Aktuelles aus der Theorie Endlicher Automaten

10. Theorietag der Fachgruppe Automaten und Formale Sprachen der GI

Wien, September 2000

1 Zustandskomplexität von DFA/NFA–Operationen

Für $L \in \mathrm{REG}$ ist $D(L)$ die Zustandszahl des minimalen deterministischen Automaten für L, und $N(L)$ die eines kleinsten nichtdeterministischen Automaten für L.

1.1 NFA und DFA

Wir wissen, daß $D(L)$ höchstens exponentiell größer sein kann als $N(L)$, da wir einen DFA für L aus der Potenzmengenkonstruktion erhalten, und diese $2^{N(L)}$ Zustände erzeugt.

Andererseits wird diese Schranke auch erreicht: für $\Sigma = \{0, 1\}$ setzen wir

$$L_n = \Sigma^* 0 \Sigma^n$$

d. h. das n–letzte Zeichen muß eine 0 sein. Ein NFA für L kann nun einfach raten, daß er gerade das n–letzte Zeichen der Eingabe gelesen hat. Dazu braucht er n Zustände. Der DFA muß sich aber immer die n zuletzt gelesenen Zeichen im Gedächtnis behalten, damit er nicht vom plötzlichen Wort–Ende überrascht wird. Er braucht dazu 2^n Zustände.

Es gibt auch Sprachen, die für DFA und NFA gleich schwer (gleich einfach) sind: wir benutzen im Folgenden die unären Sprachen

$$T_p = (a^p)^*$$

d. h. die Wort–Länge ist durch p teilbar. Für diese gilt $D(T_p) = N(T_p) = p$.

1.2 NFA: Vereinigung und Durchschnitt

Für beliebige Sprachen gilt sicher

$$N(L_1 \cup L_2) \leq N(L_1) + N(L_2)$$

Wir können nämlich einen NFA bauen, der zunächst rät, ob er L_1 oder L_2 lesen will, und dann entsprechend fortsetzt. (Für DFA ist die Vereinigung im Allgemeinen teuer, siehe unten.)

Den Durchschnitt zweier Automaten bestimmen wir nach der Kreuzprodukt–Konstruktion. Damit gilt

$$N(L_1 \cap L_2) \leq N(L_1) \cdot N(L_2)$$

Wenn wir so zwei DFA multiplizieren, erhalten wir wieder einen DFA, also

$$D(L_1 \cap L_2) \leq D(L_1) \cdot D(L_2)$$

Wenn p und q verschiedene Primzahlen sind, kann man zeigen, daß die Schranken bei $T_p \cap T_q$ tatsächlich erreicht werden.

Die Kreuzprodukt–Konstruktion können wir ebenfalls benutzen, um die Vereinigung zweier DFA zu bestimmen (wir wählen dabei lediglich die akzeptierenden Zustände anders als für den Durchschnitt). Damit folgt

$$D(L_1 \cup L_2) \leq D(L_1) \cdot D(L_2)$$

1.3 NFA: Komplement

Wenn $n = N(L)$, wie groß kann $D(L)$ werden? Wir können den DFA für L bestimmen, er hat die Größe $\leq 2^n$, und dort akzeptierende und ablehnende Zustände vertauschen. Deswegen

$$N(\overline{L}) \leq D(\overline{L}) \leq 2^{N(L)}$$

Das funktioniert z. B. für die oben erwähnte Sprachen L_n.

Allerdings sind die NFA für das Komplement wieder sehr klein.

Mit der folgenden Konstruktion kommt man weiter:

Für paarweise verschiedene Primzahlen $p_1, \ldots, p_n$ setzen wir

$$L = T_{p_1} \cup \ldots \cup T_{p_n}$$

d. h. die Wortlänge ist durch irgendeine dieser Primzahlen teilbar. Wir haben $N(L) = p_1 + \ldots + p_n$ nach der offensichtlichen Konstruktion.

Das Komplement von L ist

$$\overline{L} = \overline{T_{p_1}} \cap \ldots \cap \overline{T_{p_n}}$$

Die einzelnen NFA haben die Größe p_i, und nach der Kreuzprodukt–Konstruktion ist

$$N(\overline{L}) = p_1 \cdot \ldots \cdot p_n$$

Man muß dazu natürlich zeigen, daß das nicht besser geht. Das ist machbar.

Jetzt nehmen wir an, wir haben z Primzahlen, und alle sind von der Größe $z \log z$. (Besser geht es aus zahlentheoretischen Gründen nicht.)

$$n = N(L) = z^2 \log z; \overline{n} = N(\overline{L}) = z^z \log^z z$$

Wir wollen $\overline{n}$ als Funktion von n darstellen: mit $z \approx \sqrt{n}$ gilt

$$\overline{n} = (n/z)^z \approx \sqrt{n}^{\sqrt{n}} \approx 2^{\sqrt{n} \log n}$$

Das ist zwar mehr als jedes Polynom, aber noch nicht 2^n.

1.4 Offene Fragen

1. Zeige für unäre Sprachen, daß das wirklich nicht besser geht (d. h. 2^n ist in diesem Fall unerreichbar).

2. Finde eine binäre Sprache mit $N(\overline{L}) = 2^{\theta(n)}$.

2 REG $\subseteq$ CRCL?

Eine Church–Rosser–Sprache $L \subseteq \Sigma^*$ ist gegeben durch ein endliches, längenverkürzendes, konfluentes Wortersetzungssystem R auf einem Alphabet $\Gamma \supseteq \Sigma$, und zwei Wörtern $l, r \in \Gamma^*$ sowie einen Buchstaben $Y \in \Gamma \setminus \Sigma$, so daß

$$\forall w \in \Sigma^* : w \in L \iff w \to_R^* Y$$

Der Begriff stammt von McNaughton, Narendran und Otto 1988. (Literatur hier: `http://elba.db.informatik.uni-kassel.de/~niemann/research.html`.)

Im Allgemeinen beschreibt er deterministische (wegen der Konfluenz), wachsende (wegen der Längenverkürzung) kontextsensitive (wegen l und r) Sprachen.

Eine stark eingeschränkte Klasse sind Church–Rosser–Kongruenz–Sprachen. Das sind Vereinigungen von Kongruenzklassen von längenverkürzenden, konfluenten Wort–Ersetzungs–Systemen.

$L \in$ CRCL, wenn es ein System $R = \{(u_1, v_1), \ldots, (u_n, v_n)\}$ mit $\forall i : |u_i| > |v_i|$ (d. h. R terminiert) und R konfluent (d. h. jedes Wort hat bzgl. R genau eine Normalform) und eine endliche Menge M von R–Normalformen gibt, so daß

$$\forall w \in \Sigma^* : w \in L \iff \exists m \in M : w \to_R^* m$$

Hier gibt es also kein zusätzliches Alphabet Γ und auch keine Markierungen l, r.

2.1 Beispiele

Wir fixieren das Alphabet $\Sigma = \{a, b\}$.

Die Sprache a^{2*} ist in CRCL. Wir setzen $R = \{aa \to \epsilon\}$ und $M = \{\epsilon\}$.

Lokale Konfluenz prüft man leicht: es gibt nur ein kritisches Paar aaa, aber dieses ist konfluent. Wegen Längenverkürzung haben wir Termination, also globale Konfluenz.

Die Sprache $\{a^n b^n \mid n \geq 0\}$ ist in CRCL, wie man am System $\{aabb \to ab\}$ und den ausgezeichneten Normalformen $\{\epsilon, ab\}$ sieht. Übung: warum geht $\{ab \to \epsilon\}$ nicht?

Wir betrachten nun die Sprachen

$$L(p, q) = (\Sigma_p^q)^* \quad \text{mit} \quad \Sigma_p = \{a_1, \ldots, a_p\}$$

d. h. ein Alphabet mit p Buchstaben, und die Wortlänge soll durch q teilbar sein. Das Beispiel oben war Ł$(1, 2)$.

Für $L(2, 2)$ finden wir nach längerem Suchen das Ersetzungssystem

$$aa \to \epsilon, bab \to b, bbb \to b$$

das ist offensichtlich längenverkürzend, und (nicht so ganz offensichtlich) konfluent. Es hat nur endlich viele Normalformen

$$\{\epsilon, a, b, ab, ba, bb, aba, abb, bba, abba\}$$

von denen wir die mit gerader Länge auswählen.

2.2 Offene Fragen

Zeige, daß $L(3, 2)$ und $L(2, 3)$ in CRCL sind.

Zeige, daß jede reguläre Sprache in CRCL ist.

3 Zählen in Bildern

Grundsätzliche Definitionen: Giammarresi, Restivo, Seibert und Thomas. Dieses Resultat: Klaus Reinhardt, `http://www-fs.informatik.uni-tuebingen.de/~reinhard/picgen.html`.

Wir betrachten Bilder $b : \mathbb{Z}^2 \to \Sigma$.

Ein Bild–Automat ist eine endliche Menge M von endlichen Bausteinen mit Markierungen in einem Zustandsalphabet Γ.

Ein Automat akzeptiert ein Bild, wenn es sich komplett mit Bausteinen aus M überdecken läßt.

Eine Bildsprache heißt lokal, wenn sie so akzeptiert werden kann.

Eine Bildsprache heißt erkennbar, wenn sie homomorphes Bild einer lokalen Sprache ist.

Da es um endliche Bilder gehen soll, müssen wir deren Ränder mit speziellen Symbolen markieren. Das ist aber hier nicht wesentlich.

Beispiel: Die Sprache aller Rechtecke der Seitenlängen $n \times 2^n$ ist erkennbar. Wir können im Innern des Rechtecks binär zählen, und schließlich die Zählerbits durch den Morphismus löschen.

3.1 Resultate

Lokale Sprachen sind genau die durch Sätze erster Ordnung beschreibbaren, und erkennbare Sprachen gestatten Existentielle Logik zweiter Ordnung.

Es läßt sich zeigen, daß für die lokalen Sprachen bereits Muster der Größe 2×1 und 1×2 ausreichen.

Beschränken wir uns auf Bilder der Breite 1, erhalten wir als erkennbare Sprachen offensichtlich genau die regulären Wortsprachen. (Wir raten einfach die Zustandsfolgen einer akzeptierenden Berechnung.)

Betrachten wir die untere Kante beliebiger Bilder, dann erhalten wir alle aufzählbaren Sprachen. Wir können nämlich in einem Bild die Rechnung einer Turingmaschine simulieren: jede Zeile entspricht einer Konfiguration, und von oben nach unten läuft die Zeit. Die Korrektheit der Rechnung läßt sich lokal verifizieren.

3.2 Zähler

Die Menge der Bilder, an deren unteren Kante gleich viele a wie b stehen, ist erkennbar. Wir können auf sehr anschauliche Weise einen Ableitungsbaum einer kontextfreien Grammatik nachbilden. Wir zeichnen Verbindungen zwischen zusammengehörenden Klammernpaaren.

Die Frage ist, ob die Menge der rechteckigen Bilder, die als Ganzes genausoviele a wie b enthalten, akzeptierbar ist.

Wir geben zunächst für die Kantensprache einen geschickteren Akzeptor: der Abstand einer Strecke von der Kante kodiert die Anzahl der noch zu schließenden Klammern.

Diese Konstruktion übertragen wir in die Ebene und durchlaufen dabei ein Quadrat mit einer Hilbertkurve, tatsächlich mit mehreren, verschieden wertigen. In den Kreuzungen müssen wir geschickt die Information verteilen. Klaus Reinhard hat gezeigt, daß es geht.

4 Endliche Quanten–Automaten (QFA)

Übergangsmatrizen für DFA

Die Übergangsfunktion eines vollständigen DFA mit Zustandsmenge Q und Eingabealphabet Σ können wir auffassen als $\Sigma \to (Q^2 \to \{0,1\})$ d. h. zu jedem Buchstaben $a \in \Sigma$ gibt es eine Matrix M_a mit genau einer 1 in jeder Zeile. Einen Zustand beschreiben wir durch einen Vektor mit genau einer 1 (und sonst 0), und Zustandsübergänge sind dann Matrizenmultiplikationen.

Stochastische Matrizen, Stochastische Automaten

Wir gehen zu stochastischen Automaten und beschreiben Zustände durch eine Wahrscheinlichkeitsverteilung, sowie Übergänge durch stochastische Matrizen (d. h. alle Einträge zwischen 0 und 1 sowie alle Zeilensummen 1). Wir lassen den Automaten mit vorgegebener Startverteilung laufen und bestimmen schließlich den Erwartungswert des Anteils der akzeptierenden Zustände bei Wortende.

Ein stochastischer Automat akzeptiert eine Sprache L mit Schranke λ, wenn jedes Wort in L mit Wahrscheinlichkeit $> \lambda$ akzeptiert wird, und jedes Wort nicht in L mit Wahrscheinlichkeit $\leq \lambda$. Wir können noch einen Abstand $\epsilon > 0$ fordern, d. h. entweder $\lambda + \epsilon$ oder $\lambda - \epsilon$.

Es gibt klassische Resultate (60er Jahre) über die derart akzeptierbaren Sprachen. Sprachen mit Abstand sind immer regulär, aber die Sprachen ohne Abstand können komplizierter sein.

Das liegt ungefähr daran, daß die Wahrscheinlichkeiten Zahlen sind, und diese ziemlich lang werden können (und die allerletzten Stellen immer noch signifikant sind). Es ist also gar kein “endlicher” Automat mehr.

Unitäre Matrizen, QFA

Jetzt nicht Wahrscheinlichkeitsverteilungen, sondern Überlagerungen von Quantenzuständen. Die Übergänge sind dann keine stochastischen, sondern unitäre Matrizen, d. h. die Einträge sind komplexe Zahlen und die l_2–Norm jeder Zeile ist 1.

Wenn wir wieder nur am Wortende die Verteilung messen, bekommen wir ungefähr die gleichen Klassen wie für stochastische Automaten.

Beachten wir dabei, daß die Matrizen immer Inverse besitzen, d. h. Rechnungen sind umkehrbar. (Das ist quantenphysikalisch zu erwarten.)

Jetzt gestatten wir Messungen während der Rechnung. In der Quantenphysik sind aber “reine” Messungen nicht möglich, jede Messung beeinflußt den Systemzustand (projiziert die Wellenfunktion auf einen Unterraum) (und ist damit nicht umkehrbar). Deswegen sehen QFA–Algorithmen etwas umständlich aus.

Quantenphysik und Informationstheorie

Die klassische Vorstellung ist: Informatik wird durch Elektronik implementiert, und ist daher Teil der Physik (jedenfalls die elementarsten Algorithmen werden physikalisch beschrieben).

Die Quantenphysik zeigt: es gibt keine wirkungsfreien Messungen. Der Zustand eines physikalischen Systems kann davon abhängen, wieviel Information aus dem System herausgeholt wird. Damit werden die elementarsten physikalischen Zusammenhänge durch Informationstheorie beschrieben! Das ist praktisch eine Umkehr der gewohnten Verhältnisse.