Orakel-DEA für ein Wort

Orakel-DEA für w

- besitzt als Grundstruktur Pfad mit Beschriftung w,
- enthält für jeden Faktor von w einen Pfad vom Startknoten,
- hat w als einzigen Pfad der Länge $|w|$,
- ist einfach und schnell zu konstruieren.
- Nicht jeder Pfad vom Startknoten entspricht einem Faktor.
- Anwendung: Schnelle Suche nach dem Wort P mittels Orakel-Automat für P^r.

Konstruktion des Orakel-DEA für ein Wort

Gegeben sei $w \in \Sigma^*$ mit $|w| = m$.

- Konstruiere iterativ die Orakel-DEA für die Präfixe von w.
- Orakel-DEA für ε: Knoten 0, keine Kanten.
- i-te Erweiterung (von $w[1 \dots i-1]$ zu $w[1 \dots i]$):
  1. Füge Knoten i und Kante $(i-1, w[i], i)$ hinzu.
  2. Füge Kanten mit Beschriftung $w[i]$ zum Knoten i derart hinzu, dass für jedes Suffix von $w[1 \dots i]$ ein Pfad von 0 existiert.
- Zur effizienten Ausführung von Schritt 2:
  Benutze Knoten $S[i]$, wobei $S[i]$ das Ende des längsten Suffixes von $w[1 \dots i]$ ist, dessen Pfad nicht in i endet.
  $S[i]$ wird während der i-ten Erweiterung bestimmt.
  Definition: $S[0] := -1$.

Algorithmus: Orakel-DEA für ein Wort

Algorithmus 1 Konstruktion des Orakel-DEA für ein Wort

Eingabe: Wort $w \in \Sigma^*$, $|w| = m$
Ausgabe: Orakel-DEA für w

$Z \leftarrow \{0\}$; $\delta \leftarrow \emptyset$; $S[0] \leftarrow -1$;
for $i \leftarrow 1$ **to** m
    $Z \leftarrow Z \cup \{i\}$; $\delta \leftarrow \delta \cup \{(i-1, w[i], i)\}$;
    $j \leftarrow S[i-1]$;
    while ($j \neq -1$ **and** $\delta(j, w[i])$ nicht definiert)
        $\delta \leftarrow \delta \cup \{(j, w[i], i)\}$; $j \leftarrow S[j]$;
    if $j = -1$ **then** $S[i] \leftarrow 0$;
                else $S[i] \leftarrow \delta(j, w[i])$;
$E \leftarrow \{m\}$; $e \leftarrow S[m]$;
while $e > 0$
    $E \leftarrow E \cup \{e\}$; $e \leftarrow S[e]$;
return $A = (\Sigma, Z, \delta, 0, E)$;

Algorithmus: Suche mit Orakel-DEA

Algorithmus 2 Backward Oracle Matching (BOM-Algorithmus)

Eingabe: Wörter P, T über Σ mit $|P| = m, |T| = n$
Ausgabe: Menge S der Vorkommen von P in T

Konstruiere den Orakel-DEA für P^r $A = (\Sigma, Z, \delta, z_0, F)$;
$S \leftarrow \emptyset$; $z \leftarrow z_0$; $k \leftarrow m$;
while $k \leq n$
  $j \leftarrow k$;
  while $\delta(z, T[j])$ existiert
    $z \leftarrow \delta(z, T[j])$; $j \leftarrow j - 1$;
  if $j = k - m$ **then** $S \leftarrow S \cup \{k - m + 1\}$; $k \leftarrow k + 1$;
    else $k \leftarrow j + m$;
return S;

Orakel-DEA für eine Menge von Wörtern

Orakel-DEA für $\mathcal{W} = \{w_1, w_2, \ldots, w_k\}$ mit
$|w_1| = |w_2| = \ldots = |w_k| = m$

- besitzt als Grundstruktur $\mathit{Trie}(w_1, \ldots, w_k)$.
- enthält für jeden Faktor von $\mathcal{W}$ einen Pfad vom Startknoten,
- hat die Wörter aus $\mathcal{W}$ als einzige Pfade der Länge m.
- Nicht jeder Pfad vom Startknoten entspricht einem Faktor.
- Konstruktion analog zu Orakel-DEA für ein Wort.
- Zur Suche nach einer Menge $\mathcal{P} = \{P_1, P_2, \ldots, P_k\}$ mit kürzester Länge μ benutze Orakel-Automat für die Präfixe der Länge μ von P_i^r, $1 \leq i \leq k$.

Algorithmus: Orakel-DEA für eine Menge von Wörtern

Algorithmus 3 Konstruktion des Orakel-DEA für eine Menge

Eingabe: Menge $\mathcal{W} = \{w_1, \ldots, w_k\} \subset \Sigma^*$, $|w| = m$ für $w \in \mathcal{W}$

Ausgabe: Orakel-DEA für $\mathcal{W}$

(1) Konstruiere $\mathit{Trie}(\mathcal{W})$ mit Knotenmenge $Z = \{0, 1, \ldots, M\}$ in BFS-Ordnung und Kantenmenge δ; $S[0] \leftarrow -1$;

(2) **for** $i \leftarrow 1$ **to** M

(3) $j \leftarrow S[i-1]$;

(4) **while** ($j \neq -1$ **and** $\delta(j, w[i])$ nicht definiert)

(5) $\delta \leftarrow \delta \cup \{(j, w[i], i)\}$; $j \leftarrow S[j]$;

(6) **if** $j = -1$ **then** $S[i] \leftarrow 0$;

(7) **else** $S[i] \leftarrow \delta(j, w[i])$;

(8) $E \leftarrow \emptyset$;

(9) **foreach** Knoten f der Tiefe m

(10) $e \leftarrow f$;

(11) **while** $e > 0$

(12) $E \leftarrow E \cup \{e\}$; $e \leftarrow S[e]$;

(13) **return** $A = (\Sigma, Z, \delta, 0, E)$;

Suche mit Orakel-DEA für Mengen

Set Backward Oracle Matching (SBOM-Algorithmus)

Suche $\mathcal{P} = \{P_1, P_2, \ldots, P_r\}$, wobei $\min\{|P_i| : 1 \leq i \leq r\} = \mu$.

Konstruiere Orakel-DEA für $\{w_1, w_2, \ldots, w_r\}$ mit $w_i = P_i^r[1 \ldots \mu]$.
$A = (\Sigma, Z, \delta, z_0, E)$.

Suche nach Vorkommen mit Ende an Position k:

(1) $j \leftarrow k$; $z \leftarrow z_0$;
(2) **while** $\delta(z, T[j])$ existiert
(3) $\quad z \leftarrow \delta(z, T[j])$; $j \leftarrow j - 1$;
(4) **if** $j > k - \mu$ **then** $k \leftarrow j + \mu$;
(5) **else** weiter prüfen (Suffix der Länge μ kommt vor);

Für das Überprüfen kann man $\mathit{Trie}(P_1^r, P_2^r, \ldots, P_r^r)$ nutzen.