数理解析研究所講究録
1138巻 2000年 180-188

戦術に忠実な並列Buchberger算法

岩手大学工学部　鈴木　正幸(Masayuki Suzuki) *

1　はじめに

グレブナー基底を計算するBuchberger算法は，代数方程式の求解の基本算法である．算法の改良や計算機の性能向上に伴い，大きな問題が解けるようになってはきたが，実用に用いられるためには，性能向上が望まれており，そのため，Buchberger算法の並列化が試みられている．

Buchberger算法では，戦術により，基底を構成する候補のペア数や数係数膨張が大きく変化し，計算時間に大きく影響することが知られている．したがって通常の並列化とは異なり，戦術に忠実な並列化を行うことが重要である．

戦術に忠実な並列化には[1, 6]がある．[1]では，中間基底候補のペアに関する粗粒度の並列化を行う．マスタプロセスは中間基底を広報し，複数のワーカプロセスがS多項式の計算と正規化を担当する．マスタは戦術を保持するためにワーカの実行順序を制御する．基底の生成順序に依存する逐次性のために，並列性が大きくできないことが問題である．

[6]では，S多項式をある一つの中間基底で簡約する計算を並列化する．多項式の数係数が大きい場合に有効である．逐次計算部分が残ることと，細粒度の並列化であるために，並列度を大きくした場合に通信時間が問題となる．

上記の二つの戦術に忠実な並列化には，それぞれに性能限界が存在する．本稿では，この二つの並列性を組み合わせた並列Buchberger算法を提案し，より大きな並列性を引き出せるかの検討を行う．

また，斉次なイデアルのグレブナー基底を求める場合は，中間基底が次数毎に求められることが知られている[2]．非斉次な場合は，sugar次数について同様の事が言える．この次数構造は，中間基底計算に独立性を持ち込み，次数毎に十分な中間基底が存在すれば，大きな並列化が期待できる．並列度の向上のためにGröbner基底の次数構造についても考察する．

以下[2]に従い，本稿で用いる記法を説明する．

係数体 R 上の多項式環を $R[\underline{x}] = R[x_1, \ldots, x_n]$ と表す．

*suzuki@cis.iwate-u.ac.jp

T は $x_1,\ldots,x_n$ からなる項の集合を表す．T 上の admissible な順序を $<$ とし，$t = x_1^{\alpha_1}\cdots x_n^{\alpha_n} \in T$ に対し，全次数を $\deg(t) \stackrel{def}{=} \sum_{i=1}^n \alpha_i$ とする．

$f = \sum_{i=1}^n c(\alpha_1,\cdots,\alpha_n)x_1^{\alpha_1}\cdots x_n^{\alpha_n} \in R[\underline{x}]$ に対し，

- f の monomial の集合: $M(f) \stackrel{def}{=} \{c(\alpha_1,\cdots,\alpha_n)x_1^{\alpha_1}\cdots x_n^{\alpha_n} | c(\alpha_1,\cdots,\alpha_n) \neq 0\}$,
- 項の集合: $T(f) \stackrel{def}{=} \{x_1^{\alpha_1}\cdots x_n^{\alpha_n} | c(\alpha_1,\cdots,\alpha_n) \neq 0\}$,
- 全次数: $deg(f) \stackrel{def}{=} \max\{\deg(t) | t \in \mathrm{T}(f)\}$,
- 頭項: $\mathrm{HT}(f) \stackrel{def}{=} \max(\mathrm{T}(f))$, 頭 **monomial**: $\mathrm{HM}(f) \stackrel{def}{=} \max(M(f))$, 頭係数: $\mathrm{HC}(f) \stackrel{def}{=} \mathrm{HM}(f)$ の係数, とする．

$F \subset R[\underline{x}]$ に対し，$\mathrm{HT}(F) \stackrel{def}{=} \{\mathrm{HT}(f) | f \in F\}$, $\mathrm{HM}(F) \stackrel{def}{=} \{\mathrm{HM}(f) | f \in F\}$, $\mathrm{T}(F) \stackrel{def}{=} \{\mathrm{T}(f) | f \in F\}$, とする．

$f, g, p \in R[\underline{x}]$ に対し，

- f は p で g に簡約可能であることを $f \xrightarrow{p} g$ と表し，$\exists t \in \mathrm{T}(f), \exists s \in T\ s * \mathrm{HT}(p) = t$ かつ $g = f - \frac{a}{\mathrm{HC}(p)} * s * p$ であることする．ここで a は f の中の t の係数とする．
- $f \xrightarrow{P} g$は，$\exists p \in P\ f \xrightarrow{p} g$ であることとし，
- $\xrightarrow{P}{}^*$ は $\xrightarrow{P}$ の推移的閉包を意味することとする．f の $\xrightarrow{P}{}^*$ による正規形を $f\downarrow_P$ と表す．
- f は P で g に t 頭項簡約可能であるとは，$f \xrightarrow{P} g$ かつ $\mathrm{HT}(g) < \mathrm{HT}(f)$ とする．

f と g に対する **S** 多項式を

$$\mathrm{SPOL}(f,g) \stackrel{def}{=} \mathrm{HC}(g)\frac{lcm\ (\mathrm{HT}(f),\mathrm{HT}(g))}{\mathrm{HT}(f)}f \ - \ \mathrm{HC}(f)\frac{lcm\ (\mathrm{HT}(f),\mathrm{HT}(g))}{\mathrm{HT}(g)}g$$

とする．

sugar 次数 [5] $\deg_S$ は演算に定義され，

- 初期多項式 f_i に対し，$\deg_S(f_i) \stackrel{def}{=} \deg(f_i)$
- $\deg_S(p+q) \stackrel{def}{=} \max(\deg_S(p), \deg_S(q))$
- $t \in T$ に対し，$\deg_S(t*p) \stackrel{def}{=} (\deg(t) + \deg_S(p))$.

とする．

2 Buchberger算法と並列化

以下に，$f_1, \ldots, f_l$ が作るイデアルの Gröbner 基底を計算する Buchberger 算法を示す．

Buchberger 算法

Input: $F = \{f_1, \ldots, f_l\}$
Output: Gröbner 基底 G of $Ideal(F)$

$PairQ \longleftarrow \phi$;
$G \longleftarrow \phi$;
foreach $(f_i \in F)$ {
  $PairQ \longleftarrow UpdatePairQ(PairQ, f_i, F)$;
  $G \longleftarrow UpdateBase(G, f_i)$;
}
while ($PairQ \neq \phi$) {
  $(g_i, g_j) \longleftarrow$ *select an element of* $PairQ$;
  $PairQ \longleftarrow PairQ \setminus \{(g_i, g_j)\}$;
  $g_k \longleftarrow \mathrm{SPOL}(g_i, g_j)\downarrow_G$;
  if $g_k \neq 0$ {
    $PairQ \longleftarrow UpdateQ(PairQ, g_k, G)$;
    $G \longleftarrow UpdateBase(G, g_k)$;
  }
}

算法の概要と戦術

- G は中間的な基底の集合，$PairQ$ は新たな基底を構成可能な中間基底の組 (ペア) の集合，を表している．
- $PairQ$ から一つのペア (g_i, g_j) を選ぶ．この選び方を選択戦術と呼ぶ．
- $\mathrm{SPOL}(g_i, g_j)$ の現在の中間基底での正規形 g_k を求める．簡約基底の選び方の順序や簡約法を簡約化戦術と呼ぶ．
- g_k が 0 でなければ，

- ペア削除戦術により，新たなペアの生成と，不必要なペアの削除をおこない ($UpdateQ$)，
- 中間基底に追加し，基底削除戦術により不必要な中間基底の削除をおこなう ($UpdateBase$)，

- $PairQ$ が空になった時点で算法は停止し，G に Gröbner 基底が求まる．

2.1 Buchberger 算法の並列性

Buchberger 算法の計算上の問題点は，ペアの個数の組み合わせ的な膨張と，中間基底の数係数の膨張である．ペアの個数の膨張を防ぐために，いくつかの選択戦術が考えられており，選択戦術を保持したまま，ペアの個数に関する並列性の導入が必要となる [1]．

野呂ら [6] は, 数係数の膨張による計算時間の増大を，並列計算により減らせることを示した．筆者 [8] は，共有メモリを用いて更に高速化を行った．一つの基底による S 多項式の簡約 ($\mathrm{SPOL}(g_i, g_j)\downarrow_{\{g_k\}}$) を $\mathrm{SPOL}(g_i, g_j)$ や g_k を分割し，並列計算する．これを一簡約並列と呼ぶ．この方式では，

- 全ての戦術を保持したまま並列計算が可能であるが,
- 細粒度の並列化であり，有効となるのは数係数が大きくなった場合に限る，
- 逐次部分が残る．

この方式は，大規模な Gröbner 基底計算 [7] において，並列度が中規模 (≤ 20) 程度であれば良い性能を示している [6, 8]．しかし計算の逐次部分，通信コストのために，性能限界を持つ．

[1] では，選択戦術を忠実に守りつつ，ペアに関する簡約 ($\mathrm{SPOL}(g_i, g_j)\downarrow_G$) を並列に行っている．$G$ を共有し，複数のワーカが別々の簡約を行う．以後，この並列化をペア並列と呼ぶ．ペア並列では，

- 逐次部分がないが，
- 中間基底の生成順序を保つため，S 多項式の生成，簡約化に待ちが生ずる，
- 無駄な計算 (0 簡約される基底を用いたペア) が生ずる．

この方式では, 中間基底の生成順序による待ちがボトルネックとなり，様々な問題に対して性能限界が生じることが報告されている．この論文中，斉次な基底計算の場合，生成順序による待ちが大幅に減らせ，高い並列性能を示すことが言及されているが，その性能は示されていない．

3 並列算法の組合わせによる並列度の向上

前章の二つの並列化算法はそれぞれ性能限界を持つ．しかし，その限界を持つ原因は異なるので，二つを組み合わせることにより，並列性能向上が期待できる．

提案する算法の基本的な考え方は，

- ペア並列度を検出し，
- ペア並列度が低い場合に，一簡約並列を行う

であるが，ペア並列度の検出は計算中には行えない．そこで，まず同じ戦術の modular 計算を行い，0 簡約される基底，基底の生成順序と簡約依存性をあらかじめ求める．この手法は，[3] で用いられていて，ペア並列度は低いことが報告されている．つまり，ペア並列度だけでは高い性能向上は見込めない．そこで，

- modular 計算により基底の生成順序と簡約依存性をあらかじめ求め，並列計算可能なブロックに分ける．(これを並列計算のシナリオと呼ぶ)
- シナリオにより，ブロック内をペア並列実行するが，並列度が投入できるプロセッサ台数より小さい場合，全プロセッサが計算に参加できるように，一簡約並列を併用する．

4 d-Gröbner 基底によるペア並列度の向上

選択戦術として斉次化あるいは sugar を用いる場合には，あらかじめ決めることができるペア並列度が存在する．

4.1 d-グレブナー基底

定義 1 *S* 多項式の全次数 *(*または *sugar* 次数*)* d で打ち切った *Buchberger* 算法の結果を G_d とする．この G_d のことを *d*-グレブナー基底という．

定理 2 斉次多項式 $f_1, \ldots, f_n$ に対する d-グレブナー基底は以下の性質を持つ：

1. $\deg(f) < d$ な f に対し，$\xrightarrow{G_d}{}^*$ が定義される．
2. $\forall p \in \mathcal{I}\ \deg(p) \leq d \Rightarrow p \xrightarrow{G_d}{}^* 0$
3. $\forall f, g \in G_d\ \deg(\mathrm{HT}(f), \mathrm{HT}(g)) \leq d$ に対し，$\mathrm{SPOL}(f, g) \xrightarrow{G_d}{}^* 0$

$\forall d > d_\infty\ G_d = G_{d_\infty}$ となる d_∞ が存在する．□

系 **3** 任意の多項式に対し，定理 *2* の deg を $\deg_S$ で置き換えて，性質 *1, 2, 3* および d_{infty} の存在が成り立つ．

定理より d-グレブナー基底は，

$$G_0 \to G_1 \to \cdots \to G_d \to G_{d+1} \to \cdots \to G_{d_\infty} = \cdots$$

のように計算でき，$G_d = G_{d-1} + \{d\text{-次式}\}$ となる．

4.2 d-グレブナー基底の並列性

前節の定理より，G_{d-1} が求まっていて，G_d を求める場合は，次の事が言える．

1. G_d に追加される基底は，$\mathrm{SPOL}(g_i, g_j)$, $g_i, g_j \in G_{d-1}$, より作られ，基底候補の S 多項式に依存性はない．
2. $\mathrm{SPOL}(g_i, g_j)\downarrow_{G_{d-1}}$ の計算にも依存性はない．
3. 上の計算後，$\mathrm{SPOL}(g_i, g_j)\downarrow_{G_d}$ の計算は，1,2 で作られた d-次基底のみの相互簡約で求められる．

つまり，S 多項式の並列生成，G_{d-1} に関する並列簡約，が可能である．d-次基底の相互簡約には基底間の依存性が存在するが，これは一簡約並列実行可能である．

5 実装と性能 (予測)

前章により，斉次あるいは sugar を用いた並列算法は，

- modular 計算によりシナリオを作成し，
- d-の S 多項式 s_i を並列生成し，
- $s_i\downarrow_{G_{d-1}}$ を並列計算する．ペア並列度が足りない場合に，一簡約並列を併用する．
- $s_i\downarrow_{G_{d-1}}$ 同士の相互簡約を一簡約並列計算する．

sugar 値	基底数	時間
11	14	21.22
12	24	89.32
13	37	359.4
14	63	2962
15	101	84620
16	168	572100
17	1	28900
18	1	12800
20	1	30000
total	442	731800

表 16: McKay 計算の sugar 毎の基底数と実行時間 (秒)

d-Gröbner	gr_mod_main	asir F_4
180 (409)	240 ()	126 (432)

表 17: McKay 計算の modular 計算時間 (秒) 比較

となる．asir 上で逐次版の d-グレブナー基底計算を実装し，その実行過程を検討し，並列版を現在実装中である．

表 16 に，McKay[7] 問題に対し，選択戦術として sugar 戦術をもちいて実行した結果をしめす．8 台の場合の一簡約並列性能は，5.6, ほぼ 7 割である．表中の基底数が，シナリオを用いて計算した場合のペアの並列度になる．計算時間のもっともかかる，sugar 値 15, 16 辺りのペア並列度はかなり大きい．sugar 値 17 以上では, ペアの並列度は 1 で，ペア並列だけでは十分な性能向上ははかれないことがわかる．

表 17 に，同じ問題の modular 基底を，d-Gröbner 基底算法を用いて計算した結果を，asir の gr_mod_main, F_4 の結果とともに示す．括弧内は g.c. 時間である．この計算は並列化のシナリオを作成する部分に相当する．まだ asir F_4 の性能には及ばないが，gr_mode_main に比べて数割早くなっていることがわかる．

表 18 に，d-グレブナー基底計算中の各 S 多項式の G_{d-1} に関する簡約時間，d-次の基底間の相互簡約にかかる時間を示す．G_{d-1} に関する簡約時間が支配的であり，並列化した場合，ペア並列度が実行時間に大きく寄与することがわかる．

一簡約並列算法 (共有メモリ版) の性能は，12 のプロセッサで 8 程度の並列性能を得ている [8]．d-グレブナー基底計算の並列版は実装中であるので，算法の組合わせによる全体

sugar 次数 d	全実行時間		SPOL(,)$\downarrow_{G_{d-1}}$		$\downarrow_{G_d \setminus G_{d-1}}$	
total	180.7	(409.3)	148.1	(321.2)	32.2	(87.3)
11	0.8	(3.3)	0.7	(3.1)	0.1	(0.2)
12	2.5	(9.6)	2.2	(8.4)	0.3	(1.2)
13	7.6	(27.7)	6.8	(24.5)	0.8	(3.2)
14	19.4	(58.7)	16.6	(49.4)	2.7	(9.1)
15	48.7	(134.4)	39.6	(104.3)	9.0	(29.8)
16	74.6	(150.3)	54.9	(106.2)	19.4	(43.6)
17	25.2	(22.7)	25.2	(22.7)	0.0	(0.0)
18	1.0	(0.9)	1.0	(0.9)	0.0	(0.0)
19	0.1	(0.0)	0.1	(0.0)	0.0	(0.0)
20	0.3	(0.2)	0.3	(0.2)	0.0	(0.0)
21	0.4	(0.3)	0.4	(0.3)	0.0	(0.0)

表 18: d-Gröbner 基底計算時間 (秒) の内訳

性能を示すことはできないが，相互簡約の部分の並列化，ペア並列性の低い部分，が高速化でき，良い性能が得られるることは明らかだろう．

参 考 文 献

[1] Attardi, G., Tracerso, C.,: Strategy–Accurate Parallel Buchberger Algorithms, J.Symb. Comp., 21/4-6 (1997), 411–426

[2] Beker,T., Weispfenning, V.: Gröbner Bases. GTM bf 141, Springer-Verlag, 1993

[3] Faugére, J.C.: Parallelization of Gröbner basis *Proc. PASCO'94*, 1994, 124–132

[4] Faugére, J.C.: A new efficient algorithm for computing Gröbner bases (F_4), *Journal of Pure and Applied Algebra* **139**(1–3), 1999, 61–88

[5] Giovini, A., Mora, T., Niesi, G., Robbiano, L., Traverso, C.: "One sugar cube, please" OR Slection strategies in the Buchberger algorithm, Proc. ISSAC'91, 1991, 49–54

[6] Noro, M., Kando, T., Takeshima, T.: Solving a large scale problem by parallel algebraic computation on AP3000, *Research Report ISIS-RR-97*, FUJITSU LABS, 1997

[7] Noro, M., Mckay, J.: Computation of replicable functions on Risa/Asir, *Proc. PASCO'97*, ACM Press, 1997, 130–138

[8] 鈴木正幸: 分散共有メモリを用いた並列 Gröbner 基底計算の性能評価, 第 8 回数式処理大会, 1999