Theoretische Grundlagen der Informatik

Tutorium 7 | Wintersemester 2014/15

Moritz Klammler | 20. Januar 2015

Tagesthemen

- Syntaxbäume
- Chomsky-Normalform für kontextfreie Grammatiken
- Cocke-Younger-Kasami-Algorithmus

Syntaxbäume

Gegeben eine Grammatik $G = (T, V, S, P)$ und ein Wort $w \in T^*$ sodass $S \Rightarrow^* w$, also $w \in L(G)$.

Ein **Syntax- oder Ableitungsbaum** für w ist ein Baum, dessen

- Wurzel S ist,
- Blätter Terminale aus T sind, sodass eine Tiefensuche, auf der man die Blätter einsammelt, genau w ergiebt,
- innere Knoten Nichtterminale aus V sind und
- ein Knoten v nur dann die Kinder $u_1, \ldots, u_n$ (für ein $n \in \mathbb{N}$) hat, wenn $(v, u_1 \cdots u_n) \in P$.

Syntaxbäume

- Zu jeder Ableitung gehört genau ein Syntaxbaum.
- Zu jedem Syntaxbaum gehört genau ein Wort.

Die Umkehrung der beiden Aussagen gilt in der Regel nicht.

- Eine **Grammatik** G heißt **eindeutig**, wenn es für jedes Wort $w \in L(G)$ genau einen Syntaxbaum gibt.
- Eine **Sprache** L heißt **eindeutig** wenn es eine eindeutige Grammatik G gibt, sodass $L(G) = L$.
- Eine Sprache, die nicht eindeutig ist, heißt **inhärent mehrdeutig**.

Dangling Else

Zwei Interpretationen desselben C-Programms. (Einrückung hat keine syntaktische Relevanz.)

```
void
go_for_it(
    bool like_cookies_p,
    bool have_money_p)
{
  if (like_cookies_p)
    if (have_money_p)
      buy_cookies();
    else
      steal_cookies();
}
```

```
void
go_for_it(
    bool like_cookies_p,
    bool have_money_p)
{
  if (like_cookies_p)
    if (have_money_p)
      buy_cookies();
  else
      steal_cookies();
}
```

CHOMSKY-Normalform

Sei $G = (T, V, S, P)$ eine kontextfreie Grammatik. G ist in **CHOMSKY-Normalform (CNF)** genau dann wenn für alle $(\alpha, \beta) \in P$ gilt, dass

- $\alpha \in V$ und
  - $\beta \in V^2 \cup T$ oder
  - $\alpha = S$ und $\beta = \epsilon$.

Für jede kontextfreie Grammatik $G_1 = (T, V_1, S_1, P_1)$ existiert eine kontextfreie Grammatik $G_2 = (T, V_2, S_2, P_2)$ in CHOMSKY-Normalform, sodass $L(G_1) = L(G_2)$.

Kontextfreie Grammatik in CNF überführen

1. Falls $(S, \epsilon) \in P$: Neues Startsymbol S' einführen und neue Produktion $S' \to S \mid \epsilon$ hinzunehmen.

2. Terminale $\mathtt{a}$ in nicht-terminierenden rechten Seiten durch neue Nichtterminale $X_{\mathtt{a}}$ ersetzen und neue Produktionen $X_{\mathtt{a}} \to \mathtt{a}$ hinzunehmen.

3. Wiederholen bis nicht mehr anwendbar: $A \to BCD$ ersetzen durch $A \to Y_{BC}D$ und neue Produktion $Y_{BC} \to BC$ hinzunehmen.

4. ϵ-Produktionen $A \to \epsilon$ (für $A \neq S$) streichen und für $L \to AR$ neue Produktion $L \to R$ hinzunehmen.

5. Kettenregeln $A \to B$ streichen und für $B \to w$ neue Produktion $A \to w$ hinzunehmen.

Cocke-Younger-Kasami-Algorithmus

;; Eingabe: Grammatik $G = (T, V, S, P)$ *in CNF und Wort* $w \in T^*$ *mit* $|w| = n$
;; Ausgabe: Aussage ob $w \in L(G)$
FOR $i \leftarrow 1, \ldots, n$
    $\text{table}[i][1] \leftarrow \emptyset$
    FOREACH $(\alpha, \beta) \in P$ *;; terminierende Produktion*
        IF $\beta = w[i]$
            $\text{table}[i][1] \leftarrow \text{table}[i][1] \cup \{\alpha\}$
FOR $j \leftarrow 2, \ldots, n$
    FOR $i \leftarrow 1, \ldots, n - j$
        $\text{table}[i][j] \leftarrow \emptyset$
        FOR $k \leftarrow 1, \ldots, j$
            FOREACH $(\alpha, (\beta_1, \beta_2)) \in P$ *;; nicht−terminierende Produktionen*
                IF $\beta_1 \in \text{table}[i][k] \wedge \beta_2 \in \text{table}[i+k][j-k]$
                    $\text{table}[i][j] \leftarrow \text{table}[i][j] \cup \{\alpha\}$
RETURN $S \in \text{table}[1][n]$

```
SUBROUTINE DAXPY(N,DA,DX,INCX,DY,INCY)
DOUBLE PRECISION DA
INTEGER INCX,INCY,N
DOUBLE PRECISION DX(*),DY(*)
INTEGER I,IX,IY,M,MP1
INTRINSIC MOD
IF (N.LE.0) RETURN
IF (DA.EQ.0.0d0) RETURN
IF (INCX.EQ.1 .AND. INCY.EQ.1) THEN
   M = MOD(N,4)
   IF (M.NE.0) THEN
      DO I = 1,M
         DY(I) = DY(I) + DA*DX(I)
      END DO
   END IF
   IF (N.LT.4) RETURN
   MP1 = M + 1
   DO I = MP1,N,4
      DY(I) = DY(I) + DA*DX(I)
      DY(I+1) = DY(I+1) + DA*DX(I+1)
      DY(I+2) = DY(I+2) + DA*DX(I+2)
      DY(I+3) = DY(I+3) + DA*DX(I+3)
   END DO
ELSE
   IX = 1
   IY = 1
   IF (INCX.LT.0) IX = (-N+1)*INCX + 1
   IF (INCY.LT.0) IY = (-N+1)*INCY + 1
   DO I = 1,N
    DY(IY) = DY(IY) + DA*DX(IX)
    IX = IX + INCX
    IY = IY + INCY
   END DO
END IF
RETURN
END
```

Quelle: BLAS-Referenzimplementierung (http://www.netlib.org/blas/)