Komplexität von Erzeugen in Algebren

H. JÜRGENSEN

1. Das Problem

$\mathfrak{A}=(A, F)$ sei eine endliche Algebra; A ist die endliche Trägermenge von $\mathfrak{A}$, und F ist die endliche Menge von Operationszeichen. Die Abbildung $\nu: F \to \mathbf{N}_0 = \mathbf{N} \cup \{0\}$ ordnet jedem Operationszeichen $f \in F$ seine Stelligkeit ν_f zu. Jedes $f \in F$ definiert eine Operation auf A, die ebenfalls mit f bezeichnet sei, d.h., eine Abbildung $f: A^{\nu_f} \to A$. Für $M \subseteq A$ sei

$$F(M) := \{a \mid a \in A \wedge \exists f \in F \exists a_1, \ldots, a_{\nu_f} \in M: a = f(a_1, \ldots, a_{\nu_f})\},$$

und für $G \subseteq F$ und $N \subseteq A$ sei

$$G(M, N) = \left\{ a \,\middle|\, \begin{array}{l} a \in A \wedge \exists f \in G \exists i, 1 \leqq i \leqq \nu_f \exists a_i \in M \\ \exists a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_{\nu_f} \in N: a = f(a_1, \ldots, a_{\nu_f}) \end{array} \right\}.$$

Ferner seien $\mathfrak{M}, \mathfrak{M}'$ die kleinsten Unteralgebren von $\mathfrak{A}$, die M enhalten und gegen F bzw. gegen $G(\cdot, A)$ abgeschlossen sind. $\mathfrak{M}$ ist also die von M erzeugte Unteralgebra; $\mathfrak{M}'$ könnte man als das von M erzeugte G-Ideal bezeichnen.

In der vorliegenden Arbeit untersuchen wir den Aufwand zur Bestimmung von $\mathfrak{M}$ oder $\mathfrak{M}'$ aus $\mathfrak{A}$, M und G. Als Maschinenmodell für die Realisierung der Algorithmen verwenden wir die RAM (=random access machine); diese Wahl erlaubt es, die Kosten einigermaßen realistisch auch für „relativ" kleine Algebren abzuschätzen, weil keine wesentlichen Kosten für die Adreßrechnungen anfallen.

Um das Problem genau zu stellen, müssen wir noch festlegen, wie $\mathfrak{A}$, M und G dargestellt werden: Die Elemente von A können abstrakt, d.h., etwa als natürliche Zahlen $1, 2, \ldots, n := |A|$ gegeben sein oder konkret, etwa als Transformationen einer endlichen Menge. In vielen Fällen ist die Kenntnis von n und $\mathfrak{A}$ für den Erzeugungsalgorithmus irrelevant, in anderen wird man $\mathfrak{A}$ mit $\mathfrak{M}$ identifizieren können. Wir setzen die Existenz eines Programms zur Berechnung einer Bijektion χ von A auf $[1:n] := \{1, \ldots, n\}$ voraus, dessen Kosten mit konstant h_A angesetzt werden. Die Operation $f \in F$ kann sich auf die χ-Bilder der Argumente oder auf die Argumente selbst beziehen. Für f setzen wir konstant die Kosten c_f an. Als typische Realisierung von χ könnte man an ein hash-Verfahren denken mit im wesentlichen linearen Kosten relativ zur Größe der Darstellung der Elemente. Als Realisierung

für f käme einerseits ein Tabellenzugriff in Frage oder andererseits z.B. ein tatsächliches Rechnen mit den konkreten Elementen, etwa den Transformationen.

Wir wollen Platz- und Zeitaufwand verschiedener Verfahren diskutieren; zu diesem Zweck vereinbaren wir, daß die Kostenfunktion Φ eines Problems jeweils die zwei Varianten Φ^p und Φ^z für die Platz- und die Zeitkomplexität hat.

Verwandte Fragen werden in [3, 4, 6] behandelt. Dabei diskutiert [6] allerdings keine Komplexitätsfragen. Die Aufsätze [3, 4], soweit sie thematisch einschlägig sind, arbeiten mit einem anderen Algorithmusbegriff, der es erlaubt die „Buchführungskosten" zu ignorieren. In [7] wird ein Algorithmus zum Erzeugen von Normalteilern in Gruppen angegeben, dessen Aufwand mit $O(n^2)$ abgeschätzt werden kann, wenn n die Gruppenordnung ist; allerdings werden auch bei dieser Abschätzung alle Buchführungskosten vernachlässigt. Weitere einschlägige Arbeiten findet man in der Bibliographie des Übersichtsartikels [5]. Im allgemeinen werden in diese keine Komplexitätsaussagen gemacht. In [2] schließlich wird gezeigt, daß einige natürliche Probleme für Permutationsgruppen in polynomialer Zeit lösbar sind.

2. Die naiven Algorithmen

Die Trägermenge $\langle M\rangle$ von $\mathfrak{M}$ erhält man mit Hilfe der folgenden trivialen Bemerkung:

2.1. Bemerkung. Sei $M_0 := M$ und $M_{i+1} := M_i \cup F(M_i)$ für $i = 0, 1, \ldots$. Es sei k minimal mit $M_{k+1} = M_k$. Dann ist

$$\langle M\rangle = \bigcup_{i=0}^{k} M_i.$$

Dabei gilt $k \leqq |\langle M\rangle| - |M| \leqq |A| - |M|$.

Sei $m := |M|$, $m_i := |M_i|$, $\tilde{m} := |\langle M\rangle|$, $n := |A|$. Zur Bestimmung von M_{i+1} aus M_i werden $\sum_{f\in F} m_i^{v_f}$ Polynome berechnet. Jedes Ergebnis ist in eine Menge M' mit $M_i \subseteq M' \subseteq M_{i+1}$ einzusortieren. Bei Realisierung der M_i durch Listen kommt man ohne Berücksichtigung der Kosten für die Abbildung χ und die Operationen f mit einem Zeitaufwand

$$O\left(m_{i+1} \sum_{f\in F} m_i^{v_f}\right)$$

für die Bestimmung von M_{i+1} aus M_i aus. Insgesamt ergibt $m_i = O(\tilde{m}) = O(n)$ und $\tilde{m} - m + 1 = O(\tilde{m}) = O(n)$ die (sehr grobe) obere Schranke

$$(\tilde{m} - m + 1)\, O\Big(\sum_{f\in F} \tilde{m}^{v_f+1}\Big) = O\Big(\sum_{f\in F} \tilde{m}^{v_f+2}\Big) = O(\tilde{m}^{\mu+2}) = O(n^{\mu+2})$$

für den Zeitaufwand mit

$$\mu := \max(v_f \mid f \in F).$$

Der Platzaufwand hat ohne Berücksichtigung des zur Definition von $\mathfrak{A}$ erforderlichen Platzes die Größenordnung $O(\tilde{m})$; es wird nur Platz zur Abspeicherung der M_i benötigt, und M_{i+1} kann jeweils als Verlängerung von M_i realisiert werden.

Für die Trägermenge $[M]$ von $\mathfrak{M}'$, gegeben durch $M \subseteq A$ und $G \subseteq F$, hat man die folgende konstruktive Definition:

2.2. Bemerkung. Sei $M'_0 := M$, $M'_{2i+1} := M'_{2i} \cup F(M'_{2i})$, $M'_{2i+2} := M'_{2i+1} \cup G(M'_{2i+1}, A)$ für $i = 0, 1, \ldots$. Es sei k minimal mit $M'_{2k+2} = M'_{2k}$. Dann ist

$$[M] = \bigcup_{i=0}^{2k} M'_i.$$

Dabei gilt $k \leqq |[M]| - |M| \leqq n - m$.

Sei $m'_i := |M'_i|$, $\tilde{m} := |[M]|$. Unter denselben Bedingungen wie oben erhält man als Zeitschranke für die Bestimmung von $[M]$:

$$\sum_{i=0}^{m'-m} \Big(m'_{2i+1} \cdot \sum_{f \in F} m'^{\nu_f}_{2i} + m'_{2i+2} \cdot \sum_{\substack{f \in F \\ \nu_f > 1}} m'_{2i+1} \cdot n^{\nu_f - 1} \cdot \nu_f \Big)$$

$$= (\tilde{m}' - m + 1) O(\tilde{m}'^{\mu+1} + \mu' \tilde{m}'^2 n^{\mu'-1})$$

$$= O(\tilde{m}'^{\mu+2} + \tilde{m}'^3 n^{\mu'-1})$$

$$= O(n^{\mu+2} + n^{\mu'+2}) = O(n^{\mu+2})$$

mit

$$\mu' := \max(\nu_f | f \in G),$$

wobei $\mu' > 1$ vorausgesetzt sei (sonst ist $[M] = \langle M \rangle$). Der Platzaufwand kann wieder durch $O(\tilde{m}')$ abgeschätzt werden.

Für den Speziallfall von Halbgruppen oder Ringen ergeben diese zugegebenermaßen sehr naiven Abschätzungen die folgenden Aussagen über den Zeitaufwand:

Halbgruppen: Erzeugen einer Unterhalbgruppe der Ordnung $\tilde{m}$: $O(\tilde{m}^4)$.
Erzeugen eines Ideals der Ordnung $\tilde{m}'$: $O(\tilde{m}'^3 n)$.
Erzeugen eines einseitigen Ideals der Ordnung $\tilde{m}'$: $O(\tilde{m}'^3 n)$.

Ringe: Erzeugen eines Teilrings der Ordnung $\tilde{m}$: $O(\tilde{m}^4)$.
Erzeugen eines Ideals der Ordnung $\tilde{m}'$: $O(\tilde{m}'^3 n)$.

3. Der Vorteil sorgfältiger Buchführung

Wir wollen jetzt den durch Bemerkung 2.1 gegebenen Algorithmus sorgfältiger studieren; dabei übernehmen wir die oben eingeführte Bezeichnung und definieren zusätzlich

$$Q_{i+1} := F(M_i) \setminus M_i \quad \text{für} \quad i = 0, 1, \ldots.$$

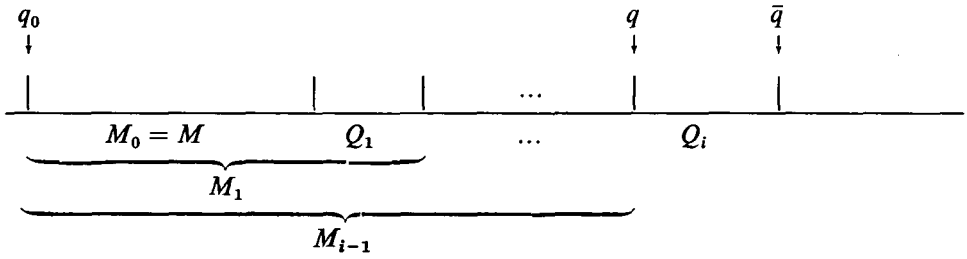
Dann ist offenbar

$$M_{i+1} = M_i \cup F(Q_i, M_i) \quad \text{für} \quad i = 1, 2, \ldots.$$

Zur Bestimmung von $F(Q_i, M_i)$ müssen genau

$$\sum_{f \in F} m_i^{\nu_f} - m_{i-1}^{\nu_f}$$

verschiedene Polynomwerte berechnet werden. Wir wollen wieder voraussetzen, daß die M_i in Listenform gespeichert sind:



Man benötigt vier Zeiger:

q_0 verweist auf den Anfang der Liste, $\bar{q}_0$ hinter das Ende,

q verweist auf den Anfang von Q_i, $\bar{q}$ hinter das Ende.

Zu beginn der Berechnung von M_{i+1} ist $\bar{q}_0 = \bar{q}$. Mit jedem neuen Element wird $\bar{q}_0$ „nach hinten" verschoben. Der Algorithmus endet, wenn nach der Berechnung von M_{i+1} gilt: $\bar{q} = \bar{q}_0$. Andernfalls setzt man $q := \bar{q}$, $\bar{q} := \bar{q}_0$ und fährt mit der Berechnung von M_{i+2} fort.

Sei nun $f \in F$ und $v_f > 0$. Zur Bestimmung der Argumente $(a_1, \ldots, a_{v_f}) \in M_i^{v_f} \setminus M_{i-1}^{v_f}$ benutze man z.B. v_f Laufvariable $p_1, \ldots, p_{v_f}$ mit

$$q \leqq p_1 < \bar{q} \quad \text{und} \quad q_0 \leqq p_j < \bar{q} \quad \text{für} \quad j > 1.$$

Zu jedem derartigen v_f-tupel $(p_1, \ldots, p_{v_f})$ betrachte man ferner sämtliche Permutationen der Form

$$(p_j, p_2, p_3, \cdots, p_{j-1}, p_1, p_{j+1}, \cdots, p_{v_f}),$$

für die $p_j < q$ ist. Bei sorgfältiger Buchführung über die p_j mit $p_j < q$ (z.B. durch gekettete Speicherung dieser p_j) kann man diese Permutationen insgesamt mit dem Aufwand $O(t+1)$ bestimmen, wobei t die Anzahl dieser p_j ist. Der gesamte Zeitaufwand für alle v_f-tupel ist daher proportional der Anzahl der betrachteten v_f-tupel, und diese ist

$$q_i \cdot \sum_{j=0}^{v_f-1} m_{i-1}^{v_f-j-1} \cdot q_i^j \cdot \binom{v_f-1}{j} \cdot (v_f - j) \leqq \frac{(v_f+1)^2}{4 v_f} (m_i^{v_f} - m_{i-1}^{v_f}) =$$

$$= O\big(v_f (m_i^{v_f} - m_{i-1}^{v_f})\big).$$

Der Platzaufwand hat die Größenordnung $O(v_f)$ für die Laufvariablen — diesen Aufwand hatten wir in den „naiven" Abschätzungen vernachlässigt.

Für jedes v_f-tupel $(a_1, \ldots, a_{v_f})$ sind die folgenden Operationen auszuführen:

(1) Berechnung von $a := f(a_1, \ldots, a_{v_f})$.

(2) Prüfen, ob a im bisher aufgebauten Teil von M_{i+1} liegt, und, wenn nein, a in Q_{i+1} einfügen.

Für (1) treten jedesmal die Kosten c_f auf. Die Aufgabe (2) läßt sich durch Suchen in einer zu m_{i+1} proportionalen Zeit erledigen. Eine Verbesserug ist nur bei Abänderung der Speicherungsmethode für die M_i zu erzielen. Diese Möglichkeit soll später weiter verfolgt werden.

In der bisherigen Version haben wir also die Zeitschranke

$$\Phi^z := \sum_{f\in F} m_0^{\nu_f}\cdot c_f\cdot m_1 + \sum_{i=1}^{\tilde{m}-m+1}\sum_{\substack{f\in F\\ \nu_f>0}} \frac{(\nu_f+1)^2}{4\nu_f}\cdot (m_i^{\nu_f}-m_{i-1}^{\nu_f})\cdot c_f\cdot m_{i+1}$$

und die Platzschranke

$$\Phi^p := O(\tilde{m}+\mu).$$

Φ^z wird maximal für $m_1=m_2=\ldots=\tilde{m}$, d.h., $m_1-m_0=\tilde{m}-m$. Dies entspricht der Situation, daß man zunächst alle Elemente von $\langle M\rangle\setminus M$ erhält und danach nur noch kostspielige Überprüfungen gemäß (2) mit negativem Ergebnis macht. Dies ergibt

$$\Phi^z \leqq \sum_{f\in F} m^{\nu_f}\cdot\tilde{m}\cdot c_f + \sum_{\substack{f\in F\\ \nu_f>0}} \frac{(\nu_f+1)^2}{4\nu_f}\cdot(\tilde{m}^{\nu_f}-m^{\nu_f})\cdot\tilde{m}\cdot c_f$$

$$\leqq \sum_{\substack{f\in F\\ \nu_f>0}} \frac{(\nu_f+1)^2}{4\nu_f}\cdot\tilde{m}^{\nu_f+1}\cdot c_f + \sum_{\substack{f\in F\\ \nu_f=0}}\tilde{m}\cdot c_f =$$

$$= \begin{cases} O(\mu\cdot\tilde{m}^{\mu+1}\cdot c), & \text{falls } \mu>0,\\ O(\tilde{m}\cdot c), & \text{falls } \mu=0,\end{cases}$$

mit $c:=\max(c_f|f\in F)$. Für festes μ und c hat man also $\Phi^z=O(\tilde{m}^{\mu+1})=O(n^{\mu+1})$. Diese Schranke erhält man — etwa für $\mu=2$ — auch aus dem trivialen Erreichbarkeitsalgorithmus für gerichtete Graphen (vgl. [1], S. 207).

Wie angekündigt, wollen wir jetzt die Speicherungsmethode ändern. Ein einfacher Übergang auf hash-Tabellen oder boolesche Vektoren, der die Vereinigungsoperation (2) erheblich beschleunigen würde, ist nicht zu empfehlen, weil mit dieser Beschleunigung eine Verlangsamung der Bestimmung der ν_f-tupel eingehandelt würde. Man beachte, daß der Zugriff auf die M_i gleichzeitig sequentiell und indiziert möglich sein sollte. Zwei Lösungswege mit unterschiedlichen Auswirkungen hinsichtlich der Kosten liegen nahe: Man kann neben dem Listenspeicher für die M_i vorsehen

(a) ein Feld der Größe $O(n)$ zur Markierung der in M_i vorhandenen Elemente von A (etwa als hash-Tabelle) oder

(b) eine zusätzliche Abspeicherung der M_i in Form balancierter Bäume.

In beiden Fällen wird zu jedem Element ein zusätzlicher Rechenaufwand zur Indexberechnung bei (a) oder für Vergleiche bei (b) nötig, dessen Zeitbedarf durch h_A abgeschätzt werden kann. Im Falle (a) sind je Element $O(1)$ Zugriffe ausreichend; im Falle (b) muß man für Suchen, Einfügen und Restrukturieren $O(\log m_i)$ Schritte ansetzen. Für (a) erhält man somit die Zeitschranke

$$\Psi^z := \sum_{f\in F} m_0^{\nu_f}\cdot c_f\cdot h_A + \sum_{i=1}^{\tilde{m}-m+1}\sum_{\substack{f\in F\\ \nu_f>0}} \frac{(\nu_f+1)^2}{4\nu_f}\cdot(m_i^{\nu_f}-m_{i-1}^{\nu_f})\cdot c_f\cdot h_A + O(n) =$$

$$= O(\tilde{m}^{\mu}\cdot c\cdot h_A + n)$$

und die Platzschranke

$$\Psi^p := O(n+\tilde{m}+\mu).$$

Der zusätzliche Zeitaufwand $O(n)$ ist erforderlich, um die Anfangsbesetzung der hash-Tabelle zu organisieren. Dies spart man bei (b). Dort hat man die Kosten

$$\Omega^z := \sum_{f\in F} m_0^{\nu_f}\cdot c_f\cdot h_A\cdot\log m_1+\sum_{i=1}^{\tilde{m}-m+1}\sum_{\substack{f\in F\\ \nu_f>0}}\frac{(\nu_f+1)^2}{4\nu_f}\cdot(m_i^{\nu_f}-m_{i-1}^{\nu_f})\cdot c_f\cdot h_A\cdot\log m_{i+1}=$$

$$= O(\tilde{m}^\mu\cdot c\cdot h_A\cdot\log\tilde{m})$$

und

$$\Omega^p := O(\tilde{m}+\mu).$$

Zusammenfassend erhalten wir:

3.1. Satz. Die Erzeugung von $\mathfrak{M}$ ist möglich mit dem Aufwand

$$\Psi^z = O(\tilde{m}^\mu+n),\quad \Psi^p = O(n)$$

beziehungsweise mit dem Aufwand

$$\Omega^z = O(\tilde{m}^\mu\cdot\log\tilde{m}),\quad \Omega^p = O(\tilde{m}).$$

Für die Erzeugung von $\mathfrak{M}'$ aus $M\subseteq A$ und $G\subseteq F$ kann man ähnlich vorgehen. Sei

$$Q'_{2i+1} := F(M'_{2i})\setminus M'_{2i},$$

$$Q'_{2i+2} := G(M'_{2i+1}, A)\setminus M'_{2i+1}.$$

Dann ist also

$$M'_{2i+1} = M'_{2i}\cup F(Q'_{2i}\cup Q'_{2i-1}, M'_{2i})$$

und

$$M'_{2i+2} = M'_{2i+1}\cup G(Q'_{2i+1}\cup Q'_{2i}, A).$$

Wie oben schätzt man den Zeitaufwand und den Platzaufwand ab und erhält:

3.2. Satz. Die Erzeugung von $\mathfrak{M}'$ ist möglich mit dem Aufwand

$$\Psi'^z = O(\tilde{m}'^\mu+\tilde{m}'n^{\mu'-1}+n),\quad \Psi'^p = O(n)$$

beziehungsweise mit dem Aufwand

$$\Omega'^z = O(\tilde{m}'^\mu\log\tilde{m}'+\tilde{m}'n^{\mu'-1}\log\tilde{m}'),\quad \Omega'^p = O(\tilde{m}').$$

Es ist klar, daß die Schranken Ψ^z und Ψ'^z der Größenordnung nach optimal sind, weil ohne genauere Kenntnis von $\mathfrak{A}$ jeweils sämtliche Operationen f auf sämtliche mögliche ν_f-tupel angewendet werden müssen.

An dieser Stelle ist anzumerken, daß die obigen Aussagen stillschweigend voraussetzen, daß $|F|=O(1)$ ist für die einmal gewählte Klasse von Algebren. Es ist offensichtlich, welche Verallgemeinerungen für den Fall, daß $|F|\neq O(1)$ ist, durchzuführen sind. Wir beschränken uns darauf, für diese Situation ein repräsentatives Beispiel anzugeben: $\mathfrak{A}$ sei eine Gruppe; um auch die Normalteilereigenschaften ausdrücken zu können, wählen wir

$$F = \{\cdot, {}^{-1}\}\cup\{\gamma_a | a\in A\}$$

mit

$$v_\cdot = 2, \quad v_{-1} = 1, \quad v_{\gamma_a} = 1$$

und

$$\gamma_a(b) = a^{-1} \cdot b \cdot a.$$

Die Unteralgebren von $\mathfrak{A}$ sind gerade die Normalteiler, und geeignete Modifikation von Satz 3.1 ergibt die Aufwandsschranken

$$\Psi^z = O(n\tilde{m}+\tilde{m}^2+n), \quad \Psi^p = O(n)$$

beziehungsweise

$$\Omega^z = O(n\tilde{m}\log\tilde{m}+\tilde{m}^2\log\tilde{m}), \quad \Omega^p = O(\tilde{m}).$$

Die entsprechenden Schranken ohne Berücksichtigung der Buchführungskosten findet man in [7].

4. Kosten der Uniformisierung

Der Ablauf der im vorigen Abschnitt vorgeführten Algorithmen ist in hohem Maße von den Eingabedaten M und gegebenenfalls G abhängig. In diesem Abschnitt untersuchen wir an den Spezialfällen endlicher Halbgruppen und endlicher Ringe die Kosten eines von den Eingabedaten unabhängigen Erzeugungsverfahrens.

Für die angesprochenen Spezialfälle ergeben sich aus 3.1 und 3.2 die folgenden Schranken:

Halbgruppen: $\Psi^z = O(\tilde{m}^2+n) = O(n^2)$, $\Psi^p = O(n)$;

$$\Psi'^z = O(\tilde{m}'n) = O(n^2), \quad \Psi'^p = O(n) \quad \text{mit} \quad G = F = \{\cdot\}.$$

Ringe: Wie Halbgruppen, mit $G = \{\cdot\}$.

Wir werden zeigen, wie sich die Erzeugungsaufgaben in Halbgruppen und Ringen auf die Multiplikation von $n \times n$-Matrizen über dem booleschen Halbring **B** zurückführen lassen.

Sei also jetzt $\mathfrak{A}$ eine endliche Halbgruppe. Zu $a \in A$ sei

$$\Lambda_a = (\lambda^a_{b,c})$$

die durch die innere Linkstranslation von a definierte $n \times n$-Matrix über **B**, d.h.,

$$\lambda^a_{b,c} = \begin{cases} 1, & \text{falls} \quad ab = c, \\ 0, & \text{falls} \quad ab \neq c. \end{cases}$$

Für $M \subseteq A$ sei

$$\Lambda_M := \sum_{a \in M} \Lambda_a.$$

Sei weiter Λ^0_M die $n \times n$-Einheitsmatrix über **B** und

$$\Lambda^*_M := \sum_{j=0}^{\infty} \Lambda^j_M.$$

Man beweist leicht, daß Λ^*_M an der Stelle (b, c) genau dann eine 1 hat, wenn $b = c$ ist oder wenn $c = a_1 a_2 \ldots a_j b$ für geeignete $j \in \mathbf{N}$ und $a_1, \ldots, a_j \in M$ ist.

Sei nun π_M der n-komponentige Zeilenvektor über $\mathbf{B}$, dessen b-te Komponente genau dann 1 ist, wenn $b \in M$ ist. Damit gilt:

4.1. Lemma. $c \in \langle M \rangle$ genau dann, wenn die c-te Komponente von $\pi_M \Lambda_M^*$ gleich 1 ist.

Die zur Berechnung von Λ_M^* erforderliche Zeit hat bekanntlich dieselbe Größenordnung wie die zur Multiplikation zweier $n \times n$-Matrizen über $\mathbf{B}$ erforderliche Zeit $\mathrm{Mult}^z(n, \mathbf{B})$. Der Zeitaufwand für die Herstellung von Λ_M ist durch $O(n^2)$ beschränkt. Damit folgt:

4.2. Satz. Die Unterhalbgruppe $\mathfrak{M}$ von $\mathfrak{A}$ kann mit einem durch $O(\mathrm{Mult}^z(n, \mathbf{B}))$ beschränkten Zeitaufwand (uniform) erzeugt werden.

Natürlich hat auch der Platzaufwand für die Erzeugung von $\mathfrak{M}$ die Größenordnung des für die Berechnung von Λ_M^*, d.h., für die Berechnung der transitiven Hülle einer binären Relation erforderlichen Platzaufwandes.

Für die Idealerzeugung geht man ein wenig anders vor:

4.3. Satz. Die Berechnung des von M erzeugten Linksideals (Rechtsideals, Ideals) der Halbgruppe $\mathfrak{A}$ kann (uniform) in der Zeit $O(n^2)$ durchgeführt werden.

Zum Beweis hat man nur zu beachten, daß c genau dann im von M erzeugten Linksideal liegt, wenn die c-te Komponente von $\pi_M \Lambda_A^*$ gleich 1 ist. Zur Berechnung von Λ_A^* ist jedoch keine Multiplikation, sondern nur die Addition

$$\Lambda_A^* = \Lambda_A + \Lambda_A^0$$

erforderlich. Mit der dualen Aussage erhält man auch die Behauptung für Rechtsideale und Ideale.

Es ist ohne Schwierigkeiten möglich, diese Überlegungen auf Ringe $\mathfrak{A}$ zu übertragen. Neben der Assoziativität für die beiden Operationen $+$ und $\cdot$ nutzt man dabei nur noch die Distributivitätsgesetze

$$a \cdot (b+c) = a \cdot b + a \cdot c, \quad (a+b) \cdot c = a \cdot c + b \cdot c$$

aus. Den durch $M \subseteq A$ bezüglich $+$ und $\cdot$ bewirkten Linkstranslationen entsprechen die Matrizen $\Lambda_{+,M}$ und $\Lambda_{\cdot,M}$. Man beweist leicht, daß $c \in \langle M \rangle$ genau dann gilt, wenn die c-te Komponente von $\pi_{M_1} \Lambda_{+,M_1}^*$ gleich 1 ist, wobei M_1 durch $\pi_{M_1} = \pi_M \Lambda_{\cdot,M}^*$ definiert ist. Damit erhält man aus Satz 4.2:

4.4. Korollar. Der Teilring $\mathfrak{M}$ von $\mathfrak{A}$ kann mit einem durch $O(\mathrm{Mult}^z(n, \mathbf{B}))$ beschränkten Zeitaufwand (uniform) erzeugt werden.

Für die Erzeugung von Idealen tritt, weil im allgemeinen auf die Erzeugung der durch M erzeugten Unterhalbgruppe bezüglich $+$ nicht verzichtet werden kann, ebenfalls der Aufwand $O(\mathrm{Mult}^z(n, \mathbf{B}))$ auf.

Ein uniformes Erzeugungsverfahren für beliebige universelle Algebren wird schon vom Ansatz her erheblich komplizierter als für die Spezialfälle von Halbgruppen und Ringen. Man beachte insbesondere, daß die so bequeme Darstellung durch Matrizen über $\mathbf{B}$ im allgemeinen wenig Vorteile bringen dürfte; in den Beispielen von Halbgruppen und Ringen ist durch die Matrixdarstellung etwas zu gewinnen, weil der Assoziativität in der Algebra die Assoziativität der Matrixmultiplikation korrespondiert.

Abstract

Time and space complexity of algorithms for generating algebras is studied; the bounds derived are essentially optimal.

DEPARTMENT OF COMPUTER SCIENCE
THE UNIVERSITY OF WESTERN ONTARIO
LONDON, ONTARIO
CANADA N6A 5B7

Literatur

[1] A. V. AHO, J. E. HOPCROFT, J. D. ULLMAN: The Design and Analysis of Computer Algorithms. Addison-Wesley Publ. Co., Reading, 1975.
[2] M. FURST, J. HOPCROFT, E. LUKS: Polynomial-Time Algorithms for Permutation Groups Proc. 21st FOCS Symp., 1980, 36—41.
[3] A. GORALČÍKOVÁ, P. GORALČÍK, V. KOUBEK: Testing Properties of Finite Algebras. Proc. ICALP 1980. Lecture Notes in Computer Science *85*, Springer-Verlag, Berlin, 1980, 273—281.
[4] P. GORALČÍK, A. GORALČÍKOVÁ, V. KOUBEK, V. RÖDL: Fast Recognition of Rings and Lattices. Proc. FCT 1981. Springer Lecture Notes in Computer Science *117*. Springer-Verlag, Berlin, 1981, 137—145.
[5] H. JÜRGENSEN: Computers in Semigroups. Semigroup Forum *15* (1977/78), 1—20.
[6] T. KREID: The Determination of Subalgebras of a Given Algebra. Demonstratio Mathemat. *8* (1975), 269—279.
[7] R. E. Tarjan: Determining whether a Groupoid is a Group. Information Proc. Letters *1* (1972), 120—124.

Received May 9, 1983.