日本ソフトウェア科学会第 29 回大会 (2012 年度) 講演論文集

正規言語上の Abstract Numeration System の文字列圧縮への応用

新屋 良磨

Abstract Numeration System (ANS) $\mathcal{S} = (L, A, <)$ は，全単射 $\mathrm{val}_{\mathcal{S}} : L \to \mathbb{N}$ とその逆写像 $\mathrm{rep}_{\mathcal{S}} : \mathbb{N} \to L$ によって，数と文字列を一対一対応づける記数法である．$\mathrm{val}_{\mathcal{S}}(w) = n \Leftrightarrow \mathrm{rep}_{\mathcal{S}}(n) = w$ は，長さ—辞書順において文字列 $w \in L$ が n 番目の要素であることを意味する．本稿では，正規言語上の $ANS\,\mathcal{S} = (L, A, <), \mathcal{S}' = (K, A', <')$ について，$\underset{\mathcal{S}\to\mathcal{S}'}{P} : L \to K$, $\underset{\mathcal{S}\to\mathcal{S}'}{P}(w) := \mathrm{rep}_{\mathcal{S}'}(\mathrm{val}_{\mathcal{S}}(w))$ を言語 L から K への基数変換として定義し，これが圧縮に応用できることを示す．また，具体的なアルゴリズムとその計算量，および圧縮率に関する重要な定理を与える．

1 はじめに

Goldberg と Sipser [5] は，与えられた言語について文字列の順番 — *Ranking* による圧縮を提案し，文脈自由文法においては *Ranking* の計算が多項式時間で計算可能なことを示した．それ以降，形式文法の文法クラスにおける *Ranking* の並列計算可能性[†1]や効率の良いアルゴリズムの研究がいくつか行われている [6] [3] [9]．文献 [8]§7.2 には *Ranking* の圧縮への応用についての情報が纏められている．

一方，*Ranking* とは独立に，言語と数の全単射によって定められる記数法 *Abstract Numeration System (ANS)* が Lecomte と Rigo [7] によって提案され，その表現力や性質が研究されている [1]．

本研究では，正規言語上の *ANS* について，任意の正規言語 L, K 上の基数変換 $\underset{\mathcal{S}\to\mathcal{S}'}{P} : L \to K$ を定義し，$\underset{\mathcal{S}\to\mathcal{S}'}{P}$ が圧縮の条件を満たす場合について考察を行なった．本研究における貢献を挙げる．

- *ANS* に関するアルゴリズムの提案とその計算量の見積り (§ 3)
- 正規言語上の *ANS* による基数変換の提案 (§ 4. 1)
- 圧縮率の定義とその計算方法 (§ 4. 2)

本稿は本章を含んで 5 章から構成される．§ 2 は本稿で使用する記法及び前提となる定義・定理について紹介する．§ 3 にて，*ANS* のアルゴリズム (擬似コードを含む) とその計算量を示し，§ 4 で圧縮への応用を提案する．§ 5 を本稿のまとめとする．

2 準備

本稿における記法及び定義，いくつかの定理を紹介する．読者には特に前提知識を仮定しない．本章で扱う定義や定理の詳細は文献 [10] [2] [12] [4] 等に依る．

2. 1 正規言語とオートマトン

有限文字集合 A において，A^n を A 上の長さが n である文字列の集合として用いる．さらに，A 上の全ての文字列の集合を

$$A^* := \bigcup_{i=0}^{\infty} A^i$$

によって表す．A^* の部分集合 L を，A 上の言語と呼ぶ．特に断らない限り，$w \in A^*, \sigma \in A$ の記号を用いる．さらに，文字列においては，その長さを $|w|$ とし，$1 \le i \le |w|$ について $w_i \in A$ を i 番目の文字とする．

Text Compression using Abstract Numeration System on a Regular Language

Ryoma Sin'ya, 東京工業大学情報理工学研究科数理・計算科学専攻, Department of Mathematical and Computing Sciences, Graduate School of Information Science and Engineering, Tokyo Institute of Technology.

†1 *NC–computability*

定義 2.1. A 上の正規言語の集合は，以下のように再帰的に定義される．

- 空言語 $\emptyset$，空文字列の単集合 $\{\epsilon\}$，及び A の元のみからなる単集合 $\forall\sigma \in A \ \ \{\sigma\}$ は全て正規言語である．
- 正規言語 K, L において，下記の演算を適用した結果も正規言語である．
  - $L \cup M$ (和集合)
  - $KL := \{kl \in A^* \mid k \in K, l \in L\}$ (連接)
  - $L^* := \{\epsilon\} \cup L \cup LL \cup \cdots$ (Kleene 閉包) □

定義 2.2. 有限オートマトン $\mathcal{A}$ は 5 つ組 $\mathcal{A} = (Q, A, \delta, I, F)$; 状態の有限集合 Q, 入力文字の有限集合 A, 遷移関数 $\delta : Q \times A \to \mathfrak{P}(Q)$, 初期状態集合 $I \subseteq Q$, 受理状態集合 $F \subseteq Q$ によって定義される． □

オートマトン $\mathcal{A}$ の大きさ $|\mathcal{A}|$ を，その状態集合の元の個数 $\#Q$ とする．さらに，説明のため以下に定義される拡張遷移関数 $\widehat{\delta} : Q \times A^* \to \mathfrak{P}(Q)$ を用いる．

$$\begin{aligned} \widehat{\delta}(q, \sigma w) &:= \bigcup_{q' \in \delta(q,\sigma)} \widehat{\delta}(q', w) \\ \widehat{\delta}(q, \epsilon) &:= \{q\} \end{aligned}$$

ここで ϵ は空文字を表す．$\mathcal{A}$ において，$p \in \widehat{\delta}(q, w)$ となる q 及び文字列 w が存在する場合，この w による状態遷移を $q \xrightarrow[\mathcal{A}]{w} p$，誤解のない場合は $q \xrightarrow{w} p$ と表記する．

オートマトン $\mathcal{A}$ 上の計算 (*computation*) c は状態遷移列を意味し，

$$c := q_0 \xrightarrow{\sigma_1} q_1 \xrightarrow{\sigma_2} q_2 \cdots \xrightarrow{\sigma_n} q_n$$

と記述する．$\mathcal{A}$ において q_0 が初期状態，q_n が受理状態である場合，文字列 w を受理文字列と呼ぶ．$\mathcal{A}$ の全受理文字列の集合を

$$L(\mathcal{A}) = \left\{ w \mid \exists q \in I, \exists p \in F \left[q \xrightarrow{w} p \right] \right\}$$

として記述する．オートマトン $\mathcal{A}, \mathcal{A}'$ が等価であるとは，$L(\mathcal{A}) = L(\mathcal{A}')$ を満たすことである．$L(\mathcal{A}) = L$ であるとき，$\mathcal{A}$ は L を認識するという．

定義 2.3. 以下の条件をみたす場合，$\mathcal{A}$ は無曖昧オートマトン (*Unambiguous, UFA*) であるという．

1. すべての状態対 $(q, p) \in Q \times Q$、全ての文字列 $w \in A^*$ について，計算 $c = q \xrightarrow{w} p$ がたかだか一つ存在する．
2. 全ての受理文字列 $w \in L(\mathcal{A})$ について，$q \xrightarrow{w} p$ を満たす初期状態 $q \in I$ と受理状態 $p \in F$ が一意に存在する．

特に，$\mathcal{A}$ の初期状態集合及び遷移状態関数の像が単集合:

$$|I| = 1 \ \ \wedge \ \ \forall q \in Q, \forall \sigma \in A \ \ [|\delta(q, \sigma)| = 1]$$

を満たす場合は，$\mathcal{A}$ は決定性オートマトン (*Deterministic, DFA*) であるという．*DFA* であることを強調したい場合，オートマトンを $\mathcal{D}$ で記す □

DFA においては遷移関数の像が集合である必要はないので，簡約のため $\delta : Q \times A \to Q$ として扱う．

この節の最後に，オートマトンのベクトル及び行列による表現形式を定義する．オートマトンは有向グラフと見ることができるため，その隣接行列を考えることでオートマトンから自然数上の正方行列を導出することができる．

定義 2.4. オートマトン $\mathcal{A}$ について，その隣接行列 $\boldsymbol{M}(\mathcal{A}) \in \mathbb{N}^{|\mathcal{A}| \times |\mathcal{A}|}$ を，全ての q 行 p 列要素に対して

$$\boldsymbol{M}(\mathcal{A})_{qp} := \# \left\{ \sigma \mid \exists q, p \in Q \ \ \left[q \xrightarrow{\sigma} p \right] \right\}$$

と定義する．また，初期ベクトル (行ベクトル) 及び受理ベクトル (列ベクトル)$\boldsymbol{V}_I(\mathcal{A}), \boldsymbol{V}_F(\mathcal{A})^T \in \mathbb{B}^{|\mathcal{A}|}$ を

$$\begin{aligned} \boldsymbol{V}_I(\mathcal{A})_q &:= \text{if } \ q \in I \ \text{ then 1 else } 0 \\ \boldsymbol{V}_F(\mathcal{A})_q &:= \text{if } \ q \in F \ \text{ then 1 else } 0 \end{aligned}$$

と定義する．また，$\mathcal{A}$ が固定されている場合は，それぞれ簡単に $\boldsymbol{M}, \boldsymbol{V}_I, \boldsymbol{V}_F$ と記述する．さらに，隣接行列において M_q とした場合は，M における q 行目の行ベクトルを表すとする． □

2.2 Combinatorial Complexity

形式言語理論における *Combinatorial Complexity* は，言語 L における数え上げ関数[†2] $C_L : \mathbb{N} \to \mathbb{N}$ に関する複雑さとして定式化されている [12] [1] [3] .

定義 2.5. 言語 L について，長さが n 及び n 以下である文字列の個数をそれぞれ

$$\begin{aligned} C_L(n) &:= \#\{w \in L \mid n = |w|\} = \#(L \cap A^n) \\ C_L^{\leq}(n) &:= \sum_{i=0}^{n} C_L(i) \end{aligned}$$

で表す． □

†2 *Counting function* 又は *Complexity function* とも呼ぶ

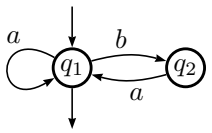

図 1 *DFA* $\mathcal{D}_{fib}$; $L(\mathcal{D}_{fib}) = \{a, ba\}^*$

C_L の計算は，前述の *UFA* を用いて計算することができる．

補題 2.1. L を認識する *UFA* $\mathcal{A}$ について，

$$C_L(n) = \boldsymbol{V}_I \boldsymbol{M}^n \boldsymbol{V}_F$$

が成り立つ．

証明. $\mathcal{A}$ の隣接行列 $\boldsymbol{M}$ について，$\boldsymbol{M}^n_{qp}$ は状態 q から p への文字列の長さが n の計算の総数にほかならない．これに初期ベクトルを掛けた結果，ベクトルの各要素 $(\boldsymbol{V}_I \boldsymbol{M}^n)_p$ は初期状態から状態 p への長さ n の計算の総数となる．さらに，受理ベクトルを掛けることで，補題の式から「初期状態から受理状態への長さ n の計算の総数」が得られる．

ここで $\mathcal{A}$ は無曖昧であるので，計算の文字列において重複はない．よって補題が証明された． □

例 2.1. 図 1 の *DFA* $\mathcal{D}_{fib}$ について，C_L を考る．$n \leq 4$ までを例示すると

$$\begin{aligned} C_L(0) &= \#\{\epsilon\} = 1, C_L(1) = \#\{a\} = 1 \\ C_L(2) &= \#\{aa, ba\} = 2 \\ C_L(3) &= \#\{aaa, aba, baa\} = 3 \\ C_L(4) &= \#\{aaaa, aaba, abaa, baaa, baba\} = 5 \end{aligned}$$

である．$\mathcal{D}_{fib}$ の隣接行列 及び初期ベクトル，受理ベクトルはそれぞれ

$$\boldsymbol{M} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}, \boldsymbol{V}_I = \boldsymbol{V}_F^T = (1, 0)$$

となる．$\boldsymbol{M}$ の固有値 $\alpha = \frac{1+\sqrt{5}}{2}, \beta = \frac{1-\sqrt{5}}{2}$ から

$$\boldsymbol{M} = \boldsymbol{P}\boldsymbol{D}\boldsymbol{P}^{-1} = \frac{1}{\sqrt{5}} \begin{pmatrix} \alpha & \beta \\ 1 & 1 \end{pmatrix} \begin{pmatrix} \alpha & 0 \\ 0 & \beta \end{pmatrix} \begin{pmatrix} 1 & -\beta \\ -1 & \alpha \end{pmatrix}$$

と対角化可能である．ここで，補題 2.1 を適用すると

$$\begin{aligned} C_L(n) &= \boldsymbol{V}_I \boldsymbol{M}^n \boldsymbol{V}_F = \boldsymbol{V}_I \boldsymbol{P} \boldsymbol{D}^n \boldsymbol{P}^{-1} \boldsymbol{V}_F \\ &= \frac{1}{\sqrt{5}} \boldsymbol{V}_I \begin{pmatrix} \alpha^{n+1} - \beta^{n+1} & \alpha^n - \beta^n \\ \alpha^n - \beta^n & \alpha^{n-1} - \beta^{n-1} \end{pmatrix} \boldsymbol{V}_F \\ &= \frac{1}{\sqrt{5}} \left\{ \left(\frac{1+\sqrt{5}}{2} \right)^{n+1} - \left(\frac{1-\sqrt{5}}{2} \right)^{n+1} \right\} \end{aligned}$$

となり，フィボナッチ数の $n+1$ 番目が得られる． □

次に，数え上げ関数の漸近に関する次の重要な定理を紹介する．この定理は § 4.2 における圧縮率の見積りで必要となる．

定理 2.1 (Shur [11][12]). *DFA* $\mathcal{D}$ によって受理される任意の正規言語 $L = L(\mathcal{D})$ について，

$$C_L(n) = \Theta(n^m \alpha^n)$$

が成り立つ．ここで α は $\mathcal{D}$ の隣接行列のフロベニウス根[†3]，$m+1$ は $\mathcal{D}$ の隣接行列において α を持つ，単一路で連結された強連結成分の最大個数である，

補足 2.1. ここでは定理 2.1 の証明は行わない．頂点数 n 及び辺の数が e である有向グラフについて，その隣接行列のフロベニウス根を $\Theta(n^3)$，全ての強連結成分を求める強連結成分分解は $\mathrm{O}(n+e) = \mathrm{O}(n^2)$ の時間計算量で求められることを補足とする． □

例 2.2. 再び図 1 の $\mathcal{D}_{fib}$ について考る．$\mathcal{D}_{fib}$ の隣接行列におけるフロベニウス根は $\alpha = \frac{1+\sqrt{5}}{2}$ である．さらに，$\mathcal{D}$ は唯一の強連結成分から構成されているので，$m+1 = 1$ となる．例 2.1 で求めた結果から

$$C_L(n) = \Theta\left(\left(\frac{1+\sqrt{5}}{2} \right)^n \right)$$

を満たすことが分かる． □

2.3 Abstract Numeration System

記数法 (*Numeration System*) とは，数を文字列として表現するシステムを指す．その表現力や代数演算に関する性質など，それ自体 1 つの数学の研究分野として多く研究されている [1][7].

ここでは，非常に柔軟な記数法である *Abstract Nu-*

†3 非負の正方行列における最大の固有値．詳しくは [2] を参照.

meration System について説明を行う．基本となるアイディアは，順序付けられた文字集合上の言語と自然数間の全単射である．

定義 2.6. 全順序が定められた文字集合 $(A,<)$ において，長さ–辞書順[†4] $\prec$ を次のように定める．文字列 $u,v\in A^*$ において，$|u|<|v|$ である場合 (長さ順) に $u\prec v$ である．さらに，$|u|=|v|$ かつ適当な文字列 $p,q,r\in A^*$ 及び文字 $a,b\in A$ $a<b$ によって $u=paq, v=pbr$ の形に分解できる場合 (辞書順) も $u\prec v$ である．$\prec$ は A^* における全順序となる． □

定義 2.7. Abstract Numeration System(ANS) $\mathcal{S}$ は 3 つ組み $(L,A,<)$; 文字の全順序集合 $(A,<)$ 上の言語 L によって定義される．$\mathbb{N}$ から言語 L への全単射 $\mathrm{rep}_{\mathcal{S}}:\mathbb{N}\to L$ は，自然数 $n\in\mathbb{N}$ を，L を長さ–辞書順によって整列した時の n 番目の文字列に写す写像である．$\mathrm{rep}_{\mathcal{S}}$ の逆写像を $\mathrm{val}_{\mathcal{S}}:L\to\mathbb{N}$ で表す． □

本稿では正規言語上の ANS のみについて考える．ANS の仕組みは非常に単純であるが，記数法としての柔軟性は非常に高い．2 進法や 10 進法も ANS として表現することができる (一般には記数法は負数，さらには実数をも扱えるが，本稿では自然数との対応のみを扱う)．

例 2.3. ANS $\mathcal{S}_2=(L,\{0,1\},<); L=\{0\}\cup\{1\}\{0,1\}^*, 0<1$ について考える．L を長さ–辞書順によって整列すると $\{0,1,10,11,100,\cdots\}$ が得られ，$\mathrm{rep}_{\mathcal{S}_2}$ 及び $\mathrm{val}_{\mathcal{S}_2}$ は 2 進法を定める．つまり

$$\mathrm{rep}_{\mathcal{S}_2}(0)=0,\mathrm{rep}_{\mathcal{S}_2}(1)=1,\mathrm{rep}_{\mathcal{S}_2}(2)=10,\cdots$$
$$\mathrm{val}_{\mathcal{S}_2}(0)=0,\mathrm{val}_{\mathcal{S}_2}(1)=1,\mathrm{val}_{\mathcal{S}_2}(10)=2,\cdots$$

は $n\in\mathbb{N}$ とその 2 進表現を一対一に対応付ける．

同様にして，任意の正整数 k について ANS $\mathcal{S}_k=(L,\{0,1,\cdots,\sigma_{k-1}\},<)$;
$L=\{0\}\cup\{1,2,\cdots,\sigma_{k-1}\}\{0,1,\cdots,\sigma_{k-1}\}^*$, $0<1,1<2,\cdots,\sigma_{k-1}<\sigma_k$ は k 進法を定める． □

3 *DFA* を用いた *ANS* のアルゴリズム

本章では，任意の DFA $\mathcal{D}$ の認識する言語 $L=L(\mathcal{D})$ 上の ANS $\mathcal{S}=(L,A,<)$ について，$\mathrm{rep}_{\mathcal{S}}$ と $\mathrm{val}_{\mathcal{S}}$ のアルゴリズムを定める．アルゴリズムの中心となるのは，DFA $\mathcal{D}$ の隣接行列とベクトルの演算である．

3.1 $\mathrm{val}_{\mathcal{S}}$ の計算

$\mathrm{val}_{\mathcal{S}}$ のアルゴリズムの説明に，次のベクトルの記法を用いる．

定義 3.1. ベクトル $\boldsymbol{V}_q^{<\sigma}\in\mathbb{N}^{|\mathcal{D}|}$ を，状態 q から σ 未満の入力文字による遷移の総数

$$(\boldsymbol{V}_q^{<\sigma})_p:=\#\{\sigma'\in A\mid\sigma'<\sigma\ \wedge\ q\xrightarrow{\sigma'}p\}$$

と定義する． □

定理 3.1. DFA $\mathcal{D}=(Q,A,\delta,I,F); L=L(\mathcal{D})$ となる ANS $\mathcal{S}=(L,A,<)$ において

$$\mathrm{val}_{\mathcal{S}}(w) = \sum_{i=1}^{|w|}\left(\boldsymbol{V}_{q_i}^{<w_i}+\boldsymbol{V}_I\right)\boldsymbol{M}^{|w|-i}\boldsymbol{V}_F$$
$$\text{where}\quad q_1\in I, q_i\xrightarrow[\mathcal{D}]{w_i}q_{i+1}$$

が成り立つ．

証明. 定義より，

$$\begin{aligned}\mathrm{val}_S(w) &= \#\{f\in L\mid f\prec w\}\\ &= \#\{f\in L\cap A^{|w|}\mid f\prec w\}+C_L^{\leq}(|w|-1)\end{aligned}$$

さらに，補題 2.1 より

$$C_L^{\leq}(|w-1|)=\sum_{i=0}^{|w|-1}\boldsymbol{V}_I\boldsymbol{M}^i\boldsymbol{V}_F \tag{1}$$

が成り立つ．なので

$$\#\{f\in L\cap A^{|w|}\mid f\prec w\} \tag{2}$$

について，式を求めれば良い．これは，長さが $|w|$ かつ辞書順において w 未満の文字列による，$\mathcal{D}$ の全て受理計算にほかならない．よって

$$(2)=\sum_{i=1}^{|w|}(\boldsymbol{V}_{q_i}^{<w_i})\boldsymbol{M}^{|w|-i}\boldsymbol{V}_F$$

が満たされる．(1), (2) より

$$\begin{aligned}\mathrm{val}_{\mathcal{S}}(w) &= (1)+(2)\\ &= \sum_{i=0}^{|w|-1}\boldsymbol{V}_I\boldsymbol{M}^i\boldsymbol{V}_F+\sum_{i=1}^{|w|}(\boldsymbol{V}_{q_i}^{<w_i})\boldsymbol{M}^{|w|-i}\boldsymbol{V}_F\\ &= \sum_{i=1}^{|w|}\left(\boldsymbol{V}_{q_i}^{<w_i}+\boldsymbol{V}_I\right)\boldsymbol{M}^{|w|-i}\boldsymbol{V}_F\end{aligned}$$

□

系 3.1. 式 1 のように簡単な変形を行うことで，$\mathrm{val}_{\mathcal{S}}(w)$ は

- $2|w|-1$ 回のベクトル和
- $|w|-1$ 回のベクトル—行列積
- 1 回のベクトル—ベクトル積

によって計算可能である． □

†4 *length-lexicographic* 又は *radix* order とも呼ぶ

式 1　定理 3.1 の変形式 (Horner 法)

$$\begin{aligned}\mathrm{val}_{\mathcal{S}}(w) &= \sum_{i=1}^{|w|}\left(\boldsymbol{V}_{q_i}^{<w_i}+\boldsymbol{V}_I\right)\boldsymbol{M}^{|w|-i}\boldsymbol{V}_F\\ &= (\boldsymbol{V}_{q_1}^{w_1}+\boldsymbol{V}_I)\boldsymbol{M}^{|w|-1}\boldsymbol{V}_F+(\boldsymbol{V}_{q_2}^{w_2}+\boldsymbol{V}_I)\boldsymbol{M}^{|w|-2}\boldsymbol{V}_F+\cdots+(\boldsymbol{V}_{q_{|w|}}^{w_{|w|}}+\boldsymbol{V}_I)\boldsymbol{M}^{0}\boldsymbol{V}_F\\ &= \left(\cdots\left(\left((\boldsymbol{V}_{q_1}^{w_1}+\boldsymbol{V}_I)\,\boldsymbol{M}+(\boldsymbol{V}_{q_2}^{w_2}+\boldsymbol{V}_I)\right)\boldsymbol{M}+(\boldsymbol{V}_{q_3}^{w_3}+\boldsymbol{V}_I)\right)\boldsymbol{M}+\cdots+\left(\boldsymbol{V}_{q_{|w|}}^{w_{|w|}}+\boldsymbol{V}_I\right)\right)\boldsymbol{V}_F\end{aligned}$$

3.2 $\mathrm{rep}_{\mathcal{S}}$ の計算

基本的に $\mathrm{rep}_{\mathcal{S}}(n)$ の計算は，$\mathrm{val}_{\mathcal{S}}(w)$ の計算式 1 について各ベクトル $\boldsymbol{V}_{q_i}^{w_i}$ を満たす w_i を文字集合 A について線形探索をするだけである．ここで重要なのは，入力 n から対応する w の長さ $|w|$ を求めることである．$|w|$ を同定しないかぎり，ベクトルの線形探索はできない．n から $|w|$ を求めるには，

$$\max_{\ell}\left(C_L^{\leq}(\ell-1)\leq n\right)$$

を求めればよい．補題 2.1 において，$C_L(n)$ の計算が冪行列とベクトルの積のみで計算できることを示した．この節では $C_L^{\leq}(n)$ も同様の計算で行えることを示す．そのために，以下の性質を持つ正規言語を使う．

定義 3.2. 正規言語 $L\subseteq A^*$ について，その総和言語 $\widetilde{L}\subseteq(A\cup\{\$\})^*;\$\notin A$ を

$$\widetilde{L}:=L\{\$\}^*$$

と定義する．□

補題 3.1. 任意の正規言語 L について，

$$C_{\widetilde{L}}(n)=C_L^{\leq}(n)$$

が成り立つ．

証明. 全ての n について，長さが L の n 以下の全ての文字列 w に対応する，長さ n の文字列 $\widetilde{w}\in\widetilde{L}$ が存在し，かつそれ以外が $\widetilde{L}$ に含まれないことを示せば良い．

$$\forall n\in\mathbb{N}\quad\forall w\in L\left[|w|\leq n\Rightarrow\widetilde{w}=w\overbrace{\$\cdots\$}^{|w|-n}\in\widetilde{L}\right]$$

によって順方向を示せる．また，$\widetilde{w}$ は必ず $\$$ を含まない文字列 v の後ろに $\$$ が続く形になるので，末尾の $\$$ を除いた文字列 v は必ず L の要素である．よって逆方向も示せた．□

補題 3.2. 任意の *UFA* $\mathcal{A}=(Q,A,\delta,I,F)$ について，その総和言語を認識する *UFA* $\widetilde{\mathcal{A}};L(\widetilde{\mathcal{A}})=\widetilde{L}(\mathcal{A})$ は，たかだか 2 つの状態の追加で構成可能である．

証明. A に含まれない文字 $\$\notin A$ について，$\widetilde{\mathcal{A}}=(\widetilde{Q},A\cup\{\$\},\widetilde{\delta},\widetilde{I},\widetilde{F})$ を構成する．まず，$\mathcal{A}$ が空文字を受理しない場合 ($\epsilon\notin L(\mathcal{A})$)

$$\begin{aligned}\widetilde{Q} &:= Q\cup\{\widetilde{q}\}\\ \widetilde{\delta}(q\in Q,\sigma) &:= \begin{cases}\delta(q,\sigma)\cup\{\widetilde{q}\} & \delta(q,\sigma)\cap F\neq\emptyset\\ \delta(q,\sigma) & otherwise\end{cases}\\ \widetilde{\delta}(\widetilde{q},\$) &:= \{\widetilde{q}\}\\ \widetilde{I} &:= I\\ \widetilde{F} &:= \{\widetilde{q}\}\end{aligned}$$

$\mathcal{A}$ が空文字を受理する場合は，上の構成の後に，$\widehat{Q},\widetilde{I},\widetilde{F}$ にそれぞれ q_ϵ を追加する．

以上の構成から，$\mathcal{A}$ が受理状態に遷移する場合は $\widetilde{\mathcal{A}}$ も受理状態に遷移する．また，$\widetilde{\mathcal{A}}$ は一度受理状態 $\widetilde{q}$ に遷移すると，文字 $\$$ によってその後も (唯一) 自身に遷移することができる．よって $L(\widetilde{\mathcal{A}})=\widetilde{L}(\mathcal{A})$ を満たす．□

3.3 計算量

前節で説明したアルゴリズムを擬似コードとしてアルゴリズム 1,2 に，対応する計算量を表 1 に示す．計算量については，任意の桁数の整数乗算が定数時間で行える理想的な場合と，実際の乗算アルゴリズムで知られている計算量の両方を記述した．n 桁の整数の乗算において，$\mathrm{O}(n\log n\log\log n)$ の計算量を持つ Schnhage–Strassen アルゴリズム等が知られている．長整数乗算の計算量の比較は文献 [4]§9.5.8 が詳しい．

4 *ANS* による文字列圧縮

本章では，*ANS* による正規言語の文字列と数の一対一対応を応用して，正規言語間での文字列の一対一対応を提案し，圧縮に関する考察を行う．

アルゴリズム 1 $\mathrm{val}_{\mathcal{S}}$

入力: $w \in L$

結果: $n = \#\{f \in L \mid f \prec w\}$

1: $q \leftarrow q_0 \in I$
2: $\boldsymbol{V} \leftarrow \boldsymbol{V_0} \in \mathbb{N}^{|w|}$
3: **for** $i = 1 \to |w|$ **do**
4: $\quad \boldsymbol{V} \leftarrow \boldsymbol{V} + \boldsymbol{V}_q^{\sigma_i} + \boldsymbol{V}_I$
5: $\quad q \leftarrow \delta(q, w_i)$
6: $\quad$ **if** $i < |w|$ **then**
7: $\quad\quad \boldsymbol{V} \leftarrow \boldsymbol{V}\boldsymbol{M}$
8: $\quad$ **end if**
9: **end for**
10: $n \leftarrow \boldsymbol{V}^t \boldsymbol{V}_F$

アルゴリズム 2 $\mathrm{rep}_{\mathcal{S}}$

入力: $0 <= n < \#L$

結果: $w \in L \ \ s.t. \ \ \#\{w' \in L \mid w' \prec w\} = n$

1: $q \leftarrow q_0 \in I$
2: $\ell \leftarrow \max_{\ell} \left(C_L^{\leqq}(\ell - 1) \leq n \right)$
3: $m \leftarrow n - C_L^{\leqq}(\ell - 1)$
4: $w \leftarrow \epsilon$
5: **for** $i = \ell \to 1$ **do**
6: $\quad \sigma \leftarrow \min_{<} A$
7: $\quad$ **while** $\boldsymbol{M}_{\delta(q,\sigma)}^{\ell - i}{}^t \boldsymbol{V}_F \leq m$ **do**
8: $\quad\quad m \leftarrow m - \boldsymbol{M}_{\delta(q,\sigma)}^{\ell - i}{}^t \boldsymbol{V}_F$
9: $\quad\quad \sigma \leftarrow \mathrm{suc}(\sigma)$
10: $\quad$ **end while**
11: $\quad w \leftarrow w\sigma$
12: $\quad q \leftarrow \delta(q, \sigma)$
13: **end for**

表 1 計算量

	時間計算量	行列積	行列–ベクトル積	ベクトル積
$\mathrm{val}_{\mathcal{S}}(w)$	$\Theta(\lvert w\rvert \times \lvert\mathcal{D}\rvert^2)$ $\left[\mathrm{O}(\lvert w\rvert^2 \log \lvert w\rvert \log\log \lvert w\rvert \times \lvert\mathcal{D}\rvert^2)\right]$	0	$\lvert w\rvert - 1$	1
$\mathrm{rep}_{\mathcal{S}}(n)$	$\mathrm{O}(\ell \log \ell \times \lvert\mathcal{D}\rvert^3)$ $\left[\mathrm{O}(\ell^2 \log^2 \ell \log\log \ell \times \lvert\mathcal{D}\rvert^3)\right]$	$\mathrm{O}(\ell \log \ell)$	0	$\mathrm{O}(\ell \times \lvert A\rvert)$

$\ell = |rep_{\mathcal{S}}(n)|$ [] 内は乗算コストを考慮した場合．行列積, 行列–ベクトル積, ベクトル積は演算回数を表す．

4.1 *ANS* 間の基数変換

定義 4.1. *ANS* $\mathcal{S} = (L, A, <), \mathcal{S}' = (K, A', <')$ において，$\mathcal{S}$ から $\mathcal{S}'$ への基数変換 $P_{\mathcal{S}\to\mathcal{S}'} : L \to K$ を

$$\underset{\mathcal{S}\to\mathcal{S}'}{P}(w) := \mathrm{rep}_{\mathcal{S}'}(\mathrm{val}_{\mathcal{S}}(w))$$

によって定める． □

前章において，$\mathrm{val}_{\mathcal{S}}$ は文字列の長さに対して線形時間，$\mathrm{rep}_{\mathcal{S}}$ は対応する文字列の長さに対して順線形時間で計算可能なことを示した．よって，基数変換は両方の計算量の和で計算可能である．

補 足 4.1. *ANS* $\mathcal{S} = (L(\mathcal{D}), A, <), \mathcal{S}' = (L(\mathcal{D}'), A', <')$ において $P_{\mathcal{S}\to\mathcal{S}'}(w)$ の計算量は

$$\Theta(|w| \times |\mathcal{D}|^2) + \mathrm{O}(\ell \log \ell \times |\mathcal{D}'|^3)$$
$$\text{where} \quad \ell = |\,\mathrm{rep}_{\mathcal{S}'}(\mathrm{val}_{\mathcal{S}}(w))|$$

となる． □

4.2 圧縮率

定義 4.2. *ANS* $\mathcal{S} = (L, A, <), \mathcal{S}' = (K, A', <')$ について，基数変換 $P_{\mathcal{S}\to\mathcal{S}'} : L \to K$ の圧縮率 CR を

$$CR(P_{\mathcal{S}\to\mathcal{S}'}, n) := \frac{\log C_L^{\leqq}(n)}{\log C_K^{\leqq}(n)}$$

さらにその極限を

$$CR(P_{\mathcal{S}\to\mathcal{S}'}) := \lim_{n\to\infty} CR(P, n)$$

と定義する．特に，$CR(P_{\mathcal{S}\to\mathcal{S}'}) < 1$ を満たすような $P_{\mathcal{S}\to\mathcal{S}'}$ を圧縮と呼ぶ． □

$CR(P_{\mathcal{S}\to\mathcal{S}'}, n)$ は長さ n の文字列についての情報量の比である．単に圧縮率といった場合は，その極限を指す．圧縮率に関しては，明らかに次の系が成り立つ．

系 4.1. 任意の *ANS* $\mathcal{S} = (L, A, <)$ に対して，*ANS* $\mathcal{S}' = (K, A', <')$ への基数変換 $P_{\mathcal{S}\to\mathcal{S}'}$ は $K = A'^*$ において最も圧縮率が高い． □

ここで，数え上げ関数に関する以下の補題を説明する．これは次の定理の証明に用いられる．

補題 4.1. 任意のオートマトン $\mathcal{A}$ について，認識する言語を $L = L(\mathcal{A})$，その隣接行列のフロベニウス根を α とすると，

$$C_L^{\leq}(n) = \begin{cases} \Theta(n \times C_L(n)) = \Theta(n^{m+1}) & \alpha = 1 \\ \Theta(C_L(n)) = \Theta(n^m \alpha^n) & \alpha > 1 \end{cases}$$

が成り立つ．

証明. まず，$\alpha = 1$ について証明する．定理 2.1 の簡単な式変形より，

$$\begin{aligned} C_L^{\leq}(n) &= \sum_{i=0}^{n} \Theta(i^m \alpha^i) = \Theta\left(\sum_{i=0}^{n} i^m\right) \\ &= \Theta(n^{m+1}) \end{aligned}$$

また，$\alpha < 1$ のとき

$$C_L(n) = \Theta(n^m \alpha^n) \leq C_L^{\leq}(n) \tag{3}$$

さらに，

$$\begin{aligned} C_L^{\leq}(n) &= \sum_{i=0}^{n} \Theta\left(i^m \alpha^i\right) \\ &\leq n^m \sum_{i=0}^{n} \Theta(\alpha^i) = n^m \Theta(\alpha^n) \\ &= \Theta(n^m \alpha^n) \end{aligned} \tag{4}$$

(3), (4) より，上下界が漸近的に一致するため

$$C_L^{\leq}(n) = \Theta(C_L(n))$$

となる． □

準備が揃ったので，この説の最後に圧縮率に関する重要な定理を示す．

定理 4.1. 任意の *UFA* $\mathcal{A}, \mathcal{A}'$ の認識する言語上の，任意の *ANS* $\mathcal{S} = (L = L(\mathcal{A}), A, <), \mathcal{S}' = (K = L(\mathcal{A}'), A', <')$ において，$\mathcal{S}$ から $\mathcal{S}'$ への基数変換 $P_{\mathcal{S}\to\mathcal{S}'} : L \to K$ の圧縮率は

$$CR(P_{\mathcal{S}\to\mathcal{S}'}) = \begin{cases} \frac{\log \alpha}{\log \alpha'} & \alpha > 1, \alpha' > 1 \\ \frac{m+1}{m'+1} & \alpha = 1, \alpha' = 1 \\ \infty & \alpha > 1, \alpha' = 1 \\ 0 & \alpha = 1, \alpha' > 1 \end{cases}$$

となる．ここで，$\alpha(\alpha')$ は $\mathcal{A}(\mathcal{A}')$ の隣接行列におけるフロベニウス根，$m(m')$ は $\mathcal{A}(\mathcal{A}')$ の $\alpha(\alpha')$ を持つ，単一路で連結された強連結成分の最大数である．

証明. $\alpha > 1, \alpha' > 1$ について考える．圧縮率の定義と補題 4.1 より

$$\begin{aligned} CR(P_{\mathcal{S}\to\mathcal{S}'}) &= \lim_{n\to\infty} \frac{\log C_L^{\leq}(n)}{\log C_K^{\leq}(n)} = \lim_{n\to\infty} \frac{\log \Theta\left(C_L(n)\right)}{\log \Theta\left(C_K(n)\right)} \\ &= \lim_{n\to\infty} \frac{\log \Theta\left(n^m \alpha^n\right)}{\log \Theta\left(n^{m'} \alpha'^n\right)} \\ &= \lim_{n\to\infty} \frac{m \log n + n \log \alpha + \log c}{m' \log n + n \log \alpha' + \log c'} \\ &= \frac{\log \alpha}{\log \alpha'} \end{aligned}$$

が得られる．ここで c, c' は Θ における定数項である．

$\alpha = 1, \alpha' = 1$ や他の場合も，補題 4.1 から同様に計算することで各圧縮率が得られる． □

補足 4.2. 圧縮率 $CR(P)$ の計算は，$\mathcal{A}, \mathcal{A}'$ のフロベニウス根の計算に帰着される．補足 2.1 で述べたように，その計算量は $\mathrm{O}(|\mathcal{A}|^3), \mathrm{O}(|\mathcal{A}'|^3)$ である． □

5 まとめ

本稿では *ANS* における圧縮を提案し，その圧縮率を与えられた任意の正規言語について計算できることを示した．また，*ANS* における基本的な計算 $\mathrm{val}_{\mathcal{S}}, \mathrm{rep}_{\mathcal{S}}$ 及び数え上げ関数 $C_L, C_L^{\leq}$ のアルゴリズムの提案も行なった．

参 考 文 献

[1] Berth, V. and Rigo, M.: *Combinatorics, Automata and Number Theory*, Cambridge University Press, New York, NY, USA, 1st edition, 2010.

[2] Brouwer, A. E. and Haemers, W. H.: *Spectra of graphs (to appear).*, Berlin: Springer, 2012.

[3] Choffrut, C. and Goldwurm, M.: Rational Transductions and Complexity of Counting Problems, *Proceedings of the 17th International Symposium on Mathematical Foundations of Computer Science*, MFCS '92, London, UK, UK, Springer-Verlag, 1992, pp. 181–190.

[4] Crandall, R., Pomerance, C., Crandall, R., and Pomerance, C.: *Prime numbers: a computational perspective. Second Edition*, 2005.

[5] Goldberg, A. and Sipser, M.: Compression and ranking, *Proceedings of the seventeenth annual ACM symposium on Theory of computing*, STOC '85, New York, NY, USA, ACM, 1985, pp. 440–448.

[6] Hemachandra, L. A. and Rudich, S.: On the Complexity of Ranking., *J. Comput. Syst. Sci.*, Vol. 41, No. 2(1990), pp. 251–271.

[7] Lecomte, P. B. A. and Rigo, M.: Numeration systems on a regular language, *CoRR*, Vol. cs.OH/9903005(1999).

[8] Li, M. and Vitnyi, P. M.: *An Introduction to Kolmogorov Complexity and Its Applications*, Springer Publishing Company, Incorporated, 3 edition, 2008.

[9] Mkinen, E., Mkinen, E., and Mkinen, E.: Ranking and unranking left Szilard languages, Technical report, ISO/IEC JTC1/SC29/WGll/N2467, Atlantic City, 1997.

[10] Sakarovitch, J.: *Elements of Automata Theory*, Cambridge University Press, New York, NY, USA, 2009.

[11] Shur, A. M.: Combinatorial complexity of regular languages, *Proceedings of the 3rd international conference on Computer science: theory and applications*, CSR'08, Berlin, Heidelberg, Springer-Verlag, 2008, pp. 289–301.

[12] Shur, A. M.: Combinatorial Characterization of Formal Languages, *CoRR*, Vol. abs/1010.5456(2010).