1997年度日本オペレーションズ・リサーチ学会
春季研究発表会

等式系の基底解の列挙

01605630　東京都立大学　松井 泰子　MATSUI Yasuko

1　はじめに

本稿では、行列 A のすべての基底の逆行列と、等式系 $A\boldsymbol{x}=\boldsymbol{b}$ の基底解を列挙する解法を提案する。行列 A のサイズを $r\times n$、ランクを r とし、与えられた等式系は、少なくとも1つは解を持つものとする。提案する解法は、$\mathrm{O}(rn^2+\beta_A r^2)$ 時間、記憶容量 $\mathrm{O}(rn)$ で実行する。

A の基底に対応する、列ベクトル集合の族を $\mathcal{B}_A$ とする。以下では $|\mathcal{B}_A|=\beta_A$ とする。A の列ベクトルの集合を E_A とする。$(E_A,\mathcal{B}_A)$ は、マトロイドの公理系を満たすので、等式系 $A\boldsymbol{x}=\boldsymbol{b}$ の基底解の列挙は、行列マトロイドの基の列挙とみなす事が出来る。マトロイドの基の列挙に関しては、Avis & Fukuda が提案した解法が存在する [1]。彼らの解法は、基本サーキットオラクルを用いる事によって、基の列挙を $\mathrm{O}(\beta_A n(T+r))$ 時間と $\mathrm{O}(rn)$ 記憶容量で実行する。ただし、n は台集合に含まれる要素の個数、$\mathrm{O}(T)$ は基本サーキットオラクルに要する計算量である。彼らの解法を用いて、等式系の基底解の列挙を行なうと、基底解を一つ出力するたびに、$\mathrm{O}(nr^2)$ 時間を要する。

提案する解法は、基底解を一つ出力するのに必要な計算量は、$\mathrm{O}(r^2)$ であり、行列の列の数に依存せずに、基底解を出力する事が出来る。また、行列のランクが列の数よりも十分に小さい時、彼らの解法に比べて理論的計算量に関し、効率が良いと言える。

2　定義

サイズが $r\times n$ の行列 A の任意の基底を B とする。任意の列ベクトル $e\notin B$ に対し、$\mathcal{C}(B|e)\subseteq E_A$ は、$B+e$ 中に唯一のサーキットであるとする。任意の列ベクトル $e\in B$ に対し、$(E_A\setminus B)+e$ に含まれる唯一のコサーキットを co-$\mathcal{C}(B|e)$ と書く。

$E_A=\{e_1,\ldots,e_n\}$ を列ベクトルとする。任意の列ベクトル e_j に対し、e_j の添え字を j とする。index(e) で列ベクトル e の添え字を表す。与えられた行列 $A'\subseteq A$ に対し、添え字最小である A' の列ベクトルを A' のトップベクトルと呼び、top(A') と書く。同様に、添え字最大である A' の列ベクトルを A' のボトムベクトルと呼び、btm(A') と書く。行列のペア A',A'' に対し、以下の条件を満たすとき、A' は A'' より辞書的順序が大きいといい、$A'>_{\mathrm{lex}}A''$ あるいは $A''<_{\mathrm{lex}}A'$ と書く。(1) $0\leq k<j\leq n$ である任意の k に対し、$e_k\in A'$ である必要十分条件は $e_k\in A''$ であり、(2) $A'\ni e_j\notin A''$ を満たす整数 j が存在する。

本稿では、以下の性質を仮定する。

仮定1　次の条件をすべて満たす、整数列 $(j_0, j_1,\ldots,j_t)$ が存在する。(1) $0=j_0<j_1<\cdots<j_t=n$, (2) E_A の分割 $(G_1,G_2,\ldots,G_t)$ で、部分集合 $G_s=\{e_{j_{s-1}+1},e_{j_{s-1}+2},\ldots,e_{j_s}\}$ は、G_1 が A の基底であり、

$$\forall s\in\{2,\ldots,t\},\ G_s\ \text{は、}E_A\setminus(G_1\cup G_2\cup\cdots\cup G_{s-1})$$

の極大独立集合である。

以下簡単のために G_1 を B^* と書く。明らかに、B^* は、A の辞書的順序最大の基底である。

提案する解法では、基底の列挙に、列挙木という木構造を用いる。基底 B^* とは異なる任意の基底 B' に対し、$B'-f+g$ を $\phi(B')$ と書く。ただし、$f=\mathrm{btm}(B')$ で、$g=\mathrm{top}(\text{co-}\mathcal{C}(B'|f))$ とする。もし $B'\neq B^*$ かつ $\phi(B')=B'-f+g$ ならば、$g\in B^*\not\ni f$ である。よって、任意の基底 $B'\neq B^*$ に対して、$\phi(B')>_{\mathrm{lex}}B'$ であり、$\phi^t(B')=B^*$ を満たす正の整数 t が存在する。与えられた基底 $B'(\neq B^*)$ に対し、基底 B' は $\phi(B')$ の子であり、$\phi(B')$ は B' の親であると定義する。ここで、ノード集合 $\mathcal{B}$ と、枝集合 $\{(B,B')\in\mathcal{B}\times(\mathcal{B}\setminus\{B^*\})\mid B=\phi(B')\}$ のペア $(\mathcal{B},\phi)$ を列挙木とする。明らかに、基底 B^* は列挙木のルートに対応する。提案する解法では、暗に列挙木をたどりながら、A の基底を列挙する。

© 日本オペレーションズ・リサーチ学会. 無断複写・複製・転載を禁ず.

3 基底の列挙解法

A の任意の基底を B とし、$B' \ni f \notin B$ である B の子の基底 B' が存在するならば、列ベクトル f を B の子の枢軸ベクトルと呼ぶ。また、$B' \not\ni g \in B$ である B の子の基底 B' が存在するならば、列ベクトル g を B の親の枢軸ベクトルと呼ぶ。$B = B'-f+g$ は B' の親であるとする。g は co-$\mathcal{C}(B'|f)$ のトップベクトルであり、co-$\mathcal{C}(B'|f) =$ co-$\mathcal{C}(B|g)$ なので、g はまた、co-$\mathcal{C}(B|g)$ のトップベクトルである。部分集合 $\{e' \in B \mid e'$ はco-$\mathcal{C}(B|e')$のトップベクトル$\}$ を $\mathrm{Bst}(B)$ と書く。

ここで、基底 B の任意の親の枢軸ベクトルは、$\mathrm{Bst}(B)$ に含まれている。

次の定理は、子の集合を特徴付ける。

定理2 B を A の基底、$B' = B-g+f$ を B の子とする。列ベクトル f' が B' の子の枢軸ベクトルならば、f' はまた、B の子の枢軸ベクトルである。

定理3 B を A の基底とする。部分集合 $B' = B-g+f$ が B の子である必要十分条件は、B' は以下の条件を満たすことである。(1) B' は基底である, (2) $g \in \mathrm{Bst}(B)$, (3) $\mathtt{index}(\mathrm{btm}(B)) < \mathtt{index}(f)$。

集合 $\mathrm{Bst}(B)$ とサーキットを計算する方法があれば、基底 B のすべての子を求める事が出来る。すなわち、条件 (3) を満たす各列ベクトル f に対して、サーキット $\mathcal{C}(B|f)$ と、子の集合 $\{B-g+f \mid g \in \mathrm{Bst}(B) \cap \mathcal{C}(B|f)\}$ を生成すれば、B のすべての子を求められる。また、基底 B の部分集合 $\mathrm{Bst}(B)$ から、B の子の B' の部分集合 $\mathrm{Bst}(B')$ を簡単に構築出来る。

定理4 B を A の基底解、$B' = B-g+f$ を B の子とする。そのとき、$\mathrm{Bst}(B') = (\mathrm{Bst}(B) \setminus \mathcal{C}(B|f)) \cup \{E_A \in \mathrm{Bst}(B) \cap \mathcal{C}(B|f) \mid \mathtt{index}(e) < \mathtt{index}(g)\} \subseteq \mathrm{Bst}(B)$ である。

添え字最大である子の枢軸ベクトルに対しては、次の定理が成り立つ。

定理5 B の基底とし、F_* を添え字最大である、B の子の枢軸ベクトルとする。そのとき、F_* は、辞書的順序が最小の基底 B_* に含まれている。

<u>行列の基底の列挙解法 `base-enum`</u>

入力： 行列 A

出力： 行列 A 中のすべての基底

<u>Step 0</u>: $B := B^*, Bst := B^*, F := E_A \setminus B^*$ とする。B^* を出力し、$Q := \emptyset$ とする。

<u>Step 1</u>: $Q := \{(B, \mathrm{Bst}, F)\}$ とする。

<u>Step 2</u>: Q の最後の要素を (B, Bst, F) とし、Q から取り除く。

<u>Step 3</u>: もし、B^* に子が存在するならば、添え字が最大である B^* の子の枢軸ベクトル f_* を見つけ、Step 5 へ。

<u>Step 4</u>: Q が空ならば終了する。それ以外は Step 2 へ。

<u>Step 5</u>: $F_* := \{f \in F \mid \mathtt{index}(f) \leq \mathtt{index}(f_*)\}$, $F' := \emptyset$ とする。

<u>Step 6</u>: F_* が空ならば Step 4 へ。

<u>Step 7</u>: $f := \mathrm{btm}(F_*)$ とし、F_* から f を除く。$C := \mathcal{C}(B|f) \cap \mathrm{Bst}$ とする。

<u>Step 8</u>: C が空ならば、Step 6 へ。

<u>Step 9</u>: $F' := F' + f, \mathrm{Bst}' := \mathrm{Bst}$ とする。

<u>Step 10</u>: C が空ならば、Step 6 へ。

<u>Step 11</u>: $g := \mathrm{btm}(C)$ とし、C から g を除く。$B-g+f$ を出力し、$\mathrm{Bst}' := \mathrm{Bst}' - g$, $B := B-g+f$ とする。

<u>Step 12</u>: Q の最後に (B, Bst, F') を加える。

<u>Step 13</u>: $B := B-f+g$ とし、Step 8 へ。

参考文献

[1] D. Avis and K. Fukuda, *Reverse search for enumeration*, Discrete Applied Mathematics (1996), pp.21-46.

© 日本オペレーションズ・リサーチ学会. 無断複写・複製・転載を禁ず.