Gramáticas de Atributos

Marcelo Arroyo
marroyo@dc.exa.unrc.edu.ar

Departamento de Computación
Facultad de Cs. Exactas, Fco-Qcas y Naturales
Universidad Nacional de Río Cuarto

Temario

1. Introducción y aplicaciones
2. Principales características
3. Un ejemplo
4. Técnicas de evaluación
5. Extensiones
6. Relación con otros formalismos
7. Herramientas basadas en Gramáticas de Atributos

Introducción

- Propuestas por D. Knuth en 1966
- Extensión de las gramáticas libres de contexto. Los símbolos contienen propiedades (computables) llamadas *atributos*
- Principales aplicaciones:
  - Especificar la semántica de lenguajes
  - Procesamiento de lenguajes: especificaciones de computaciones sobre árboles
- Características:
  - Modularidad de las gramáticas libres de contexto
  - Expresividad de un lenguaje funcional

ejemplo:memoria requerida por una declaración (con *agcc*)

REGULAR EXPRESSIONS:

id : <[a-z][a-z0-9]*>

integer: <"int">

character:<"char">

END

SEMANTIC DOMAINS:

.+.: int,int->int;

size:(INT|CHAR)->int;

END

SYNTHETIZED ATTRIBUTES:

m: int of id, lid, decl

t: (INT|CHAR) of type, decl

END

INHERIT ATTRIBUTES:

t:(INT|CHAR) of id,lid, decl

END

```
BEGIN GRAMMAR:
RULE r1: decl -> type lid
COMPUTE
    decl.t = type.t
    decl.m = lid.m
    lid.t = type.t
END
RULE r2: type -> integer
COMPUTE
    type.t = INT
END
RULE r3: type -> character
COMPUTE
    type.t = CHAR
END
RULE r4: lid -> id
COMPUTE
    id.t = lid.t
    lid.m = size(lid.t)
END
RULE r5: lid -> lid id
COMPUTE
    lid[1].t = lid[0].t
    lid[0].m = lid[1].m+size(id.t)
    id.t = lid[0].t
END

END GRAMMAR
```

Gramáticas de atributos: definiciones

Una *gramática de atributos* $GA =< G, A, D, F, R >$, donde

- G es una gramática libre de contexto $G =< T, N, P, S >$
- A es un conjunto de *atributos*. $A = \bigcup_{X \in (T \cup N)} A(X)$,
  $A(X) = H(X) \cup S(X)$, $H(X) \cap S(X) = \emptyset$
  $H(X)$, $S(X)$: conj. de *atributos heredados* y *sintetizados* de X, resp.
- $D = \bigcup_{a \in A} Dom(a)$, es el conjunto de dominios semánticos
- $R = \bigcup_{p \in P} R_p$: conjunto de *reglas semánticas* de la forma

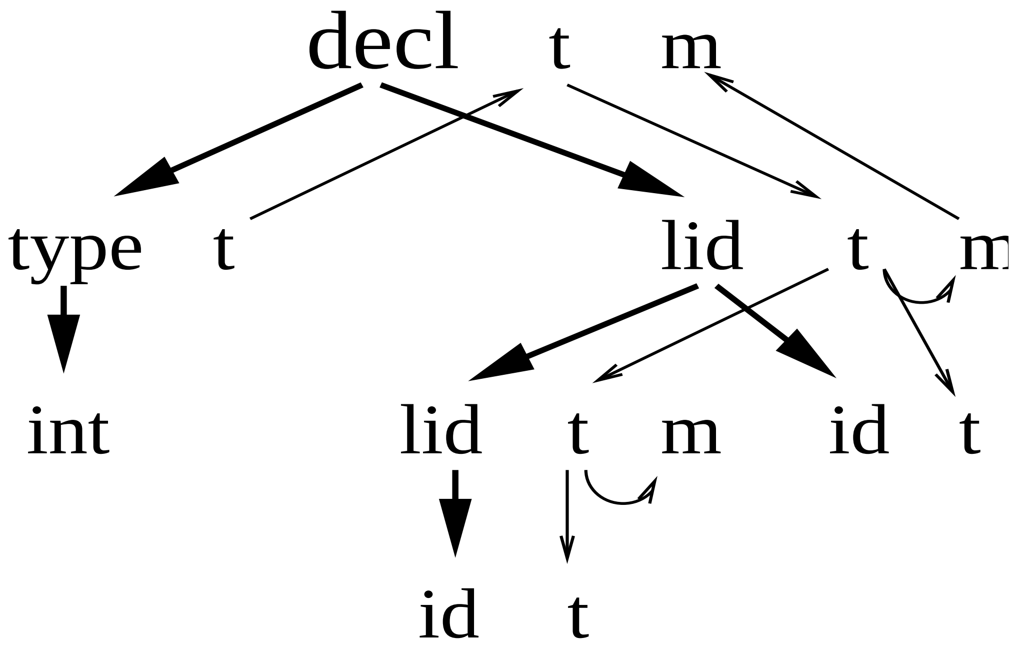
$$X_0.a_0 = f(X_1.a_1, \ldots, X_k.a_k)$$

  – diremos que $X_0.a_0$ *depende* de $X_1.a_1, \ldots, X_k.a_k$
- $F = \{f : Dom(a_1) \times \ldots \times Dom(a_k) \to Dom(a_0)\}$ son las *funciones semánticas*

Gramáticas de atributos bien definidas

- ***árbol atribuido***: $AT =< Node, Childs >$, donde $Node = \{< X, a_1, \ldots, a_k >\}$, $a_i \in A(X), 1 \leq i \leq k$, $(X \in (V \cup T))$ y $Childs : Node \rightarrow \{Node\}$
- *Grafo de dependencias*: relación (clausura) definida por las dependencias entre las instancias de los atributos en un árbol atribuido
- $eval : AG \times AT \rightarrow AT'$ es una *función de evaluación de atributos.* AT' se denomina ***árbol decorado***
- Una GA es *bien definida* sii no es posible generar un árbol atribuido correspondiente a GA cuyo grafo de dependencias entre los atributos sea circular
- *Test de circularidad:* complejidad exponencial

Arbol atribuido



Técnicas de evaluación

- Evaluación estática:
  - generación de planes de evaluación (ej: secuencias de visita)
  - definiciones de subclases (restricciones sobre las dependencias)

$$SAG \subseteq LAG \subseteq OAG \subseteq ANCAG \subseteq WFAG$$

$$NC(1) \subseteq NC(m) \subseteq NC(\infty)$$

- Evaluación dinámica: construcción del grafo de dependencias para el árbol atribuido de entrada y la evaluación sigue el orden topológico
- Evaluación secuencial vs concurrente
  - combinaciones de las anteriores

Extensiones

- Esquemas de traducción (introducen secuencia explícitas de evaluación)
- Gramáticas de atributos condicionales
  $A.x :=$ if *cond* then $f(\ldots B.y \ldots)$ else $g(\ldots C.z \ldots)$
- Parallel Attribute Grammars
- Affix Grammars (AG de dos niveles, los atributos son operadores)
- Extended Affix Grammars (los atributos son expresiones)
- Gramáticas de atributos de alto orden (HAG)
- Gramáticas de atributos acopladas (HAG's de dos niveles)

Relación con Lenguajes de Programación

- Programación funcional
- Programación lógica
- Programación orientada a objetos

Herramientas basadas en gramáticas de atributos

- YACC: Yet Another Compiler Compiler
- FCN-2 Attribute Grammar System
- TOOLS
- ELI
- NCEval
- AGCC