Attribute Grammars and Automatic Complexity Analysis

Marni Mishna

Projet ALGO, INRIA Roquencourt

A simple technique for specifying the meaning of languages

(Knuth)

Problem:

An explicit, yet simple approach for the analysis of decomposable structure properties

$\downarrow$

Automatic average-case complexity analysis

Program description AND Multivariate GFs (moments, distribution)

Structures

- Specification (`combstruct`)
- Generating Functions
- Random Generation [FlGoSt90, FlZiVa94]
- Substructure Counting
  - Multivariate GFs
- Object Grammars [DuFe98]

Algorithm Analysis

- On combstruct grammars
- Recursive program description
- Average case analysis
  - LUO Univariate GFs [FlSt84, FlSaZi91]

Attribute Grammars

- Introduced in 1968 by D. Knuth
  - Consistency check
  - Simplifying compiler descriptions
- Program description/ Functional programming
- Executable e.g. FNC-2 [Jourdan, ...]
- Useful for combinatorics!
  - q-series enumeration [DeFe94]
  - Random generation [DuFe98]

Binary Trees

$T = N \mid N \cdot T \cdot T$ $\qquad (T_0 = N \mid N \cdot T_1 \cdot T_2)$

Attribute Productions

$$
\begin{aligned}
\mathrm{size}(T) &= 1|1 + \mathrm{size}(T_1) + \mathrm{size}(T_2) \\
\mathrm{size}(N) &= 1 \\
\mathrm{ht}(T) &= 1|\max(\mathrm{ht}(T_1), \mathrm{ht}(T_2)) + 1 \\
\mathrm{pl}(T) &= 0|\mathrm{size}(T_1) + \mathrm{size}(T_2) + \mathrm{pl}(T_1) + \mathrm{pl}(T_2) \\
\mathrm{depth}(N) &= \mathrm{depth}(T_0) \\
\mathrm{depth}(T_1) &= \mathrm{depth}(T_0) + 1 \\
\mathrm{depth}(T_0) &= 0 \\
\mathrm{dpsum}(T_0) &= \mathrm{depth}(N)|\mathrm{depth}(N) + \mathrm{dpsum}(T_1) + \mathrm{dpsum}(T_2)
\end{aligned}
$$

Decomposable Structures and Grammars

- specification: a rule $T_i = \Phi_i(T_1, T_2, \ldots, T_m)$ where Φ_i is ϵ, Atom (Z), $|$, $\cdot$, $*$, Set, Multiset, Cycle
  grammar: a set of specifications
- Each T_j required is a descendant of T_i and T_i is an ancestor of each T_j.
- either labelled or unlabelled

Common Examples:	
Non-Plane Trees	$T = Z \cdot \mathrm{Set}(T)$
Functional Graphs	$G = \mathrm{Set}(\mathrm{Cycle}(T))$
Integer Partitions	$I = \mathrm{Set}(Z^*)$ (unlabelled)
Permutations	$P = \mathrm{Set}(\mathrm{Cycle}(Z))$ (labelled)

Attribute Grammars

- ⋆ attribute: a function which assigns values to structures.

- ⋆ attribute grammar: a three-tuple (G, A, P); G is a grammar, A is a set of attributes and P is a set of productions defining the values of the attributes.

- ⋆ synthesized: an attribute which is a function of descendants
  inherited: an attribute which is a function of its ancestors.

- ⋆ well-defined: no circular dependencies amongst the attributes.

- ⋆ linear attribute: a linear function of the attributes of the descendants.

Binary Trees

$T = N \mid N \cdot T \cdot T \qquad (T_0 = N \mid N \cdot T_1 \cdot T_2)$

Attribute Productions

$$
\begin{aligned}
\mathrm{size}(T) &= 1|1 + \mathrm{size}(T_1) + \mathrm{size}(T_2) \leftarrow \text{Synthetic, Linear} \\
\mathrm{size}(N) &= 1 \\
\mathrm{ht}(T) &= 1|\max(\mathrm{ht}(T_1), \mathrm{ht}(T_2)) + 1 \leftarrow \text{Non-linear} \\
\mathrm{pl}(T) &= 0|\mathrm{size}(T_1) + \mathrm{size}(T_2) + \mathrm{pl}(T_1) + \mathrm{pl}(T_2) \\
\mathrm{depth}(N) &= \mathrm{depth}(T_0) \\
\mathrm{depth}(T_1) &= \mathrm{depth}(T_0) + 1 \leftarrow \text{Inherited} \\
\mathrm{depth}(T_0) &= 0 \\
\mathrm{dpsum}(T_0) &= \mathrm{depth}(N)|\mathrm{depth}(N) + \mathrm{dpsum}(T_1) + \mathrm{dpsum}(T_2)
\end{aligned}
$$

Allowable Attributes I

- ⋆ First consider linear, synthesized, well-defined
- ⋆ Attributes extended to set/sequence/cycle: iterative (ϕ)
- ⋆ e.g. Non-plane trees $T = N \cdot \mathrm{Set}(T)$: $\mathrm{pl}(T) = \phi(\mathrm{size}(T) + \mathrm{pl}(T))$

General Specificaton:

$$A = \Phi_1(B_1^1, \ldots, B_{k_1}^1) | \ldots | \Phi_m(B_1^m, \ldots, B_{k_m}^m)$$

General Attribute:

$$F_i(A) = \bigcup_m \phi(\delta_i^m + \sum_{j,k} \alpha_{ij}^m F_j(B_k^m)) + \gamma_i$$

Examples

Pathlength of non-plane trees

$T = Z \cdot \mathrm{Set}(T) \qquad \mathrm{pl}(T) = \phi(\mathrm{pl}(T) + \mathrm{size}(T))$

Area of Dyck paths

$P = M^* \qquad \mathrm{area}(P) = \phi(\mathrm{area}(M))$

$M = xPy \qquad \mathrm{area}(M) = \mathrm{area}(P) + \mathrm{len}(P) + 1$

Number of cycles in a permutation

$P = \mathrm{Set}(\mathrm{Cycle}(Z)) \qquad \mathrm{cycles}(P) = \phi(1)$

Number of parts in a partition

$P = \mathrm{Set}(Z^*) \qquad \mathrm{parts}(P) = \phi(1)$

Generating Functions

$$A(z) = \sum_{a \in A} z^{|a|} = \sum_{n > 0} a_n z^n, \; A(z) = \sum_{n > 0} \frac{a_n}{n!} z^n$$

Folk Theorem: Generating function translations of constructions

C	$C(z)$	Universe
$A\vert B$	$A(z) + B(z)$	either
$A \cdot B$	$A(z)B(z)$	either
A^*	$\frac{1}{1-A(z)}$	either
Set(A)	$\exp(A(z))$	labelled
Set(A)	$\exp(\sum_{k>0} (-1)^{k-1} \frac{A(z^k)}{k})$	unlabelled
Cycle(A)	$\ln(\frac{1}{1-A(z)})$	labelled
Cycle(A)	$\sum_{k>0} \ln(\frac{1}{1-A(z^k)})$	unlabelled
MultiSet(A)	$\exp(\sum_{k>0} \frac{A(z^k)}{k})$	unlabelled

AG GF Theorem

$$\boxed{A(\mathbf{z}) = \sum_{a \in A} z_1^{|a|} z_2^{F_2(a)} \ldots z_k^{F_k(a)}}$$

$$A = \Phi_1(B_1^1, \ldots, B_{k_1}^1) | \ldots | \Phi_n(B_1^n, \ldots, B_{k_n}^n)$$

$$F_i(A) = \bigcup_m \phi(\delta_i^m + \sum_{j,k} \alpha_{ij}^m F_j(B_k^m)) + \gamma_i$$

$$\boxed{A(\mathbf{z}) = \sum_m \mathbf{z}^{\gamma^m} \mathcal{G}_{\Phi_m}(\mathbf{z}^{\delta^m} B_k^m(\mathbf{z}^{\alpha^m}))}$$

where $\mathcal{G}_{\Phi_m}$ is the generating function transformation

from the folk thm, $\mathbf{z}^\gamma = z_1^{\gamma_1} z_2^{\gamma_2} \cdots z_k^{\gamma_k}$, and

$$\mathbf{z}^\alpha = (\prod_{i=1}^k z_i^{\alpha_{i1}}, \ldots, \prod_{i=1}^k z_i^{\alpha_{ik}}).$$

Examples

Pathlength of non-plane trees

$T = Z \cdot \mathrm{Set}(T)$ $\quad$ $\mathrm{pl}(T) = \phi(\mathrm{pl}(T) + \mathsf{size}(T))$

$$T(z,u) = ze^{T(zu,u)}$$

Area of Dyck paths

$P = M^*$ $\quad$ $\mathrm{area}(P) = \phi(\mathrm{area}(M))$

$M = xPy$ $\quad$ $\mathrm{area}(M) = \mathrm{area}(P) + \mathrm{len}(P) + 1$

$$P(z,u) = \frac{1}{1-M(z,u)}$$

$$M(z,u) = \frac{z^2u}{Path(zu,u)}$$

Number of cycles in a permutation

$P = \mathrm{Set}(\mathrm{Cycle}(Z))$ $\quad$ $\mathrm{cycles}(P) = \phi(1)$

$$P(z,u) = e^{u\ln(\frac{1}{1-z})}$$

Number of parts in a partition

$P = \mathrm{Set}(Z^*)$ $\quad$ $\mathrm{parts}(P) = \phi(1)$

$$P(z,u) = e^{\sum_{k>0} \frac{(-1)^{k-1}u^k}{(1-z^k)k}}$$

Proof of AG GF Theorem

Idea can be illustrated with production: $A = \Phi(B)$ $(n = 1)$

- ⋆ linear attributes corresponds to q-linear series

$$\begin{aligned}
A(\mathbf{z}) &= \sum_{a \in A} z_1^{F_1(a)} z_2^{F_2(a)} \ldots z_k^{F_k(a)} \\
&= \sum_{a \in A} \prod_i z_i^{\gamma_i} \cdot \prod_{b \in a} \prod_i z_i^{\delta_i + \sum_{j=1}^k \alpha_{ij} F_j(b)} \\
&= \mathbf{z}^\gamma \sum_{a \in A} \mathbf{z}^\delta \prod_{b \in a} \prod_j (\prod_i z_i^{\alpha_{ij}})^{F_j(b)} \\
&=^* \mathbf{z}^\gamma \mathcal{G}_\Phi(\mathbf{z}^\delta B(\mathbf{z}^\alpha))
\end{aligned}$$

* $B(\mathbf{z}) = \sum_{b \in B} \prod z_j^{F_j(b)}$, $A(z) = \sum_{a \in A} \prod_{b \in a} z^{|b|} = \mathcal{G}(B(z))$

Allowable Attributes II

Selective attribute θ: Value of Set is linear attribute value of a randomly chosen element (LUO)

Self-Referential attribute

- $\star$ Must check for circularity
- $\star$ As a shortcut
- $\star$ Within an iterative

Are these q-linear??

Automatic Complexity Analysis

- ⋆ LUO lets the user describe algorithms on decomposible structures using: sequence of subprograms, test on unions, partial descent, component iteration

- ⋆ LUO returns the average complexity of $P(a:A)$ on structures of size n $[z^n]\tau_P(z)/[z^n]A(z)$

$$\mathrm{LUO} \subseteq \mathrm{AG}$$

- ⋆ Given a LUO procedure can define an attribute which evaluates to the complexity of the program on that input by a map of primitives

- ⋆ $\tau_P(z) = \frac{\partial A}{\partial u}(z,u)|_{u=1}$ for program $\mathrm{P}(a:A)$/attribute $\mathrm{P}(A)$

Differentiation

Find the average size of an expression of size n after differentiation.

```
>gram:= {Exp = Union( zero, one, x,
              Prod(plus, Exp, Exp),
              Prod(times, Exp, Exp),
              Prod(e, Exp))
           zero = Atom, x= Atom, one= Atom,
           plus = Atom, mult = Atom, e = Atom }:

>ag:= {DS(Exp) = Union( 0, 0, 1,
            Prod(1, DS(Exp), DS(Exp)),
            Prod(3, DS(Exp)+Size(Exp),
                            DS(Exp)+Size(Exp)),
            Prod(2, DS(Exp)+Size(Exp)))}:
>eqns:= agfeqns(gram, ag,
                unlabelled,[[u, DS]], z);
```

$$eqns := [\mathrm{Exp}(z,\, u) = z + z + zu + zu$$

$$\mathrm{Exp}(z,\, u)^2 + zu^3\,\mathrm{Exp}(z\,u,\, u)^2$$

$$+zu^2\,\mathrm{Exp}(z\,u,\, u), \ldots]$$

```
> num:= size_solve(eqns, [z,u],map(lhs, gram));
> avg:= avg_solve(eqns, [z,u],map(lhs, gram));
```

$$num := \{\mathrm{Exp}(z,\,1) = \frac{1}{4}\,\frac{1 - z - \sqrt{1 - 2\,z - 23\,z^2}}{z}$$

$$\mathrm{one}(z,\,1) = z,\ \mathrm{p}(z,\,1) = z,$$

$$\mathrm{m}(z,\,1) = z,\ \mathrm{e}(z,\,1) = z,\ \mathrm{x}(z,\,1) = z,\ \mathrm{zero}(z,\,1) = z\}$$

$$avg := \{e_2(z,\,1) = 2\,z,\ x_2(z,\,1) = z,\ zero_2(z,\,1) = z,$$

$$Exp_2(z,\,1) = \frac{1}{4}\,\frac{2\,z\,\%1 + 12\,z^2\,\%1 + 1 - 3\,z - 34\,z^2 - \%1}{z\,(-1 + 2\,z + 23\,z^2)},$$

$$Exp_1(z,\,1) = -\frac{1}{4}\,\frac{-1 + \%1 + z}{z^2\,\sqrt{1 - 2\,z - 23\,z^2}}...\}$$

$$\%1 := \sqrt{1 - 2\,z - 23\,z^2}$$

```
> num_exp:= subs(num, Exp(z,1)):
  avg_exp:=subs(avg, Exp[2](z,1)):
> num_coef:=equivalent(num_exp, z, n):
  avg_coef:=equivalent(avg_exp, z, n):
  gdev(avg_coef/num_coef, n=infinity, 2);
  evalf(%);
```

$$8 \frac{(\frac{13}{46} - \frac{3}{46}\sqrt{6} - \frac{17}{2}(-\frac{1}{23} + \frac{2}{23}\sqrt{6})^2)\sqrt{\pi}\, n^{(3/2)}}{(\frac{2}{23} - \frac{4}{23}\sqrt{6} - 46(-\frac{1}{23} + \frac{2}{23}\sqrt{6})^2)\sqrt{-\frac{2}{23} + \frac{4}{23}\sqrt{6} + 46(-\frac{1}{23} + \frac{2}{23}\sqrt{6})^2}} + \mathrm{O}(n)$$

$$.8042175440\, n^{(3/2)} + \mathrm{O}(n)$$

```
> alleqns[0]:=eqns:alleqns[1]:=num:alleqns[2]:=avg:
> moment_solve(alleqns, [z,u], names, 2):
  facmom:=equivalent(subs(%, Exp[2,2](z,1)),z,n):
  evalf(gdev((facmom+avg_coef)/num_coef
                  -(avg_coef/num_coef)^2,
            n=infinity, 2));
```

$$796.0402864\, n^{(3/2)}\, e^{(.005827030\, n)} + \mathrm{O}(\sqrt{n}\, e^{(.005827030\, n)})$$

Quicksort

```
Quicksort(L)
if |L| = 1 return L
else
   [L1, c, L2] = partition(L)
   [Quicksort(L1), c, Quicksort(L2)]
fi
end
```

- ⋆ Random input (permutations of [1,2,...,n])
- ⋆ Partition on the first element

$$T = \epsilon | \min(Z) \cdot T \cdot T$$

Calculations

Need new operator: min

```
{T=Union(Epsilon, `combstruct/Int`(Prod(Theta(Z), T, T)))}:
{PL(T)=Union(0, `combstruct/Int`(Prod(0, PL(T)+Size(T),PL(T)+Size(T))))}:
```

$$T(z,u) = Z(z,u) + \int \left(\frac{\partial}{\partial z} z \right) (T(zu,u))^2 \, dz \implies T(z,1) = \frac{1}{1-z}$$

$$T_u(z,u)|_{u=1} = \frac{x - 2\ln(1-z)}{(1-z)^2}$$

$$2H_n - 3 + \frac{H_n}{n} = [z^n]T_u(z,u)|_{u=1}/[z^n]T(z,1)$$

Future

- Random generation for structures with given attribute values (Dutour, Fédou)
- Distribution Analysis
- Other attribute types e.g. head/tail for sequences
- Composition of Functions
  - Generate structures with grammar
  - Grammar coupling