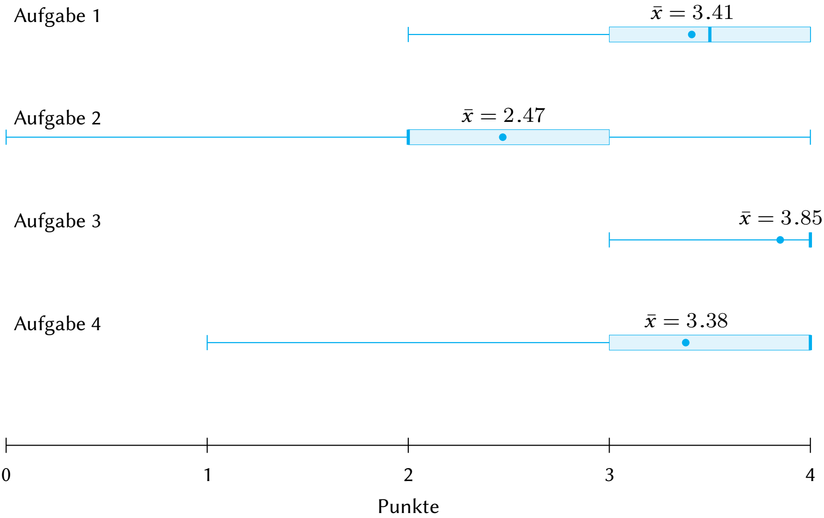
Theoretische Grundlagen der Informatik

Tutorium 11 — Wintersemester 2013/14

Moritz Klammler — 12. November 2013

1. Übungsblatt (16 gültige Abgaben)



Anmerkungen zum 1. Übungsblatt

- Übungsblätter bitte sorgfältig aufbewahren (Beweismittel für Euch).
- Bitte Platz für Korrektur lassen.
- Deckblatt wäre nett – zB für Punktesumme...
- Legt besonderen Wert auf **korrekte Beweisführung**.

Tagesthemen

- Beweistechniken
- Pumping-Lemma für reguläre Sprachen
- Kontextfreie Grammatiken
  - Chomsky-Normalform
  - Cocke-Younger-Kasami-Algorithmus

Pumping-Lemma für reguläre Sprachen

Sei Σ ein Alphabet und $L \subset \Sigma^*$ eine reguläre Sprache.

Dann existiert ein $p \in \mathbb{N}$ sodass für alle $w \in L$ mit $|w| \geq p$ gilt: es existieren $x, y, z \in \Sigma^*$, sodass

1. $w = xyz$,
2. $y \neq \epsilon$,
3. $|xy| \leq p$ und
4. $\forall i \in \mathbb{N}_0 : xy^i z \in L$.

Chomsky-Normalform (CNF)

Sei $G = (T, N, S, P)$ eine kontextfreie Grammatik. G is in CNF genau dann wenn für alle $(\alpha, \beta) \in P$ gilt

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

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 ∈ T* mit |w| = n
;; Ausgabe: Aussage ob w ∈ L(G)
FOR i ← 1,...,n
    table[i][1] ← ∅
    FOREACH (l,r) ∈ P
        IF r = w[i]
            table[i][1].add(l)
FOR j ← 2,...,n
    FOR i ← 1,...,n − j
        table[i][j] ← ∅
        FOR k ← 1,...,j
            FOREACH (l,(r1,r2)) ∈ P ;; nicht−terminierende Produktionen
                IF r1 ∈ table[i][k] ∧ r2 ∈ table[i + k][j − k]
                    table[i][j].add(l)
RETURN S ∈ table[1][n]
```