Eigenschaften und Minimierung von Baumautomaten

Seminar Automata Theory

Frank Schüssele

Definitionen

- Endlicher Baumautomat: $A = (Q, F, Q_f, \Delta)$
  - Q: Zustände
  - F: Funktionssymbole
  - $Q_f \subseteq Q$: Finale Zustände
  - Δ: Transitionen der Form $f(q_1, ...q_n) \to q$ $(f \in F_n, q, q_1, ...q_n \in Q)$
- A ist deterministisch (DFTA), falls es $\forall f \in F_n, q_1, ..., q_n \in Q$ höchstens ein Regel $f(q_1, ..., q_n) \to q \in \Delta$ gibt, ansonsten nicht-deterministisch (NFTA)
- A ist vollständig, falls es $\forall f \in F_n, q_1, ..., q_n \in Q$ mindestens eine Regel $f(q_1, ..., q_n) \to q \in \Delta$ gibt
- A heißt reduziert, falls alle Zustände im Automaten erreichbar sind

Definitionen

- Für zwei Automaten $A_1 = (Q_1, F, Q_{f1}, \Delta_1)$ und $A_2 = (Q_2, F, Q_{f2}, \Delta_2)$ definiere:

- $$\Delta_1 \times \Delta_2 := \{f((q_1, q_1'), ...(q_n, q_n')) \to (q, q') \mid \\ f(q_1, ..., q_n) \to q \ \in \Delta_1 \wedge \\ f(q_1', ..., q_n') \to q' \in \Delta_2\}$$

Äquivalenz DFTA / NFTA

- DFTA und NFTA gleich mächtig

$\Rightarrow$ Determinisierung möglich
(Algorithmus ähnlich zur Potenzmengenkonstruktion für NEA's)

Abgeschlossenheit

Theorem:
Die Menge der akzeptierbaren Baumsprachen ist bezüglich Vereinigung, Schnitt und Komplement abschlossen.

Abgeschlossenheit - Komplement

- Gegeben ein DTFA $A = (Q, F, Q_f, \Delta)$
  (NFTA muss zuerst determinisiert werden)
- Konstruiere $A' = (Q, F, Q \setminus Q_f, \Delta)$

$\Rightarrow L(A') = \overline{L(A)}$

Abgeschlossenheit - Vereinigung

- Gegeben $A_1 = (Q_1, F, Q_{f1}, \Delta_1)$ und $A_2 = (Q_2, F, Q_{f2}, \Delta_2)$
- Falls beide vollständig, konstruiere:
  $A = (Q_1 \times Q_2, F, Q_{f1} \times Q_2 \cup Q_1 \times Q_{f2}, \Delta_1 \times \Delta_2)$
  (deterministisch falls A_1 und A_2 deterministisch)
- Ansonsten, konstruiere: $A' = (Q_1 \cup Q_2, F, Q_{f1} \cup Q_{f2}, \Delta_1 \cup \Delta_2)$

$\Rightarrow L(A) = L(A') = L(A_1) \cup L(A_2)$

Abgeschlossenheit - Schnitt

- Gegeben $A_1 = (Q_1, F, Q_{f1}, \Delta_1)$ und $A_2 = (Q_2, F, Q_{f2}, \Delta_2)$
- Konstruiere $A = (Q_1 \times Q_2, F, Q_{f1} \times Q_{f2}, \Delta_1 \times \Delta_2)$

$\Rightarrow L(A) = L(A_1) \cap L(A_2)$

Abgeschlossenheit - Beispiel

- Gegeben $A_1 = (Q_1, F, Q_{f1}, \Delta_1)$ und $A_2 = (Q_2, F, Q_{f2}, \Delta_2)$, mit
  - $F = \{a, f(), g()\}$
  - $Q_1 = \{q, q_f\}$
  - $Q_{f1} = \{q_f\}$
  - $\Delta_1 = \{a \to q, f(q) \to q_f, f(q_f) \to q_f, g(q) \to q, g(q_f) \to q\}$
  - $Q_2 = \{q', q'_f\}$
  - $Q_{f2} = \{q'_f\}$
  - $\Delta_2 = \{a \to q', f(q') \to q', f(q'_f) \to q'_f, g(q') \to q'_f,$
    $g(q'_f) \to q'_f\}$

Abgeschlossenheit - Beispiel

- $Q_1 \times Q_2 = \{(q, q'), (q, q'_f), (q_f, q'), (q_f, q'_f)\}$
- $\Delta_1 \times \Delta_2 = \{a \to (q, q'), f((q, q')) \to (q_f, q'),$
  $f((q_f, q')) \to (q_f, q'), f((q, q'_f)) \to (q_f, q'_f),$
  $f((q_f, q'_f)) \to (q_f, q'_f), g((q, q')) \to (q, q'_f),$
  $g((q, q'_f)) \to (q, q'_f), g((q_f, q')) \to (q, q'_f),$
  $g((q_f, q'_f)) \to (q, q'_f)\}$
- $Q_{f1} \times Q_{f2} = \{(q_f, q'_f)\}$
- $Q_{f1} \times Q_2 \cup Q_1 \times Q_{f2} = \{(q, q'_f), (q_f, q'), (q_f, q'_f)\}$

- $A_\cup = (Q_1 \times Q_2, F, Q_{f1} \times Q_2 \cup Q_1 \times Q_{f2}, \Delta_1 \times \Delta_2)$
- $A_\cap = (Q_1 \times Q_2, F, Q_{f1} \times Q_{f2}, \Delta_1 \times \Delta_2)$

Probleme und Komplexität

- Größe eines Baumautomaten $A = (Q, F, Q_f, \Delta)$:

$$||A|| := |Q| + \sum_{f(q_1,\ldots q_n)\to q \in \Delta} (n+2)$$

- Größe eines Terms t:
  $||t||$: Anzahl der Knoten im Baum

Probleme und Komplexität - Wortproblem

Problem: Gegeben ein Baumautomat A und ein Term t.
Wird t von A akzeptiert?

- A deterministisch: ein Lauf muss überprüft werden
  $\Rightarrow$ entscheidbar in $O(||A|| + ||t||)$ (Linearzeit)

- A nicht-deterministisch: Berechnung erreichbarer Zustände, Vergleich am Ende mit finalen Zuständen
  $\Rightarrow$ entscheidbar in $O(||A|| \cdot ||t||)$ (polynomielle Zeit)

Probleme und Komplexität - Leerheitsproblem

Problem: Gegeben ein Baumautomat A. Ist die Sprache, die A erkennt, leer?

- Reduziere A (nicht erreichbare Zustände entfernen), Laufzeit $O(||A||)$
- Dann gilt: $L(A) = \emptyset \Leftrightarrow Q_f = \emptyset$

$\Rightarrow$ entscheidbar in $O(||A||)$

Probleme und Komplexität - Äquivalenz

Problem: Gegeben zwei DFTA's A_1 und A_2. Gilt $L(A_1) = L(A_2)$?

- Es gilt: $L(A_1) = L(A_2)$
  $\Leftrightarrow \overline{L(A_1)} \cap L(A_2) = \emptyset \wedge \overline{L(A_2)} \cap L(A_1) = \emptyset$
- Konstruiere Automaten, die $\overline{L(A_1)} \cap L(A_2)$ und $\overline{L(A_2)} \cap L(A_1)$ erkennen und überprüfe auf Leerheit

$\Rightarrow$ entscheidbar in $O(||A_1|| \cdot ||A_2||)$

(Für NFTA's in exponentieller Zeit entscheidbar, wegen Determinisierung)

Minimierung - Algorithmus

Data: a complete and reduced DTFA $A = (Q, F, Q_f, \delta)$
$P' := \{Q_f, Q \setminus Q_f\}$
repeat
  $P := P'$
  // Refine the relation P'
  $qP'q'$ if qPq' and $\forall f \in F_n : \forall q_1, ..., q_{i-1}, q_{i+1}, ..., q_n \in Q :$
  $\delta(f(q_1, ..., q_{i-1}, q, q_{i+1}, ..., q_n))P\delta(f(q_1, ..., q_{i-1}, q', q_{i+1}, ..., q_n))$
until $P = P'$;
$Q_{min} := \{[q] \mid q \in Q\}$ (equivalence classes of P)
$\delta_{min} := \{f([q_1], ...[q_n]) \to [q] \mid f(q_1, ...q_n) \to q \in \delta\}$
$Q_{min_f} := \{[q] \mid q \in Q_f\}$

Result: the minimum DFTA of A: $A_{min} = (Q_{min}, F, Q_{min_f}, \delta_{min})$

Minimierung - Markierungsalgorithmus DEA

- Zuerst äquivalente Zustände finden und zusammenführen
- Ähnlich wie bei Markierungsalgorithmus für DEA's

- Markiere zuerst alle Zustandspaare $\{q, q_f\}$ mit $q \notin Q_f, q_f \in Q_f$
- $\{q, q'\}$ wird markiert, falls gilt:
  $\exists a \in \Sigma$:
  $\{\delta(q, a), \delta(q', a)\}$
  ist markiert
- Wiederhole, bis sich nichts mehr ändert
- Nicht-markierte Zustandspaare können zusammengefasst werden

Minimierung - Markierungsalgorithmus DFTA

- Zuerst äquivalente Zustände finden und zusammenführen
- Ähnlich wie bei Markierungsalgorithmus für DEA's

- Markiere zuerst alle Zustandspaare $\{q, q_f\}$ mit $q \notin Q_f, q_f \in Q_f$
- $\{q, q'\}$ wird markiert, falls gilt:
  $\exists f \in F_n, q_1, ..., q_{i-1}, q_{i+1}, ..., q_n \in Q :$
  $\{\delta(f(q_1, ..., q_{i-1}, q, q_{i+1}, ..., q_n)), \delta(f(q_1, ..., q_{i-1}, q', q_{i+1}, ..., q_n))\}$
  ist markiert
- Wiederhole, bis sich nichts mehr ändert
- Nicht-markierte Zustandspaare können zusammengefasst werden

Minimierung - Beispiel

- Gegeben $A = (Q, F, Q_f, \Delta)$ mit
  - $F = \{a, b, c, f(,)\}$
  - $Q = \{q_1, q_2, q_3\}$
  - $Q_f = \{q_3\}$
  - $\Delta = \{a \to q_1, b \to q_2, c \to q_3, f(q_1, q_1) \to q_1, f(q_2, q_2) \to q_2,$
    $f(q_3, q_3) \to q_3, f(q_1, q_2) \to q_1,$
    $f(q_2, q_1) \to q_2, f(q_1, q_3) \to q_1,$
    $f(q_3, q_1) \to q_3, f(q_2, q_3) \to q_2,$
    $f(q_3, q_2) \to q_3\}$

Minimierung - Beispiel

- Es gilt: $[q_1] = [q_2] = \{q_1, q_2\}$ und $[q_3] = \{q_3\}$

  $\Rightarrow A_{min} = (Q_{min}, F, Q_{min_f}, \delta_{min})$ mit
  - $Q = \{[q_1], [q_3]\}$
  - $Q_f = \{[q_3]\}$
  - $\Delta = \{a \to [q_1], b \to [q_2], c \to [q_3], f([q_1], [q_1]) \to [q_1],$
    $f([q_3], [q_3]) \to [q_3], f([q_1], [q_3]) \to [q_1],$
    $f([q_3], [q_1]) \to [q_3]\}$

Quellen


H. Comon, M. Dauchet, R. Gilleron, C. Löding, F. Jacquemard, D. Lugiez, S. Tison, and M. Tommasi, *Tree automata techniques and applications*, Available on: `http://www.grappa.univ-lille3.fr/tata`, 2008, release November, 18th 2008.