# Zusammenfassung zur Vorlesung Basismodul Computerlinguistik

**Perfektes Hashing**

Speicherung von Zusatzinformationen für einzelne Lexikonwörter

02.12.2021

# Zusatzinformationen Speichern

**Recap:**

- Mittels Daciuk-Algorithmus wird ein Lexikon in einem minimalen Lexikonautomaten gespeichert.

**Aber:** In der Regel wird ein Lexikon dazu noch benötigt, die Zusatzinformationen zu einzelnen Wörtern zu speichern,

z.B. wollen wir die **Frequenz** und den **Part of Speech** der Wörter speichern.

| **WORT** | **FREQUENZ** | **POS** |
|---|---|---|
| Apfel | 23 | NN |
| spannend | 12 | ADJ |
| arbeiten | 18 | VB |
| ... | ... | ... |

# Hashing

- **Hashing** ist eine Methode, um Schlüssel-Wert-Paare zu speichern und wieder auf sie zuzugreifen.
- Eine **Hash-Funktion** h(k) rechnet aus dem **Schlüssel** die zugehörige **Zeile (Hash-Wert)** aus.
- Eine **Tabelle oder Matrix** enthält die Informationen zeilenweise gespeichert:

| **Zeile (Hash-Wert)** | **Schlüssel (Wort)** | **Wert (Information)** |
|---|---|---|
| $0 \leftarrow h(Apfel)$ | Apfel | (23, NN) |
| ... | ... | |
| $7 \leftarrow h(arbeiten)$ | arbeiten | (18, VB) |
| ... | ... | ... |
| $16 \leftarrow h(spannend)$ | spannend | (12, ADJ) |
| ... | ... | ... |

Table: Beispiel Hash-Tabelle

# Perfektes Hashing

## Was ist Perfektes Hashing?

- Jedem Wort muss nur ein eindeutiger Hash-Wert zugeteilt werden.
- Hash-Tabelle darf keine Lücke haben.

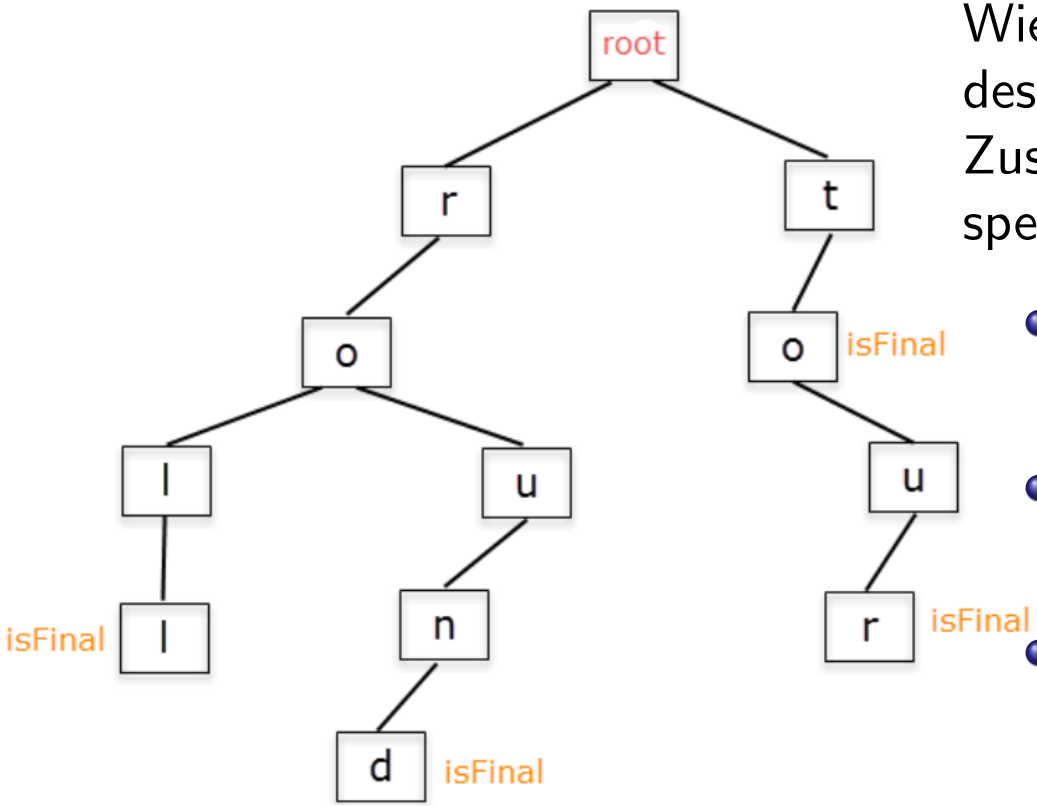
Beispiel für eine Hash-Funktion für **Wörter**: $h(wort) = len(word)$

| **Zeile (Hash-Wert)** | **Schlüssel (Wort)** | **Wert (Information)** |
|---|---|---|
| 0 | | |
| ... | ... | |
| 5 | Apfel | (23, NN) |
| ... | ... | ... |
| 8 | **arbeiten, spannend** (**Kollision!**) | (18, VB), (12, ADJ) |

- Diese Hash-Funktion ist **nicht perfekt**.

# Perfektes Hashing bei nicht-minimalen Lexikonautomaten

Wir haben einen Lexikonautomaten, in dem jedes Wort an einem eigenen Zustand endet.

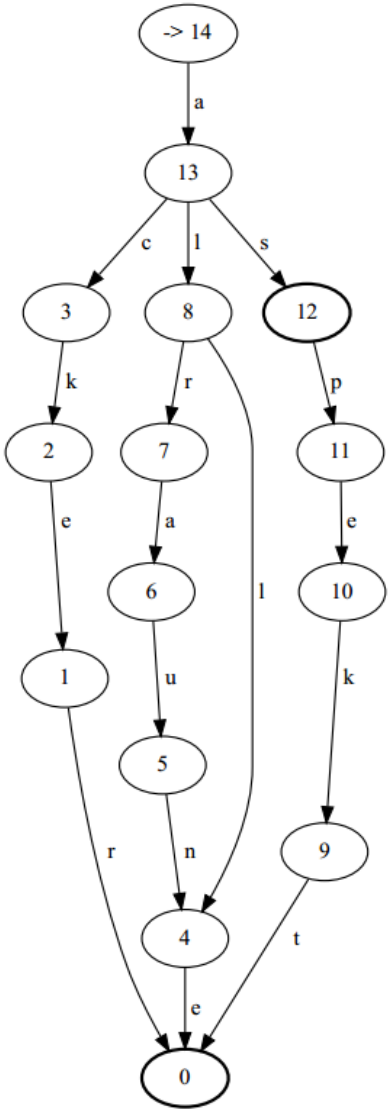



Wie kann man Perfektes Hashing mittels des linken Trie erzeugen und Zusatzinformationen zu jedem Wort speichern?

- Jedes Wort hat einen eigenen Finalzustand.
- Der Finalzustand kann auf die Informationen zeigen.
- So eine Form von Hashing erzeugt: Die Nummer des Finalzustandes ist ein eindeutiger Hash-Wert.

# Perfektes Hashing bei minimalen Lexikonautomaten

In der Vorlesung haben wir anhand Daciuk-Algorithmus einen minimalen Lexikonautomaten aufgebaut.




- **Problem**:
  Wir können jetzt nicht mehr einfach an jedem Finalzustand die Informationen speichern, weil die meisten Wörter auf den gleichen Finalzustand zeigen.
- Wir müssen nun den Hash-Wert des Wortes aus dem Lexikonautomaten rekonstruieren.
- **Ziel**:
  Erzeuge eine perfekte Hash-Funktion, durch die jedes Wort auf seine Position in der sortierten Wortliste abgebildet wird als Hash-Wert.

# Perfektes Hashing mittels Lexikonautomaten

**Grundidee:**
Wenn wir den Hash-Wert für ein Wort $w$ aus dem Lexikonautomaten wollen, dann müssen wir auf dem Weg durch den Automaten **mitzählen**, wieviele Wörter **alphabetisch** vor $w$ existieren.

Dazu werden 2 Kennzahlen eingeführt:

- **Zustandszahl** eines Zustands: die Größe der **Rechtssprache** eines Zustands
- **Label-Zahl** eines Übergangs: die Anzahl der Wörter, die von dem Zustand des Labels aus mit lexikographisch kleineren Labels gelesen werden.

## Zustandszahlen

Sei $z(q)$ die Zustandszahl an Zustand $q$ mit Nachfolgezuständen $p_0$ bis $p_n$.
Sei weiterhin $f(q) = 1$ falls $q$ final ist, ansonsten 0.
Dann ist $z(q) = f(q) + \Sigma_{i=0}^{n} z(p_i)$

## Labelzahlen

Sei $\sigma$ das Label eines Übergangs von Zustand $q$, und sei $\mathcal{T}$ die Menge der Labels der alphabetisch kleineren Übergänge von $q$ auf einen Zustand.
Dann ist die Labelzahl $l(q, \sigma) = f(q) + \Sigma_{\tau \in \mathcal{T}} z(\delta(q, \tau))$

# Berechnung des Hash-Wertes

- Im Lexikonautomaten wird ein Wort durch Labels auf einem Pfad vom **Startzustand** bis zu einem **Finalzustand** repräsentiert.
- Will man den Hash-Wert eines Wortes berechnen, dann soll man nur die **Labelzahlen** des Labels auf dem das Wort repräsentierenden Pfad **mitzählen**.
- Damit ist das Ergebnis genau der **perfekte Hash-Wert**, also die Position des Wortes in der Wortliste.

# Tafelübung

Gegeben sei das sortierte Lexikon $\{abgelaufen, abladen, ablauf, lauf, laufen\}$. Berechne zuerst den minimalen DEA für das Lexikon nach Daciuk-Algorithmus, markiere dann die Labelzahlen für jeden Übergang, die für Perfektes Hashing zur Verfügung gestellt werden.