Programmierkurs Python II, Sommersemester 2012, Übungsblatt 7

1 Chomsky-Normalform (2 + 1 Punkte)

Implementiere eine Funktion `cnf(rules)`, die eine Grammatik als Eingabe nimmt und eine äquivalente Grammatik in Chomsky-Normalform zurückgibt (die nur Regeln enthält, deren rechte Seite entweder genau zwei Nichtterminale oder genau ein Terminalsymbol enthält).

Folgende Fälle sollen betrachtet werden:

1. Kettenregeln (A $\rightarrow$ B)
2. Regeln mit mehr als 2 Nichtterminalen auf der rechten Seite (A $\rightarrow$ B C D)
3. Bonus: Tilgungsregeln (A $\rightarrow \epsilon$)

Die Eingabegrammatik soll als Liste von Regeln repräsentiert werden, und Regeln als 2-stelliges Tupel, dessen erstes Element die linke Seite (ein String) und dessen zweites Element die rechte Seite (eine Liste von Strings) der Regel darstellen.

Der Einfachheit halber kannst Du davon ausgehen, dass das eigentliche Lexikon (Regeln wie DET $\rightarrow$ der) vom Rest der Grammatik getrennt wurde, in den Regeln selbst also keine Terminal-Symbole vorkommen.

2 CYK-Erkenner (3 + 2 Punkte)

Implementiere den CYK-Algorithmus als Erkenner.

Die Grammatik soll im gleichen Format angegeben werden wie in Aufgabe 1. Implementiere eine Methode `recognize_cyk(sentence, rules, lexicon)`, die als Eingabe einen Satz, eine Grammatik (Regeln) und ein Lexikon nimmt und `True` zurückgibt, falls der Eingabesatz in der Sprache der Grammatik enthalten ist (sonst `False`).

Bonus: Erweitere die Implementierung bzw. den Algorithmus, so dass auch Grammatiken mit Kettenregeln der (A $\rightarrow$ B) direkt verwendet werden können.

3 CYK-Parser (3 Punkte, Bonus)

Erweitere die Implementierung aus Aufgabe 2 zu einem Parser. Der Parser soll alle Ableitungsbäume für den Eingabesatz als Ausgabe liefern.

Abgabe bis Donnerstag, 2012-06-28, 08:30 Uhr