

Enumeration der Sprache einer kontextfreien Grammatik

Aufgabenstellung

In dieser Aufgabe haben Sie eine kontextsensitive Grammatik G und eine natürliche Zahl n gegeben und sollen die Menge $\{w \in \Sigma^* \mid G \rightarrow^* w \wedge |w| \leq n\}$ berechnen, d.h. alle von der Grammatik erzeugten Wörter mit Länge maximal n .

Diese Aufgabenstellung wird nachfolgend präzisiert.

Erläuterung

Am Anfang der Vorlesung haben Sie die *kontextsensitiven Grammatiken* kennengelernt. Wir definieren diese hier (**leicht abweichend zur Vorlesung**) folgendermaßen: Sei $G = (V, \Sigma, P, S)$ eine Grammatik. Wir nennen G kontextsensitiv wenn folgende Bedingungen erfüllt sind:

- $S \rightarrow \varepsilon \in P$ ist erlaubt, aber **nur, wenn S nicht auf der rechten Seite irgendeiner anderen Produktion auftaucht**.
- Für alle anderen Produktionen $\alpha \rightarrow \beta \in P$ muss gelten $|\alpha| \leq |\beta|$.

Die Grammatik $G = (\{S, A\}, \{a, b\}, P, S)$ mit den Produktionen

$$S \rightarrow \varepsilon \quad S \rightarrow aA \quad A \rightarrow bS$$

ist also nicht kontextsensitiv. Die äquivalente Grammatik $G' = (\{S, A, B\}, \{a, b\}, P', S)$ mit den Produktionen

$$S \rightarrow \varepsilon \quad S \rightarrow B \quad A \rightarrow b \quad A \rightarrow bB \quad B \rightarrow aA$$

hingegen *ist* kontextsensitiv. Ebenso ist die (nicht dazu äquivalente) Grammatik $G'' = (\{S, A\}, \{a, b\}, P'', S)$ mit den Produktionen

$$S \rightarrow A \quad S \rightarrow a \quad A \rightarrow S$$

kontextsensitiv, da S zwar auf der rechten Seite von $A \rightarrow S$ auftaucht, aber $S \rightarrow \varepsilon$ nicht enthalten ist.

Eine (in diesem Sinne) kontextsensitive Grammatik hat die Eigenschaft, dass – im Gegensatz zu den Typ 0-Grammatiken – kein Ableitungsschritt (außer dem trivialen $S \rightarrow \varepsilon$) die Länge des Wortes verringern kann. Somit kann man in endlicher Zeit entscheiden, ob ein Wort der Länge n erzeugbar ist, indem man nach und nach ausgehend vom Startsymbol alle Wörter mit maximaler Länge n erzeugt.

Hinweise

- Wenn Sie das Template verwenden, muss das leere Wort ε immer als leerer String `""` repräsentiert werden, *nicht* etwa als `"ε"`.
- Die Ausgabe der Wörter muss jeweils mit einem Wort pro Zeile sein. Die Wörter müssen aufsteigend nach Länge sortiert sein (und bei gleicher Länge aufsteigend lexikographisch sein). Auch dies übernimmt das Template für Sie.
- Startsymbol und Menge der Produktionen sind gegeben (bzw. sind im Template realisiert als Felder der Klasse `ContextSensitiveGrammar`). Die Mengen Σ der Terminale und V der Nichtterminale sind der Einfachheit halber wieder einfach alle Kleinbuchstaben/Zahlen bzw. alle Großbuchstaben. Sie können hierfür die Methoden `isTerminal` bzw. `isNonTerminal` verwenden.
- Sie müssen nicht prüfen, ob die Grammatik wirklich kontextsensitiv ist. Sie können annehmen, dass dies der Fall ist.

Implementierung

Machen Sie sich mit den Klassen `Production` und `ContextSensitiveGrammar` vertraut. Dann:

1. Implementieren Sie die Methode `Production.apply`, die für ein Wort $w \in (N \cup \Sigma)^*$ und eine Produktion p alle möglichen Folgewörter berechnet, die sich ergeben, wenn man die Produktion p auf w anwendet, also $\{w' \in (N \cup \Sigma)^* \mid w \rightarrow_p w'\}$.
2. Implementieren Sie die Methode `ContextSensitiveGrammar.enumerateWordsRaw`, die alle Wörter in $(N \cup \Sigma)^*$ mit Länge maximal n berechnet, die von G erzeugt werden. Nutzen Sie hierfür `Production.apply`.
3. Implementieren Sie die Methode `ContextSensitiveGrammar.enumerateWords`, die alle Wörter in Σ^* mit Länge maximal n berechnet, die von G erzeugt werden. Nutzen Sie hierfür `ContextSensitiveGrammar.enumerateWordsRaw`.

Der Grund für die Existenz von `ContextSensitiveGrammar.enumerateWordsRaw` ist, dass Sie dadurch eventuell Fehler leichter erkennen können, weil Sie dadurch auch alle Zwischenergebnisse sehen können.

Eingabe

Das Wort `Raw` oder `Normal`, gefolgt von einer kontextsensitiven Grammatik in der Form, die an den Beispieleingaben abzulesen ist, und einer einzelnen Zahl, die die maximale Wortlänge angibt.
Das Einlesen der Eingabe ist für Sie bereits in den Vorlagen implementiert.

Ausgabe

Für jede Grammatik: Die sortierte Folge aller erzeugten Wörter bis zu der gegebenen Länge, ein Wort pro Zeile, gefolgt von `END`. Die Wörter müssen in Anführungszeichen eingeschlossen sein und sortiert in der zuvor beschriebenen Weise. Das leere Wort wird einfach als `" "` ausgegeben.

Sample Input 1

```
Normal
CSG
Start symbol: S
Productions:
S ->
S -> T
T -> aa
T -> bb
T -> aTa
T -> bTb
END
6
```

Sample Output 1

```
" "
"aa"
"bb"
"aaaa"
"abba"
"baab"
"bbbb"
"aaaaaa"
"aabbba"
"abaaba"
"abbbba"
"baaaab"
"babbab"
"bbaabb"
"bbbbbb"
END
```

Sample Input 2

```
Normal
CSG
Start symbol: S
Productions:
S ->
S -> T
T -> aBC
T -> aTBC
CB -> CZ
CZ -> WZ
WZ -> WC
WC -> BC
aB -> ab
bB -> bb
bC -> bc
cC -> cc
END
15
```

Sample Output 2

```
" "
"abc"
"aabbcc"
"aaabbbccc"
"aaaabbbbccccc"
"aaaaabbbbbccccc"
END
```

Sample Input 3

```
Normal
CSG
Start symbol: S
Productions:
S -> DTA
S -> ab
T -> DTa
T -> Da
Da -> abD
Db -> bD
DA -> Ab
A -> a
END
12
```

Sample Output 3

```
"ab"
"abbabb"
"abbabbbbabb"
END
```

Sample Input 4

```
Normal
CSG
Start symbol: S
Productions:
S -> H
Ia -> aaI
IR -> aaK
Sa -> JaI
SR -> JaK
H -> JL
K -> aL
L -> M
L -> N
aQ -> Qa
M -> O
JQ -> Pa
aO -> QR
JO -> PR
P -> S
aT -> Ta
N -> T
JT -> Ua
U -> a
END
8
```

Sample Output 4

```
"aa"
"aaaa"
"aaaaaaaa"
END
```

Sample Input 5

```
Normal
CSG
Start symbol: A
Productions:
A ->
A -> X
X -> XX
X -> aab
X -> aaXb
X -> aXab
X -> aXaXb
X -> aXbXa
X -> abXa
X -> aXba
X -> aba
X -> bXaXa
X -> baXa
X -> bXaa
X -> baa
END
6
```

Sample Output 5

```
" "
"aab"
"aba"
"baa"
"aaaabb"
"aaabab"
"aaabba"
"aabaab"
"aababa"
"aabbba"
"abaaab"
"abaaba"
"ababaa"
"abbbaa"
"baaaab"
"baaaba"
"baabaa"
"babaaa"
"bbaaaa"
END
```

Sample Input 6

```
Normal
CSG
Start symbol: A
Productions:
A ->
A -> X
X -> XX
X -> aab
X -> aba
X -> baa
END
6
```

Sample Output 6

```
" "
"aab"
"aba"
"baa"
"aabaab"
"aababa"
"aabbbaa"
"abaaab"
"abaaba"
"ababaa"
"baaaab"
"baaaba"
"baabaa"
END
```

Sample Input 7

```
Normal
CSG
Start symbol: A
Productions:
A ->
A -> X
X -> XaY
X -> aY
X -> YaX
X -> Ya
Y -> XaXbX
Y -> aXbX
Y -> XabX
Y -> XaXb
Y -> abX
Y -> aXb
Y -> abX
Y -> ab
Y -> XbXaX
Y -> bXaX
Y -> XbaX
Y -> XbXa
Y -> baX
Y -> bXa
Y -> baX
Y -> ba
Y -> YY
END
7
```

Sample Output 7

```
" "
"aab"
"aba"
"baa"
"aabab"
"aabba"
"abaab"
"ababa"
"abbba"
"baaba"
"babaa"
"aaaabb"
"aaabab"
"aaabba"
"aabaab"
"aababa"
"aabbbaa"
"abaaab"
"abaaba"
"ababaa"
"abbbaa"
"baaaab"
"baaaba"
"baabaa"
"babaaa"
"bbaaaa"
"aababab"
"aababba"
"aabbbaab"
"aabbaba"
"abaabab"
"abaabba"
"ababaab"
"abababa"
"ababbba"
"abbaaba"
"abbabaa"
"baababa"
"baabbba"
"babaaba"
"bababaa"
END
```