) = L_i(b(F)) \cup (\bigcup (L_{i+1}^{(p)}(b(F)) \mid p \in P))$;

$$R_{i+1}(b(F)) = R_i(b(F)) \cup (\bigcup (R_{i+1}^{(p)}(b(F)) \mid p \in P)).$$

If both $L_{i+1}(b(F)) = L_i(b(F))$ and $R_{i+1}(b(F)) = R_i(b(F))$ for all $F \in N$, $b \in I(F)$, then let $R(b(F)) = R_i(b(F))$, else set $i = i+1$ and repeat step (f).

(4) Output "no" if there exist $F \in N$ and $a \in \nu(F)$ such that $a(F) \in R(a(F))$. Otherwise output "yes". Halt.

Theorem 3.3. The atomic (uniform) m-pass property can be decided in polynomial time.

Proof. By Theorem 3.2 it is enough to prove that Algorithms 3.1 and 3.2 are both polynomial. The size of $\mathcal{G}$ will be expressed in the parameters $n = \|N\|$, $p = \|P\|$, $a = \|A\|$ and x, which is the maximum number of nonterminal occurrences in a production. First consider Algorithm 3.2. One execution of step (b) requires $O(pxa^2)$ time, and it must be executed at most $\|R\| < na^2$ times. Thus, the total amount of time required for step (1) is $O(pxna^4)$. In step (2) consider the sum

$$S_i = \sum (\|L_i(a(F))\| + \|R_i(a(F))\| \mid F \in N,\ a \in S(F)) \quad \text{for each } i.$$

Since $2na$ is an upper bound for all the S_i, step (d) must be executed at most $O(na)$ times. One execution of step (d) requires $O(pxa^2)$ time, so step (2), as well as step (3), needs $O(pxna^3)$ amount of time. Thus, the complexity of Algorithm 3.2 is $O(pxna^4)$. Now consider Algorithm 3.1. One execution of step (b) requires $O(pxa^2)$ time, and by Lemma 3.3 it must be executed at most $i_0 < na$ times. Similarly, step (d) must be executed at most $n_0 < na$ times, and one execution needs $O(pxa^2)$ time. Step (3) is not relevant, thus the complexity of Algorithm 3.1 is $O(pxna^4)$, too.

4. Implementation

As it is usual, a node u of a derivation tree is considered an object consisting of the following data.

— *u.prod*: number of production applied at u;
— *u.j*: reference to the j-th son of u;
— *u.a*: attribute a at u for each $a \in \nu(u)$;
— *u.on_c* and *u.off_c*: Boolean flags for each atom $c \in \mu(u)$.

The initial value of all the flags is false. *u.on_c* can be set to true by procedure *activate* (u, c), while procedure *release* (u, c) sets *u.on_c=false* and *u.off_c=* *=true*. Boolean functions *active* (u, c) and *done* (u, c) are used to test the value of *on_c* and *off_c* at u, respectively. For each $F \in N$ we have a procedure *F_pass* (u), which makes one visit to an F-labelled node u as follows.

```
Procedure F_pass(u) node u
begin
    for each $c \in \mu(F)$
    if active (u, c) then
    begin
        case u.prod of
         ⋮
         p: comment p: $F_0 \to w_0 F_1 \ldots F_k w_k$
            for $j=1$ to $k$
            begin
                for each $d \in \varrho_p(c) \cap \mu(F_j)$
                if not done (u.j, d) then
                begin
                    activate (u.j, d);
                    evaluate attributes $\chi_{F_j}(d) \cap I(F_j)$ at u.j
                end;
                $F_j$_pass (u.j)
            end;
            evaluate attributes $\chi_{F_0}(c) \cap S(F_0)$ at u
         ⋮
        esac;
        release (u, c)
    end
end
```

Procedure *evaluate* (u) controls the evaluation of a complete derivation tree with root u.

```
Procedure evaluate (u) node u
comment $\pi_0 = (\mathscr{A}_1, \ldots, \mathscr{A}_m)$
begin
    activate (c, u) for all $c \in \mathscr{A}_1$;
    ⋮
    Z_pass(u);

    activate (c, u) for all $c \in \mathscr{A}_m$;
    Z_pass (u)
end
```

5. Conclusion

The main advantage of the atomic characterization is that the evaluation procedure can be implemented in a simple and efficient way. It is not practial to base the evaluation procedure directly on a uniform top-down assignment of partitions, since the number of possible partitions is exponential. Using the atomic characterization, however, the size of implementation becomes polynomial, and the characterization itself can be carried out in polynomial time. Unfortunately, the atomic characterization of uniform multi-visit (i.e. anc) AG is not so useful (although it is possible to do it) because it can be proved that the decision problem of the uniform m-visit property is P-space complete. On the other hand, if we consider subclasses of anc AG in which the way of walking through the derivation trees is fixed independently of the grammar, then the atomic characterization is always useful. For example, it is worth doing the atomic characterization of ASE-like anc AG in which the attributes are evaluated in alternative left-to-right and right-to-left passes.

Abstract

A natural characterization of uniform multi-pass attribute grammars is introduced. An easy algorithm is given to test the uniform multi-pass property, and it is proved that the uniform m-pass property for fixed $m \in \mathrm{N}$ is decidable in polynomial time.

RESEARCH GROUP ON THEORY OF AUTOMATA
HUNGARIAN ACADEMY OF SCIENCES
SOMOGYI U. 7.
SZEGED, HUNGARY
H—6720

DEPT. OF COMPUTER SCIENCE
A. JÓZSEF UNIVERSITY
ARADI V. TERE 1.
SZEGED, HUNGARY
H—6720

References

[1] ENGELFRIET, J. and G. FILE, Simple multi-visit attribute grammars, Journal of Computer and System Sciences, v. 24, 1982. pp. 283—314.
[2] ENGELFRIET, J. and G. FILE, Passes and paths of attribute grammars, Information and Control, v. 49, 1981, pp. 125—169.
[3] KENNEDY, K. and S. K. WARREN, Automatic generation of efficient evaluators for attribute grammars, Conf. Record of the Third ACM Symp. on Principles of Programming Languages, 1976, pp. 32—49.
[4] KNUTH, D. E., Semantics of context free languages, Math. Systems Theory, v. 2, 1968, pp. 127—145.
[5] NIELSON, H. R., Computation sequences: a way to characterize classes of attribute grammars, Acta Informatica, v. 19, 1983, pp. 255—268.

Received July 14, 1984