COMS3261: Computer Science Theory

Fall 2013

Mihalis Yannakakis

Lecture 15, 10/28/13

ALGORITHMS for CFLs

- We will discuss some more algorithms:
- Cleaning algorithms for CFG
  e.g. elimination of useless variables and productions
- Transformation to a simple form: Chomsky Normal Form

Algorithm for Computing the Generating Variables

- Generating variable: can derive some terminal string

- Initialization (Basis): K := T
- Loop (Induction): while ($\exists$ production $X \rightarrow \beta$ such that $X \notin K$ but all symbols of $\beta \in K$) $K := K \cup \{X\}$
- Return the variables in K

- Time: straightforward: $O(|G|^2)$, where $|G|$ is the size of the grammar (includes sum of lengths of the productions)
- Can do in $O(|G|)$ time with more care with appropriate data structure – see book, Sec 7.4.3

Example

- $S \rightarrow ABE \mid AC$
  $A \rightarrow 1B \mid 0C$
  $B \rightarrow 0D$
  $C \rightarrow 1$
  $D \rightarrow AB$
  $E \rightarrow 0$

- K ={0,1}
- Add C, E, A, S
- $\Rightarrow$ B, D not generating

Reachable Variables

- A variable X is called reachable if there is a derivation from start symbol $S \Rightarrow^* \alpha X \beta$ for some strings α, β

- Theorem: In every derivation of a terminal string from S, all the variables that appear in the derivation are generating and reachable
- Proof: Suppose X appears in a derivation of a terminal string w from S: $S \Rightarrow \ldots \Rightarrow \alpha X \beta \Rightarrow \ldots \Rightarrow w$

  Then X reachable (since $S \Rightarrow^* \alpha X \beta$) and X generating (derives a substring of w since $\alpha X \beta \Rightarrow^* w$)

$\Rightarrow$ Nongenerating and unreachable variables are useless:
If we remove these variables and all the productions where they appear (in head or body), then clearly language does not change.

Algorithm for Reachable Variables

Initialization (Basis): $R = \{S\}$

Loop (Induction): while ($\exists$ production $X \rightarrow \beta$ such that $X \in R$ but not all variables of β are in R) add all variables of β to R

- Correctness: Easy inductions (HW)

- Time: Straightfoward $O(|G|^2)$
  With little more care, $O(|G|)$.

Reachable variables

- Can reduce also to Graph Reachability
- Construct graph:
- Nodes = variables
- Edges = $A \to B$ if $\exists$ production $A \to \ldots B \ldots$
- Scan the productions and make for each variable A an adjacency list Adj(A) of the adjacent nodes = variables that appear in bodies of productions (B may appear several times).
- #edges = sum of lengths of Adj lists = sum |bodies|
- Apply a graph searching algorithm out of S
- O(n) time, where n = length of CFG description

Example

- $S \to ABE \mid AC$ $\quad$ $A \to 1B \mid 0C$

  $B \to 0D$ $\quad$ $C \to 1$

  $D \to AB$ $\quad$ $E \to 0$

- All variables reachable: R:={S}; S adds A,B,C,E; then B adds D
- If we remove first nongenerating variables B,D and their productions where they appear (in head or body), we get:

  $S \to AC$ $\quad$ $A \to 0C$

  $C \to 1$ $\quad$ $E \to 0$
- Reachable R ={S,A,C}. E is not reachable any more

Order of removal

- If we first remove all nongenerating variables (and productions that contain them) and then all unreachable variables, then no useless variables.
- Proof: All remaining variables reachable.

  Suppose some remaining variable X now nongenerating. Before the 2nd round, it was generating, i.e. $X \Rightarrow^* w$ for some terminal string. Since X not removed in 2nd round, reachable $\Rightarrow$ all the variables that appear in the derivation $X \Rightarrow^* w$ also marked reachable $\Rightarrow$ X still generating.

Other order may not work: if we first remove unreachable, then nongenerating, we may get more unreachable (cf. example)

Chomsky Normal Form

- Chomsky normal form (CNF): All productions are of the form $A \to BC$ or $A \to a$ (and can also ensure that there are no useless symbols)
- If $\varepsilon \in L(G)$ then we allow also the production $S \to \varepsilon$ and require that S not appear in the body of a production

- Transforming a CFG into Chomsky Normal Form:
- Can shorten the bodies of the productions to length ≤ 2; in fact either ≤ 2 variables or 1 terminal in each body
- Transform grammar to get rid of ε-productions (i.e., $A \to \varepsilon$) except that if $\varepsilon \in L$ then we lose it; can retain if we include $S \to \varepsilon$
- Transform to get rid of unit productions $A \to B$
- And can eliminate useless variables.

Eliminate terminals from bodies > 1

- If a terminal a appears in a body of length ≥ 2, then introduce a new variable X_a, add production $X_a \rightarrow a$, and use X_a in place of a in all the bodies of length ≥ 2
- After this, no "mixed" bodies: all bodies of length ≥ 2 have only variables
- Bodies of length 1 may have a variable or a terminal

Breaking up long bodies

- For each production $A \rightarrow X_1 X_2 \ldots X_k$ with $k \geq 3$, introduce new variables $Y_1, \ldots, Y_{k-2}$ (new for each production)

and new productions that replace the original

$A \rightarrow X_1 Y_1$

$Y_1 \rightarrow X_2 Y_2$

....

$Y_{k-2} \rightarrow X_{k-1} X_k$

Example: $A \rightarrow BCBD$ becomes $A \rightarrow BY_1$

$Y_1 \rightarrow CY_2$

$Y_2 \rightarrow BD$

Nullable variables

- Variable A nullable if $A \Rightarrow^* \varepsilon$

- Algorithm for computing nullable variables
- Initialization (Basis): $N = \{ A \mid A \to \varepsilon$ is in $P\}$
- Induction: while ($\exists$ production $A \to \beta = B_1 B_2 \ldots B_k$ such that all $B_i \in N$ but $A \notin N$) $N := N \cup \{A\}$
- Linear time : data structure same as for generating

Elimination of ε-productions

- Each production $A \to X_1 X_2 \ldots X_k$ that has some – say m nullable symbols in body, replaced by 2^m productions obtained by omitting in body any subset of nullable symbols, except if m=k we do not delete all m symbols
- Delete all ε-productions

- Example: $A \to BC \mid CD$, where B,C are nullable:
  becomes: $A \to B \mid C \mid BC \mid D \mid CD$

Cost: potentially exponential if we have long bodies

But if we first reduce to bodies of size ≤ 2; then a production will be replaced by at most 3 productions $\Rightarrow$ linear cost

Elimination of ε-productions ctd

- Theorem: If we eliminate the ε-productions from cfg G to get cfg G' as above, then $L(G') = L(G) - \{\varepsilon\}$

Proof:

$\subseteq$ In G' no ε-productions, so ε is not in L(G')

Every production of G' can be simulated in G by the production that generated it combined with derivation of ε from the omitted nullable symbols

$\supseteq$ By induction: If $A \Rightarrow X_1 X_2 \dots X_k \Rightarrow^* w = w_1 w_2 \dots w_k \neq \varepsilon$

some of the w_i may be ε but can use the production of A that produces only the X_i for the other i and by induction derive their w_i, and get w

- If $\varepsilon \in L(G)$, then to retain it in the language, we add new start symbol S' and productions

  $S' \to S \mid \varepsilon$

Unit productions and unit pairs

- Unit production: $A \to B$ for two variables A,B
- They do wasteful work: may have long sequence

  $A \to B \to C \to D \dots$

- Unit pair (X,Y): $X \Rightarrow^* Y$ via a sequence of unit productions

- Observation: If no ε-productions then $X \Rightarrow^* Y$ iff (X,Y) is a unit pair, because once we introduce a second symbol in a derivation, we can't get rid of it.
- Not true if there are ε-productions

Computation of unit pairs

- Directed graph $D = (V,E)$: nodes =variables,
- Edges = unit productions $A \to B$
- Reachable pairs = unit pairs

Algorithm:

Basis: $U = \{(A,A) \mid A \in V\}$

Induction: If $(A,B) \in U$ and $B \to C$ a production then add (A,C) to U.

At the end : U = set of unit pairs.

Time: all pairs reachability $O(|V| \cdot \#\text{unit productions})$

Example of unit pairs

$I \to a \mid b \mid Ia \mid Ib \mid I0 \mid I1$

$F \to I \mid (E)$

$T \to F \mid T * F$

$E \to T \mid E + T$

Unit productions: $F \to I$, $T \to F$, $E \to T$

Unit pairs: (F,I), (T,F), (T,I), (E,T), (E,F), (E,I), and self-pairs

Elimination of unit productions

- For each unit pair (A,B) and each production $B \to \beta$, add production $A \to \beta$
- Remove the unit productions

- Example

$I \to$ a | b | Ia | Ib | I0 | I1

$F \to$ (E) | a | b | Ia | Ib | I0 | I1

$T \to$ T * F | (E) | a | b | Ia | Ib | I0 | I1

$E \to$ E +T | T * F | (E) | a | b | Ia | Ib | I0 | I1

Elimination of unit productions may create useless symbols

- Example: $\{S \to A, A \to a\}$ becomes $\{S \to a, A \to a\}$
  A is useless now (unreachable)

Summary: Transformation to CNF

[optional: eliminate useless symbols]

- Eliminate terminals from bodies of length >1
- Eliminate bodies of length >2

- Eliminate ε-productions
- Eliminate unit productions
- Eliminate useless symbols
  a. eliminate nongenerating variables
  b. eliminate unreachable variables

Order important for correctness

The first group of transformations could be done after the second – but if there are long bodies that contain nullable symbols, better to break them up first

Chomsky Normal Form Theorem

- Theorem: Given a CFG G, we can construct a CNF grammar G' such that L(G') = L(G)

- All productions of the form $A \to BC$ or $A \to a$ and no useless symbols
- If ε is in L(G), we allow also $S \to \varepsilon$ and S does not appear in the body of any production

- Time of the algorithm is polynomial in the size of G