Eine Chomsky-Grammatik für die Fibonacci-Zahlen

Agnes Schubert

Seminarbeitrag *Molekulare Algorithmen*

FSU Jena
26. Juni 2017

rekursive Berechnung

Definition

Die *Fibonacci-Folge* ist rekursiv folgendermaßen definiert:

$$f(0) = 0$$
$$f(1) = 1$$
$$f(n) = f(n-1) + f(n-2) \qquad \forall n > 1.$$

Beginn der Folge:

$$0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, \ldots$$

explizite Berechnung

Definition

Die *Fibonacci-Folge* ist nach der Formel von Moivre-Binet explizit folgendermaßen definiert:

$$f(n) = \frac{\phi^n - \psi^n}{\phi - \psi}.$$

Dabei gilt $\phi = \frac{1+\sqrt{5}}{2}$ und $\psi = \frac{1-\sqrt{5}}{2}$.

Vortragsgliederung

Vortragsgliederung

Ziel

Wir suchen eine Grammatik, die folgende Sprache L erzeugt:

$$\mathcal{L} = \{0^x | \exists n \in \mathbb{N} : x = f(n)\}.$$

Dabei sei 0^x die Konkatenation von x 0en.
Beginn der Folge:

Fibonacci-Zahl	korrespondierendes Element in $\mathcal{L}$
0	ε
1	0
1	0
2	00
3	000
5	00000
8	00000000
13	0000000000000

Regeln $\mathcal{R}$:

$$
\begin{aligned}
S &\rightarrow \varepsilon | 0 | 00 | LYR \\
BX &\rightarrow XA \\
AX &\rightarrow XAB \\
LX &\rightarrow LYAB \\
YA &\rightarrow AY \\
YB &\rightarrow BY \\
YR &\rightarrow XR | 00 \\
A0 &\rightarrow 00 \\
B0 &\rightarrow 00 \\
L0 &\rightarrow 00
\end{aligned}
$$

$$G = \{N, T, \mathcal{R}, \mathcal{S}\}$$

mit

$$\begin{aligned} N &= \{S, A, B, L, R, X, Y\} \\ T &= \{0\} \\ \mathcal{R} & \quad \text{siehe letzte Folie} \\ \mathcal{S} &= S \end{aligned}$$

Ein Beispiel: Ableitung von $0^{f(5)} = 0^5$

Regeln $\mathcal{R}$:

$$\begin{aligned}
S &\to \varepsilon | 0 | 00 | LYR \\
BX &\to XA \\
AX &\to XAB \\
LX &\to LYAB \\
YA &\to AY \\
YB &\to BY \\
YR &\to XR | 00 \\
A0 &\to 00 \\
B0 &\to 00 \\
L0 &\to 00
\end{aligned}$$

Ableitung für $f(5)$:

$$\begin{aligned}
S &\to LYR \\
LYR &\to LXR \\
LXR &\to LYABR \\
LYABR &\to LAYBR \\
LAYBR &\to LABYR \\
LABYR &\to LAB00 \\
LAB00 &\to LA000 \\
LA000 &\to L0000 \\
L0000 &\to 00000
\end{aligned}$$

Ein Beispiel: Ableitung von $0^{f(5)} = 0^5$

Ableitung für $f(5)$:

$$
\begin{aligned}
S &\to LYR \\
LYR &\to LXR \\
LXR &\to LYABR \\
LYABR &\to LAYBR \\
LAYBR &\to LABYR \\
LABYR &\to LAB00 \\
LAB00 &\to LA000 \\
LA000 &\to L0000 \\
L0000 &\to 00000
\end{aligned}
$$

Analyse des Beispiels:

- falls weder S noch YR im Wort enthalten sind, kann immer nur genau eine Regel angewendet werden
- Aufbau der Wortlänge erfolgt durch die Regeln $LX \to LYAB$ und $AX \to XAB$
- vor Umwandlung der Nichtterminal- in Terminalsymbole wandert das Y von links nach rechts

Ein Beispiel: Ableitung von $0^{f(5)} = 0^5$

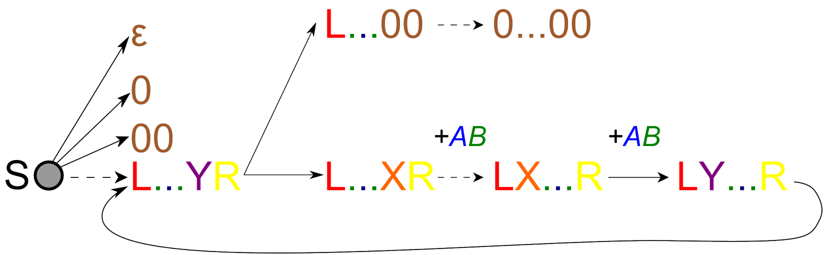
Ableitung für $f(5)$:

$$
\begin{aligned}
S &\rightarrow LYR \\
LYR &\rightarrow LXR \\
LXR &\rightarrow LYABR \\
LYABR &\rightarrow LAYBR \\
LAYBR &\rightarrow LABYR \\
LABYR &\rightarrow LAB00 \\
LAB00 &\rightarrow LA000 \\
LA000 &\rightarrow L0000 \\
L0000 &\rightarrow 00000
\end{aligned}
$$

Analyse des Beispiels:

- es gibt in jedem Wort der Ableitung höchstens ein Y oder ein X
- Rs kommen ausschließlich am rechten Wortrand vor
- Ls kommen ausschließlich am linken Wortrand vor

Ableitungsbaum der Grammatik



Vortragsgliederung

Wir definieren uns zwei Hilfsfunktionen:

$$a(v) = \begin{cases} 1 & \text{falls } v \in \{A, L, X, Y\} \\ 0 & \text{falls } v \in \{B, R\} \\ a(v_1) + a(v_2) & \text{falls } v = v_1 v_2 \end{cases}$$

$$b(v) = \begin{cases} 1 & \text{falls } v \in \{B, R\} \\ 0 & \text{falls } v \in \{A, L, X, Y\} \\ b(v_1) + b(v_2) & \text{falls } v = v_1 v_2 \end{cases}$$

Die Funktion a zählt also die Nichtterminalsymbole A, L, X und Y, die Funktion b die Symbole B und R.

Wir wollen folgende Aussage durch Induktion über m zeigen:

Erzeugt die m-te Anwendung der Regel $YR \rightarrow XR$ innerhalb einer Ableitung ein Wort $w \in N^*$, so gilt: $a(w) = f(m+2)$ und $b(w) = f(m+1)$.

Induktionsanfang

Sei $m = 1$. Es ist zu zeigen:
Bei der ersten Anwendung der Regel $YR \to XR$ innerhalb einer Ableitung entsteht ein Wort $w \in N^*$ und es gilt: $a(w) = f(1+2)$ und $b(w) = f(1+1)$.
Um $YR \to XR$ anwenden zu können, muss die Ableitung mit $S \to LYR$ starten. Direkt danach kann (und muss) $YR \to XR$ angewendet werden. Also gilt $w = LXR$ und damit
$a(w) = a(L) + a(X) + a(R) = 1 + 1 + 0 = 2 = f(3) = f(1+2)$
sowie
$b(w) = b(L) + b(X) + b(R) = 0 + 0 + 1 = 1 = f(2) = f(1+1)$.

Induktionsvoraussetzung & -behauptung

IV: Die Aussage gilt für $m = l$, d.h. bei der l-ten Anwendung der Regel $YR \rightarrow XR$ innerhalb einer Ableitung entsteht ein Wort $w \in N^*$ mit: $a(w) = f(l+2)$ und $b(w) = f(l+1)$.

IB: Die Aussage gilt dann auch für $m = l+1$, d.h. bei der $(l+1)$-ten Anwendung der Regel $YR \rightarrow XR$ innerhalb einer Ableitung entsteht ein Wort $w \in N^*$ mit:
$a(w) = f(l+1+2) = f(l+3)$ und
$b(w) = f(l+1+1) = f(l+2)$.

Induktionsschritt

Sei $m = l + 1$ und sei w das aus l-ter Anwendung der Regel $YR \to XR$ in der Ableitung entstandene Wort sowie w' jenes Wort, das aus $(l+1)$-ter Anwendung dieser Regel in der Ableitung entsteht.
Dann gilt nach IV: $a(w) = f(l+2)$ und $b(w) = f(l+1)$. Da w aus Anwendung der Regel $YR \to XR$ entstanden ist und Rs ausschließlich am rechten Wortrand vorkommen können, hat w die Form $\dots XR$. Da außerdem Ls nur am linken Wortrand und X bzw. Y nur einmal pro Wort vorkommen können, hat w die Form $L \dots XR$. Wobei die Zeichen in der Wortmitte alle As und Bs sind.

Induktionsschritt

$$w = LN_1N_2 \dots N_{f(l+3)-3}XR$$

mit $N_i \in \{A, B\}$.
Nun gilt

$$\begin{aligned} f(l+2) &= a(w) \\ &= a(L) + a(X) + a(R) + a(N_1) + a(N_2) + \cdots + a(N_{f(l+3)-3}) \\ &= 2 + a(N_1) + a(N_2) + \cdots + a(N_{f(l+3)-3}) \\ &= 2 + \sum_{i=1, N_i=A}^{f(l+3)-3} 1 \end{aligned}$$

$$\rightarrow \#A(w) = f(l+2) - 2$$

Induktionsschritt

$$w = LN_1N_2 \dots N_{f(l+3)-3}XR$$

mit $N_i \in \{A, B\}$.
Analog gilt:

$$\begin{aligned} f(l+1) &= b(w) \\ &= b(L) + a(X) + a(R) + a(N_1) + a(N_2) + \dots + a(N_{f(l+3)-3}) \\ &= 1 + a(N_1) + a(N_2) + \dots + a(N_{f(l+3)-3} \\ &= 1 + \sum_{i=1, N_i=B}^{f(l+3)-3} 1 \end{aligned}$$

$$\rightarrow \#B(w) = f(l+1) - 1$$

Induktionsschritt

$$w = L \dots XR$$

mit $\dots \in \{A, B\}^*$ und $\#A(w) = f(l+2) - 2$, $\#B(w) = f(l+1) - 1$.

Es müssen nun mehrfach die Regeln $BX \rightarrow XA$ und $AX \rightarrow XAB$ angewendet werden, bis das X an zweiter Stelle des Wortes steht. Sei w_1 das so entstandene Wort. Dann gilt:

- für jedes Bin w enthält w_1 ein A (Regel $BX \rightarrow XA$)
- für jedes Ain w enthält w_1 ein A und ein B (Regel $AX \rightarrow XAB$)

$\rightarrow$ $\#A(w_1) = \#A(w) + \#B(w) = (f(l+2)-2) + (f(l+1)-1) = f(l+3) - 3$

$\rightarrow$ $\#B(w_1) = \#A(w) = f(l+2) - 2$

Induktionsschritt

$$w_1 = LX \dots R$$

mit $\dots \in \{A, B\}^*$ und $\#A(w_1) = f(l+3) - 3$, $\#B(w_1) = f(l+2) - 2$.

Es muss nun die Regel $LX \rightarrow LYAB$ angewendet werden. Dabei entsteht das Wort w_2 mit:

- $\#A(w_2) = f(l+3) - 2$
- $\#B(w_2) = f(l+2) - 1$

Induktionsschritt

$$w_2 = LY \ldots R$$

mit $\cdots \in \{A, B\}^*$ und $\#A(w_2) = f(l+3) - 2$, $\#B(w_2) = f(l+2) - 1$.

Durch mehrfache Anwendung der Regeln $YA \to AY$ und $YB \to BY$ wandert das Y nach links, wobei die Anzahl der einzelnen Zeichen in den abgeleiteten Worten sich nicht ändert. Sei w_3 das so entstandene Wort, bei dem das Y an der zweiten Stelle von rechts steht. Dann gilt:

- $\#A(w_3) = \#A(w_2) = f(l+3) - 2$
- $\#B(w_3) = \#B(w_2) = f(l+2) - 1$

Induktionsschritt

$$w_3 = L \dots YR$$

mit $\dots \in \{A, B\}^*$ und $\#A(w_3) = f(l+3) - 2$, $\#B(w_3) = f(l+2) - 1$.

Auf w_3 muss nun eine der Regeln $YR \to XR$ und $YR \to 00$ angewendet werden. Bei Anwendung der Regel $YR \to 00$ kann allerdings nie wieder die Regel $YR \to XR$ angewendet werden, da das Y unwiderruflich aus dem Wort entfernt wird.
Uns interessiert allerdings die $(l+1)$-te Anwendung der Regel $YR \to XR$. Diese muss also w_3 als Ausgangswort verwenden. Das erzeugte Wort ist gerade das gesuchte w'.

Induktionsschritt

$$w' = L \dots X R$$

mit $\dots \in \{A, B\}^*$ und $\#A(w') = f(l+3) - 2$, $\#B(w') = f(l+2) - 1$.

Wir überprüfen nun die Aussage:

$$\begin{aligned} a(w') =& a(L) + a(Y) + a(R) + \#A(w') \cdot a(A) + \#B(w') \cdot a(B) \\ =& 1 + 1 + 0 + \#A(w') \cdot 1 + \#B(w') \cdot 0 \\ =& 2 + \#A(w') \\ =& 2 + (f(l+3) - 2) \\ =& f(l+3) \end{aligned}$$

Induktionsschritt

$$w' = L \dots X R$$

mit $\dots \in \{A, B\}^*$ und $\#A(w') = f(l+3) - 2$, $\#B(w') = f(l+2) - 1$.

Wir überprüfen nun die Aussage:

$$\begin{aligned} b(w') &= b(L) + b(Y) + b(R) + \#A(w') \cdot b(A) + \#B(w') \cdot b(B) \\ &= 0 + 0 + 1 + \#A(w') \cdot 0 + \#B(w') \cdot 1 \\ &= 1 + \#B(w') \\ &= 1 + (f(l+2) - 1) \\ &= f(l+2) \end{aligned}$$

Zusammenfassung

Wir haben gezeigt:
Erzeugt die m-te Anwendung der Regel $YR \to XR$ innerhalb einer Ableitung ein Wort $w \in N^*$, so gilt: $a(w) = f(m+2)$ und $b(w) = f(m+1)$.
Sei nun v das Wort, auf das die Regel $YR \to XR$ angewendet wird, um w zu erzeugen. Da die Anwendung dieser Regel lediglich ein Y durch ein X ersetzt und $a(Y) = a(X)$ sowie $b(Y) = b(X)$ gilt, so gilt auch $a(w) = a(v)$ und $b(w) = b(v)$.

Zusammenfassung

$$v = L \dots YR$$

mit $a(v) = f(m+2)$ und $b(v) = f(m+1)$.

Um nun Terminalsymbole zu erzeugen, müssen auf ein solches Wort v die Regel $YR \to 00$, anschließend mehrfach die Regeln $A0 \to 00$ und $B0 \to 00$ sowie abschließend die Regel $L0 \to 00$ angewendet werden. Offensichtlich hat das entstehende Wort (nur aus 0en) genauso viele Zeichen wie v. Sei $n(v)$ die Anzahl der Zeichen in v. Dann gilt:
$n(v) = a(v) + b(v) = f(m+2) + f(m+1) = f(m+3)$.

Zusammenfassung

Kann in einer Ableitung die Regel $YR \to XR$ zum m-ten mal angewendet werden, so kann alternativ das Wort $0^{f(m+3)}$ abgeleitet werden.

Aus der Struktur der Regeln ist ersichtlich, dass Ableitungsfolgen mit beliebig vielen Anwendungen von $YR \to XR$ möglich sind.

$\to$ Für beliebige $m \in \mathbb{N}, m > 0$ kann $0^{f(m+3)}$ abgeleitet werden.

$\to$ Für $n > 3$ kann $0^{f(n)}$ abgeleitet werden.

Zusammenfassung

Was ist mit $0^{f(n)}$ für $n \leq 3$?
Das geht auch:

Ableitung für $n = 0$
$0^{f(0)} = \varepsilon$

$S \to \varepsilon$

Ableitung für $n = 1, 2$
$0^{f(1)} = 0^{f(2)} = 0$

$S \to 0$

Ableitung für $n = 3$
$0^{f(3)} = 00$

$S \to 00$

Zusammenfassung

- Wir können $0^{f(n)}$ für jede Fibonacci-Zahl $f(n)$ ableiten.
- Da wir nur beim Startsymbol S und bei Vorkommen von YR am Ende eines Wortes eine Wahl zwischen verschiedenen Regeln haben, können wir keine anderen Worte ableiten.

$\rightarrow$ Die Grammatik erzeugt genau die gewünschten Worte.

Vortragsgliederung

von $f(n-1)$ zu $f(n)$

Wir gehen davon aus, dass jede Regelanwendung genau eine Zeiteinheit benötigt.
Um ein Wort der Länge $f(n)$ aus einem Wort der Länge $f(n-1)$ abzuleiten, benötigen wir (für $n > 4$):

- 1 Zeiteinheit für die Anwendung von $YR \to XR$
- $f(n-1) - 3$ Zeiteinheiten für die Anwendungen von $BX \to XA$ und $AX \to XAB$
- 1 Zeiteinheit für die Anwendung von $LX \to LYAB$
- $f(n) - 3$ Zeiteinheiten für die Anwendung von $YA \to AY$ und $YB \to BY$

$\to$ insgesamt $f(n-1) + f(n) - 4 \approx f(n+1)$ Zeiteinheiten

von S zu $f(n)$

Um ein Wort der Länge $f(n)$ aus S abzuleiten ($n > 4$), benötigen wir ungefähr:

- 1 Zeiteinheit für ein Wort der Länge $f(4)$
- $f(6)$ Zeiteinheiten für ein Wort der Länge $f(5)$
- $f(7)$ Zeiteinheiten für ein Wort der Länge $f(6)$
- ...
- $f(n+1)$ Zeiteinheiten für ein Wort der Länge $f(n)$

Das sind zusammen

$$1 + \sum_{i=6}^{n+1} f(i) = f(n+3) - 12 \approx \left(\frac{1+\sqrt{5}}{2}\right)^{n+3} - 12 \approx 1,62^{n+3} - 12$$

Zeiteinheiten! [IJON], [LP03]

$\rightarrow$ exponentielle Zeitkomplexität: $Time(G) \in \mathcal{O}(1,62^n)$

Speicherbedarf der Grammatik

Um $0^{f(n)}$ abzuleiten, erzeugen wir als Zwischenergebnisse nur Worte der Länge $\leq f(n)$. Zusätzlich benötigen wir konstant viel Speicherplatz für die Regeln etc.
Also brauchen wir insgesamt nur konstant mehr Speicherplatz als $f(n) \approx \left(\frac{1+\sqrt{5}}{2}\right)^n \approx 1,62^n$. [LP03]

$\rightarrow$ exponentielle Platzkomplexität: $Space(G) \in \mathcal{O}(1,62^n)$

Formel von Moivre-Binet

$$f(n) = \frac{\phi^n - \psi^n}{\phi - \psi}.$$

Dabei gilt $\phi = \frac{1+\sqrt{5}}{2}$ und $\psi = \frac{1-\sqrt{5}}{2}$.

- Rechenzeit zur Erzeugung von $f(n)$: $\mathcal{O}(1)$
- Speicherplatz zur Erzeugung von $f(n)$: $\mathcal{O}(\log n)(?)$
- aber: Die Formel enthält irrationale Zahlen!
  - $\rightarrow$ Damit numerische Fehler nicht zu groß werden, ist eine sehr genaue (also aufwendige) Gleitkomma-Arithmetik notwendig.

Vortragsgliederung

Simulation

Wie in der Vorlesung vorgestellt kann die angegebene Grammatik durch ein Splicing-System simuliert werden.

Ende

Vielen Dank für Ihre Aufmerksamkeit!

Literatur

- Lovász, Laszló et al. *Diskrete Mathematik*. Springer Verlag, Berlin/Heidelberg, 2003.
- *ijon.de/mathe/fibonacci* (abgerufen am 10.06.2017)